FACULTAD DE ESTUDIOS ESTADÍSTICOS

MÁSTER EN MINERÍA DE DATOS E INTELIGENCIA DE NEGOCIOS

Curso 2018/2019

Trabajo de Fin de Máster

TITULO:

LEGAL DATA MINING: Análisis y predicción de sentencias judiciales.

Alumna: Mélani Espinosa Villar

Tutora: Aida Calviño Martínez.

Septiembre de 2019

UNIVERSIDAD COMPLUTENSE
MADRID

Índice

Pág

Índice de Figuras

Índice de Tablas

1. Introducción

Hoy en día, es muy común relacionar la Minería de Datos con el sector bancario, o con otros campos como la medicina o el marketing, pero no se limita a estos, sino que se aplica en sectores tan dispares como el Derecho, conociéndose en este caso como *Legal Data Mining*. De hecho, en 2018 tuvo lugar el primer "*International Workshop on Legal Data Analytics - LeDAM",* en Turín, donde se definió a la minería de datos legales como la subárea de la minería de datos aplicada a textos legales, tales como legislación, jurisprudencia, patentes y trabajos académicos. Los sistemas de minería de datos legales son clave para proporcionar un acceso más fácil a la ley tanto para personas no pertenecientes al mundo del Derecho como para profesionales legales.

También encontramos otras definiciones relacionadas con este área que vienen a suponer algo similar y utilizan nombres diferentes, como *Legal Prediction*, *Legal Data Analysis*, y la más conocida, *Jurimetrics*.

La Jurimetría es la aplicación de métodos cuantitativos, generalmente estadísticos, a la Ley. Al utilizar un enfoque cuantitativo para analizar las decisiones judiciales, es posible identificar patrones y valores atípicos, lo que permite pronosticar el resultado de una decisión judicial y, por lo tanto, hacer que la ley sea más predecible (Armonas et al., 2017).

Debido a su naturaleza, la Jurimetría requiere un enfoque multidisciplinar en el que son necesarios conocimientos legales, estadísticos y computacionales, que generalmente no posee una misma persona y hace necesario el trabajo en equipo de expertos de las distintas ramas.

Dicho enfoque también requiere sistemas informáticos capaces de almacenar y procesar la decisión judicial, lo cual exige el acceso en tiempo real a las resoluciones judiciales que se llevan a cabo cada día en el territorio en el que deseemos aplicar esta técnica.

Aunque lo parezca, la definición de jurimetría no es algo nuevo, pues en el siglo pasado se definía como "la investigación científica de los problemas legales" (Loevinger, 1971). Sin embargo, en España, debido a la llegada tardía de documentos legales electrónicos y la dificultad para acceder a la información de los tribunales, no fue hasta 2017 cuando la empresa Wolters Kluwer, con la ayuda de Google, lanzó la herramienta "Jurimetría" y comenzó así a conocerse la existencia de esta posibilidad.

Wolters Kluwer define "Jurimetría" como «*la herramienta de analítica jurisprudencial más innovadora del mercado que permite definir la estrategia procesal más idónea para el éxito del caso, a través de indicadores gráficos interactivos, basados en el análisis cognitivo de millones de decisiones judiciales*». Dicha herramienta evalúa los parámetros del caso, del Juez o Magistrado (posturas del juez en función del tipo de caso, veces que ha dictado sentencia sobre un determinado tema…) y del abogado (cuántos casos similares ha llevado y su resultado). Pese a ello, aún son muchos, por desgracia la mayoría de los agentes legales, los que desconocen las grandes posibilidades estratégicas para el sector legal que supone aplicar la minería de datos al Derecho.

1.1 Aclaraciones sobre conceptos jurídicos

Para comprender bien la envergadura de este proyecto es necesario aclarar una serie de conceptos que pueden no ser conocidos por todo el mundo y nos servirán para justificar por qué vamos a limitar el ámbito de actuación y el por qué tomar con cautela las investigaciones y resultados que se hayan obtenido hasta ahora en otros países.

La mayoría de las investigaciones a las que se ha podido acceder pertenecen a investigadores estadounidenses. El modelo de acceso de datos judiciales en Estados Unidos es inmenso, no sólo se dispone del acceso a las sentencias, sino a todo el expediente judicial (demandas, escritos procesales, grabaciones, pruebas...). La cantidad de contenidos disponibles permiten hacer estadísticas sobre múltiples campos, permitiendo, no sólo trabajar en la predictibilidad de las sentencias, si no también en estadísticas de éxito de abogados, jueces, peritos... por tipologías de asuntos, temáticas... todo un filón para la minería de datos, sobre todo si lo comparamos con el acceso a la información judicial con la que contamos en España, donde sólo se tiene acceso a las resoluciones del Tribunal Supremo, Tribunales Superiores de Justicia y Audiencias Provinciales y algunas primeras instancias. Con esta información, se pueden extraer algunas estadísticas que nos permitan analizar tendencias sobre cómo resuelven los tribunales nacionales pero parece que no de manera tan precisa como se podría hacer disponiendo de los mismos datos que ofrecen en otros países.

La Jurimetría está muy valorada y desarrollada en otros países como EEUU, Canadá, Reino Unido… pero estos países no siguen el mismo derecho que en España. Existen dos grandes sistemas legales aplicados en el mundo: *Civil Law* y *Common Law*. El primero se aplica en toda Europa y Latinoamérica (a excepción de Guyana y Belice) y el segundo en los países anglosajones (Estados Unidos, Canadá, Reino Unido y Australia), entre otros. Existen otros dos sistemas legales menos extendidos como el Derecho Musulmán y un Sistema Bijudicial que mezcla el *Common Law* y el *Civil Law* en los que no nos detendremos porque no resultan relevantes para este estudio.

Es importante conocer esta distinción porque las investigaciones existentes en *Legal Predictive* o *Jurimetrics* que vamos a exponer posteriormente se han realizado en países donde está implantado el sistema del *Common Law*, originario de la tradición inglesa, el cual dista mucho de parecerse a las del *Civil Law*, desarrollado a partir del Derecho Romano y por el que se rige nuestro sistema legal.

Pese a que hay opiniones de todo tipo, no podemos considerar un sistema mejor que otro, pero sí explicar las diferencias.

El *Civil Law*, como ya hemos dicho, embebe del Derecho Romano y se basa en los códigos legales, en la Ley escrita. Son uno o varios jueces/magistrados los que enjuician los litigios y sólo en casos tasados de Derecho Penal participa el jurado popular de legos en Derecho. Es el juez el que dirige en todo momento el proceso y decide, en base a la ley, el orden en el que se celebra el proceso, cuándo habla cada una de las partes, etc. Además, utiliza un razonamiento deductivo (de lo general – la Ley - a lo específico – el caso concreto-). Siempre debe enjuiciar en función de la Ley y no de su propia opinión, pues uno de los deberes de todo juez es la imparcialidad y el sometimiento a la Ley.

Por su parte, el *Common Law* se basa en la Ley del precedente, esto es, el juez aplica un razonamiento inductivo: si existe ya un caso similar con una sentencia se aplica lo

mismo para el presente caso (se conoce como "*case law*"). Aquí el Juez hace el Derecho mientras que en el Civil Law el Juez aplica el Derecho, pues los códigos que existen son la fuente principal del Derecho. La mayoría de leyes y códigos que existen en países con *Common Law* se han creado con la recopilación de las decisiones que los jueces han ido tomando para cada tipo de caso, existiendo únicamente principios muy laxos que no se pueden obviar. Para predecir los resultados la Ley de Precedente es muy útil ya que si el caso se amolda a algún precedente sabrás cuál es el fallo que dará el magistrado, no así en el *Civil Law*, donde una pequeña circunstancia puede cambiar el sentido del fallo, al igual que las leyes, que se modifican continuamente, no pudiendo aplicarse normativa no vigente a un caso presente.

En el caso del tribunal, también se aplica para delitos, pero no en casos tasados y de manera obligatoria como en el Derecho español, sino que es el acusado el que elige si quiere que le enjuicie un jurado popular o un juez y son los abogados quienes eligen al jurado (existiendo departamentos de selección de testigos en los grandes despachos que se encargan de investigar a los candidatos a jurado para escoger los más convenientes para su caso concreto utilizando, entre otras cosas, la huella digital de éstos). Los abogados (y si son necesarios fiscales) adquieren más protagonismo en el *Common Law*. El Jurado siempre elije en función de lo que les parece más justo, sin basarse en ninguna ley. El juez del *Common Law* debe respetar una reglas escritas pero si no chocan con éstas puede estimar lo que estime oportuno para el caso, teniendo más margen a la subjetividad e interpretación que en el *Civil Law*.

Por todo lo expuesto, es necesario delimitar el ámbito de actuación para nuestro estudio, ya que no se podrá aplicar una forma uniforme de análisis y predicción para todas las jurisdicciones e instancias del Derecho español. Dicha delimitación se llevará a cabo en el epígrafe de metodología.

1.2 Delimitación del ámbito de actuación

Para realizar la extracción de los datos acudiremos a una base de datos jurídica y realizaremos técnicas de *web scrapping* que nos permitan obtener una gran cantidad de sentencias en poco tiempo. La traducción literal de *scrapping* es "raspado" y con esta técnica de *web scrapping* se permite navegar de manera automática por una *web* y extraer de ella la información que deseamos. Al *software* programado para *scrapear* se le suele denominar *bot, spider* (o araña en español) o *crawler*.

Un ejemplo habitual de *web scrapping* son los comparadores de precios. Por ejemplo *Skyscanner*, que es un comparador de vuelos, procesa la consulta que le hace el usuario y lo traduce realizando consultas instantáneas en todas las webs de compañías aéreas. Como resultado nos devuelve un listado con todas las opciones que tenemos según nuestros criterios para que las podamos comparar. Mientras que realizar esa acción de comprobar una a una todas las páginas de aerolíneas con los filtros que deseamos nos llevaría bastantes minutos, estos comparadores lo hacen en cuestión de segundos. Si esto lo llevamos al trabajo que queremos realizar, donde las sentencias se cuentan por cientos, realizar la búsqueda y extracción manual de los datos una a una nos llevaría no solo unos minutos u horas, sino días. Con el uso de *web scrapping* podemos reducir esta tarea a minutos o pocas horas.

La única base de datos jurídica gratuita pertenece al Centro de Documentación Judicial (CENDOJ) del Consejo General del Poder Judicial[1]. En ella, se puede filtrar por numerosos criterios, entre los que encontramos la jurisdicción (penal, civil, social, etc.), el tipo de resolución judicial (auto, sentencia, providencia), el tipo de órgano (juzgados de primera instancia, Audiencia Provincial, Tribunal Superior de Justicia, etc.), entre otros. El resultado que devuelve son las resoluciones judiciales que cumplen con los criterios establecidos, debiendo acceder a cada una de ellas para conocer el contenido y siendo la respuesta a cada *click* en ellas la descarga del *pdf* relativo a la resolución seleccionada. Esto supone una desventaja de tratamiento de los datos, ya que tendríamos que contar con un *crawler* que permitiera la descarga de todos los documentos en formato *pdf*, lo cual supone además una alta capacidad de memoria del equipo en el que vamos a trabajar, a lo que hay que añadir herramientas necesarias para procesar texto posteriormente desde documentos en *pdf*.

Ante la poca utilidad que proporcionan documentos en formato *pdf*, dado que no sabemos la utilidad de cada uno de ellos a priori y la necesidad de depuración, se opta por utilizar otra base de datos alternativa y disponible en las bases de datos legales de la Universidad Complutense de Madrid para evitar los problemas de almacenamiento y procesamiento intentando encontrar resoluciones judiciales que puedan ser consultadas directamente desde la web y que permitan crear nuestra base de datos de manera automática con un programa, según encontremos una sentencia que cumpla con nuestros criterios de búsqueda y agregando a dicha futura base de datos únicamente los campos que consideremos oportunos.

Para cumplir con todo lo anterior, tenemos que definir en primer lugar las características que van a poseer las resoluciones judiciales que queremos analizar para posteriormente encontrar una base de datos jurídica que nos permita acceder de manera automática para aplicar técnicas de *web scrapping* y crear, a su vez, la que será nuestra base de datos con los campos correspondientes para cada uno de los registros que cumplan nuestros requisitos.

Para el primero de nuestros requisitos hay que exponer las diferentes jurisdicciones que existen dentro de nuestro ordenamiento, es lo que se conoce como criterio material:

- Penal: para enjuiciar los delitos que se cometen, distinguiendo en este caso la parte de instrucción (que es aquella en la que se investiga el delito) y la fase de enjuiciamiento que es, como su propio nombre indica, en la que se trata de enjuiciar el asunto, no puedo ser el mismo juez el que instruye y enjuicia.
- Civil: que regula las relaciones jurídicas relacionadas con Derecho Civil, siendo a nivel general aquel que regula la relaciones entre particulares o empresas. Se trata de asuntos de contratos, obligaciones, familia, sucesiones, etc.
- Contencioso-Administrativa: para los problemas que surgen entre el ciudadano y las Administraciones Públicas.
- Social: conoce de los asuntos que tienen lugar entre los trabajadores y empresas privadas (ya que las relaciones entre trabajadores y empresas públicas se llevan por el orden contencioso-administrativo) y asuntos relacionados con las

[1] http://www.poderjudicial.es/search/indexAN.jsp

prestaciones del Sistema de la Seguridad Social (pensiones, accidentes de trabajo, etc.).

Existen otras dos jurisdicciones menos relevantes y frecuentes que son la Militar (sólo aplicable a militares) y la Constitucional (para supervisar la constitucionalidad de las leyes y el respeto de los Derechos Fundamentales contenidos en la Constitución).

En estas jurisdicciones, las leyes y criterios aplicables son muy diferentes, por lo que no se puede analizar desde la misma perspectiva cada una de ellas y habría que inspeccionar cada orden jurisdiccional por separado. A esto hay que añadir que algunas comunidades autónomas como Cataluña o el derecho foral de País Vasco difieren en gran medida del Derecho común aplicable al resto de España. Por estos motivos y por la ingente cantidad de sentencias disponibles en las Bases de Datos Jurídicas vamos a limitarnos a estudiar los litigios que tienen lugar en la jurisdicción penal, pues son los casos más habituales a los que se tiene que enfrentar cualquier letrado y en ellos no existen diferencias en cuanto a las comunidades autónomas a diferencia del orden civil.

Otra de las cuestiones a tener en cuenta es el tipo de tribunal con el que nos podemos encontrar. En derecho existe la famosa pirámide del Poder Judicial (Figura 1) para conocer los diferentes escalones en los que se puede enjuiciar un determinado asunto.

Figura 1: Pirámide del Poder Judicial

Dicha pirámide sigue un orden ascendente:

1. En la base tenemos los Juzgados de Paz, cuyas funciones se recogen en el artículo 100 de la Ley Orgánica del Poder Judicial (en adelante LOPJ) y que, en lo que se refiere al orden penal que es lo que nos interesa, expone la ley que "*conocerán en primera instancia de los procesos por faltas que les atribuya la ley. Podrán intervenir, igualmente, en actuaciones penales de prevención, o por delegación, y en aquellas otras que señalen las leyes*"[2].

[2] El término "faltas" ha desaparecido con la reforma del Código Penal de 2015, recogiendo desde entonces el código sólo los delitos y delitos leves (siendo muchos de estos últimos las antiguas faltas). Según el

2. En el caso del siguiente escalón, encontramos diferentes Juzgados:
   - El artículo 87 de la LOPJ expone las competencias de los Juzgados de Instrucción, relativo al orden penal, que, a grandes rasgos, supone conocer de los delitos que se han cometido o de adoptar medidas urgentes.
   - Por lo que se refiere a los Juzgados de Violencia sobre la Mujer, las competencias que se le atribuyen en el orden penal están expuestas en el art.87.1 ter de la misma Ley y, en términos generales, son las relativas a temas relacionados con violencia de género.
   - Por último, otro de los posibles Juzgados con el que nos podemos encontrar en nuestro análisis es el Juzgado de lo Penal, cuyos poderes rezan en el art.89.2 bis de la LOPJ y se encarga principalmente de enjuiciar los delitos que ya han sido instruidos por los Juzgados de Instrucción, ya que el juez o jueces que instruyen (investigan un determinado caso) no pueden enjuiciar dicho caso por entender la Ley que tienen ya una opinión formada y pueden estar "contaminados" por la investigación, de ahí que se necesiten otros jueces que no hayan estado implicados en la investigación y que únicamente lean las conclusiones de la misma para que enjuicien de manera objetiva y cumplan así con el principio de imparcialidad.

El resto de la pirámide lo componen órganos colegiados, es decir, órganos formados por más de un juez. Existe un ponente, que es el representante de todos los jueces o magistrados presentes en ese procedimiento, y es quien figura en las bases de datos jurídicas. Esto plantea un problema a la hora de realizar el *webscrapping*, ya que no se cuenta con un identificador que recoja el nombre de todos los jueces presentes en el caso y puede plantear problemas si queremos utilizar ese campo. A esto hay que añadir que las sentencias no se conforman por unanimidad, si no por mayoría, por lo que pueden existir jueces con opiniones contrapuestas ante una controversia. Si forman parte de la minoría cuyo veredicto no ha triunfado, pueden redactar un "voto particular", que es la opinión divergente hecha contar por escrito, pero no es obligatoria y no siempre se da. Puede ser porque el magistrado difiera completamente de la decisión tomada (recibiendo el nombre en este caso de voto discrepante) o porque pese a estar de acuerdo no lo está con la argumentación jurídica formulada (siendo este conocido como voto concurrente).

Con todos los problemas que esto puede acarrear a la hora de analizar los datos, se decidió finalmente filtrar las resoluciones judiciales al ámbito penal y a la vez al segundo escalón de la pirámide del poder judicial relativo a los Juzgados de Primera Instancia e Instrucción, Juzgados de lo Penal y Juzgados de Violencia sobre la Mujer, por ser órganos unipersonales, para ver si además se pueden observar diferencias en función de criterios tales como el género del juez o la fecha en la que se dictó la resolución. Además, el tipo de resolución serán sentencias, que son las que ponen fin al procedimiento y las que se pronuncian en nuestro caso sobre el supuesto delito. En el caso de los autos y providencias, que son las otras resoluciones judiciales disponibles,

artículo 14 de la Ley de Enjuiciamiento Criminal (LECrim) el Juzgado de Paz ya no tiene competencia para conocer los delitos leves pues atribuye la competencia de los mismos a los Juzgados de Instrucción excepto aquellos que deba conocer el Juzgado de Violencia sobre la Mujer.

se trata de cuestiones secundarias no tan relevantes para el análisis que se plantea realizar.

Una vez que hemos definido el marco de actuación, crearemos la base de datos que va a contener las sentencias que posteriormente analizaremos (los detalles de la creación se verán con profundidad en la sección 5). Tras esto realizaremos técnicas de *webscrapping* para obtener las variables para cada una de las sentencias seleccionadas y depurar la base de datos una vez esté completa.

Con la base de datos depurada, realizaremos un proceso de *stemming*[3] de las palabras para posteriormente poder aplicar una función conocida como "*Term Frequency Inverse Document Frequency (TF*IDF)"* la cual asigna puntuaciones a cada palabra atendiendo a unos determinados criterios que se explicarán más adelante y descarta aquellas palabras que no aportan información, bien porque aparecen demasiado y son irrelevantes o bien porque aparecen en contadas ocasiones.

Una vez hecho esto, y dado el gran volumen de variables resultante, es necesario aplicar una técnica de reducción de dimensionalidad denominada ACP (análisis de componentes principales) para reducir el número de campos que suponen las palabras útiles resultantes de aplicar la función *TF*IDF.* Finalmente, se aplican las técnicas de machine learning necesarias para analizar y predecir los resultados, comparando los diferentes algoritmos para diferentes conjuntos de variables y analizando el modelo ganador.

2. Estado del arte

Cada día aumentan el número de litigios en España y como consecuencia el plazo para resolverlos, siendo el tiempo medio para resolver una demanda de 296 días. Además, la mayoría de los datos legales están desestructurados y escritos en lenguaje natural y las bases de datos jurídicas gratuitas son poco intuitivas y enlazan con archivos en *pdf*, siendo muchos de estos documentos escaneados.

Fuera de nuestras fronteras no han sido pocos quienes han intentado desarrollar alguna técnica de minería de datos que ayude a predecir los resultados de una sentencia. Ya se hizo hace muchos años en EEUU (Ulmer, 1963). El análisis se realizó sin tener en cuenta lo que el Tribunal había dicho mediante el razonamiento jurídico en estos casos; se basaban únicamente en los elementos de hecho que habían sido enfatizados por los jueces en sus opiniones y en sus votos para afirmar o dejar de lado las condenas. Clasificaban aquellos casos en los que estuviesen implicados afroamericanos, comunistas y extranjeros indocumentados y, en función de los fallos, asociaban unas probabilidades a los jueces que extrapolaban con una función a otros casos, afirmando que los jueces más ancianos no cambiaban su actitud personal a la hora de enjuiciar y había que predecir en función de ese factor y no de los fundamentos jurídicos en que se basasen.

[3] El *stemming* consiste en un proceso de normalización lingüística donde las palabras se reducen a una forma común, denominada *stem*. El *stem* es la porción de la palabra que resulta de eliminar sus afijos, y que en los algoritmos que utilizan las diferentes librerías se puede tratar de su raíz o la base de la palabra que el propio algoritmo considere. Por ejemplo, el *stem* "perr" se refiere a las palabras "perro", "perros", "perrito", "perrote", etc.

Otros estudios más recientes (Martin et al., 2004) utilizan árboles de decisión para estudiar el fallo de la Corte Suprema de Estados Unidos tomando como factor relevante el Presidente del Gobierno que les ha designado, enfocado más en un sentido político que hacía predecir el resultado de los fallos. Este estudio tiene la desventaja de que asume que todos los jueces pertenecen a un grupo específico igual (por haber sido designados por el Presidente de Estados Unidos). Esto no podría extrapolarse al caso de España, ya que los jueces del Tribunal Supremo son designados por el Consejo General del Poder Judicial y no por el Presidente del Gobierno.

Siguiendo este objetivo, un estudio más actual (Katz et al., 2017) utiliza la base de datos de la Corte Suprema de Estados Unidos para crear la suya propia en base a las variables que estima oportunas y con una variable objetivo de resultado "a favor" o "en contra". Posteriormente aplican *random forest*, máquinas de soporte vectorial y redes neuronales (en concreto, perceptrón multicapa), concluyendo que el modelo que mejor predice es el primero.

También es reciente un estudio realizado por un equipo de investigadores de inteligencia artificial del University College de Londres, y las universidades de Sheffield y Pennsylvania (Aletras et al., 2016). En él, los científicos utilizaron los datos de 584 casos relacionados con los artículos 3, 6 y 8 de la Convención Europea de Derechos Humanos para aplicar su algoritmo de inteligencia artificial en busca de patrones en el texto que predijera el resultado de las sentencias del Tribunal Europeo de Derechos Humanos.

Descubrieron que los factores más fiables para predecir la decisión del tribunal son el lenguaje utilizado y los temas y las circunstancias mencionadas en el caso. Con esta información el sistema fue capaz de conseguir una precisión de acierto del 79%. Aplicaron el modelo *Bag of Words* (BoW)[4] para encontrar las palabras o pares de palabras que tenían mayor relevancia para posteriormente utilizar una Máquina de Soportes Vectoriales (SVM). Los resultados indicaron que la sección de "hechos" de un caso predice mejor la decisión real del tribunal, pues es más consistente que fundamentación legal que hay detrás de las decisiones judiciales.

De este modo, dichos estudios pueden resultar útiles para guiar esta investigación, sin obviar las diferencias existentes por el tipo de Derecho aplicable en cada lugar.

3. Objetivo

Pese a la dificultad que suponen las bases de datos jurídicas españolas que son poco informativas, el objetivo de este trabajo es intentar predecir el sentido del fallo de una sentencia en función de una serie de variables como el tribunal en el que se realizó, el juez, la fecha en que tuvo lugar… De esta forma, buscamos facilitar el trabajo de los diferentes agentes legales y permitir agilizar el tiempo de preparación y las estrategias para preparar un caso.

[4] Este modelo es una forma de representar los datos de texto para trabajar con machine learning. Con él se extraen características del texto, ya que se cuenta con un "diccionario" de palabras conocidas (denominados *tokens*), que son valoradas según su aparición. De esta forma se crea una bolsa de palabras, denominada así ya que no importa el orden o situación en las que las palabras se encuentren, si no simplemente que coincida la aparición, ya que se afirma que si un documento tiene unas palabras similares será muy parecido a otro documento, sea cual sea el orden.

Los jueces de instancia son conocidos por los abogados, sobre todo en pequeñas ciudades, el día a día los lleva a tratar con ellos y conocen no sólo la posible tendencia que puede tener el caso, sino también otros aspectos sobre cómo actuar en sala. Aplicando hoy herramientas de minería de datos, podríamos ayudar a los letrados con esas inquietudes y encontrar patrones por tipología de asuntos y asociarlos a jueces y, de estar disponible la información, también a abogados[5], lo que agilizaría el sistema judicial al haber capacidad de negociar con la parte contraria en algunos procedimientos civiles sin necesidad de la intervención de un juez.

Además, se desean comprobar otros subobjetivos:

- Si es relevante el género del juez a la hora de enjuiciar un tipo de delito.
- Si las reformas que se han llevado a cabo en el Código Penal con el cambio de delitos y penas influyen también a la hora de absolver o condenar según el tipo de delito.
- Si, dada la muestra desbalanceada de nuestra base de datos, como se verá más adelante (80% de las sentencias con fallo condenatorio y 20% con fallo absolutorio), somos capaces de mejorar el error frente a no realizar ningún modelo y suponer que el fallo va a ser condenatorio.

4. Metodología empleada

4.1. Estructura de una sentencia

Para comprender bien los datos que queremos extraer, puede resultar útil conocer cuál es la estructura de una sentencia. A modo de ejemplo expondremos una para que sea más fácil comprender posteriormente el por qué de algunas variables de la base de datos.

En primer lugar tenemos el “Encabezamiento”, se trata de la parte de la sentencia que sirve para identificarla. Como se puede observar en la Figura 2, en ella se expone el lugar en el que se está conociendo el asunto, el juez (o ponente, si se trata de un órgano colegiado), el número de la sentencia (que sirve para identificarla y poder encontrarla posteriormente en los buscadores), así como una breve descripción de los delitos que se imputan y las partes que intervienen en el juicio.

En segundo lugar encontramos los “Antecedentes de hecho” (Figura 3). En este apartado se exponen con claridad las peticiones de las partes, los hechos en los que se basan para justificar dichas pretensiones, las pruebas que hayan propuesto y los hechos que se consideran probados (en el caso de las sentencias penales, generalmente los hechos probados se exponen en otro apartado que lleva precisamente dicho nombre, “hechos, probados”).

[5] Algo que ha comenzado a realizar este año la startup “Emérita Legal”. Sitio web: https://www.emerita.legal/

De esta manera y como podemos observar en la Figura 3, encontramos enumerados los hechos, desde que se inicia el procedimiento y los pasos que se llevan a cabo por las partes.

ENCABEZAMIENTO:

JUZGADO DE LO PENAL Nº 9

MÁLAGA

C/ Fiscal Luis Portero García, s/n ("Ciudad de la Justicia").

Fax: 951939169. Tel. 951939069 (Información) Tel.: 677982525/6/7(Ejecutorias) 677982528 (Juicios).

SENTENCIA 402/2018

En Málaga, a 26 de diciembre de 2.018.

Don *IGNACIO NAVAS HIDALGO* , Magistrado-Juez del Juzgado de lo Penal número 9 de esta ciudad y su partido judicial; habiendo visto en juicio oral y público las precedentes actuaciones de **Juicio Oral 130/2.017** , por presunto delito de **HOMICIDIO POR IMPRUDENCIA GRAVE** , contra *Torcuato* , con D.N.I. nº NUM000 , cuyos demás datos personales obran en los autos, sin antecedentes penales y en situación de libertad provisional por esta causa, representado por el Procurador Sr. Abalos Guirado y defendido por el Letrado Sr. Herrera Rueda, siendo parte acusadora, en el ejercicio de la acción pública, el *MINISTERIO FISCAL* , y actuando como acusación particular, Bibiana , Camila y Emma , a su vez representadas y asistidas, la primera por el Procurador Sr. Ledesma Hidalgo y el Letrado Iranzo Ruiz, y las restantes por la Procuradora Sra. Mateo Crossa y el Letrado Sr. Agüera Lorente, y atendiendo a los siguientes,

Figura 2: Ejemplo de "Encabezamiento" de una sentencia.

ANTECEDENTES DE HECHO:

PRIMERO.- Las presentes actuaciones se iniciaron en el Juzgado de Instrucción nº 3 de DIRECCION000 en el curso de sus Diligencias Previas 489/15 (posterior Procedimiento Abreviado 62/15), en las que tras formular el Ministerio Fiscal y las acusaciones particulares escritos de calificación provisional, y decretada la apertura de juicio oral, se dio traslado a la defensa para formular escrito, turnándose a este Juzgado para su enjuiciamiento y quedando registradas bajo el número que obra en el encabezamiento de esta resolución.

SEGUNDO.- Recibidas las actuaciones y señalado día para juicio, el acto tuvo lugar en forma oral y pública, con la asistencia del Ministerio Fiscal, las acusaciones particulares, el acusado y sus respectivas direcciones letradas, habiéndose practicado las pruebas propuestas y admitidas con el resultado que figura en el soporte audiovisual que grabó su desarrollo.

TERCERO.- El Ministerio Fiscal calificó definitivamente los hechos como constitutivos de un delito de homicidio por imprudencia grave del artículo 142.1º del Código Penal , reputando responsable de tal infracción penal en concepto de autor al referido acusado, sin la concurrencia de circunstancias modificativas de la responsabilidad penal, y solicitando le fuera impuesta la pena de 30 meses de prisión, con inhabilitación especial para el derecho de sufragio pasivo durante el tiempo de la condena, junto al pago de las costas procesales. Asimismo interesó que el anterior acusado indemnizara a Camila y a Emma (hijas y legales herederas de Juan Ignacio) en la cuantía de 180.000 euros, a incrementar con los intereses legales previstos en el artículo 576 de la Ley de Enjuiciamiento Civil .

Figura 3: Ejemplo de "Antecedentes de hecho"

FUNDAMENTOS DE DERECHO:

PRIMERO.- Los hechos que se declaran probados, después de la apreciación en conciencia por este Juzgador de las pruebas practicadas en el acto de la vista conforme a los principios de oralidad, publicidad, inmediación, contradicción y demás garantías procesales y constitucionales de nuestro ordenamiento jurídico se consideran como constitutivos de un **delito de homicidio** causado por imprudencia grave, tipificado en el artículo 142. 1º del Código Penal .

De conformidad con dicho resultado probatorio, ha de concluirse que, en la escala jerárquica de las modalidades de imprudencia que se contienen en nuestro Código Penal, es la grave la que aquí concurre, en cuanto supone la

Figura 4: Ejemplo de "Hechos probados"

Por su parte en la Figura 4 encontramos un ejemplo de los hechos probados, que en este caso lleva la numeración de “único” por no haber sido probado más que un hecho.

Tras exponer los hechos que han sido probados y las actuaciones que se han llevado a cabo, el juez entra a conocer de todo lo anterior aplicando la ley vigente en ese momento para tomar una decisión. Este apartado se puede denominar “Fundamentos de Derecho” o “Fundamentos Jurídicos”.

En este apartado, si observamos la Figura 5, el juez expresa por puntos aquellas cuestiones que las partes han solicitado para que arroje luz sobre la controversia que hay, motivando el juez el por qué de su decisión, ya que es un derecho para las partes del juicio y un deber para el juez, pudiendo recurrir las partes de no justificar el juez su decisión.

ÚNICO.- Se considera probado y así expresamente se declara que alrededor de las 07:45 horas del día 8 de febrero de 2.015, el acusado, Torcuato , mayor de edad y sin antecedentes penales, se encontraba en compañía de Herminia y Matilde cuando caminaban por la CALLE000 de DIRECCION000 (Málaga).

Que en ese momento el acusado y sus acompañantes acudieron en auxilio de Julia , quien estaba siendo golpeada por Juan Ignacio e Leonor para sustraerle el bolso que portaba -hechos que son objeto de otro procedimiento-. Herminia y Matilde se quedaron en compañía de Julia mientras que el acusado salió corriendo tras Juan Ignacio para recuperar el bolso sustraído.

Que tras dar el acusado alcance a Juan Ignacio y, ante la negativa de éste de entregar el bolso, se produjo un forcejeo entre ambos en el curso del cual el acusado, siendo consciente de las graves consecuencias que podría causar con su acción aunque sin pretender ni consentir la efectiva producción de las mismas, propinó a aquél dos puñetazos, uno de ellos en la cabeza y otro, bien en el mismo lugar, bien en el cuello o en el hombro, que determinó que cayera al suelo, sufriendo un traumatismo craneoencefálico con hemorragia cerebral que le provocó la muerte el día 10 de febrero de 2015.

Juan Ignacio se encontraba divorciado y era padre de dos hijas, Camila , entonces mayor de edad, y Emma , siendo en ese momento menor de edad.

Figura 5: Ejemplo Fundamentos de Derecho

Finalmente, la sentencia concluye con el “Fallo”, que es la conclusión de la fundamentación jurídica anterior y determina si absuelve o condena al acusado (Figura 6), entre otras cuestiones que no son relevantes para este estudio.

FALLO:

Debo **CONDENAR Y CONDENO** a Torcuato , como autor criminalmente responsable del delito de ***HOMICIDIO POR IMPRUDENCIA GRAVE*** , ya definido, sin la concurrencia de circunstancias modificativas de la responsabilidad criminal, a la pena de **DOS AÑOS DE PRISIÓN** con la accesoria de inhabilitación especial para el derecho de sufragio pasivo durante el tiempo de la condena, así como al pago de las costas procesales, sin incluir las de las acusaciones particulares.

Figura 6: Ejemplo de “Fallo” de una sentencia

Adicionalmente, muchas bases de datos privadas disponen de una cabecera donde incluyen además ciertos elementos útiles. Dicha cabecera no forma parte de la sentencia, sino que se emplea para ayudar en el filtrado de información a quienes emplean dichas bases de datos para que sea más sencilla la lectura y búsqueda de la jurisprudencia que necesiten.

Tal y como muestra la Figura 7, la cabecera se trata de un pequeño resumen en el que se exponen, para el caso del orden penal, los delitos que se tratan en la sentencia, así como de un resumen del análisis jurídico llevado a cabo y es fundamental para que el juez tome la decisión final de absolver o condenar al acusado.

Además en la cabecera encontramos otros datos útiles como el “origen”, que indica el juzgado en el que se ha llevado a cabo el juicio; el “ponente”, que en este caso se trata del juez al estar ante un órgano unipersonal; la “jurisdicción”, que es penal para nuestra sentencia y para nuestro estudio, nos sirve para filtrar por los distintos órdenes penales existentes; el “tipo resolución”, que diferencia entre sentencia, auto o providencia y que de nuevo nos sirve para filtrar las sentencias en este estudio”; y otros datos menos relevantes para nuestro trabajo, como son el “número sentencia” o “número recurso”.

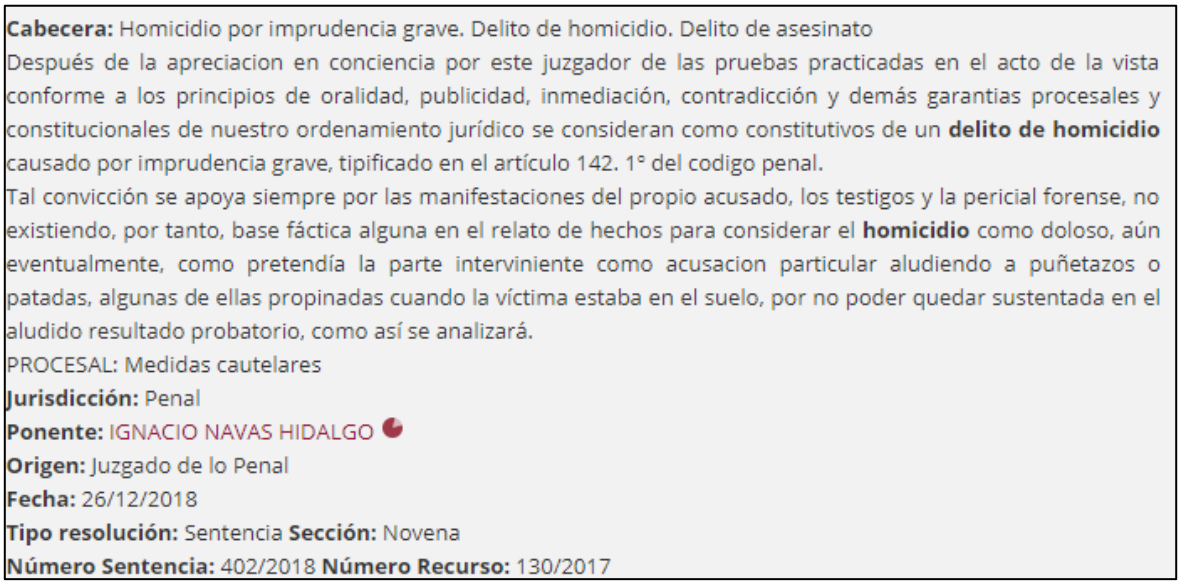

Cabecera: Homicidio por imprudencia grave. Delito de homicidio. Delito de asesinato
Después de la apreciacion en conciencia por este juzgador de las pruebas practicadas en el acto de la vista conforme a los principios de oralidad, publicidad, inmediación, contradicción y demás garantias procesales y constitucionales de nuestro ordenamiento jurídico se consideran como constitutivos de un **delito de homicidio** causado por imprudencia grave, tipificado en el artículo 142. 1° del codigo penal.
Tal convicción se apoya siempre por las manifestaciones del propio acusado, los testigos y la pericial forense, no existiendo, por tanto, base fáctica alguna en el relato de hechos para considerar el **homicidio** como doloso, aún eventualmente, como pretendía la parte interviniente como acusacion particular aludiendo a puñetazos o patadas, algunas de ellas propinadas cuando la víctima estaba en el suelo, por no poder quedar sustentada en el aludido resultado probatorio, como así se analizará.
PROCESAL: Medidas cautelares
Jurisdicción: Penal
Ponente: IGNACIO NAVAS HIDALGO
Origen: Juzgado de lo Penal
Fecha: 26/12/2018
Tipo resolución: Sentencia **Sección:** Novena
Número Sentencia: 402/2018 **Número Recurso:** 130/2017

Figura 7: Ejemplo de “cabecera” de una sentencia

Esta cabecera será la clave para crear nuestra propia base de datos, ya que la mayoría de los elementos que necesitamos se encuentran en la Figura 7: el nombre del juez, la fecha en la que se dictó la sentencia, el orden de que se trata, el juzgado, así como un resumen de la sentencia.

4.2 Procesamiento del Lenguaje Natural (NLP)

El procesamiento del lenguaje natural supone desarrollar aplicaciones y servicios comprensibles a través del lenguaje humano. Algunos ejemplos de NLP son el reconocimiento de voz, la traducción del habla, comprensión de oraciones completas, etc. Cada vez se hace más atractivo reunir y procesar este tipo de datos por la ingente cantidad disponible de ellos en la red en sitios como redes sociales, blogs, páginas web… lo que puede servir a una empresa para saber cuál es la opinión que tienen sus usuarios sobre la misma, sobre un determinado producto, sobre sus necesidades para lanzar nuevos servicios…

Para nuestro estudio, previo a utilizar modelos de *Machine Learning*, debemos procesar el texto en bruto para que se adapte mejor a dichas técnicas.

Antes de entrar a limpiar el texto hay que tener en cuenta que, en las sentencias, por desgracia, existen faltas de ortografía y además los datos han sido anonimizados, esto es, que los nombres de personas o empresas aparecen con siglas o con tres letras

similares (por ejemplo “YYY”, “AAA”), por lo que nos podemos encontrar con palabras que aparezcan sin mucho sentido. Además, existen signos de puntuación tales como comas, comillas, signos de interrogación y exclamación. También encontramos descripciones con guiones, números y puede haber marcadores de sección (por ejemplo, I, II y III).

Para que las palabras puedan utilizarse y compararse, además de realizar esta depuración inicial, debemos quedarnos con la raíz o base de la palabra, ya que, al comparar en un texto, tiene el mismo valor de cara al análisis que se emplee la palabra “gatito”, “gato”, “gata”. Esto se denomina *stemming* y será realizado empleando la librería *NLTK* de *Python*. Una vez que tenemos las palabras modificadas para poder ser analizadas, aplicaremos la función *Term Frequency – Inverse Document Frequency*.

4.3 *Term Frequency – Inverse Document Frequency* (TF*IDF).

Con la fórmula TF*IDF se puede estudiar la importancia que tienen las palabras de cada una de nuestras sentencias. Por ejemplo, imaginemos que tenemos una base de datos sobre gatos y sus características y una persona quiere buscar sobre gatos persas, por lo que en su consulta indica “el gato persa”. Debemos decidir qué documentos tenemos que devolver a este usuario de nuestra base de datos. Si hay documentos que coinciden de manera exacta con la búsqueda planteada no habría problema, pero tenemos que decidir qué hacer si no se da este caso y tenemos coincidencias parciales. Por ejemplo, pensemos que tenemos: 1. “El gato negro”; y 2. “Un gatito persa”.

La primera descripción contiene dos de las tres palabras consultadas, mientras que la segunda solo posee una de tres. Todo parece indicar que se decantaría por la primera descripción cuando a priori parece tener más similitudes la segunda. El TF*IDF ayuda a escoger la mejor opción, en este caso la segunda sobre la primera.

Por un lado, tenemos el término TF, el cual es el mismo para cada palabra. Las siglas del término TF, que significan *Term Frequency*, hacen alusión a la frecuencia de término e indica la frecuencia relativa de una palabra en un documento.

TF (t) = Número de veces que aparece la palabra “t” / Número total de palabras en el documento.

Mientras que parece lógico que los términos “gato” y “gatito” aparezcan en muchos documentos de nuestra base de datos felina (TF alto), el término “persa” aparecerá en menos documentos (TF bajo). Aquí es donde entra en juego el otro término, “IDF” (*Inverse Document Frequency*). Es útil ya que hay términos que aparecen muchas veces pero no son relevantes, como “para”, “de”, “el”. Por lo que con IDF se sopesan los términos frecuentes y se da mayor importancia a los raros, que son aquellos que pueden marcar la diferencia y, en nuestro caso, devolver el documento más apropiado para el usuario que buscaba información sobre gatos persas. De esta manera, la puntuación para “el” será muy baja, ya que al tratarse de un artículo aparecerá muchas veces,

mientras que “persa”, que es una característica no general de los felinos será muy útil para filtrar y encontrar lo que se desea, por lo que poseerá una puntuación muy alta.

IDF (t) = log (Número de documentos / Número de documentos con el término “t”)

Si tomamos esta referencia para una única palabra, por ejemplo “gato” en un documento que contiene 100 palabras y en el que la palabra “gato” aparece 3 veces. El TF para “gato” es (3/100) = 0.03. Si además suponemos que tenemos 10 millones de documentos y la palabra gato aparece en mil de ellos, el IDF será log(10.000.000 / 1.000) = 4. Por tanto, el TF*IDF será de 0.03*4 = 0.12. Esa será la relevancia que tendrá la palabra “gato”.

De un modo más preciso, las siglas del término TF, que significan *Term Frequency*, hacen alusión a la frecuencia de término e indica la frecuencia relativa de una palabra en un documento. Cuanto más extenso sea un documento, más probable será que el término aparezca más veces. En el caso de las sentencias, además, existen una serie de palabras que se pueden repetir con mucha frecuencia, como “artículo”, “ley”, “sentencia”, “jurisprudencia” y, en el caso del derecho penal, también otras como “delito”, “pena”, “reo” o “acusado”. El *term frequency* es un valor que depende del número de palabras que contiene un documento. Con este análisis comprobamos si la palabra clave aparece de forma natural o forzada en el documento analizado.

Por otro lado, el término IDF, del inglés *Inverse Document Frequency*, se refiere a la Frecuencia Inversa de Documento, es decir, este término “corrige” al término anterior, al dar la misma importancia a cualquier palabra y limitarse a calcular su valor atendiendo a su tasa de aparición. Lo que hace el IDF es atenuar el peso de las palabras comunes (*stopwords*) o de aquellas que por su frecuencia pueden considerar que no aportan información útil, para distinguir las que son relevantes de las que no. El IDF mide la importancia que tiene un término específico en un documento con respecto a la relevancia en el total de documentos, en nuestro caso, en el total de las sentencias.

De esta manera, el algoritmo TF*IDF se usa para sopesar una palabra clave y asignar la importancia de esa palabra clave en función del número de veces que aparece en el documento, comprobando qué tan relevante es dicha palabra entre todos los documentos con lo que deba compararse (*corpus*).

Para un término “t” en un documento “d”, el peso “Wt, d” del término “t” en el documento “d” viene dato por:

$$W(t,d) = TF(t,d)\log(N / DFt)$$

Donde:

- TF(t,d) es el número relativo de ocurrencias de t en el documento d.
- DF(t) es el número de documentos que contienen el término t.
- N es el número total de documentos en el *corpus*.

Con TF*IDF conseguimos tener una puntuación numérica para observar la importancia de las palabras atendiendo a su masiva o escasa aparición en las sentencias. En el caso de la librería de *Python* que se ha empleado, *sklearn*, los valores resultantes proporcionan puntuaciones entre los valores de 1 y 8, siendo aquellos valores cercanos a 1 los que se repiten demasiado (en el caso de una sentencia pueden ser las palabras "delito" o "pena", por ejemplo), y aquellos con puntuaciones cercanas a 8 aquellos más extraños que generan sentencias poco comunes o casos que no suelen ser muy habituales.

4.4 Análisis de Componentes Principales.

El análisis de Componentes Principales es un método estadístico para reducir la dimensión de un conjunto de datos. Tiene como punto de partida una matriz de datos con una serie de individuos que se caracterizan por varias variables (por eso esta técnica suele clasificarse como análisis multivariante). Es un método estadístico que permite reducir la complejidad de espacios muestrales con muchas dimensiones conservando una parte de la información que aportan. Las nuevas variables, que se denominan Componentes Principales, se obtienen en orden decreciente según su importancia, es decir, la primera componente principal recogerá la máxima información de los datos originales y a partir de esta el resto recogerá dicha información o variación en orden descendente. Esta importancia se medirá con la varianza, es decir, cuanto mayor sea la varianza de la componente mayor será la cantidad de información que lleva aparejada una componente.

Lo que se busca con este tipo de análisis es ver si las primeras Componentes Principales recogen la mayor parte de la variación de los datos originales. En caso afirmativo, dichas componentes se podrán utilizar para resumir los datos con la mínima pérdida de información. En nuestro caso, tenemos como criterio y dado el gran número de variables con las que contaremos (como se verá a continuación) que las componentes expliquen, al menos, el 50% de la información, algo que se verá con más detalle en el apartado de metodología dedicado a este análisis.

Para ello se utiliza la matriz de varianzas-covarianzas o la matriz de correlaciones. La primera tiene una principal desventaja, y es que cuando las variables tienen unidades de escala diferentes se produce un exceso de influencia por parte de las variables con mayor varianza, por ello es preferible emplear la matriz de correlaciones que está acotada entre los valores -1 y 1 estando todas las variables en la misma escala.

El resultado del análisis de componentes principales es una combinación lineal formada por p-variables. Esas p-variables están observadas en n-individuos. Cada columna es una variable y cada fila es la observación de las variables en un individuo. Es decir, se consideran una serie de variables originales (x_1, x_2, ..., x_p) sobre un grupo de individuos y se trata de calcular, a partir de ellas, un nuevo conjunto de variables (y_1, y_2, ..., y_p) incorreladas entre sí, cuyas varianzas decrecen de manera progresiva.

Tanto la matriz de varianzas – covarianzas como la de correlaciones son cuadradas y simétricas, por lo que se podrán diagonalizar, es decir, poner una matriz formada por ceros salvo en su diagonal, que será lo que se conoce como autovalores. Como ya hemos dicho, es preferible la matriz de correlaciones, para la cual se da igual importancia a todas las variables originales.

Además, con las componentes principales podemos realizar representaciones gráficas para encontrar relaciones entre variables e individuos en dos dimensiones. Sin embargo, no siempre dicha representación será útil, bien porque la primera componente acumule mucha información o porque todas acumulen más o menos lo mismo y no sea posible vislumbrar relaciones gráficamente.

4.5 Técnicas de Machine Learning.

Tras depurar nuestros datos y reducirlos a una cantidad de variables más manejables a través del análisis de componentes principales, podremos aplicar los algoritmos que consideramos más útiles para nuestro análisis.

Al tratarse de un conjunto de datos cuya variable objetivo es binaria podríamos plantearnos el uso de la regresión logística, sin embargo, dada la gran cantidad de variables con las que contamos y la complejidad de los datos, no parece útil optar por dicha opción, por lo que nos hemos decantado por otros modelos más complejos.

4.5.1 Redes neuronales

Las redes neuronales artificiales son unos algoritmos que se inspiran en las redes neuronales biológicas y tienen una serie de características que las hacen muy interesantes, como su capacidad de auto-organizarse y adaptarse, siendo más útil para encontrar relaciones no lineales a diferencia de la regresión lineal, gracias a que cuenta con nodos que trabajan en paralelo y permitiendo a la red aproximar funciones, clasificar patrones y auto-mejorarse frente al ruido, es decir, las redes neuronales tratan de imitar el funcionamiento del cerebro buscando elaborar reglas de los inputs que reciben.

Una red neuronal está formada por neuronas que a su vez están interconectadas. Cada conexión de la red neuronal se asocia a un peso que indica la importancia que tendrá esa relación en la neurona al multiplicarse por el valor de entrada. Cada neurona tiene una función de activación que define la salida de la neurona. Esta función de activación se utiliza para indicar la no linealidad en las capacidades de modelado de la red.

Para que dichas redes funcionen correctamente, es recomendable disponer de grandes cantidades de datos para que el proceso de entrenamiento sea robusto. Con entrenar la red nos referimos a proporcionarle unos *inputs* o datos de entrada y que ofrezca un *output* en función de los pesos que se hayan establecido.

De este modo se obtendrán diferentes configuraciones de las redes para encontrar la que mejor se ajuste a cada uno de los conjuntos de datos atendiendo a:

- El número de nodos.
- La función de activación.
- El algoritmo de optimización.
- La condición de parada (*early-stopping*), si es necesario, para evitar el sobreajuste.

4.5.2 Métodos basados en árboles

Los árboles de decisión son algoritmos iterativos que dividen los datos en regiones basadas en intervalos de las variables independientes. Nos sirven como modelo de segmentación de los datos, pero también como algoritmo predictivo.

Tienen la ventaja de que son flexibles, si le indicamos que aumente el número de hojas lo realizará y profundizará más en los datos; pero tiene la desventaja de que para cada región toma un valor constante, proporcionando la misma predicción.

Dentro de estos métodos podemos encontrar:

4.5.2.1 Bagging

Cabe mencionar el *bagging*, como caso particular de *random forest,* que se verá a continuación. En dicho algoritmo se seleccionan todas las variables disponibles en nuestro conjunto de datos, se construye un árbol que predice los datos test, repitiendo el proceso varias veces con muestras con reemplazamiento, lo cual quiere decir que de las n observaciones se toma una muestra aleatoriamente, por lo que pueden repetirse observaciones y otras pueden no llegar a incluirse en cada iteración. Se hace un promedio con las predicciones de todos los árboles obtenidos.

Sin embargo, el problema del *bagging* es la dependencia entre variables. Hay variables que son importantes y mandan sobre las demás, por lo que al final los árboles no son tan diferentes y, si son similares, se tiende al sobreajuste. En nuestro caso, al ser un conjunto de datos complejo donde hay muchas variables que pueden aportar información parece más útil utilizar la extensión del *bagging*, que es *random forest*, el cual procura que los árboles difieran más entre sí. Pese a ello se realizará también el modelo de *bagging* para comparar resultados.

Los parámetros que se pueden modificar son similares a *Random Forest*.

4.5.2.2 Random Forest

El *Random Forest* funciona de manera similar al *bagging* pero en cada nodo del árbol no se utilizan todas las variables como candidatas a abrir ese nodo. En cada nodo se sortea una cantidad de las mismas y se elige la que mejor se considera. De esta manera se protege de aquellas variables de gran importancia que mandan sobre las demás, consiguiendo árboles más variados. Con este azar a veces encontramos las mejores predicciones para nuestros datos. Esta aleatoriedad no es tan peligrosa, pues se trata de un algoritmo planteado para evitar el sobreajuste.

En este caso podemos modificar una serie de parámetros que son el número de variables *(mtry*), el tamaño de la muestra (*sampsize*), el número mínimo de observaciones para un nodo final del árbol (*nodesize*), el número de árboles, y si queremos que el algoritmo se haga con o sin reemplazamiento (*replace TRUE* o *FALSE* respectivamente).

4.5.2.3 Incremento Gradiente

Por otro lado, utilizaremos también el Incremento Gradiente o *Gradient Boosting*, el cual actualiza los pesos de las observaciones en función del descenso del error que se comete en la predicción. Es decir, en cada predicción de cada observación ($\hat{Y}$) se comete un error ($Y - \hat{Y}$). En función del error se construyen a continuación otros árboles ligeramente modificados, para intentar que el error sea mínimo, logrando que el resultado se ajuste a las predicciones al irse corrigiendo el error entre los propios árboles.

Podemos modificar los siguientes parámetros del Incremento gradiente en R:

- *Schrinkage* = es el parámetro de regularización, se encarga de medir la velocidad del ajuste. En función de su valor la velocidad de convergencia será mayor o menor. Si el parámetro es pequeño necesitará muchas iteraciones para converger y si es más grande convergirá antes.
- *N.minobsinnode*= es el tamaño máximo de los nodos finales y principal parámetro, es el que define la complejidad del árbol.
- *N.trees*= número de iteraciones (árboles).
- *Bag.fraction* = 1, define el porcentaje de observaciones que vamos a sortear antes de cada árbol. Es frecuente dejar ese valor al principio y posteriormente cambiarlo para ver qué sucede y evitar el sobreajuste. Cuando se cambia este parámetro se denomina Incremento gradiente estocástico, aunque a menudo no es útil modificar el mismo y es conveniente conservar el valor igual a 1.

4.5.2.4 XGBoost

Xgboost es un programa que apareció en R después de *Gradient Boosting Machine* (gbm). Sirve para conocer el concepto de regularización y para intentar luchar contra el sobreajuste que muchas personas aquejaban necesario en Incremento gradiente.

La regularización es una táctica matemática que, dentro del propio algoritmo, evita que los parámetros tomen valores excesivamente grandes o bajos, es decir, controlar que la función que estamos creando para predecir no sea muy rara. Permite controlar las cotas superiores de los parámetros.

En *XGboost* se añadió una función de la complejidad del árbol en el que se introduce el número de hojas y el valor de la complejidad de las mismas hojas. Si un árbol toma predicciones muy grandes en una hoja, eso se penaliza por un parámetro (*gamma*). XGboost permitió incluir, de este modo, los parámetros de regularización lambda y gamma.

Se pueden modificar los siguientes parámetros:

- *Eta= schrinkage* (mide la velocidad del ajuste).
- *Gamma* = parámetro de regularización.
- *Colsample_bytree*= remuestrear columnas antes del árbol, antes de empezar el árbol en este caso.
- *Min_child_weight*= número mínimo de observaciones en nodos finales.
- *Max Depth*= profundidad máxima. Tanto *Min_child_weight* como *Max Depth* controlan profundidad del árbol.
- *Subsample* = observaciones utilizadas para hacer cada árbol.

4.5.3 Máquinas de Soporte Vectorial

En las Máquinas de Soporte Vectorial (*Support Vector Machines*, más conocido por sus siglas en inglés SVM) se trata de plantear el problema de separación lineal de clases con métodos algebraicos que buscan el hiperplano de separación.

Se llama SVM porque el vector de soporte de la máquina son los puntos, porque cada observación de los datos es un vector.

Como la separación perfecta no suele existir, se permite un *soft margin*, es decir, un porcentaje de observaciones mal clasificadas por los separadores para evitar el sobreajuste, permitiendo una tasa de fallos regulada por un parámetro "C". En función de dicho parámetro, si el valor de "C" es alto se permiten menos errores y, *a sensu contrario*, los errores podrán ser más altos en los casos en los que el parámetro "C" tenga un valor menor.

Sin embargo, a veces no es suficiente con el empleo del parámetro "C" porque la separación no es lineal, por lo que se emplea en este algoritmo las funciones *Kernel*. Estas funciones resuelven el problema de clasificación transformando el espacio en el que se encuentran los datos. Existen diferentes tipos de SVM atendiendo al *Kernel* que empleemos. lo que hace el tipo de *kernel* es definir un tipo de línea o frontera entre las clases. Se observa que cuando el *kernel* es lineal se obtienen como frontera un ajuste lineal. Esa recta es lo que se llama hiperplano y lo podemos observar en la Figura 8.

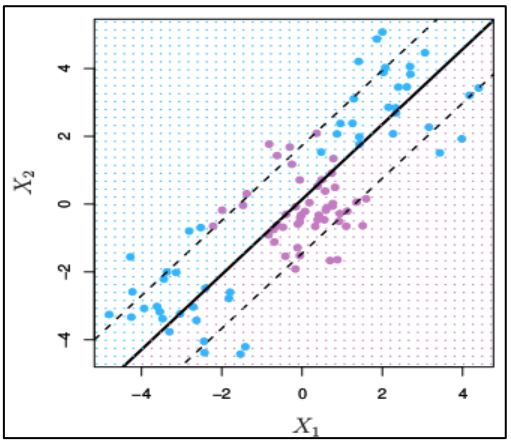


Figura 8: Ejemplo Kernel lineal[6]

En el caso del Kernel polinómico, además del parámetro C se pueden modificar otros parámetros como el grado del polinomio (*degree*). Esto es útil para buscar relaciones no lineales, aunque no se recomienda que el grado del polinomio sea superior a 5 porque puede sobreajustar. Así, podemos observar en la Figura 9 que para aquellas relaciones no lineales el SVM que emplea el *kernel* polinómico se ajustará mucho mejor a los datos, por lo que parece relevante estudiarlo para nuestro caso.

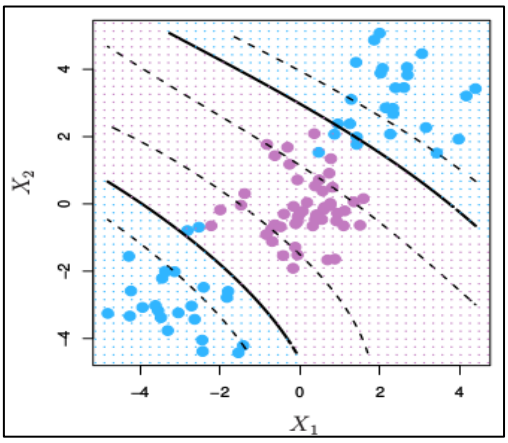


Figura 9: Ejemplo Kernel polinomial[6]

[6] Fuente: https://rpubs.com/Joaquin_AR/267926

Por último y dado que nuestra variable es binaria, utilizaremos el kernel RBF (*Radial Basis Function*) o kernel gausiano ya que aparentemente tiene como ventaja que es más flexible, tal y como observamos en la Figura 10, por lo que puede ir de un clasificador lineal a uno más complejo y tiene menos parámetros a regular que el *kernel* polinomial.

En este caso tendremos que regular dos parámetros, el ya famoso parámetro C y sigma. El valor de γ controla el comportamiento del *kernel*, cuando es muy pequeño, el modelo final es equivalente al obtenido con un *kernel* lineal, a medida que aumenta su valor, también lo hace la flexibilidad del modelo.

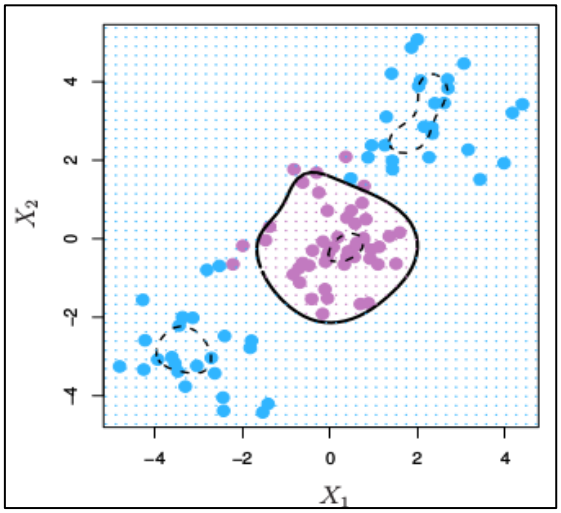


Figura 10: Ejemplo Kernel RBF[7]

4.5.4 Validación cruzada repetida

Todas las técnicas de *Machine Learning* que acabamos de mencionar serán evaluadas con validación cruzada repetida. Con la validación cruzada repetida se busca evitar el sobreajuste de los datos de entrenamiento. Consiste en dividir los datos aleatoriamente en k grupos, aislando uno de esos grupos, al que denominaremos i, el cual no se empleará para construir el modelo, que se hará con el resto de grupos (k-i). Este proceso se repetirá n veces y se estimará el error con el conjunto i. Para este trabajo se han propuesto 4 grupos y 5 repeticiones ya que dado el número de registros con los que contamos, así como el número de variables parece suficiente para evitar un sobreajuste de los datos.

5. Desarrollo del trabajo y principales resultados

5.1 Creación de la base de datos

En primer lugar se creó una base de datos SQL en MySQL pero posteriormente se decidió cambiar dicha base de datos por una SQLite. En Anexos se estudian las ventajas y desventajas de ambas opciones y por qué decidimos decantarnos finalmente por SQLite.

[7] Fuente: https://rpubs.com/Joaquin_AR/267926

Para configurar la base de datos SQLite se ha utilizado *Python*, ya que se realiza de forma rápida al no tener que configurar ningún servidor, usuario o permiso.

```
1. import sqlite3
2.
3. conn = sqlite3.connect('juridico_tfm.db')
4. c = conn.cursor()
5.
6. with open('database_vacia.sql') as fp:
7.     lines = fp.readlines()
8.
9.     for l in lines:
10.            c.execute(l)
11.
12.    conn.commit()
13.    conn.close()
```

Establecemos en primer lugar una conexión y un cursor, con lo cual al ejecutar el código se creará la base de datos si ésta no existe.

Posteriormente ordenamos que lea el fichero 'database_vacia.sql', que es una base de datos que hemos creado con los comandos necesarios para crear nuestra base de datos, la cual contará con una estructura formada por una tabla principal denominada "resoluciones judiciales", donde se volcará la información a nivel individual de cada sentencia en un pequeño resumen del caso y conclusión a la que se llega, además de la variable objetivo, el resultado, que en el ámbito de sentencias se denomina "fallo".

- **Cabecera:** se trata del resumen que proporciona la cabecera sobre el delito que se estudia y el desarrollo a lo largo de la sentencia, ya que puede resultar muy útil analizar esta parte para intentar predecir el fallo de una sentencia.
- **Fallo:** en esta variable se guardará todo el texto que se contiene en el apartado "fallo" de una sentencia, por si decidimos utilizarlo posteriormente, cosa que finalmente no sucede al contar con la variable "cabecera" que cuenta el desarrollo del juicio, algo que puede ser más relevante que el fallo en sí, que podemos generalizar acudiendo a la variable objetivo para saber si acaba en condena o absolución.
- **Sentido_fallo:** la que será nuestra variable objetivo e indicará si el fallo es "absolutorio" o "condenatorio". En un primer momento se descargaron además aquellas con el resultado "sentencia de conformidad" y "fallo doble". Las sentencias de conformidad son aquellas en las que el acusado llega a un acuerdo con el fiscal y se declara culpable a cambio de sufrir una rebaja en la pena, por lo que se acabaron incluyendo en la categoría de fallo "condenatorio". Por lo que respecta al fallo doble es el que planteaba más problemas ya que, en ocasiones, una persona puede ser acusada por más de un delito y se le puede condenar por uno de ellos y absolver de otros; también puede que más de una persona haya sido acusada por un delito (por ejemplo, el famoso caso de 'La Manada'), por lo que el resultado a veces puede variar según la persona en el juicio. Con la complejidad que supone este tipo de resultado, se opta por eliminar las sentencias que contienen un fallo doble.

- **Fecha:** la fecha en la que se dictó la sentencia.

Esta tabla principal a su vez recibirá información de:

- **Ponente:** Todos los jueces que aparecen en las resoluciones.
- **Tribunal:** definirá en qué tribunal se ha realizado la resolución, si en el penal, de instrucción, de violencia sobre la mujer, etc.
- **Tipo de resolución:** aunque a priori el trabajo está concebido para analizar sentencias, se ha estructurado la tabla de manera que también pudieran descargarse otras resoluciones judiciales. Con esta variable sabríamos si se trata de una sentencia (pone fin al procedimiento), un auto (deciden sobre cuestiones tasadas en la ley, como la admisión de pruebas o las medidas cautelares) o una providencia (se refieren a cuestiones procesales). Pero por utilidad finalmente se opta por descargar únicamente sentencias en nuestra base de datos.
- **Voces procesales/sustantivas:** Son las características del caso relacionadas con el derecho procesal (el que se aplica en el procedimiento, la forma de actuar en juicio, redactar escritos, etc.) y el derecho sustantivo (el que define los derechos y deberes de los ciudadanos).
- **Resoluciones_vprocesales/vsustantivas:** son tablas que sirven de intermediario entre la tabla de Resolución judicial y voces procesales/judiciales. Una resolución judicial tiene varias voces de cada tipo, por lo que se necesita este tipo de tablas puente para que la información quede mejor estructurada y sea más clara.

Esta será, por tanto, la estructura inicial de nuestra base de datos, recogida en la Figura 11, la cual no será la definitiva, pues posteriormente se decidirá si emplear todas estas variables o, por el contrario, optar únicamente por algunas de ellas por aportar información que pueda estar duplicada en ambas.

Figura 11: Estructura de la base de datos inicial

Una vez que tenemos el esqueleto de la base de datos, el siguiente paso es la obtención de éstos.

5.2 Obtención de datos

Para la obtención de los datos, atendiendo a los criterios anteriormente expuestos, nos decantamos por una de las bases de datos jurídicas de pago pero disponibles como estudiante de la Universidad Complutense de Madrid y se programa el raspado de datos para la misma.

En primer lugar observamos cómo al acceder a cada una de las sentencias de la base de datos, éstas cuentan con un identificador, por lo que procedemos a guardar los códigos de aquellas sentencias que sean del orden penal y de los juzgados unipersonales que ya indicamos anteriormente.

Para ello, debemos indicar a nuestra araña que realice los pasos necesarios para filtrar conforme a nuestras necesidades. En primer lugar, debe seleccionar el orden penal, posteriormente escoger los juzgados, el tipo de sentencia y accionar el botón de búsqueda.

Además, solo aparecen en pantalla un número determinado de sentencias, por lo que es necesario que el programa realice *scrolls*[8] hacia abajo para que se vaya actualizando la búsqueda y aparezcan nuevas sentencias.

Para ello, necesitamos importar determinadas librerías en *Python*:

- Módulo *re*, para expresiones regulares[9], pues queremos buscar patrones con una determinada forma donde funciones como *"find()"* o *"index()"* son limitadas y el proceso es más complejo.
- *Selenium*: es un entorno de pruebas de software portátil para aplicaciones web. En concreto utilizaremos *Selenium Webdriver* para controlar un navegador específico (del cual antes deberemos obtener su *driver*), enviando comandos y recuperando los resultados obtenidos. En nuestro caso hemos decidido utilizar el navegador Chrome, pero hay otros disponibles como Firefox o Microsoft Edge.
- *Beautiful Soup*: para encontrar y extraer fácilmente los datos que necesitamos.
- Time: aunque se usa principalmente para trabajar con fechas y horas lo utilizaremos para aplicar un formato de tiempo determinado en la ejecución de nuestro *crawler*, en concreto los segundos que el programa realizará el *scroll*.
- *Unicodedata*: es un módulo que proporciona acceso a una base de datos de caracteres Unicode y define las propiedades de los mismos, de manera que podemos leer cualquier texto aunque posean caracteres que no son comunes a todos los lenguajes. En nuestro caso es necesario para caracteres como las tildes o la letra ñ, entre otros.

[8] El *scroll* es un término inglés cuyo significado se relaciona con el desplazamiento. Es conocido principalmente en el sector de los videojuegos pero que cobra relevancia según el tipo de diseño y uso de una determinada *web*. Se denomina *scroll* al desplazamiento que generalmente un usuario realiza, ya sea hacia arriba o hacia abajo para mostrar el contenido de una *web*, *app*, etc.

[9] Las expresiones regulares son patrones que se usan para encontrar una determinada combinación de caracteres dentro de una cadena de texto.

Una vez que tenemos instaladas estas librerías y las hemos llamado, el programa en primer lugar abrirá el navegador gracias al *driver* que hemos descargado y que debe encontrarse en el mismo lugar del *script*, indicándole para ello la *url* donde se encuentra el buscador de jurisprudencia con la función *driver.get*(nuestra *url*) (Figura 12).

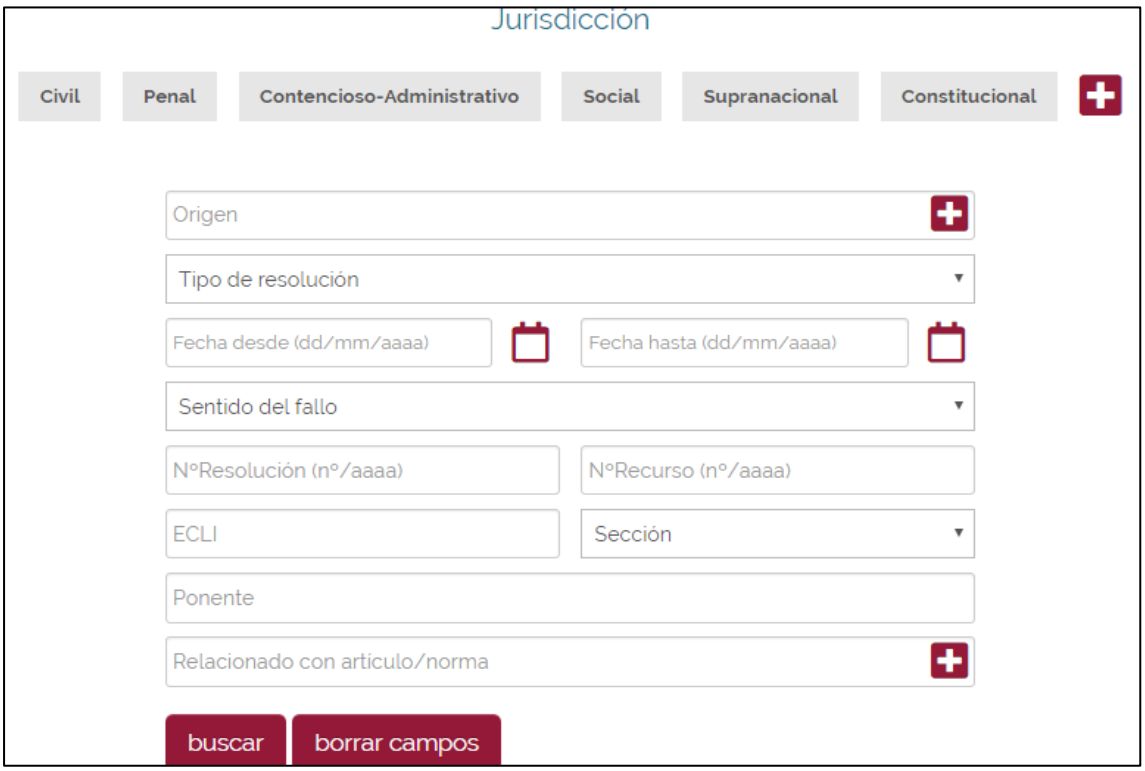


Figura 12: Aspecto del buscador de jurisprudencia

Tras esto, indicamos los filtros que ha de seguir, esto es, que seleccione las sentencias del orden penal para los Juzgados de Primera Instancia e Instrucción, Juzgados de Instrucción, Juzgados de lo Penal y Juzgados de Violencia sobre la mujer, dados los delitos a los que nos vamos a ceñir. Además, seleccionamos el tipo de resolución judicial, en nuestro caso, sentencias. Para realizar esto tenemos que indicar al programa cómo interactuar con la página encontrando cada uno de los elementos que necesitamos.

Para ello, utilizamos la función "*driver.find_element_by_id*("nuestro elemento").*click*" y podremos encontrar los elementos definidos con un id y posteriormente seleccionarlos con *".click"*. Esto nos sirve para que seleccione el tipo de juzgado que deseamos, por ejemplo. Es una manera sencilla y rápida de encontrar elementos, pero algunos de ellos no cuentan con un id, por lo que tendremos que buscar el elemento según su *xpath*[10].

```
1. chrome_options = webdriver.ChromeOptions()
2. driver = webdriver.Chrome(executable_path='./chromedriver.exe', chr
   ome_options=chrome_options)
3.
4. base_juridica = "https://bucm.idm.oclc.org/login?url=la url
   referente a la base de datos jurídica"
```

[10] XPath (XML Path Language) es un lenguaje que permite recuperar información de un documento XML. Lo que hace es definir una sintaxis para establecer partes en un documento XML, permitiendo navegar a través de sus elementos y atributos.

```
5.
6. driver.get(base_juridica)
7.
8. driver.get('https://www.tirantonline.com/tol/busquedaJurisprudencia
   /index')
9.
10.
11.    #hacemos click en el botón de Penal
12.    content = driver.find_element_by_xpath('//*[@id="boxJuris"]/labe
   l[2]').click()
13.
14.    #seleccionamos los juzgados
15.    content = driver.find_element_by_id('origen_id_84').click()
```

Como ya hemos explicado anteriormente, el número de sentencias que muestra como resultado la base de datos es limitado y necesitamos realizar varios *scrolls* para poder obtener todos los identificadores de las sentencias.

```
1. for i in range(0,2000): # ponemos el número de scrolls que
   necesitamos
2.    driver.execute_script('window.scrollBy(0, 900)')
3.    time.sleep(1)
```

Lo que hacemos con esto es que el *scroll* vaya de cero a dos mil veces y ejecuta el movimiento hacia abajo hasta la línea 900. Si no hay, empieza de nuevo en cero hasta la próxima línea 900 (desde donde lo dejó), y entre cada iteración espera un segundo para que dé tiempo a la página a recargarse.

Finalmente, una vez que el programa ha terminado de cargar todas las sentencias deseadas, debemos obtener los identificadores de las mismas para posteriormente comenzar a rellenar nuestra base de datos.

Para ello, creamos una lista de identificadores vacía y, como podemos observar en la Figura 13 en la página siguiente, unas sentencias están coloreadas en color gris claro y otras en gris oscuro de manera alternativa, lo cual nos resulta útil para encontrar los elementos por una y otra clase.

De esta manera primero buscamos los identificadores dentro de "*zebra_light*" que se corresponde con las sentencias coloreadas en gris claro, y posteriormente realizamos el mismo proceso con "*zebra dark*", esto es, las sentencias coloreadas en gris oscuro. Finalmente indicamos que vaya rellenando un txt que hemos creado, "*lista_ids.txt*" con los identificadores de las sentencias de ambos colores.

```
1. for div in content:
2.         codigo = div.get_attribute('innerHTML')
3.         soup = BeautifulSoup(codigo, 'html.parser')
4.
5.         for enlace in soup.find_all('a', href=True):
6.                 if 'map' in enlace['href']:
7.                         id = re.search("/map/(\\d+)?", enlace['href
   '])
8.                         id = id[0].replace('/map/','')
9.                         lista_ids.append(id)
```

```
10.
11.    content = driver.find_elements_by_class_name('zebra_dark')
12.    for div in content:
13.            codigo = div.get_attribute('innerHTML')
14.            soup = BeautifulSoup(codigo, 'html.parser')
15.
16.            for enlace in soup.find_all('a', href=True):
17.                    if 'map' in enlace['href']:
18.                            id = re.search("/map/(\\d+)?", enlace['h
   ref'])
19.                            id = id[0].replace('/map/','')
20.                            lista_ids.append(id)
```

Tras guardar los identificadores de las sentencias que cumplen con los filtros establecidos, podemos proceder a extraer de las mismas los datos que estimamos oportunos para crear así nuestra propia base de datos.

```
21.    with open("lista_ids.txt", "w") as text_file:
22.        for id in lista_ids:
23.            text_file.write(id + "\n")
```

Figura 13: Visualización de los resultados obtenidos tras la búsqueda.

5.3 Extracción de datos y volcado en la base de datos

Para realizar el *web scrapping* (o raspado de los datos) volveremos a utilizar librerías como *re, beautifulsoup, unicodedata* y otras como:

- *Urllib3*: para obtener todo tipo de información de la *url*.
- *SQLite3*: para que desde *Python* nos podamos conectar directamente con la base de datos.

- *Sys*: proporciona acceso a variables y funciones utilizadas por el intérprete de *Python*[11].

En este caso, en primer lugar debemos establecer la conexión a la base de datos SQLite y posteriormente definimos el cursor para realizar posteriormente las acciones deseadas.

Abrimos el archivo "txt" en el que en el paso anterior hemos ido copiando los identificadores de cada una de las sentencias para acudir directamente a ellas e indicamos al programa que vaya añadiendo al final de la *url* cada una de ellas con un bucle. Dentro de ese bucle, además, le pediremos que guarde una serie de datos que son los que conformarán nuestra base de datos.

Aunque hemos explicado anteriormente la estructura inicial de la base de datos y algunos de sus elementos, los datos con los que queremos trabajar de cada una de las sentencias son:

i) su identificador, que será el número en el orden en que aparezcan y se autoincrementará.
ii) id_web, es decir, el identificador que tenemos en lista_ids.txt que se corresponde con esa sentencia en cuestión.
iii) Cabecera: es un pequeño resumen del caso en cuestión.
iv) Motivo_resolución: por qué se está enjuiciando el caso, sobre qué delito versa.
v) Resumen: información sobre el juzgado donde se dicta la resolución.
vi) Ponente: la persona que ha enjuiciado el asunto.
vii) Tribunal: si se trata del Juzgado de Instrucción, Juzgado de Primera Instancia e Instrucción, Juzgado de lo Penal o Juzgado de Violencia sobre la Mujer.
viii) Fecha: cuándo se dictó la sentencia.
ix) Sentido_fallo: si se trata de un fallo que condena, absuelve, es un fallo mixto (condena de algunos delitos y absuelve de otros) o se trata de una sentencia de conformidad (el acusado admite su culpabilidad y como consecuencia la pena impuesta es menor).
x) Fallo: texto que recoge la conclusión a la que llega el Juez sobre el caso.
xi) Además, y como ya sabemos, contaremos con otras tablas intermedias que contendrán la información sobre las voces procesales y las voces sustantivas.

Gracias a *BeautifulSoup* podemos buscar dentro de la página cualquier elemento de la misma. En nuestro caso nos interesa una capa o "*div*". Una capa es un elemento invisible que puede contener cualquier elemento de una página, se utiliza para poner donde desees en la web un determinado contenido. En este caso estamos buscando una "*div*" específica, por lo que indicamos que la id de esta capa se tiene que denominar

[11] *Python* es un lenguaje interpretado, es decir, ejecuta las instrucciones directamente, sin una previa compilación del programa. Como tal, necesita un "intérprete", que es quien se encarga de procesar el código y ejecutar las órdenes. Un intérprete es un tipo de programa que ejecuta el código directamente, sin necesidad de compilarlo antes.

“jurisdicción”, “cabecera”, etc. con cada uno de los campos que forman parte de nuestra base de datos.

Una vez que tenemos ese “*div*”, usamos “*getText*” para obtener el texto de cada una de las capas, ya que de lo contrario la variable sería de tipo elemento web y tendría el texto HTML poco legible para nosotros. Al interesarnos únicamente por el texto es necesario utilizar la función “*getText*”.

Además, le indicamos que reemplace los saltos de línea (\n) y las tabulaciones (\t) por espacios (“ “).

```
1. with open('lista_ids.txt') as f:
2.         lineas = f.readlines()
3.         for id in lineas:
4.                 id_web = id.replace('\n', '')
5.
6.                 try:
7.                         r = http.request('GET', "http://www.[redacted].com/tol/documento/show/" + str(id_web), headers=headers)
8.                         pet1 = "http://www.[redacted].com/tol/documento/show/" + str(id_web)
9.
10.                            html = r.data
11.                            soup = BeautifulSoup(html, 'html.parser')
12.
13.                            try:
14.
15.                                    content = soup.find("div", {"id": "jurisdiccion"}).getText()
16.                                    jurisdiccion = content.replace('Jurisdicción:', '').replace('\t', '').replace('\n', '').replace('"', "")
```

En anexos se puede consultar el script completo para cada uno de los campos.

A continuación, es necesario grabar dicha información en la base de datos, en el caso de cada resumen o fallo, será distinto en función de la sentencia, pero para otros campos como el ponente o tipo_fallo, puede que se repitan los registros, por lo que en la base de datos lo primero que se comprobará es si dicho ponente o dicho tipo_fallo ya se encuentran en la base de datos; si está, se aumentará el contador en uno y, de no estar, se añadirá.

Una vez que tenemos los datos con sus respectivos identificadores podemos unirlos todos en la que será nuestra tabla principal, resoluciones_judiciales, para lo cual necesitamos añadir la línea con los datos generales en SQLite:

```
1. sql = "INSERT INTO `resoluciones_judiciales`(`id`, `id_web`, `cabecera`, `motivo_resolucion`, `resumen`, `ponente`, `tribunal`, `fecha`, `sentido_fallo`, `fallo`) VALUES (NULL,
   "+ str(int(id_web)) +", '"+ cabecera +"', '"+ motivo_resolucion +"', '"+ resumen +"', "+ str(int(id_ponente)) +",
```

```
    "+ str(int(id_tribunal)) +", '"+ fecha +"',
    "+ str(int(id_sentido_fallo)) +", '"+ fallo +"')
```

En el caso de voces_procesales y voces_sustantivas tenemos que hacer lo mismo con la tabla intermedia, en primer lugar ,observamos si las voces están, de ser así, aumenta el contador del id en uno y si no está lo añade. Tras obtener los identificadores se añade a la tabla resoluciones_vsustantivas y resoluciones_vprocesales.

Con todo esto ya tenemos nuestra base de datos inicial, con los campos deseados, de la que se han eliminado aquellos registros en los que faltaban varios campos y se han rellenado manualmente aquellos en los que sólo había un campo sin rellenar, obteniendo un total de 2461 sentencias. Sin embargo, es necesario añadir nuevas columnas que pueden resultarnos útiles o arrojar información sobre los fallos de las sentencias.

5.3.1 Creación campo fecha_reforma

En primer lugar, vamos a crear un nuevo campo denominado fecha_reforma en el cual tendremos en cuenta las reformas más relevantes que se han realizado en el Código Penal. Las reformas más relevantes han tenido lugar en mayo de 2010, marzo de 2011 y abril de 2015, donde desaparecen las faltas (el código contenía faltas y delitos).

```
1.          if anio < '2010' or (anio == '2010' and mes < '5'):
2.                  revision = 1
3.
4.          elif (anio < '2011') or (anio == '2011' and mes < '3'):
5.                  revision = 2
6.
7.          elif (anio < '2015') or (anio == '2015' and mes < '4'):
8.                  revision = 3
9.          else:
10.                    revision = 4
11.
12.             sql = "UPDATE resoluciones_judiciales SET fecha_reforma
    = '" + str(revision) + "' WHERE id = " + str(id)
13.             cur.execute(sql)
```

Así, encontraremos cuatro valores en este campo: 1, si son de fecha inferior a mayo de 2010; 2, si están entre mayo de 2010 y marzo de 2011; 3, si la fecha se sitúa entre marzo de 2011 y abril de 2015; y 4, si la sentencia tiene una fecha posterior.

5.3.2 Creación campo género

En la tabla ponente se creará un nuevo campo denominado “género” para estudiar si es relevante el sexo del juez a la hora de enjuiciar un determinado caso.

Para hacer más cómoda la tarea de manera automática, utilizamos la lista de nombres de hombres con frecuencia igual o mayor a 20 personas proporcionada por el Instituto

Nacional de Estadística[12], haciendo lo mismo para las mujeres. Las listas resultantes están en formato Excel, para nuestra comodidad las transformamos en un documento "txt" y separamos los nombres por comas.

Para crear este campo utilizaremos las librerías re, el módulo normalize de unicodedata (para eliminar tildes, o letras como la ñ no presente en otros idiomas), así como la librería sqlite3 para conectarnos con nuestra base de datos.

En primer lugar establecemos la conexión y el cursor y le indicamos que se va a ejecutar en la tabla ponente.

Creamos 4 variables diferentes:

- Contador_mujer: para que cuente los nombres que aparecen únicamente de mujer.
- Contador_hombre: para que cuente los nombres que aparecen únicamente de hombre.
- Contador_mixto: si encontramos el nombre tanto en la lista de hombres como en la de mujeres. Aunque no tiene sentido aparentemente, existen nombres que se consideran tanto de hombre como de mujer, por ejemplo, "José María" está compuesto por "José", considerado nombre masculino, y "María", como nombre de mujer. Lo mismo ocurre con "María José". Con esto veremos si nos encontramos ante casos como este para derivarlos a género masculino o femenino.
- Contador_nada: porque el nombre del juez no aparezca en la lista y tengamos que revisarlo manualmente. Esto ocurrió en pocas ocasiones en las que el respectivo juez tenía delante un "Sr." o "Sra." unido a su nombre. Al ser menos de 5 casos se modificaron manualmente en la propia base de datos.

Posteriormente comenzamos el bucle e indicamos dentro de nuestra tabla ponente dónde se encuentra el nombre de los jueces y lo separamos por comas con la función "Split()" para poder compararlo posteriormente. Creamos la variable "es_mujer" para aquellos casos en los que el nombre del juez coincida con la lista txt de nombres femeninos del INE y la variable "es_hombre" para los casos en que el nombre del juez coincida con la lista txt de nombres masculinos del INE.

Con "*normalize*" del módulo re quitamos en primer lugar las tildes, ponemos todas las letras en minúscula para poder compararlos e indicamos que, si la palabra que encontramos aparece en "es_mujer", el género será femenino y, si se encuentra en "es_hombre", el género será masculino.

Al realizar el programa hubo varios nombres que se introdujeron en "contador_mixto", como los citados "José María" y "María José". Si nos damos cuenta de esto, el primer nombre es masculino y comienza por un nombre masculino (José) mientras que el segundo nombre es femenino y comienza por un nombre femenino (María), por lo que

[12] https://www.ine.es/dyngs/INEbase/es/operacion.htm?c=Estadistica_C&cid=1254736177009&menu=resultados&idp=1254734710990

le indicamos al programa que tenga en cuenta la primera palabra para derivarlo a hombre o mujer.

```
1.                  if palabra in nombres_mujer:
2.                          es_mujer = 1
3.                          sql = "UPDATE ponente set genero = '
                                    MUJER' WHERE id=" + str(id)
4.                          cur.execute(sql)
5.                          break
6.                  if palabra in nombres_hombre:
7.                          es_hombre = 1
8.                          sql = "UPDATE ponente set genero =
   'HOMBRE' WHERE id=" + str(id)
9.                          cur.execute(sql)
10.                            break
11.
12.           if es_mujer:
13.                   contador_mujer = contador_mujer + 1
14.           elif es_hombre:
15.                   contador_hombre = contador_hombre + 1
16.           else:
17.                   contador_nada = contador_nada + 1
```

5.3.3 Pre-análisis de los datos

Para conocer si los datos que hemos volcado en nuestra base de datos se han introducido bien realizamos un preanálisis de los mismos en R con las librerías “tm” (para *text mining*) y “*wordcloud*” para crear una nube de palabras que nos permita comprender, en función del tamaño y el color de las palabras, la frecuencia de las mismas, para estudiar la coherencia de los datos, ya que es de esperar que palabras relacionadas con sentencias como “delito”, “acusado” o referencias a la ley sean frecuentes si hemos incluido bien los datos en nuestra base de datos.

Para ello en primer lugar se realiza un procesamiento del lenguaje natural, que se explicará detalladamente en el epígrafe 5.4, ya que en este caso se trata de un procesamiento más genérico consistente en eliminar tildes y caracteres extraños de las palabras, así como algunas palabras que hemos estimado oportunas y las *stopwords*, que son palabras que carecen de significado.

Con todo ello y empleando la función *wordcloud* podemos obtener la frecuencia de palabras tanto para el resumen de la cabecera (Figura 14) como para el fallo de la sentencia (Figura 15).

Observando la Figura 14 podemos comprobar que, efectivamente, los resultados tienen sentido, ya que toma como palabra más relevante “artículo”, algo que parece evidente ya que el juez se tiene que fundamentar en la Ley, por lo que continuamente hace alusión a los artículos que necesita para justificar sus decisiones.

Además, vemos otras palabras como “lesiones”, lo que parece indicar que muchas de las sentencias seleccionadas tratan sobre delitos de lesiones. Lo mismo podríamos pensar sobre el término imprudencia. Los delitos se pueden cometer con voluntariedad (y es lo que jurídicamente se conoce como “dolo”) o por imprudencia y, parece que para

las sentencias analizadas, la mayoría de los delitos sobre los que versan se han cometido por imprudencia.

Figura 14: Nube de palabras para la variable "cabecera"

Por lo que se refiere a la nube de palabras del fallo de la sentencia, también parece tener sentido y vemos cómo es evidente que las palabras más relevantes varían, pues mientras que el resumen trata de los hechos, la fundamentación y el resultado, el fallo sólo expone el resultado final de condena o absolución.

Figura 15: Nube de palabras para la variable "fallo"

En este caso, la palabra que más se repite es "sentencia" y es que es muy habitual que esta palabra aparezca en el fallo con frases como "es sentencia que se dicta en Madrid".

Además, parece evidente que palabras como “meses” se repitan en el fallo, ya que cuando se condena a alguien, bien sea a trabajos a la comunidad o a prisión, la pena se establece siempre en meses. Con la multa ocurre lo mismo, por lo que vemos que la palabra “euros” también es recurrente en el fallo. También vemos otras como “derecho”, prisión”, etc. muy frecuentes en los pronunciamientos jurídicos, por lo que parece que los datos introducidos son correctos.

Si además queremos saber en concreto y de forma gráfica cuáles son las palabras más frecuentes en el texto, podemos hacerlo con la función *findFreqTerms*.

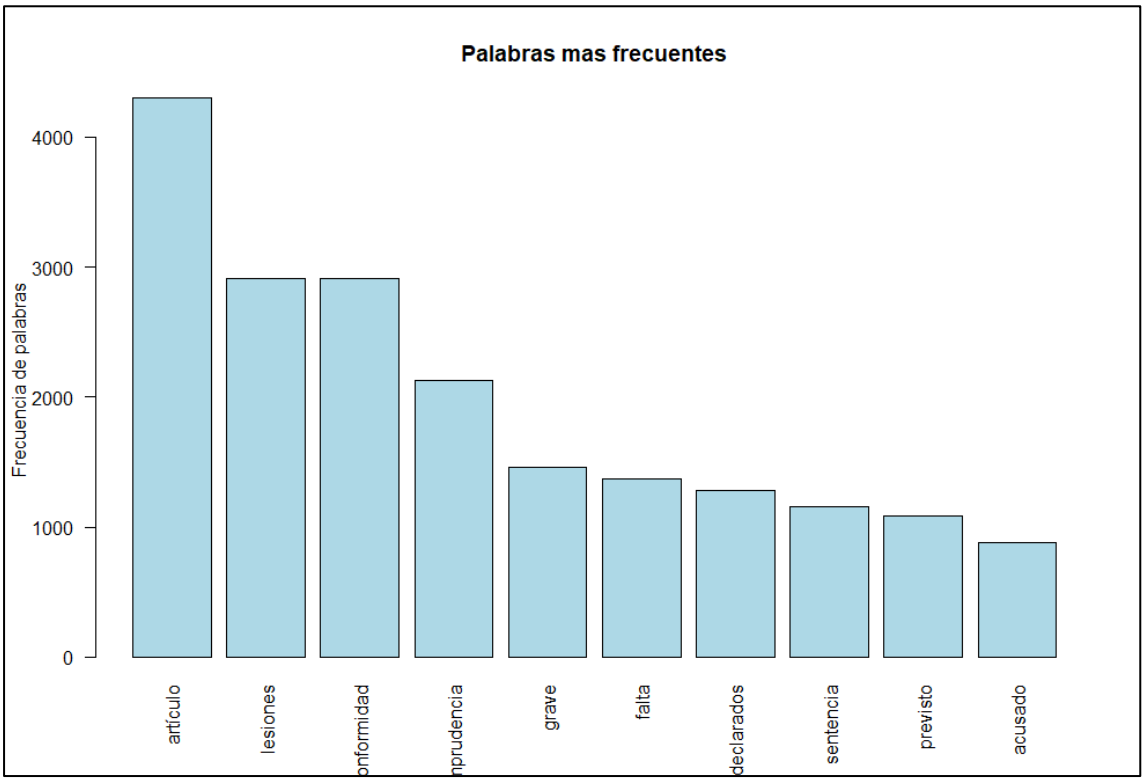


Figura 16: Gráfico de palabras más frecuentes en la variable “cabecera”

Para el caso de la variable cabecera (Figura 16) se evidencian las palabras que ya habíamos destacado en la nube de palabras, pero ahora además conocemos la frecuencia de cada una de ellas y podemos ver que, por ejemplo, en el caso de la palabra “artículo”, se repite más de 4.000 veces en las sentencias; y lesiones, conformidad o imprudencia más de 2.000.

5.4 Procesamiento del Lenguaje Natural (NLP)

Como se trata de procesar lenguaje natural, utilizaremos una librería muy popular de *Python* para ello: “*Natural language toolkit (NLTK)*”. Adicionalmente, utilizaremos las librerías *unicodedata* (para la decodificación de caracteres de forma normalizada) y sqlite3 (para incorporar el resultado a nuestra base de datos).

Debemos instalar la librería NLTK y algunos de sus paquetes, como "*punkt*", el cual contiene modelos para *tokenizar* el texto, es decir, separar el texto en *tokens*[13]; y otros como "*stopwords*[14]" llamadas también palabras vacías, que no aportan información y se eliminarán del texto.

```
1. import nltk
2. from nltk.corpus import stopwords
3. from nltk.tokenize import RegexpTokenizer
4. from nltk.stem import SnowballStemmer
5. import re
6. from unicodedata import normalize
7. import pymysql
8. import sqlite3
9.
10.    #ejecutar esto la primera vez solo
11.    nltk.download('stopwords')
12.    nltk.download('punkt')
```

En primer lugar y para trabajar más fácilmente con el texto transformamos todas las palabras a mayúscula (también podríamos haberlo hecho en minúscula). Además quitamos cualquier signo de puntuación. Tras esto, pasamos el texto a *tokens*, es decir, lo transformamos en una lista con todas las palabras. Eliminamos las palabras vacías o *stopwords*.

```
1. def resumir_texto(texto):
2.          texto = texto.upper()
3.          tokenizer = RegexpTokenizer(r'\w+')
4.          #pasamos el texto a tokens
5.
6.          tokens = tokenizer.tokenize(texto)
7.
8.
9.          #primera importamos la lista de palabras que tiene la
10.                libreria NLTK como palabras vacias
11.             palabras_a_quitar = set(stopwords.words('Spanish'))
12.             #Pasamos todas las palabras a mayusculas
13.             palabras_a_quitar = [x.upper() for x in
14.             palabras_a_quitar]
```

Una vez que tenemos únicamente las palabras que aparentemente son útiles realizamos el *stemming*, un proceso que, como hemos visto, consiste en reducir cada palabra a su raíz o base. Aunque existen muchos algoritmos de *stemming*, hemos utilizado *SnowballStemmer*, disponible en el módulo NLTK. Una vez reducidas las palabras a su

[13] Un *token* es una cadena de caracteres contiguos entre dos espacios, o entre un espacio y signos de puntuación si no eliminamos esto. Un *token* también puede ser un número. En nuestro caso se trata de transformar nuestro texto en una lista de palabras, en la que cada palabra es un elemento de la lista y está separada de las demás por una coma, como es habitual en las listas.

[14] Las *stopwords* o palabras vacías hacen referencia a aquellas palabras que carecen de significado por sí solas. Por ejemplo dichas palabras no están registradas por los principales motores de búsqueda (*Google*, *Yahoo*, *Bing*, etc.), por no aportar valor si se escriben solas o sin una palabra clave o *keyword*. Se denominan con dicho nombre porque son palabras que se filtran en el procesamiento de los datos de lenguaje natural. Los ejemplos más habituales son artículos, pronombres y preposiciones.

raíz o base, si lo deseamos podemos eliminar aquellas que estén duplicadas o aquellas que no contengan letras.

```
stemmer = SnowballStemmer('spanish')

        tokens_v4 = [stemmer.stem(i) for i in tokens_v3]

        #Si queremos quitar palabras duplicadas
        tokens_v5 = set(tokens_v4)

        #ahora quitamos todos los tokens que no contengan letras
        tokens_v6 = list(tokens_v5)
           for token in tokens_v6:
                   for c in token:
                           if not (c >= 'A' and c <= 'z'):
                                   tokens_v6.remove(token)
                                   break
```

Cuando obtenemos el resultado deseado, preparamos la lista final en tipo *string* para insertarla en la base de datos, nos conectamos a dicha base de datos y creamos un nuevo campo que será "cabecera_normalizada", actualizando la base de datos para que agregue en cada sentencia las palabras útiles y reducidas a su base o raíz. Este campo es el que recoge el resumen que aparecía en la cabecera, como ya indicamos anteriormente. Se decide utilizar únicamente este texto porque el fallo es consecuencia de dicho resumen y de la variable objetivo, y las voces procesales y sustantivas aportan una información similar en muchos casos a la aportada por el resumen de la cabecera. De incluir todas las variables que estamos mencionando estaríamos duplicando la información de manera inútil, afectando negativamente a nuestro modelo.

```
for i in busqueda:
        id = i[0]
        cabecera = i[10]
        cabecera_normalizada = resumir_texto(cabecera)

        sql = "UPDATE `resoluciones_judiciales` SET
       `cabecera_normaliz`='" + cabecera_normalizada + "' WHERE id
  =
       " + str(id)
        cur.execute(sql)

  connection.commit()
```

Con el texto ya reducido a su raíz vamos a medir la frecuencia de aparición de las palabras.

5.5 Term Frequency – Inverse Document Frequency (TF*IDF)

Hemos utilizado para este proceso la librería *sklearn* de *Python*, ya que ofrece herramientas sencillas para realizar tanto la *tokenización* – que ya hemos realizado – como la extracción de características de sus datos de texto.

Nos conectamos a nuestra base de datos y utilizamos la función "*TfidfVectorizer*" que *tokenizará* documentos, aprenderá el vocabulario y las ponderaciones inversas de

frecuencia de documentos y podremos calcular para cada una de nuestras palabras el TF*IDF.

```
#TF*IDF
vectorizer = TfidfVectorizer()

#lo pasamos por nuestra lista
vectorizer.fit(lista_fallos_nor)

#lista de puntuaciones por cada palabra
puntuaciones = vectorizer.idf_
#diccionario de palabras
  vocabulario = vectorizer.vocabulary_
```

Establecemos cuáles son los cortes que queremos realizar para quedarnos con las palabras útiles, es decir, eliminando aquellas con una puntuación muy baja y muy alta. En nuestro caso para ello primero imprimimos en pantalla las palabras y sus puntuaciones para hacerlo de la manera más coherente posible. Al realizarlo, se observan muchas palabras que contienen puntuaciones altas y bajas y menos que tienen posiciones intermedias. Hay que recordar que los valores de las palabras se encuentran comprendidos entre 1 y 8, siendo aquellas palabras con un valor bajo las que se repiten con mayor frecuencia, siendo poco importantes por este motivo; y aquellas con valor alto las que aparecen muy pocas veces y tampoco ofrecen mucha información.

En este caso hemos puesto como valores de corte a modo de prueba el mínimo de 4 y el máximo de 6 y vemos cómo de esta manera de más de 137.000 palabras considera útiles 10.000 aproximadamente. Sin embargo, hay que tener en cuenta que considera las palabras de cada uno de los documentos por separado, es decir, que la misma palabra aparece repetida en numerosas ocasiones y a lo largo de varias o muchas sentencias, por lo que posteriormente tendremos que filtrar dichas palabras cuando queramos convertir cada una de las palabras en un campo propio de cada sentencia.

Decidimos quedarnos con el intervalo de puntuación [4,6] ya probado al ofrecer una cantidad considerable de palabras y proporcionarnos aquellas que no aparecen en numerosas ocasiones, pero tampoco en una única ocasión y que pueden no resultarnos útiles. Para eliminar las palabras que se quedan fuera de este rango establecemos la condición de que la puntuación de las palabras sea o bien menor a cuatro o bien mayor que seis.

Creamos para ello una nueva columna que se denomina “palabras_utiles” que contenga únicamente las palabras que hemos decidido conservar para cada sentencia.

```
for resolucion in busqueda:
        lista_palabras = resolucion[1].split(', ')
        lista_palabras_texto = resolucion[1]
        id_resolucion = resolucion[0]


        for palabra in lista_palabras:

                total_palabras += 1
```

```
10.
11.                     try:
12.                             if vocabulario_punt[str(palabra)] < 4 or
    vocabulario_punt[str(palabra)] > 6:
13.
14.                                     lista_palabras_texto = lista_pal
   abras_texto.replace(str(palabra) + ', ', '')
15.                                     palabras_eliminadas += 1
16.                     except:
17.                             lista_palabras_texto = lista_palabras_te
   xto.replace(str(palabra) + ', ', '')
18.                             palabras_eliminadas += 1
19.
20.                     sql = "UPDATE resoluciones_judiciales SET
   palabras_utiles = '" + lista_palabras_texto + "' WHERE id=
   " + str(id_resolucion)
21.                     cur.execute(sql)
```

Del resultado inicial de 30.000 palabras que la función TF*IDF consideraba útiles, hemos ordenado al programa que se quede únicamente con aquellas que no estén repetidas, ya que dichas palabras pueden aparecer en más de una sentencia y en más de una ocasión en cada una de ellas, por lo que el resultado final es de 1.206 palabras útiles.

5.6 Creación de tabla “palabras_utiles” para realizar componentes principales

Una vez que sabemos cuáles son las palabras útiles que debemos emplear, es necesario ordenarlas en una nueva tabla.

Queremos saber la importancia de cada palabra en cada una de las sentencias a nivel individual, no teniendo en cuenta el total de documentos. Nos interesa saber el número de veces que aparece una determinada palabra en una misma sentencia para conocer su puntuación, por ello tendremos en cuenta el número de veces que aparece una determinada palabra repetida, aunque crearemos un campo único para cada palabra en la nueva tabla “palabras_utiles” que estará enlazada a la tabla principal a través del campo “id_resolucion”, gracias al cual podremos conocer la puntuación de cada una de las palabras en cada sentencia.

```
1. for palabra in palabras:
2.                 #print(palabra + ': ' +
   str(vocabulario_punt[palabra]))
3.                 try:
4.                         sql = "INSERT INTO `palabras_utiles`(`id`,
   `id_resolucion`, `palabra`, `puntuacion`) VALUES
   (NULL," + str(id_resolucion) + ",'" + palabra
   + "','" + str(vocabulario_punt[palabra]) + "')"
5.                         cur.execute(sql)
6.                 except Exception as e:
7.                         print(str(e))
```

Una vez que tenemos dicha tabla, necesitamos exportarla para poder disminuir el número de variables con las que contamos.

Para ello, en *Python* seleccionamos todos los campos de palabras útiles y creamos una lista vacía en la que incluiremos las palabras que queremos que sean campos. Una vez creada dicha tabla, con un bucle añadimos todas las palabras a la lista sin duplicarlas. Tendremos como resultado una lista de listas donde cada lista será una sentencia con la puntuación de cada palabra, apareciendo un 0 en aquellos campos que no tenga información por no aparecer dichas palabras en esa sentencia en concreto.

Finalmente lo exportamos como csv para continuar con el análisis de componentes principales en R.

5.7 Análisis de Componentes Principales

Las palabras útiles arrojan como resultado una *sparse matrix* o matriz dispersa, es decir, una matriz en la que la mayoría de sus elementos son ceros, pues muchas de las palabras no aparecen en muchas de las sentencias. Además, se trata de una matriz de gran tamaño, de 2461 filas por 1206 columnas. Se trata de una matriz con muchos ceros de la cual necesitamos reducir su dimensionalidad. Por todo esto, es conveniente emplear la librería *irlba* en R, que está diseñada para realizar el análisis de componentes principales de matrices amplias y dispersas. También se utilizarán las librerías *factoMineR* y *factoextra* para realizar alguna representación gráfica del análisis.

Realizamos el análisis con 500 componentes principales utilizando la matriz de correlaciones y obtenemos los autovalores, que nos servirán para decidir con cuántos componentes nos quedamos. Además, obtenemos el porcentaje de varianza explicada por cada uno de los componentes y el porcentaje de varianza acumulada según vamos recorriéndolos.

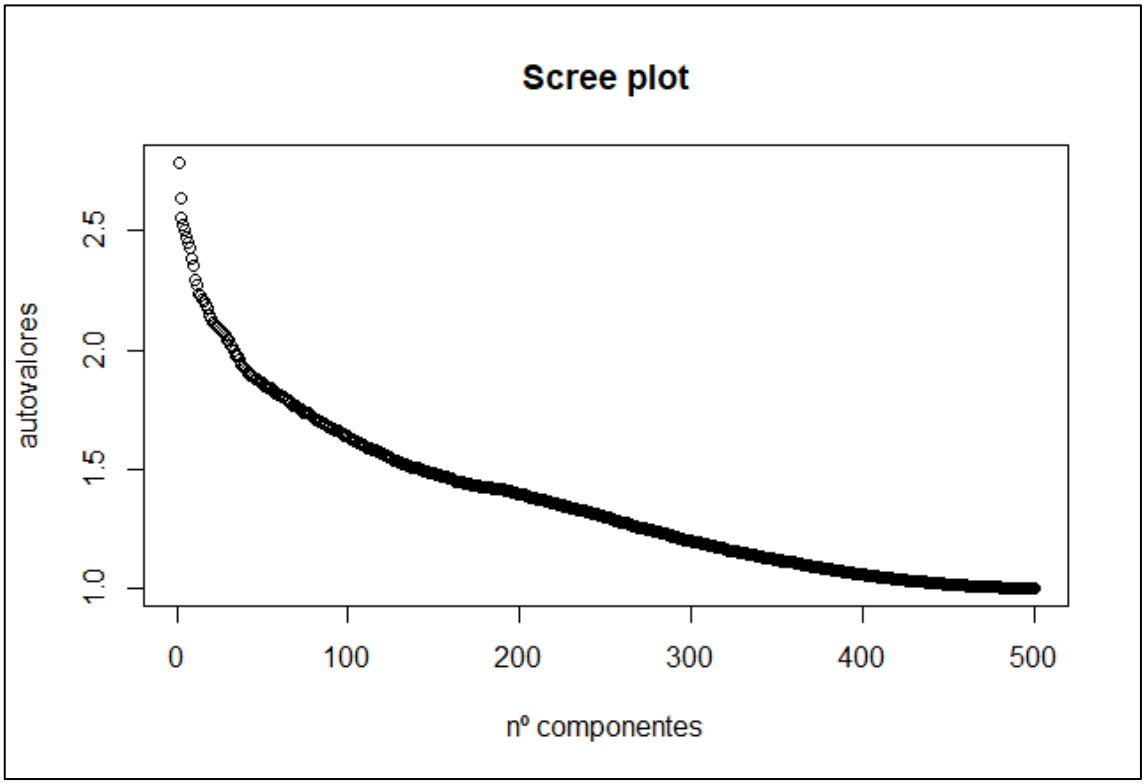


Figura 17: Scree Plot

Aunque para considerar el número de factores a retener no hay una norma básica, podemos utilizar un gráfico de sedimentación (*Scree Plot*) de los valores propios como el de la Figura 17 y determinar el número de componentes en el que observamos un "codo" o una estabilización en el descenso. En este caso, aunque no se ve nada claro parece que dicha estabilización o pequeño codo se realiza alrededor del valor 200.

Otra opción para tener en cuenta el número de componentes con el que debemos trabajar es la varianza explicada mínima que deseamos que dichos componentes expliquen (Figura 18).

Si observamos la Figura 18, podemos observar como en el lugar de la componente 200, donde el *Scree Plot* hacía el amago de descenso más pronunciado, se explican el 50% de los datos, ya que la varianza acumulada es de 0.5. Vemos que con las 500 componentes que hemos decidido investigar se explica el 80% de la varianza. Por ello, dado que parece que el descenso comienza en la componente 200 y que de 1206 variables con 200 componentes conseguimos explicar la mitad de la varianza los datos, nos quedaremos con dicho número de componentes ya que conseguimos explicar la mitad de la información. Parece que la correlación de las variables es fuerte ya que con 500 componentes conseguimos explicar el 80% de la información, cosa que no ocurriría si el peso de todas las componentes fuese el mismo, síntoma de la escasa relación que se procurarían en dicho escenario.

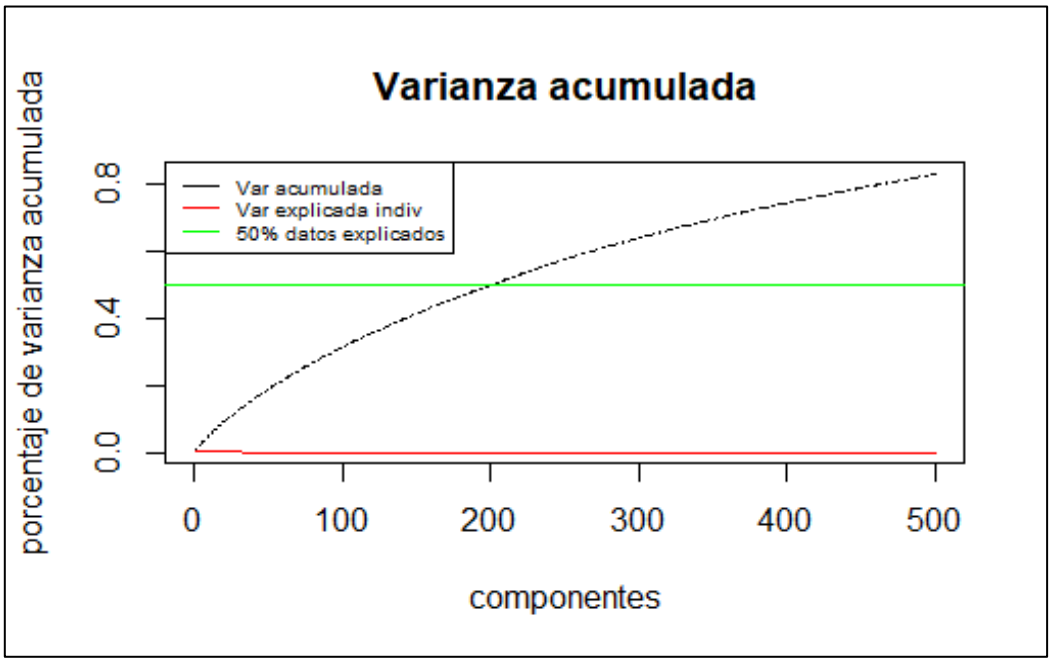


Figura 18: Porcentaje de varianza acumulada[15]

Con la cantidad de datos que tenemos no se pueden realizar las interpretaciones y los análisis clásicos, pero podemos observar cuáles son las 10 palabras con más peso de las 10 primeras componentes principales en la Tabla 1 a modo ilustrativo.

Vemos que hay palabras con un gran peso, ya que se repiten en varias componentes. En el caso de las componentes 4, 5 y 6 las palabras con más peso están relacionadas con el mundo médico. En las sentencias penales se tratan mucho este tipo de palabras

[15] La varianza acumulada es la acumulación de la varianza explicada a nivel individual.

cuando los delitos supuestamente cometidos son de lesiones, homicidio o agresiones sexuales, por lo que tiene sentido que sean relevantes en las componentes.

Tabla 1: 10 variables con más peso para las 10 primeras componentes

CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9	CP10
presum	lat	fraudulent	cicatric	apofisis	quirurg	homenaj	tenenci	nervi	diput
rotund	siguient	envenen	rehabilit	costal	falopi	recaudatori	armas	ats	aragon
ofend	botell	tributari	apofisis	cicatric	optic	tenenci	nervi	fiabl	espectrometr
ende	razon	llamp	sacr	sacr	ectop	armas	ats	elarticul	branqui
evidenci	rome	teori	costal	rehabilit	implant	port	fiabl	eman	monzon
clar	esquadr	agrav	tributari	persist	porcentaj	metr	alarm	emit	empres
indirect	acomet	siguient	llamp	homenaj	rasg	trabaj	port	tratosexpuest	somet
puest	vien	botell	fraudulent	recaudatori	camp	comun	metr	gent	remit
manifiest	reglamentari	lat	envenen	peset	recaudatori	gravesolicit	gent	vieron	emit
afect	reput	razon	teori	preci	homenaj	peset	comun	refund	elarticul

También podemos observar el ya indicado problema de “faltas de ortografía”, ya que muchas veces en las sentencias, al ser publicadas en las bases de datos, no incluyen la tabulación adecuada y dos palabras se encuentran juntas, como es el caso de “elarticul” que aparece en nuestra tabla en componentes como la 9 y 10; o “gravesolicit” y “tratosexpuest” en las componentes 7 y 9 respectivamente. Se trata, por desgracia, de errores habituales, ya que de otro modo habrían sido descartados al emplear la función “*Term Frequency – Inverse Document Frequency”*.

De este modo, conseguimos reducir las más de 1.200 variables a tan solo 200, a las que les añadiremos las variables de nuestra base de datos que consideramos que aportan utilidad para realizar la modelización, entre las cuales no se encontrará la regresión logística por el gran volumen de variables que tenemos y por su imposibilidad de interpretación al haber realizado el análisis de componentes principales, principal punto a favor de la regresión.

5.8 Modelización

Una vez que tenemos nuestros datos depurados y con las variables deseadas procedemos a modelizar. El conjunto de datos está formado por 205 variables y 2461 observaciones. Las variables continuas son las 200 componentes principales que hemos tomado a partir de las palabras útiles obtenidas en el resumen del caso. Además contamos con la variable identificadora y cuatro variables categóricas:

- Genero_juez: cuyo valor es 0 si es mujer y 1 si es hombre.
- Tribunal: cuyo valor es 1 para los Juzgados de lo Penal; 2 para los Juzgados de Primera Instancia e Instrucción; y 3 para los Juzgados de Violencia sobre la Mujer.
- Fecha_reforma: cuyo valor es 1 para aquellas sentencias anteriores a mayo de 2010; 2 para aquellas sentencias que tuvieron lugar entre mayo de 2010 y marzo

de 2011; 3 para las sentencias dictadas entre marzo de 2011 y abril de 2015; y 4 para las sentencias de abril de 2015 en adelante.

- Sentido_fallo: es la variable objetivo y los valores son 0 para las sentencias absolutorias y 1 para las sentencias condenatorias. Como ya indicamos anteriormente, se han eliminado previamente aquellas sentencias con resultados de “fallo_doble” y se han incluido en el fallo condenatorio la opción “sentencia_conformidad”, por acabar el acusado condenado igualmente.

Tras determinar qué rol posee cada una de las variables realizamos una exploración de los datos.

Como podemos observar en la Figura 19, la muestra está desbalanceada, ya que un 80 por ciento de los datos se corresponden con el evento 1, es decir, el fallo condenatorio, y el 20 por ciento restante con el fallo absolutorio. Es decir, si no realizáramos ningún análisis y ante un nuevo registro indicásemos que el resultado va a ser condenatorio, la tasa de fallos sería del 20 por ciento. Hay que tener este dato en cuenta a la hora de analizar los resultados obtenidos para saber si realmente estamos obteniendo una mejora con cada modelo.

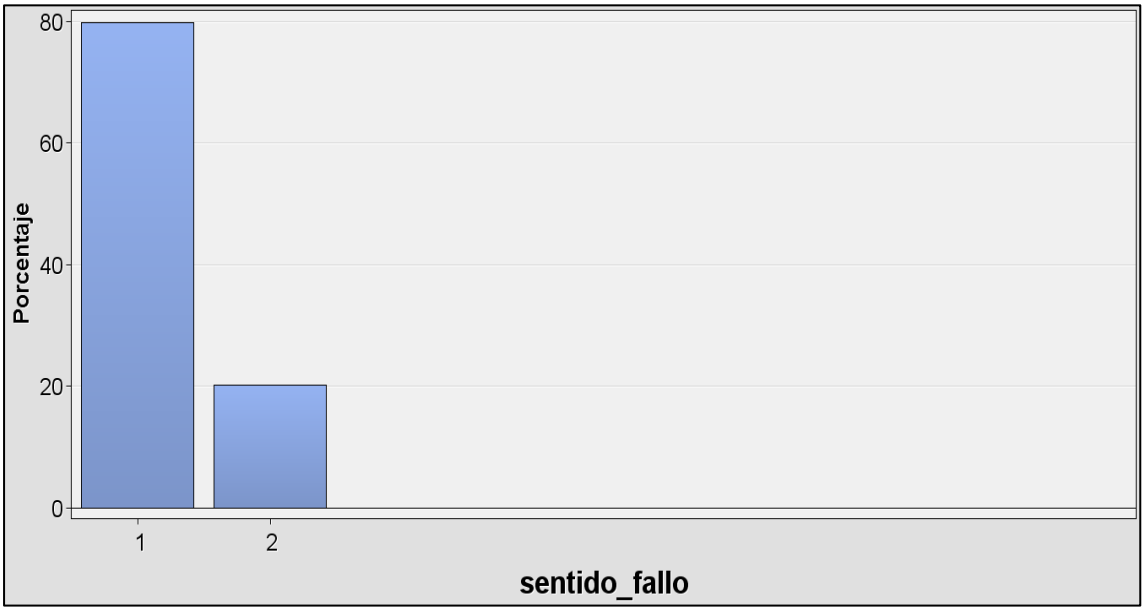


Figura 19: Porcentaje de la variable objetivo.

En cuanto a las variables categóricas, no tienen mucha relación entre sí, lo cual lo podemos observar a través de la V de Cramer, un estadístico dispuesto en intervalo de cero a uno donde los valores cercanos a cero indican escasa relación entre cada una de las variables con la variable objetivo, y los cercanos a uno suponen una relación alta entre las mismas. Tal y como podemos observar en la Figura 20, parece que las variables tribunal, genero_juez y fecha_reforma no tienen una relación muy fuerte con el sentido del fallo, aunque la variable “tribunal” parece que es la más relevante para explicar el sentido del fallo de entre todas las categóricas.

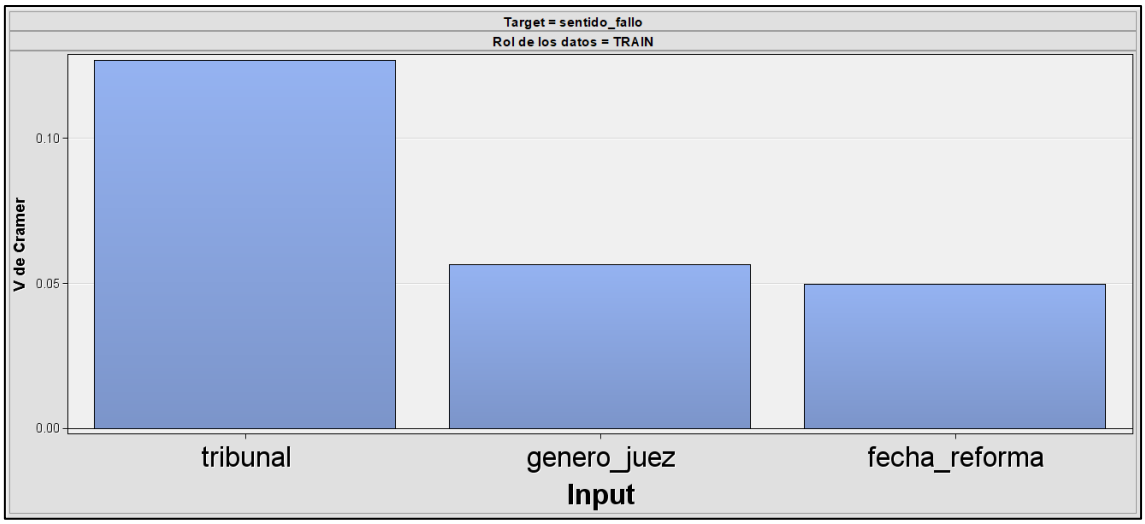


Figura 20: V de Cramer variables categóricas

Lo mismo ocurre con las variables continuas, cuyos valores son muy próximos a cero, aunque no pueden visualizarse todas a la vez, por lo que se excluye su figura por su escasa utilidad. El análisis de la V de Cramer evidencia la necesidad de emplear bastantes variables para la modelización y, pese al poco aporte, resulta útil el ranking proporcionado para saber cuáles son más relevantes para nuestra variable objetivo (Figura 21).

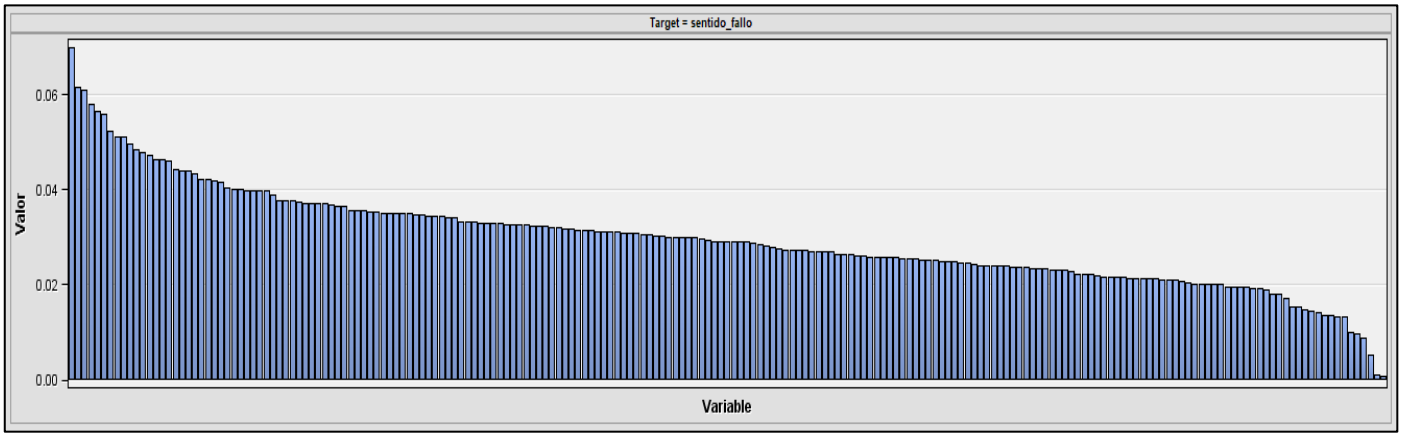


Figura 21: Ranking variables continuas

Tras el análisis inicial se comprueban que no existen errores, ya que han sido solventados a lo largo de la extracción de datos y creación de la base de datos. Tampoco existen datos *missing* ni *outliers*, por lo que realizaremos una selección de variables para realizar un ranking de variables y trabajar con más de un conjunto, comprobando los resultados que se obtienen con cada una. Para ello se realizarán diversos posibles caminos que servirán para hacer una selección de variables: creación de variables aleatorias, empleo de nodo selección, modelización con árbol de decisión, etc., y según las variables que se empleen en cada camino se crearán diferentes conjuntos/sets de variables que posteriormente serán empleados para la modelización.

En primer lugar, se han introducido dos variables aleatorias, para comprobar si hay variables que quedan por debajo de ellas, lo cual es sinónimo de relevancia nula para el futuro el modelo y podemos eliminar las variables que queden tras ellas. Se decide crear dos variables aleatorias en lugar de una con el fin de evitar la casualidad que se

pudiese dar al emplear solo una aleatoria. Como se observa en la Figura 22[16], todas las variables del explorador de estadísticos quedaban por encima de estas dos variables salvo dos: las variables genero_juez y fecha_reforma. Así, parece irrelevante teniendo en cuenta dicho resultado que el juez sea hombre o mujer para obtener un determinado resultado en la sentencia, ni tampoco las diferentes reformas que se han llevado a cabo en el Código Penal influyen para que el fallo sea condenatorio o absolutorio.

Este mismo resultado se obtuvo realizando diferentes regresiones logísticas comparando diferentes métodos y criterios de selección (*backward, forward* y *stepwise* con AIC y SBC), camino que también se decidió emplear como método de selección. En este caso el resultado obtenido para la regresión logística ganadora (una regresión logística *backward* con SBC) era el mismo que el obtenido en el empleo de las variables aleatorias como filtro, pues incluía todas las variables del modelo salvo el género del juez y la fecha en la que había tenido lugar.

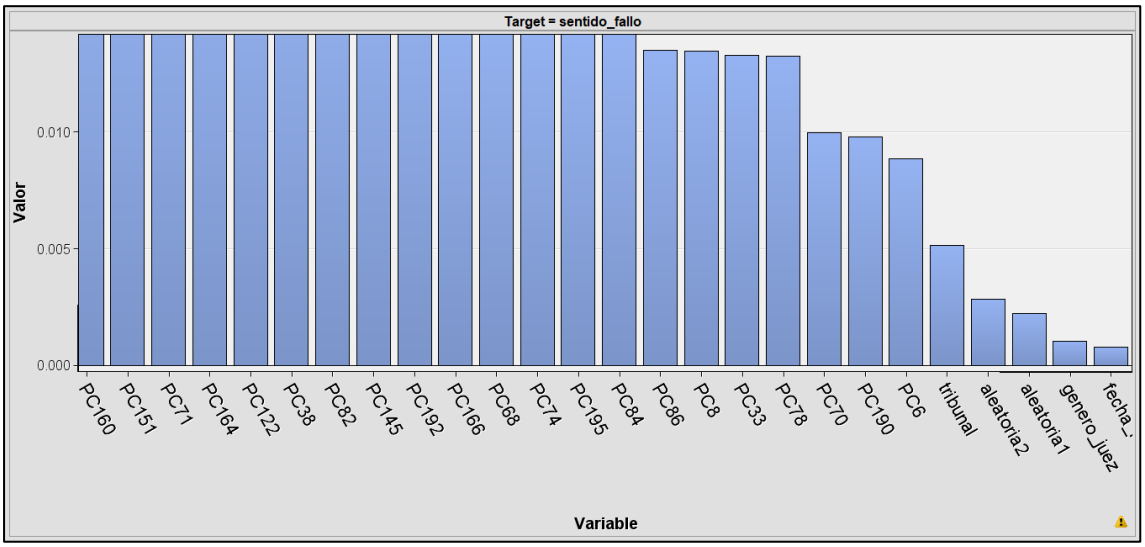


Figura 22: Selección de variables empleando aleatorias.

Otro de los métodos de selección empleados es el uso del nodo que lleva dicho nombre, el cual automatiza la decisión de si dejamos o no una variable, y lo hace en función del R cuadrado. Se puede establecer un R-cuadrado que actúe como tope y que aquellas variables que tengan menos de ese número sean rechazadas. En nuestro caso y dado que el nodo suele ser muy restrictivo y rechazar muchas variables se ha empleado un R cuadrado igual a 5.E-10.

Como podemos observar en la Figura 23, efectivamente este método de selección es mucho más restrictivo y de nuestras 205 variables considera relevantes sólo 25 de ellas.

[16] La Figura 13 es una ampliación del valor de la importancia de las variables pues al contar con tantas era imposible que apareciera el nombre de cada una de ellas si no ampliábamos hasta cierto punto la imagen. Aparecen las variables aleatorias y aquellas que quedan por detrás, por lo que sabemos que el resto están por encima y son relevantes aunque no podamos ver el orden de éstas.

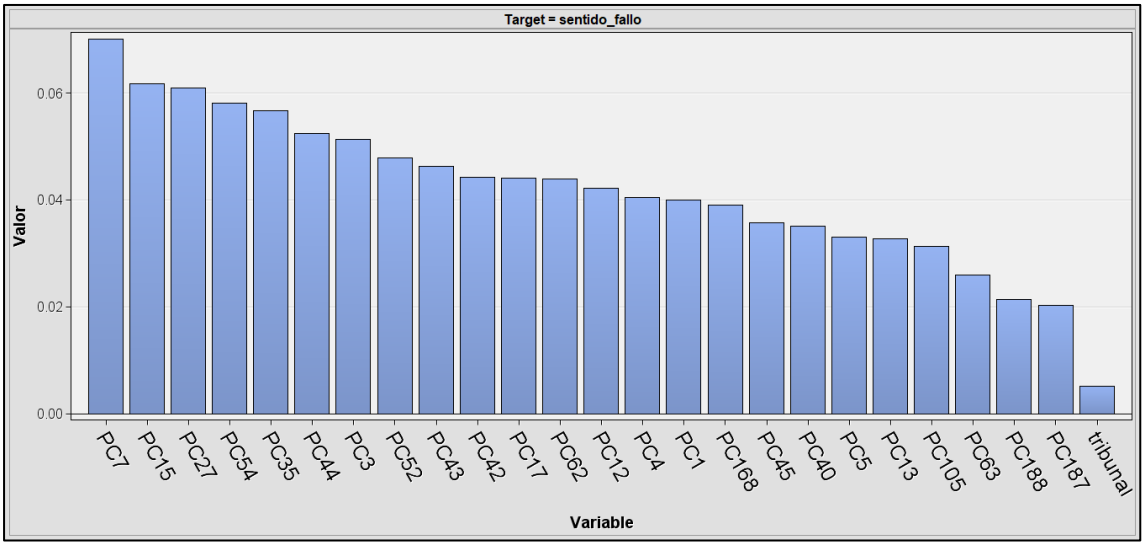


Figura 23: Selección de variables con nodo selección

Además, empleamos otros métodos de selección de variables como puede ser la realización de un árbol de clasificación, observando las variables que utiliza en su análisis.

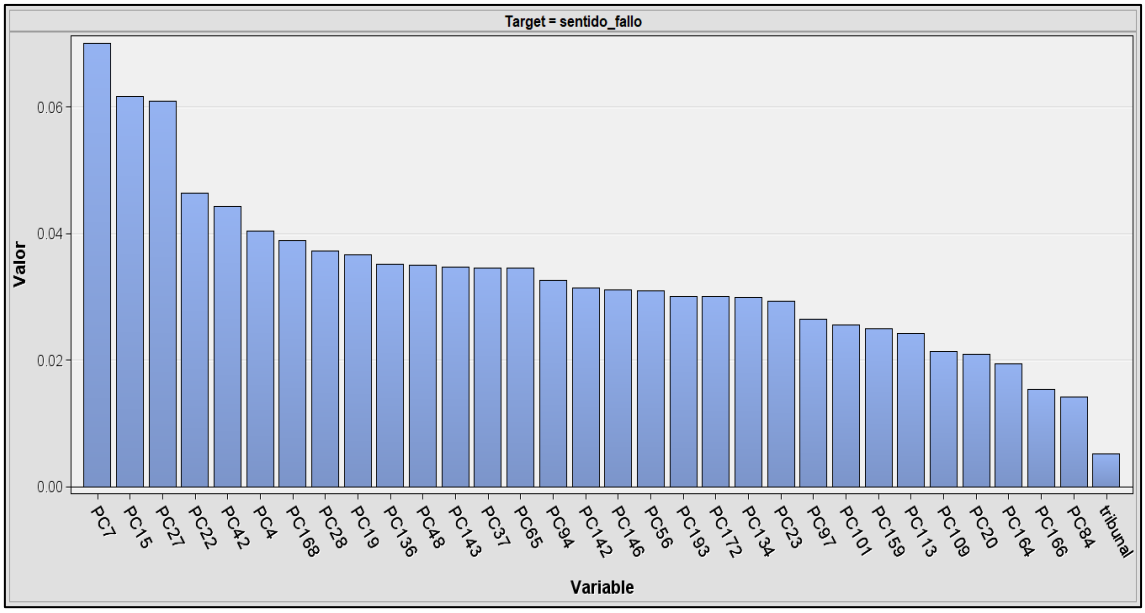


Figura 24: Selección de variables con árbol de clasificación

En este caso, tal y como se aprecia en la Figura 24, el número de variables seleccionadas es 32, escogiendo, al igual que en el caso anterior, como variables más importantes las componentes 7, 15 y 21 y como categórica la variable tribunal. Esto reafirma que efectivamente las variables género del juez y fecha de la reforma no aportan mucho valor al modelo.

Por último, hacemos lo mismo utilizando el nodo Incremento Gradiente, el cual, si nos fijamos en la Figura 25, incluye únicamente variables continuas, en concreto 48 de las 200 componentes principales.

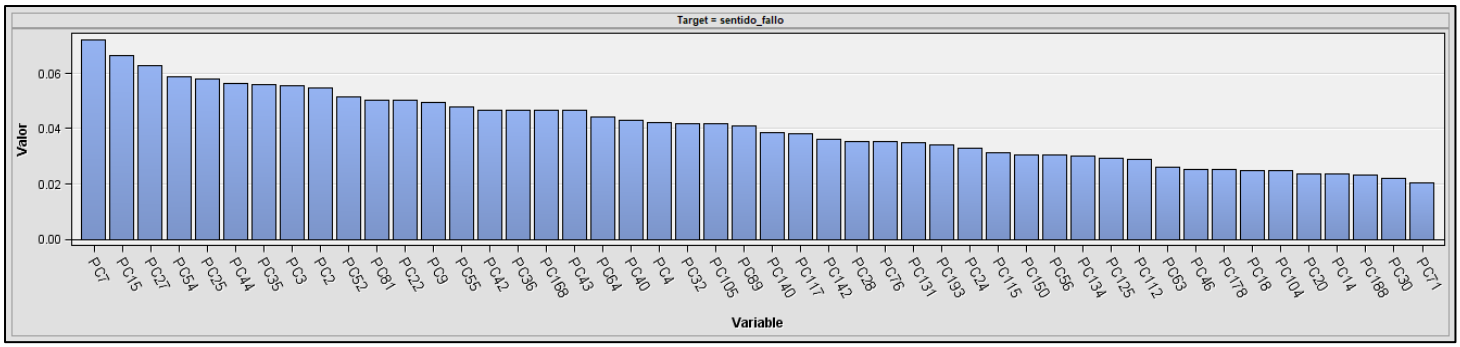


Figura 25: Selección de variables con incremento gradiente

Una vez hecho esto, obtenemos un ranking con las variables y observaremos 4 conjuntos de variables que son los siguientes:

- El primer conjunto (Set 1), conformado por las variables: PC7, PC15, PC27, PC54, PC25, PC52, PC22, PC42, PCC43, PC4, PC111, PC13, PC175, PC126, PC154, PC193, PC144, PC200, PC141, PC96, PC59, PC125, PC199, PC95, PC176, PC198, PC71, PC74, PC78, tribunal.
- El segundo conjunto (Set 2), cuyas variables son: PC7, PC15, PC27, PC54, PC25, PC44, PC35, PC3, PC2, PC52, PC61, PC22, PC9, PC55, PC42, PC36, PC166, PC43, PC64, PC40, PC4, PC32, PC105, PC69, PC140, PC117, PC142, PC26, PC76, PC131, PC193, PC24, PC115, PC150, PC56, PC134, PC125, PC112, PC63, PC46, PC176, PC16, PC104, PC20, PC14, PC166, PC30, PC71.
- El tercer conjunto (Set 3), compuesto por las variables: PC7, PC15, PC27, PC54, PC35, PC44, PC35, PC52, PC43, PC42, PC17, PC62, PC12, PC4, PC1, PC168, PC45, PC40, PC5, PC13, PC105, PC63, PC188, PC187, tribunal.
- El cuarto conjunto (Set 4): compuesto por todas las variables disponibles.

Estos conjuntos son los que utilizaremos aplicando las técnicas de *machine learning* seleccionadas, todos ellos para determinar, a posteriori, la combinación óptima de variables.

Se han empleado para la comparación de resultados tanto SAS como R, pero al ofrecer este último mayor información sobre los modelos resultantes, será el que se muestre en el cuerpo de este trabajo. No obstante, se puede consultar los resultados obtenidos en SAS en los anexos.

5.8.1 Redes neuronales

Como ya hemos indicado anteriormente, las redes neuronales artificiales se inspiran en las redes neuronales biológicas y tienen una serie de características que las hacen muy interesantes es su capacidad de auto-organizarse y adaptarse, siendo más útil para encontrar relaciones no lineales a diferencia de la regresión logística, gracias a que cuenta con nodos que trabajan en paralelo y permitiendo a la red aproximar funciones, clasificar patrones y auto-mejorarse frente al ruido.

En este caso, utilizaremos para “tunear” nuestras redes la librería *caret* de R, la cual incluye funciones que simplifican el uso de métodos de clasificación y regresión; y *nnet*,

un paquete que permite crear redes neuronales. Además, utilizaremos otras librerías como *readr*, *dummies*, *MASS* y *reshape* para tareas secundarias.

En nuestro caso, vamos a crear redes por validación cruzada repetida, ya que pese a que contamos con otros métodos de remuestreo como puede ser el training-test repetido, nuestros registros no son tan abundantes y es la técnica más fiable para el análisis de datos en estas circunstancias.

En primer lugar, importamos nuestro conjunto de datos e indicamos qué variables son categóricas y cuáles son continuas, transformando a *dummies* las variables categóricas ya que se trabaja mejor con ellas en este tipo de paquetes. Además, tenemos que estandarizar las variables continuas, ya que es primordial para poder reducir el rango de las variables input ya que si no tienen las mismas medidas puede que no lleguemos a obtener resultados o estos no sean óptimos.

En la red podemos modificar alguno de los parámetros, como puede ser el número de nodos, el cual depende de la complejidad y el número de datos. Hay que tener en cuenta que en el caso de las redes existe un criterio recomendable por la literatura, aunque no tenga que seguirse siempre, que se trata de contener al menos 20 observaciones por parámetro. Para descubrir cuántos nodos necesitan cada conjunto aplicaremos la siguiente fórmula:

$$h(k+1)+h+1=observaciones/20$$

Siendo h el número de nodos ocultos y k el número de nodos inputs, que será para cada conjunto las variables que estén incluidas en él. Así, en el caso de cada conjunto:

$$h(32+1)+h+1=2461/20$$
$$h(48+1)+h+1=2461/20$$
$$h(25+1)+h+1=2461/20$$
$$h(205+1)+h+1=2461/20$$

Para cumplir con este criterio:

- En el primer conjunto (set 1) tendríamos que emplear 3 nodos (h = 3.58).
- En el segundo (set 2), 2 nodos como máximo (h = 2.44).
- En el tercer conjunto (set 3) usaríamos 4 nodos (h = 4.5).
- En el cuarto y último conjunto (set 4), formado por la totalidad de variables, sería imposible cumplir con este criterio (h = 0.59).

Sea como fuere, en nuestro caso parece bastante limitado el empleo de un número tan escaso de nodos, por lo que probaremos con diferentes nodos para cada red atendiendo a su mayor exactitud (*accuracy*)[17].

[17] La exactitud o *accuracy* indica el grado de aproximación entre la media de los valores medidos y el valor real que se pretende medir. Próximamente, cuando hablemos de "exactitud", nos estaremos refiriendo a este concepto, más conocido como *accuracy*.

De este modo, realizamos el tuneado de las redes con el paquete caret para encontrar el número de nodos ocultos (*size*) y el *decay* adecuado. El *size* es el número de neuronas en la capa oculta, ya que estos métodos sólo crean redes neuronales con una capa oculta y tampoco podemos elegir la función de activación y el algoritmo de optimización (siendo por defecto *backpropagation*). El *decay* es el *learning rate,* y controla la regularización durante el entrenamiento de la red, es decir, determinará la longitud de la distancia cubierta en cada interacción que hará que la red se acerque al punto deseado.

En el caso de R podemos utilizar los métodos *'nnet'* o '*avnnet'*. La diferencia es que el segundo realiza varias repeticiones de entrenamiento de la red con diferentes semillas y promedia el error, mientras que *nnet* sólo lo realiza una vez, por lo que si cambia los pesos varía la predicción. Por eso no es útil usar *nnet* si no *avnnet* que sortea de manera paralela. En *avnnet* ejecutamos varias redes similares sorteando los valores de los parámetros iniciales con una medida diferente. Así se evita la dependencia del azar, del sorteo inicial de las observaciones. Por tanto, es más conveniente utilizar *avnnet* por ejecutar varias redes similares que sortean los valores de los parámetros iniciales con una medida diferente para evitar la dependencia del azar, por lo que aplicaremos este método para nuestros 4 conjuntos de variables. Esto nos da como resultado cuál es la exactitud de las redes neuronales para cada conjunto de variables en función del número de nodos y *learning rate* que hemos propuesto, quedándonos con aquella opción en la que la exactitud sea mayor para cada conjunto de variables.

Como nodos incluiremos las opciones de 3, 5, 10 y 15 nodos ya que, aunque la mayoría son excesivos para cumplir con el criterio de la literatura de 20 observaciones como mínimo por parámetro, no parecen excesivos, y no tenemos que acudir siempre a este criterio. Por otro lado, probamos con un *decay* de 0.01, 0.1 y 0.001. De este modo se han obtenido un total de 36 modelos que podemos observar en la Tabla 2, de los cuales optamos por emplear únicamente aquellos con mejores resultados para cada uno de los conjuntos.

Tabla 2: Resultados medios obtenidos en términos de exactitud para los modelos estudiados.

nº nodos	learning rate	Accuracy set 1	Accuracy set 2	Accuracy set 3
3	0.001	0.8637119	0.8705418	0.8752517
3	0.01	0.8679378	0.8721678	0.8767148
3	0.1	0.8659871	0.870459	0.8776099
5	0.001	0.8672088	0.8734637	0.8782588
5	0.01	0.8706187	0.8710276	0.8804528
5	0.1	0.8689162	0.8693203	0.8823221
10	0.001	0.8673695	0.8712693	0.8776093
10	0.01	0.8694013	0.8708654	0.8769599
10	0.1	0.8655827	0.8627399	0.8741936
15	0.001	0.8667185	0.8648491	0.8750087
15	0.01	0.8646891	0.8596493	0.8672042
15	0.1	0.8566397	0.8620856	0.86794

Según los resultados, parece que nuestros conjuntos de variables funcionan mejor con 5 nodos ocultos que con 3, que era en general el número aconsejado si queríamos cumplir con el criterio de la literatura del mínimo de observaciones por parámetro. Vemos que las diferencias no parecen muy significativas pero dado que nuestra muestra está desbalanceada optamos por emplear aquellos modelos con mejor exactitud, que son los señalados en amarillo.

La función *train* empleada para obtener los resultados de la Tabla 2 saca la mejor combinación de parámetros para cada red para posteriormente realizar la validación cruzada repetida. Para el conjunto de datos formado por todas las variables (set 4) solo se han podido obtener resultados con 3 nodos ya que no hay suficientes observaciones para estimar tantos parámetros, y solo se han podido representar los tres conjuntos restantes como podemos comprobar en la Figura 26, donde, aparentemente la red que obtiene un mejor resultado es la compuesta por las variables del nodo selección, aunque se ha obtenido la exactitud para dicho set4 con 3 nodos y se empleará dicho modelo para ese conjunto que podrá ser finalmente representado en la comparación final de todos los modelos.

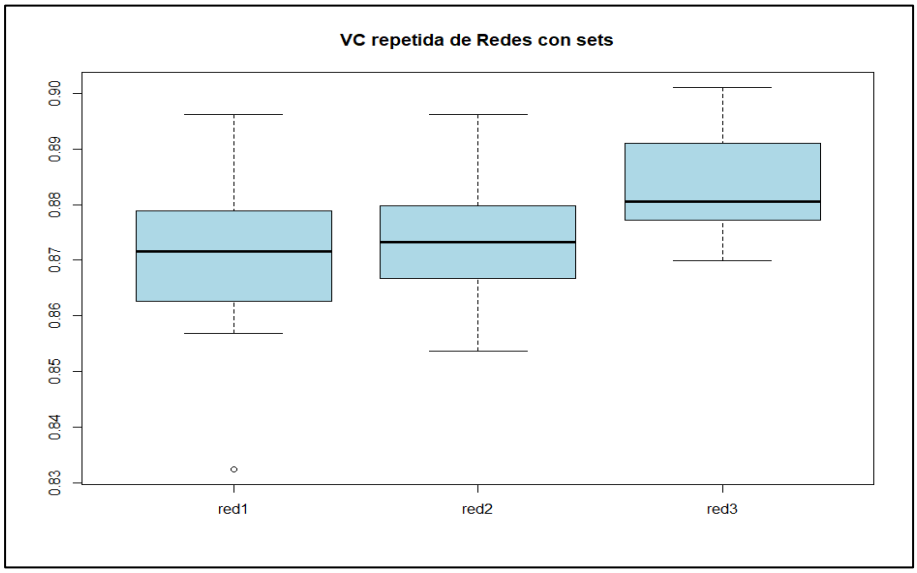


Figura 26: Boxplot representativos de la exactitud de cada mejor red

Comparar los conjuntos con la exactitud puede no ser lo más adecuado en el caso de datos desbalanceados, por lo que obtendremos también el área bajo la curva (AUC). En nuestro caso, parece más efectiva el área bajo la curva ya que nuestra muestra está desbalanceada y ésta mide cada posible punto de corte, por lo que los resultados con este indicador serán más fiables. Para obtener esto empleamos la función *"cruzadaavnnetbin*" (Portela, 2019) escogiendo para ello las redes marcadas en la Tabla 2 por ser las más precisas. De nuevo resulta imposible conseguir que la red funcione con todas las variables de nuestro modelo (set 4).

Parece que efectivamente el mejor modelo en términos de sesgo y variabilidad es el tercer conjunto (set 3), aquel que posee además menos variables. Se puede observar gráficamente en el área bajo la curva (Figura 27), donde el sesgo es menor y la variabilidad son menores, pues en el caso del primer conjunto (set 1), aunque a primera

vista pueda parecer que su variabilidad es menor, tenemos un pequeño punto atípico en torno al 0.917 que hace descartar esa concepción inicial.

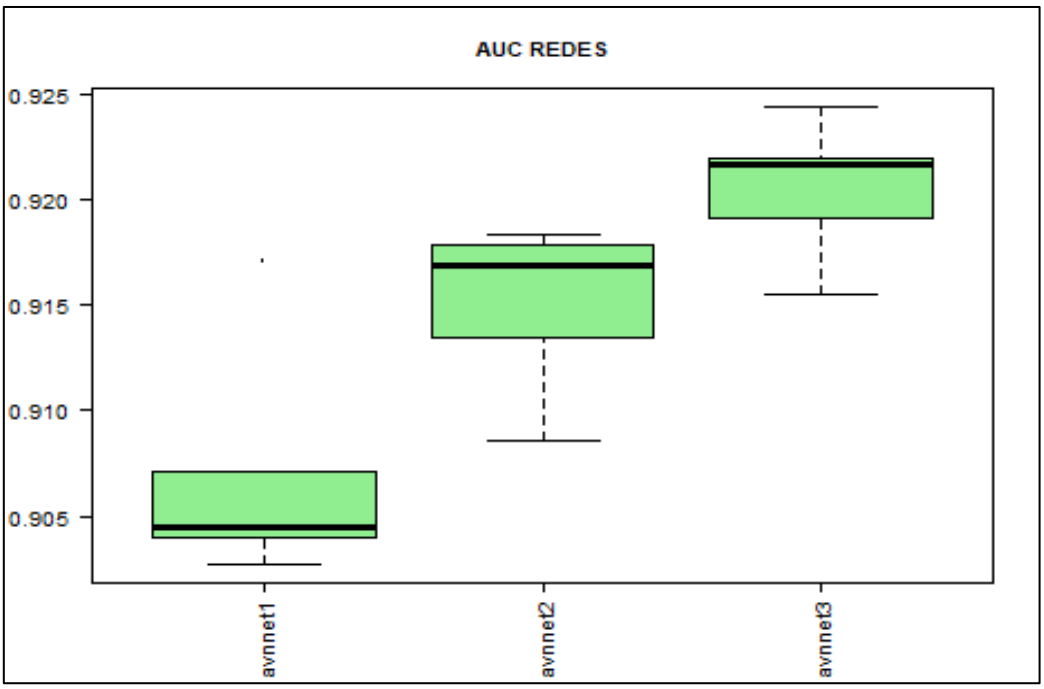


Figura 27: Área bajo la curva – redes neuronales

Se ha probado a cambiar la semilla para las redes y los resultados no difieren significativamente, en especial en el caso del conjunto 3, por lo que parece un conjunto de variables robusto.

5.8.2 Random Forest

Dados los parámetros que podemos modificar y que hemos expuesto en el epígrafe de Metodología, hemos optado por 5000 árboles con una rejilla fija que variará en función del número de variables seleccionadas para cada caso (*mtry*). Al igual que ocurría con las redes neuronales, en el caso de *random forest* también probamos con diferentes combinaciones de parámetros. Para el tamaño de la muestra se ha probado con varios valores entre 200 y 700 observaciones y para el mínimo de nodos se han probado las opciones de 6, 10 y 15 nodos. En el caso del *mtry* se prueba a incluir diferentes cantidades de variables según los conjuntos, siendo el máximo el total de variables que componen cada set (*bagging)*. Todos los árboles se han realizado con reemplazamiento.

Podemos observar en la Tabla 3 cómo, para cada conjunto de variables, el mejor *mtry* (subrayado) está cercano al empleo de todas las variables según el conjunto de datos, ya que en el caso de los conjuntos 3 y 4 la librería *caret*, empleando el método de *random forest* ("rf") ha estimado que la mejor exactitud del modelo es aquella en la que se utilizan todas las variables disponibles en este tercer y cuarto conjunto. En el caso de los conjuntos 1 y 2 no emplea todas las variables, pero está muy próximo, consiguiendo la mejor exactitud si emplea 28 de las 32 variables disponibles para el primer conjunto y 40 de las 48 existentes en el segundo conjunto.

Tabla 3: Pruebas *mtry* y exactitud para el nº de variables seleccionadas

	mtry set 1	mtry set 2	mtry set 3	mtry set 4
	3	12	3	21
	6	15	6	24
	9	18	9	30
	12	21	12	32
	15	24	15	50
	18	30	18	70
	21	32	21	100
	24	**40**	**24**	150
	28	45		180
	32	48		**205**
Accuracy	0.867046	0.8707856	0.8677795	0.8662328

La exactitud obtenida para estos modelos con los mejores *mtry* de cada set es la que aparece en la última fila de la tabla. Se ha probado además si era necesaria una condición de parada o *early stopping* por si al aumentar las iteraciones se producía sobreajuste, no siendo necesario para ninguno de los conjuntos de datos.

Como ya hemos comentado la información que proporciona la *accuracy* no es tan relevante como la que se puede obtener con el área bajo la curva. En lo que se refiere al tamaño de la muestra, los mejores resultados han sido con un número de árboles (*ntree*) igual a 200, y con un tamaño del nodo (*nodesize*) de 10. Además, se ha probado cambiando el número de árboles a 800, 2000 y 5000 y el tamaño del nodo de 6, 12 y 15 para cada uno de los conjuntos de variables.

En el caso del conjunto 4 (set4), la técnica empleada ha sido la de *bagging*, ya que se trata de un conjunto de datos que contiene todas las variables del modelo (205). Parece que, para los árboles, si observamos la tasa de fallos en la Figura 28, el menor sesgo lo obtiene el conjunto de datos número dos, que es el relativo a las 48 variables continuas seleccionadas por el nodo “incremento gradiente” en *Enterprise Miner*. Sin embargo, la variabilidad es mejor para el set3, formado por la mitad de las variables. Si observamos a la derecha el área bajo la curva ROC de los conjuntos parece que en términos de variabilidad el modelo más estable es el primer conjunto (set1), cuyas variables fueron seleccionadas realizando un árbol de clasificación. Pese a ello, en términos de sesgo es superada por el set3, que ya fue ganador en el modelo de redes neuronales. Los otros dos conjuntos restantes tienen puntos atípicos que no los hacen tan interesantes como los conjuntos ya analizados.

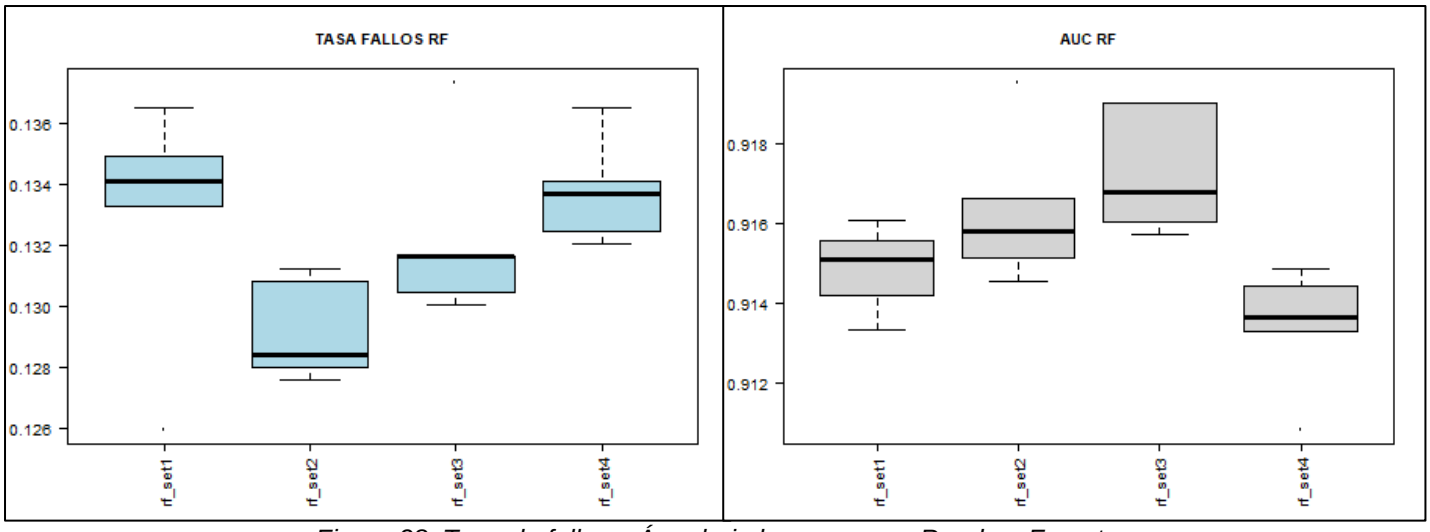


Figura 28: Tasa de fallos y Área bajo la curva para Random Forest

Cuando el algoritmo de *Random Forest* realiza un muestreo con reemplazamiento, hay observaciones que se quedan fuera (*out of bag*) y no se usan para crear el árbol. Para esas observaciones que han quedado fuera se hace una predicción y se calcula el error. Esto se hace en cada iteración para calcular el error estimado. En el caso del *bagging* en el set 4 el error cometido es del 13.41%:

```
        OOB estimate of  error rate: 13.41%
Confusion matrix:
     No  Yes class.error
No  253  236  0.48261759
Yes  94 1878  0.04766734
```

Podemos graficar dicho error (Figura 29) representando el *out of bag error* en color negro, el error al intentar predecir el fallo condenatorio (*churn = yes*) en color verde (1 – sensibilidad) y el error al intentar predecir el fallo absolutorio (*churn = no*) en color rojo (1 – especificidad), pudiendo relacionarse estas líneas con la matriz de confusión que acabamos de indicar.

Figura 29: OOB error para set 4 – bagging

Así, podemos observar que es muchos más difícil predecir los fallos absolutorios que los condenatorios, ya que el error es bastante mayor. Esto es evidente dada nuestra muestra desbalanceada, donde sólo el 20% de los datos se corresponden con el evento "no". Además, se observa que podría ser mejor empleando menos árboles, ya que si observamos la línea negra que representa el total del error llega un momento en que se aplana bastante.

Esto se ha realizado para cada uno de los conjuntos de variables y los resultados son más o menos similares, con un error entre el 12 y el 13 por ciento.

De todo lo anterior, como conclusión a este método es que el mejor modelo obtenido es aquel en el que hemos empleado 200 árboles y un *mtry* de 24 para el conjunto número 3.

5.8.3 Incremento gradiente

En primer lugar, aplicamos la función *gbm* de la librería *caret* para cada uno de los conjuntos atendiendo a distintas posibilidades en cada uno de los parámetros, como se puede ver a continuación

```
gbmgrid<-expand.grid(shrinkage=c(0.1,0.05,0.03,0.01,0.001),
                     n.minobsinnode=c(5,10,20),
                     n.trees=c(100,500,1000,5000),
                     interaction.depth=c(2))
```

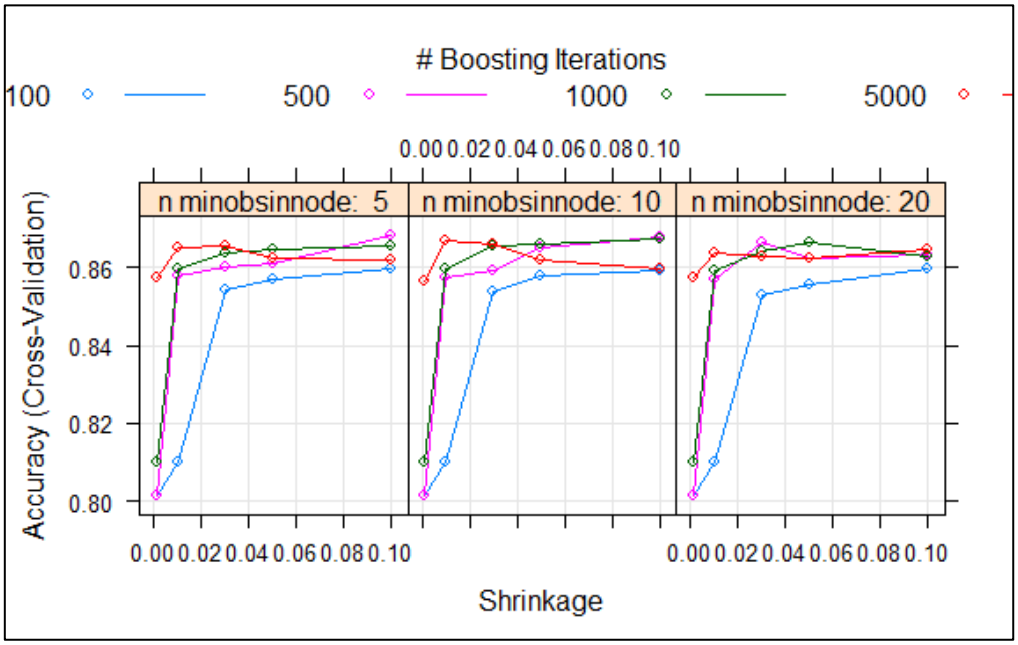


Figura 30: "Ajuste" del incremento gradiente para el set 1 en función de la exactitud

Tanto representando en modo de lista o gráficamente (Figura 30), podemos observar cuál es la customización óptima para emplear incremento gradiente en cada uno de los conjuntos atendiendo a la exactitud (*accuracy*). En el caso del set1, el mejor modelo es aquel en el que el número de árboles es 500, la profundidad 2 (que hemos fijado nosotros), el *shrinkage* 0.1 y el número mínimo de observaciones en cada nodo 5, con una exactitud de 0.8683554.

Para el conjunto 2 el modelo seleccionado es aquel formado por 1000 árboles, *shrinkage* igual a 0.05 y un mínimo de observaciones en cada nodo de 20, con una exactitud de 0.8724118.

En el caso del conjunto 3 el mejor modelo es aquel compuesto por 1000 árboles, *shrinkage* igual a 0.05 y un mínimo de observaciones en cada nodo de 20, al igual que para el conjunto número 2, con una exactitud de 0.8703767.

Por último, si empleamos todas las variables, es decir, el conjunto 4, el mejor modelo es aquel que contiene un número total de 1200 árboles, un *shrinkage* igual a 0.01 y un número mínimo de observaciones por nodo igual a 5, con una exactitud inferior al resto de conjunto de datos, 0.8289364.

Además, podemos observar la importancia de las variables para este modelo, como se puede observar en la Figura 31, donde la mayoría de las variables no resultan útiles para aplicar este algoritmo en el conjunto 1.

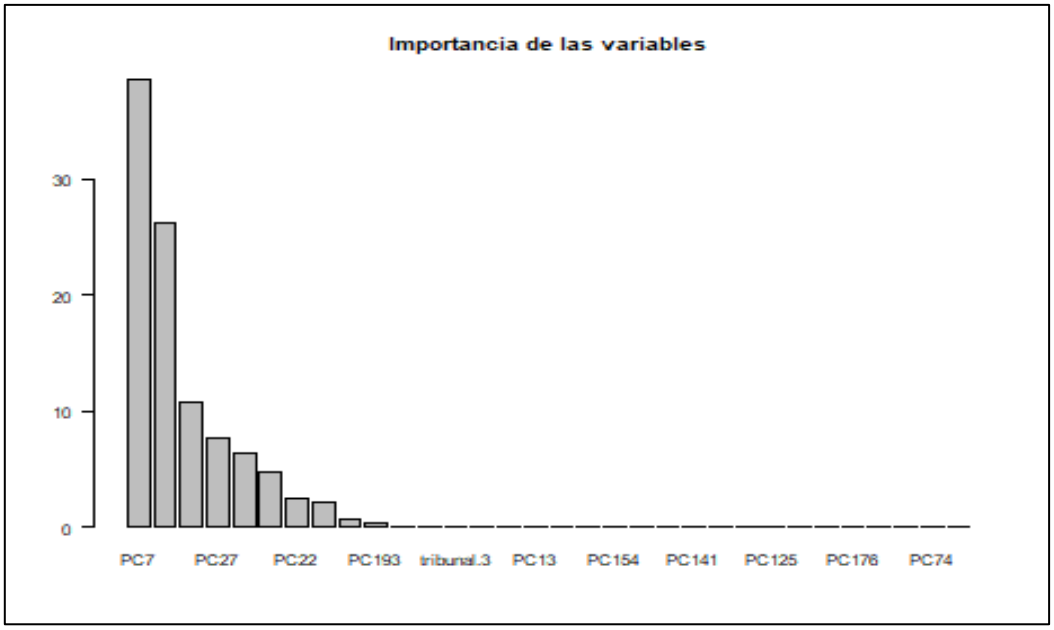


Figura 31: Importancia de las variables en set 1 aplicando Incremento gradiente

También podemos observar si es necesario o no emplear *early stopping*, ya que empleando una cantidad tan elevada de árboles podemos incurrir en sobreajuste. Si observamos la Figura 32, parece evidente cómo a partir de la iteración 800 se está produciendo un sobreajuste, pero como para este caso se han seleccionado 500 iteraciones no estamos produciendo tal sobreajuste y nuestro modelo predecirá bien tanto los datos con los que se está entrenando como los del futuro.

Si estudiamos el *early stopping* para el resto de los conjuntos, en el caso del conjunto 2 sí se necesita parar en la iteración 800, en el caso del conjunto 3 no es necesario y en el conjunto 4 se necesita parar en la iteración 1000, por lo que ajustamos de este modo los modelos para aplicar la función *cruzadagbmbin* (Portela, 2019).

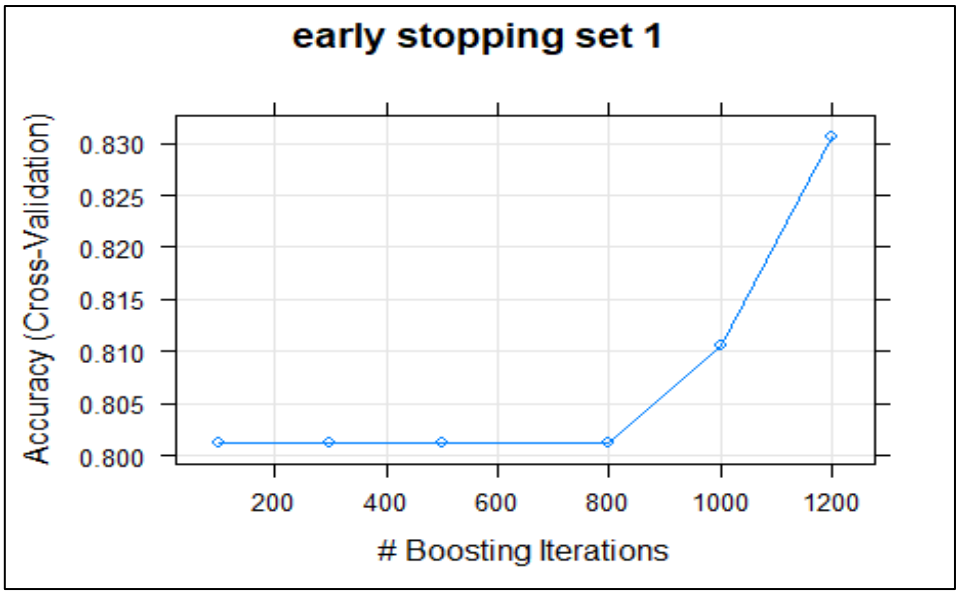


Figura 32: Early stopping set 1 – incremento gradiente

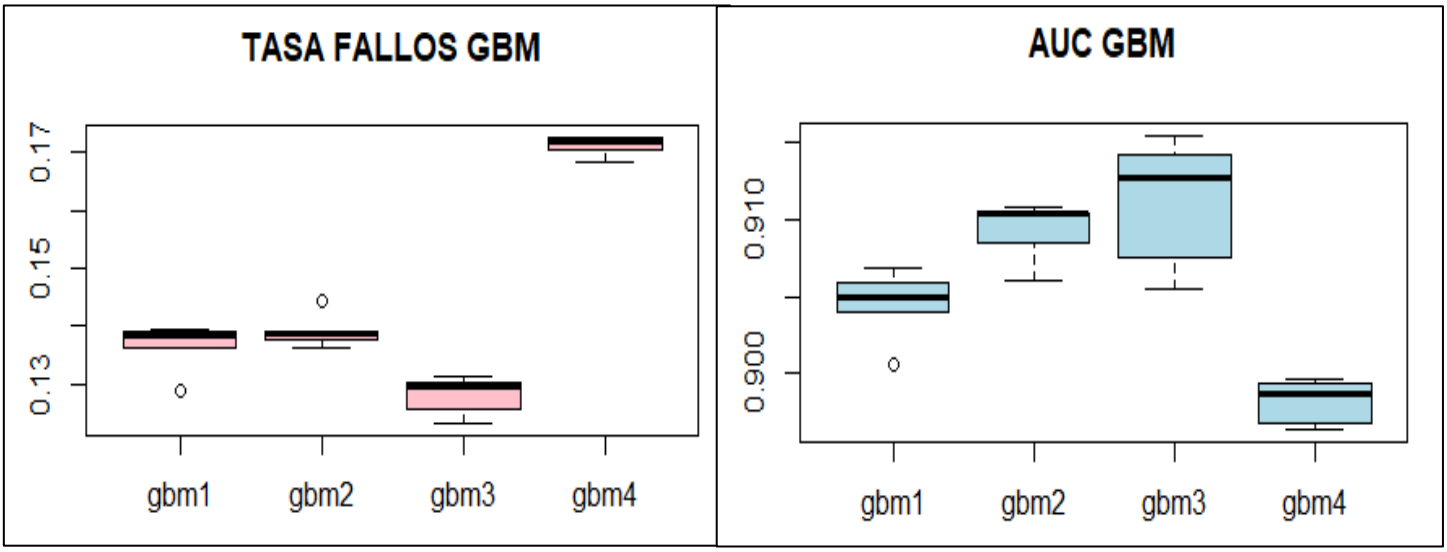


Figura 33: Tasa de Fallos y AUC empleando Incremento gradiente en los diferentes conjuntos

En este caso (Figura 33), el mejor modelo atendiendo tanto a la tasa de fallos como al área bajo la curva parece el relativo al conjunto 3, lo cual tiene cierto sentido ya que dicho conjunto está compuesto por las variables seleccionadas en *Enterprise Miner* con el nodo incremento gradiente. Sin embargo, si atendemos a la variabilidad del modelo en el caso del área bajo la curva, parece un modelo más estable el relativo al conjunto número 2, así como a nivel de sesgo, por lo que nos decantaríamos por éste.

5.8.4 XGBoost

Tal y como ocurría en *gbm*, en *Xgbm* también se calculan los parámetros para tunear el *XGboost* adecuado para cada conjunto con la librería *caret* y la función *train* aplicando el *method=xgbTree*.

Para el conjunto 2, el mejor modelo es aquel en el que son necesarios 500 árboles, con una profundidad máxima de 6, eta = 0.1, gamma = 0, y un mínimo de observaciones en nodos finales de 10.

En el caso del conjunto 1 necesitamos emplear 5000 árboles con un eta = 0.001 y resto de parámetros iguales al conjunto 2 (Figura 34).

Por lo que se refiere al conjunto 3 utilizaremos 1000 árboles con eta = 0.03 y parámetros con los mismos valores que los otros conjuntos.

Por último, en el caso del conjunto 4, necesitaremos 1000 árboles, eta = 0.03 pero un mínimo de observaciones por parámetro de 5 y no de 10 como ocurría en los conjuntos anteriores.

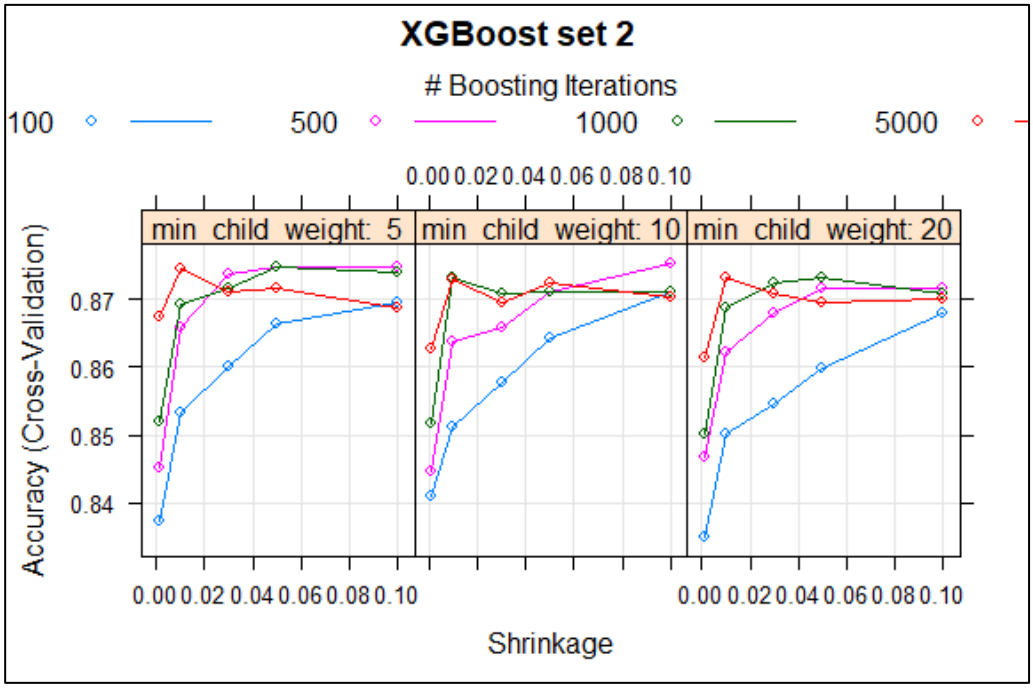


Figura 34: " Tuneado" del set 2 con XGBoost visto desde la exactitud del modelo

Estudiamos también, como para los casos anteriores, si es necesaria alguna condición de parada en cada uno de los conjuntos, no siendo necesarias para ninguno de ellos y poniendo como ejemplo la Figura 35, donde se puede verificar dicha afirmación para el conjunto 2. Esto parece coherente con el concepto de *XGboost* que busca precisamente eso, evitar el sobreajuste, por lo que se mantiene casi constante aumentando las iteraciones.

Realizamos también un análisis sobre la importancia de las variables para cada conjunto, pudiendo observar en la Figura 36 como, para dicho conjunto, al igual que ocurría en Incremento gradiente para el set 1, la variable más importante es la componente principal 7 (CP7).

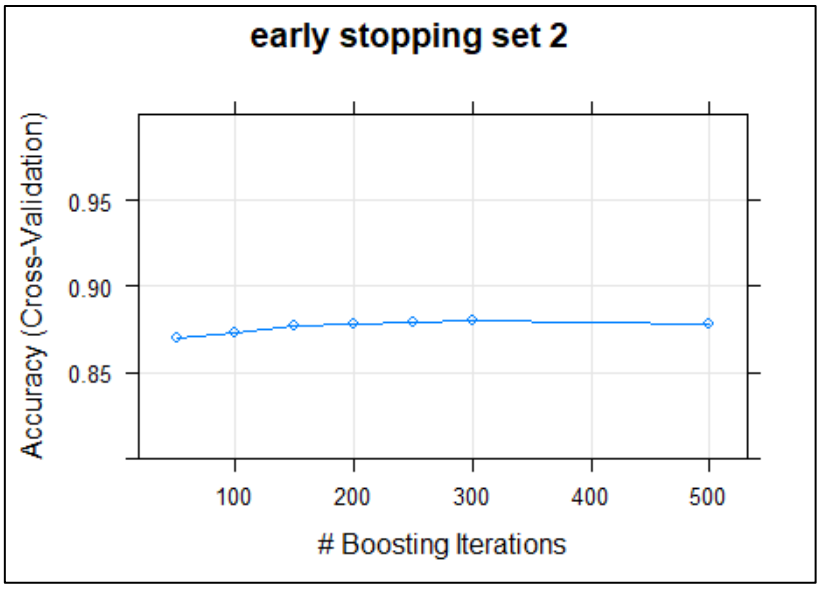


Figura 35: Estudio early stopping para set 2

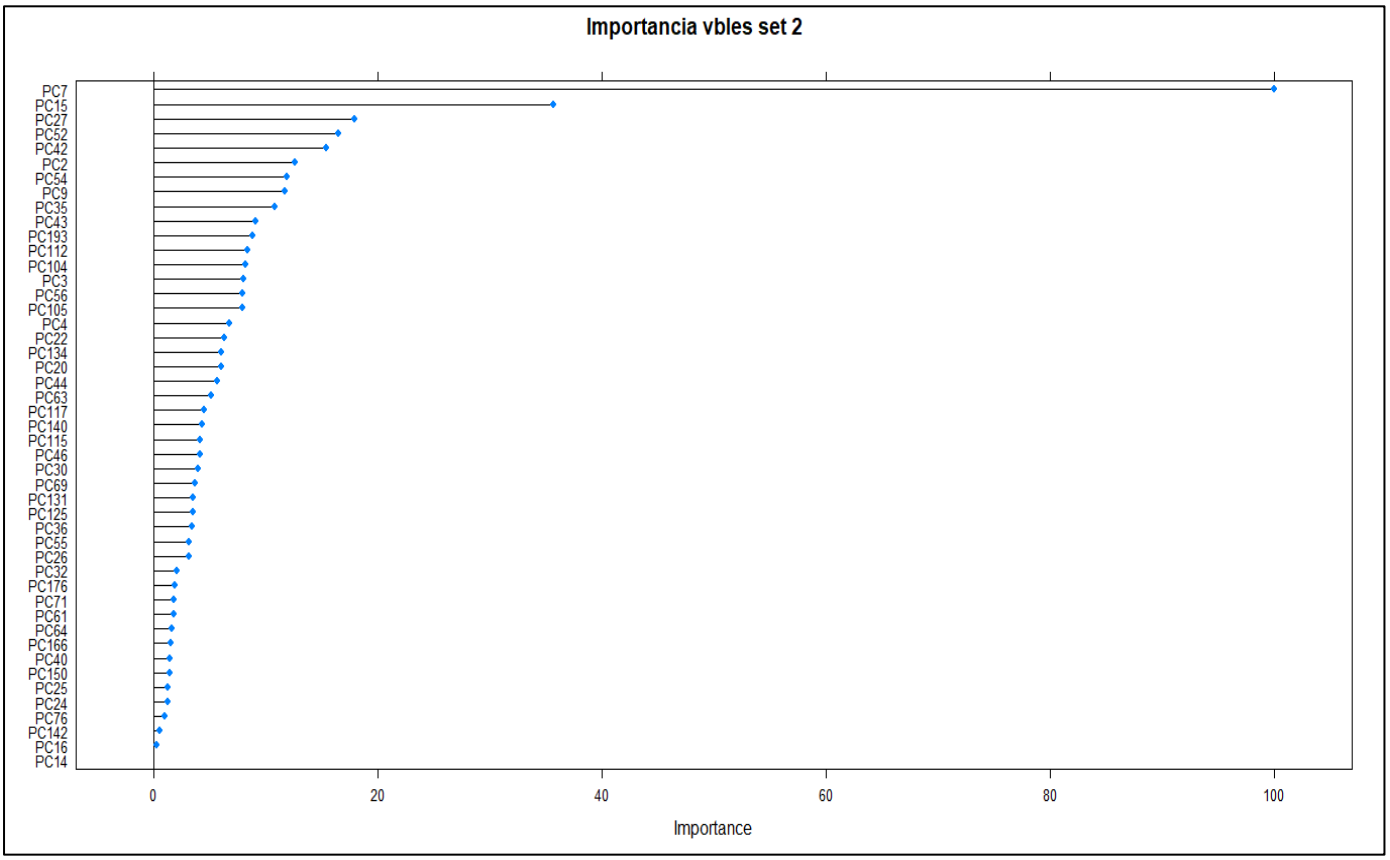


Figura 36: Importancia de las variables XGboost – set 2

Una vez realizado este estudio, aplicamos la función *cruzadaxgbmbin* (Portela, 2019) para realizar la validación cruzada repetida con los parámetros ganadores de cada conjunto y obtener su tasa de fallos y área bajo la curva (Figura 37).

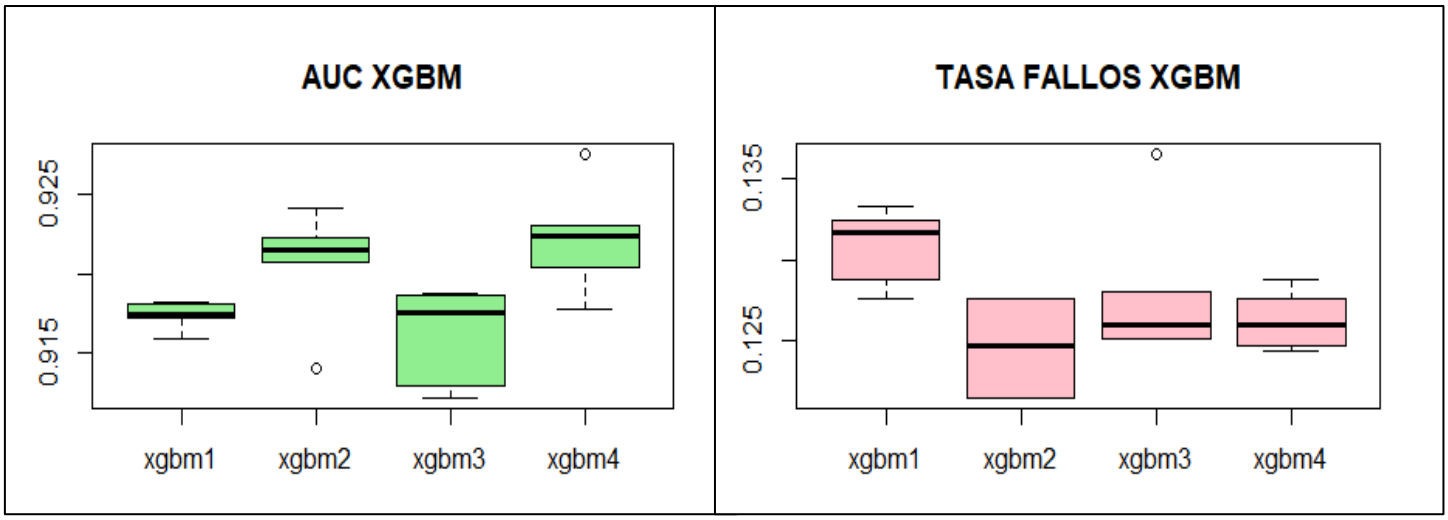


Figura 37: Área bajo la curva y Tasa de fallos para mejores modelos XGboost

Parece que, en general, a nivel de área bajo la curva, el eje Y que lo representa llega por encima de 0.92, algo que no ocurría con otros algortimos. Pese a ello parece que los modelos tienen mucha variablidad, ya que se aprecian dos puntos atípicos para el conjunto 2 por debajo de 0.915 y para el conjunto 4 por encima de 0.925, lo cual implica que el mejor modelo en términos de sesgo y variabilidad para el AUC sea el relativo al conjunto 1.

Por otro lado, en cuanto a la tasa de fallos, es el conjunto 2 quien tiene menor sesgo pero posee más variabilidad, por lo que parece que la mejor opción atendiendo a este criterio es el modelo obtenido por el conjunto 3.

De esta manera, el conjunto 3 sigue declarándose ganador para cada uno de los modelos.

5.8.5 Máquinas de Soporte Vectorial

En este caso, en primer lugar realizaremos el tuneado para SVM lineal y posteriormente para SVM polinómico y SVM RBF.

Como ya hemos indicado en el epígrafe de metodología, en este algoritmo podemos modificar el parámetro C para permitir un mayor o menor error.

Se prueba con diferentes valores para este parámetro y para cada uno de los 4 conjuntos de variables, para encontrar cuál es el mejor en términos de exactitud. En el caso del conjunto 1, que podemos observar en la Figura 38, el mejor C es el que posee un valor de 5.

```
> SVM_lin1$results
      C  Accuracy     Kappa
1  0.01 0.8378676 0.3295181
2  0.05 0.8500561 0.4130761
3  0.10 0.8529016 0.4289499
4  0.20 0.8553406 0.4395572
5  0.50 0.8565602 0.4471089
6  1.00 0.8569667 0.4481274
7  2.00 0.8569667 0.4481274
8  5.00 0.8573732 0.4491147
9 10.00 0.8569667 0.4469728
```

Figura 38: Resultados de exactitud probando diferentes parámetros C en conjunto 1

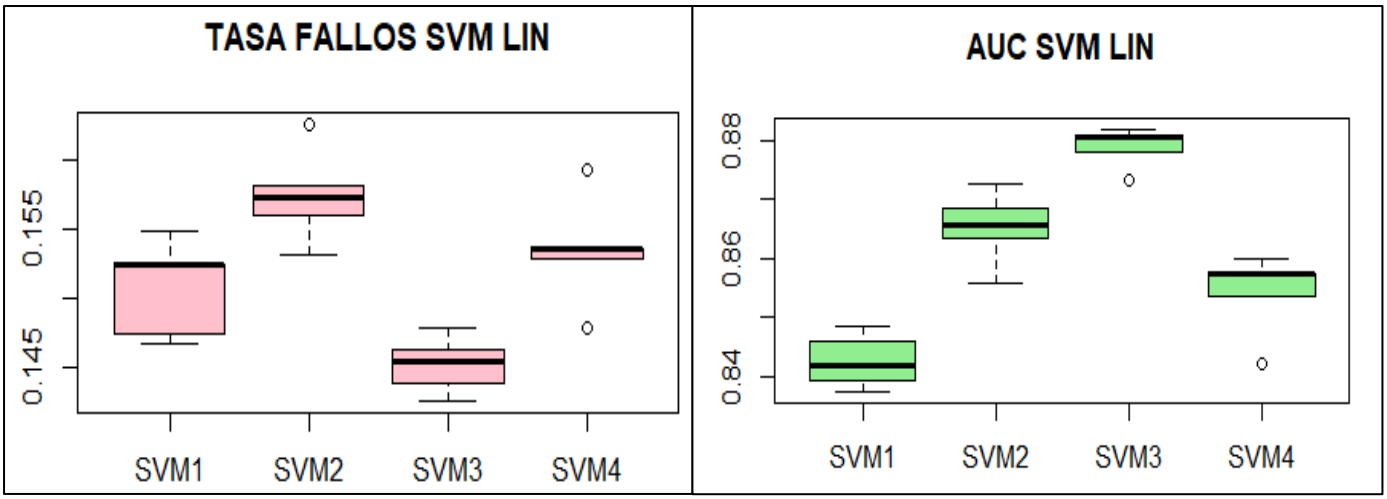


Figura 39: Tasa de Fallos y Área Bajo la Curva para SVM Lineal

Esto se ha realizado para cada uno de los conjuntos de variables para configurar la posterior validación cruzada empleando la función *cruzadaSVMbin* (Portela, 2019).

Si observamos la Figura 39, en el caso de aplicar máquinas de soporte vectorial lineales obtenemos como resultado peores áreas bajo la curva que en el caso de aplicar otros algoritmos y la tasa de fallos también es bastante superior, pese a ello sigue siendo mejor a no aplicar ningún modelo a nuestro conjunto de datos, ya que recordemos que la muestra desbalanceada era de 80-20, por lo que la tasa de fallos de decir que el resultado de la sentencia sería condenatorio es de 0.2, así que algo de ganancia se consigue aplicando este modelo. Pese a todo lo anterior, el mejor modelo atendiendo a ambos criterios (tasa de fallos y AUC) y observando su sesgo y varianza parece ser de nuevo el conjunto 3.

En el caso del Kernel polinómico, como ya expusimos en metodología, además del parámetro C se pueden modificar otros como el grado del polinomio (*degree).*

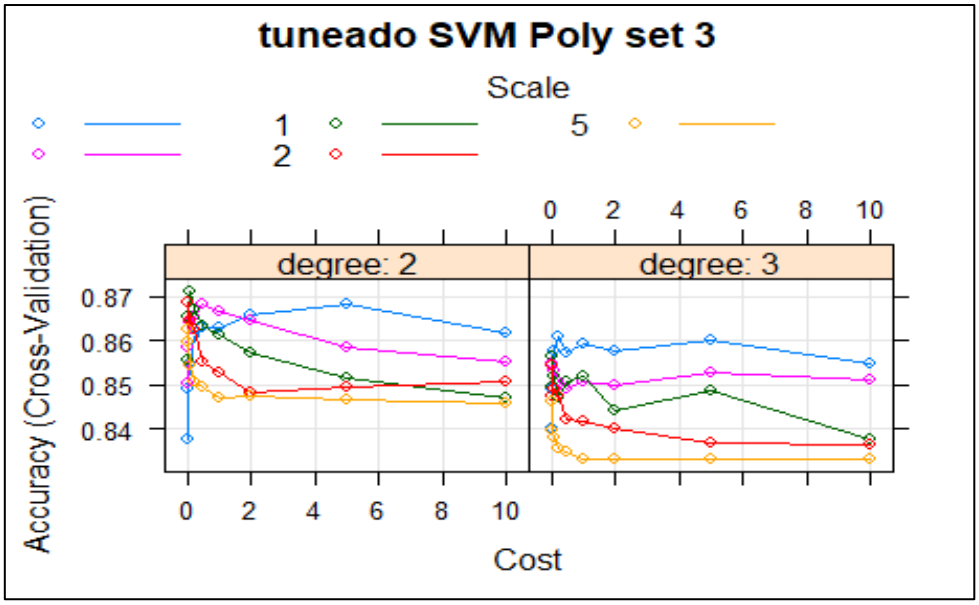


Figura 40: Tuneado SVM Kernel Polinomial – set 3

Al realizarlo, como podemos observar en la Figura 40, el mejor modelo para el conjunto 3 es aquel en el que el grado del polinomio es igual a 2, *scale* igual a 1 y el parámetro C (que en la gráfica viene definido como '*Cost*') vale 0.1, consiguiendo una exactitud de 0.8711917.

Realizamos lo mismo para el resto de los conjuntos, y atendemos a dichos resultados para modificar la función *cruzadaSVMbinPoly* (Portela, 2019) de cada conjunto atendiendo a los parámetros óptimos.

Los resultados son los que se pueden apreciar en la Figura 41, donde parece que la tasa de fallo indica que este modelo no es óptimo para nuestro conjunto de datos, ya que los errores son superiores si lo comparamos frente al no-modelo (cuya tasa de fallos sería 0.2). En el caso del área bajo la curva hay mejores resultados, pero no superiores a otros modelos ya analizados, volviendo a destacar por encima del resto el modelo obtenido con el conjunto de variables número 3.

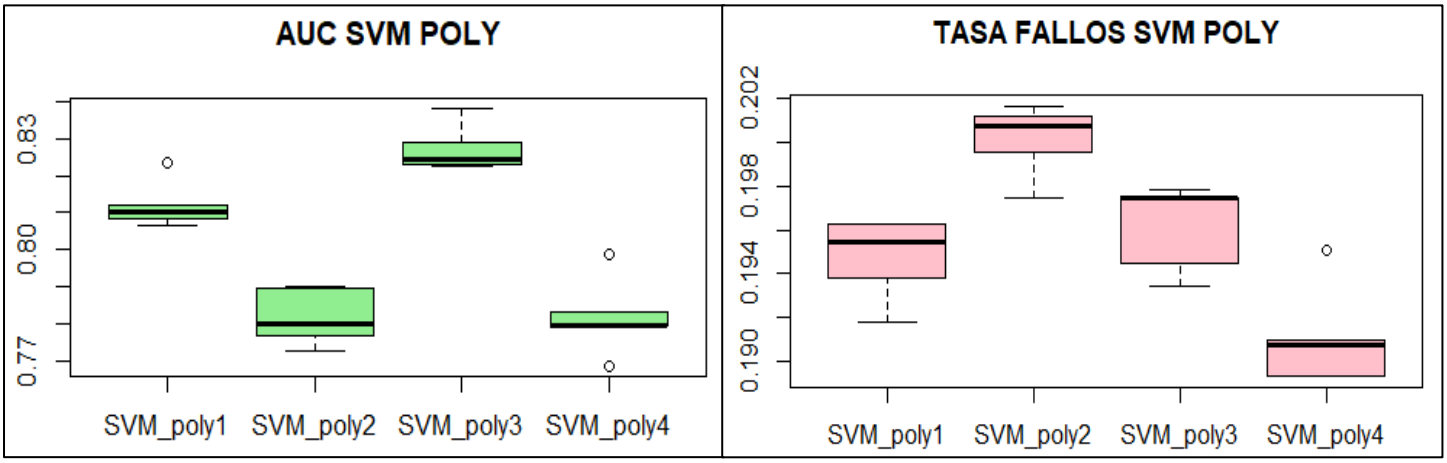


Figura 41: Resultados de AUC y Tasa de Fallos para SVM Polinomial

Si empleamos el kernel RBF, en el caso del conjunto número 1 el mejor modelo es aquel en el que el valor de σ es 0.1 y el parámetro C tiene un valor de 5.

Para el conjunto 2 el valor de σ también es de 0.1 pero el valor de C es igual a 2. Esto lo podemos observar gráficamente en la Figura 42:

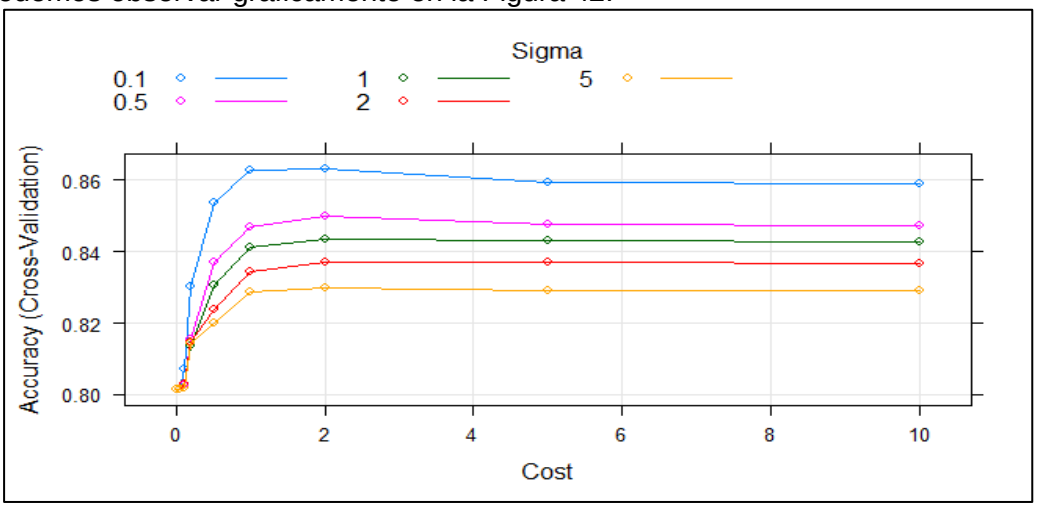


Figura 42: Tuneado de set 2 empleando kernel gausiano

Por el contrario, si atendemos al conjunto 3 debemos emplear σ = 0.1 y un C con un valor igual a 10.

El conjunto 4 requiere σ = 0.1 y un parámetro C igual a 10.

Si realizamos la validación cruzada repetida para cada uno de los conjuntos con los mejores parámetros para cada uno de ellos con la función *cruzadaSVMbinRBF* (Portela, 2019) obtenemos la Figura 45.

Tal y como podemos ver en la Figura 45, los resultados empleando el *kernel* gaussiano son mucho mejores que empleando cualquier otro *kernel.* Vemos que el área bajo la curva mejora significativamente, alcanzando niveles de 0.90 y 0.91. En el caso de la tasa de fallos también el error disminuye incluso a valores cercanos a 0.14. El peor conjunto es aquel en el que se incluyen todas las variables del modelo y el mejor conjunto en este caso es el segundo si atendemos al sesgo y variabilidad que podemos observar tanto para el área bajo la curva como para la tasa de fallos.

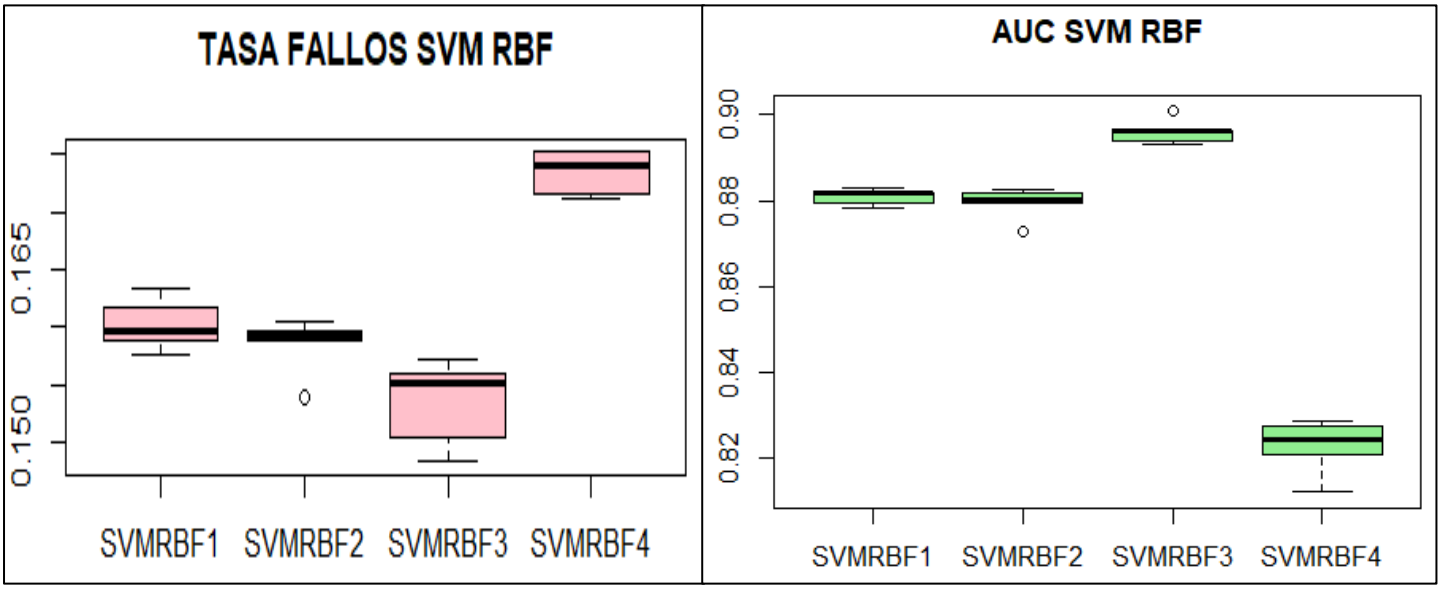


Figura 43: Resultados AUC y Tasa fallos aplicando SVM RBF

5.9 Comparación de todos los modelos

Tabla 4: Principales modelos e indicadores

Modelo	Set	AUC	Tasa Fallos
Redes Neuronales	1	0.9044	0.1259651
Redes Neuronales	2	0.9155	0.1206826
Redes Neuronales	3	0.9169	0.1214953
Redes Neuronales	4	0.9136	0.1324665
Random Forest	1	0.9151	0.1365299
Random Forest	2	0.9157	0.1316538
Random Forest	3	0.9195	0.1275904

Bagging	4	0.9136	0.1340918
Incremento gradiente	1	0.9069	0.1361235
Incremento gradiente	2	0.9055	0.1296221
Incremento gradiente	3	0.9108	0.1385616
Incremento gradiente	4	0.8986	0.1706623
Xgboost	1	0.9158	0.1316538
Xgboost	2	0.9215	0.1214953
Xgboost	3	0.9129	0.1279967
Xgboost	4	0.9231	0.1288094
SVM lineal	1	0.8419	0.1523771
SVM lineal	2	0.8655	0.1580658
SVM lineal	3	0.8819	0.1426249
SVM lineal	4	0.8806	0.145469
SVM polinomial	1	0.8085	0.195449
SVM polinomial	2	0.7727	0.201626
SVM polinomial	3	0.8225	0.197832
SVM polinomial	4	0.7797	0.1950427
SVM RBF	1	0.9045	0.1426249
SVM RBF	2	0.9171	0.1381552
SVM RBF	3	0.907	0.1328728
SVM RBF	4	0.8788	0.158472

En la Tabla 4 aparece un resumen con los mejores modelos para cada uno de los conjuntos de variables junto con sus estadísticos, algo que también podemos observar en la Figura 44 atendiendo al área bajo la curva y en la Figura 45 que especifica dicho valor únicamente para los que consideramos mejores modelos.

En este caso al atender al área bajo la curva, queremos que nuestros modelos aparezcan representados en la parte superior del gráfico, ya que es indicativo de mayor área. En el caso de las máquinas de soporte vectorial, únicamente aquellas en las que se ha aplicado el SVM Radial están por encima del 0.90, por lo que descartaríamos el resto. Esto se puede observar mejor en la Figura 45, en la cual observamos únicamente los mejores modelos, sin atender a que haya uno específico para cada conjunto y algoritmo.

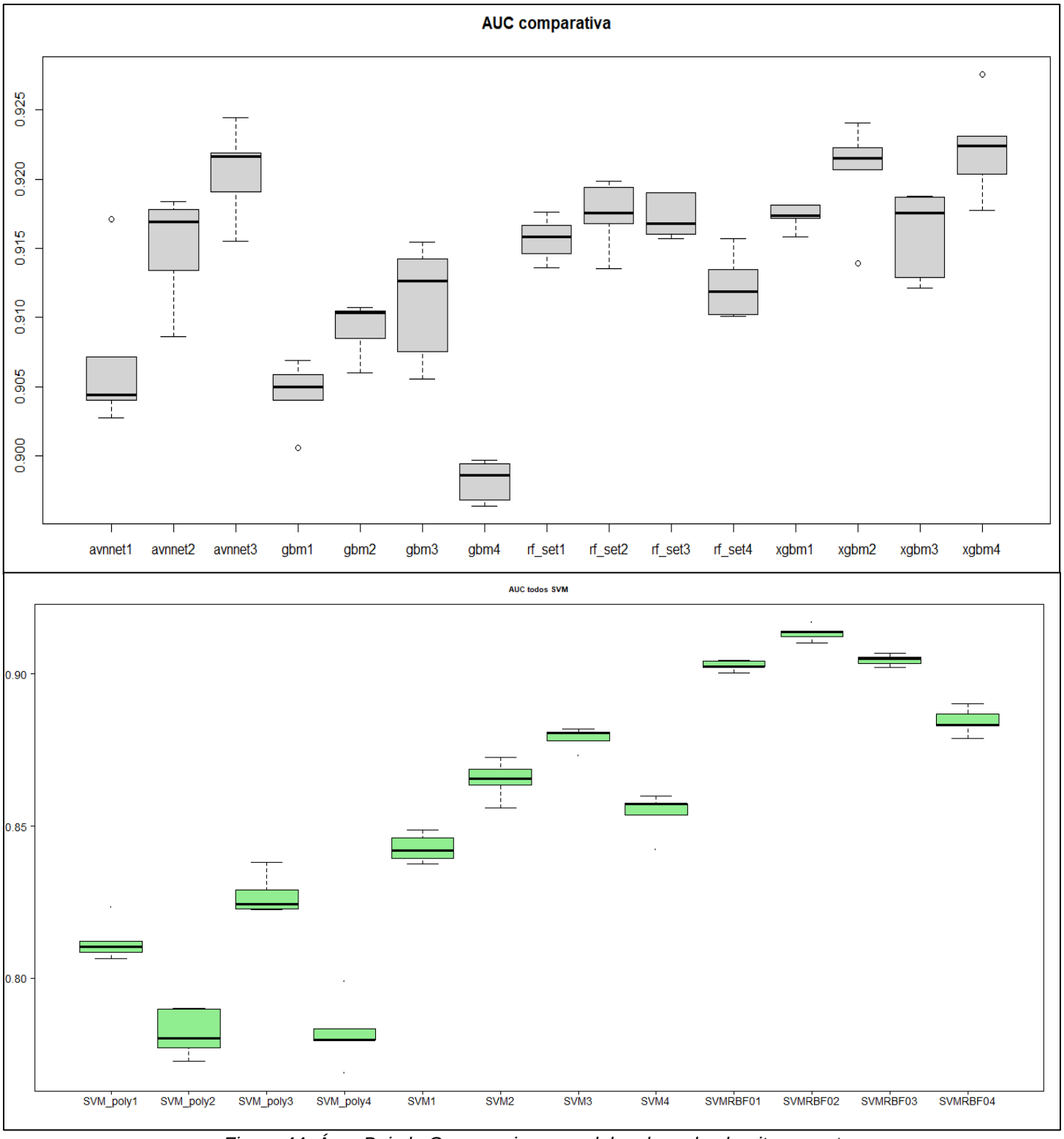


Figura 44: Área Bajo la Curva mejores modelos de cada algoritmo y set

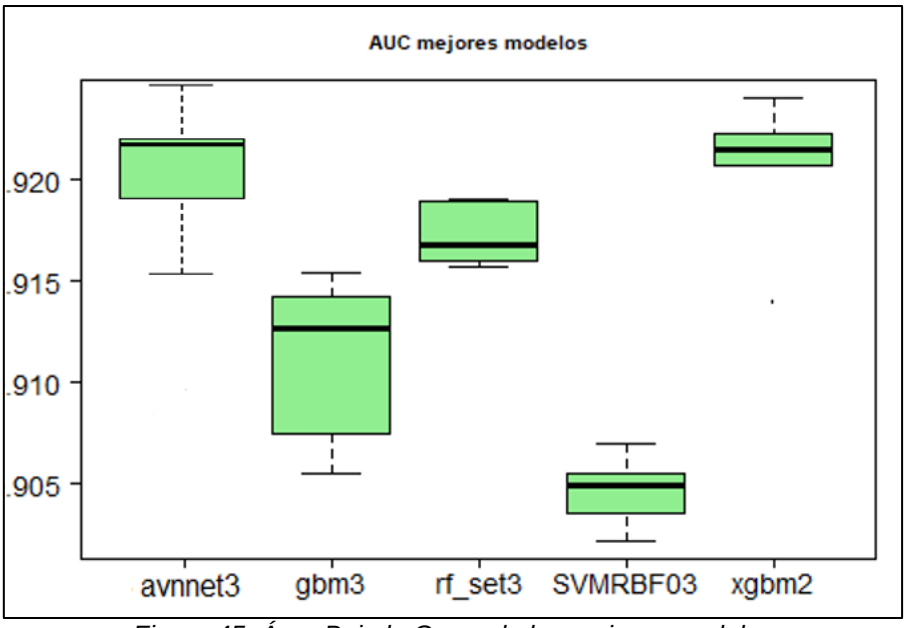


Figura 45: Área Bajo la Curva de los mejores modelos

Lo que parece evidente, es que el conjunto de datos número 3 (set3) es el conjunto que mejor funciona, aquel compuesto por 24 componentes principales y una variable categórica, el tribunal. Si además tenemos en cuenta el principio de parsimonia, parece el mejor conjunto para realizar nuestras predicciones.

Debemos elegir cuál de los demás modelos es el mejor para predecir el fallo de una sentencia. Para ello tenemos en cuenta en especial el área bajo la curva ROC, ya que tiene en cuenta todos los posibles puntos de corte y parece más fiable que emplear únicamente la tasa de fallos, que también se tiene en cuenta y se puede observar en la Tabla 4, así como la exactitud del modelo.

El modelo obtenido con *incremento gradiente* (gbm3) parece el más inestable si observamos la varianza del área bajo la curva. También el modelo realizado con redes neuronales (avnnet3) posee una gran variabilidad. A esto se suma el descarte del modelo SVM empleando kernel gaussiano (SVMRBF03) por el sesgo que posee y nos deja con los modelos *random forest* (rf_set3) y *XGboost* (xgbm2).

Si tenemos en cuenta ambas opciones de manera combinada, así como la complejidad del modelo, nos decantamos por el modelo rf_set3 como ganador, ya que el modelo xgbm2 tiene un punto atípico que se puede apreciar en negro en la imagen, en torno a .915, pese a su aparente mejor sesgo, sumado a que el número de variables es mucho mayor, 48 frente a 25.

5.10 Análisis del mejor modelo

Como acabamos de indicar el mejor modelo es aquel en el que se emplea *Random Forest* al conjunto de variables 3, que está compuesto por las variables PC7, PC15, PC27, PC54, PC35, PC44, PC52, PC43, PC42, PC17, PC62, PC12, PC4, PC1, PC168, PC45, PC40, PC5, PC13, PC105, PC63, PC188, PC187 y tribunal transformada en

dummy. Fue el resultado de aplicar la función *cruzadarfbin* al conjunto de datos 3 aplicando validación cruzada repetida y empleando un total de 200 árboles, un mínimo de nodos por parámetro de 10 y un *mtry* de 24.

La importancia de las variables para este conjunto aparece reflejada en la Figura 46, donde podemos ver que, con una diferencia muy amplia, las más relevantes son PC7, PC15 y PC42, como ya se había visto en la exploración inicial, siendo la variable categórica la variable que menos importancia tiene en el modelo.

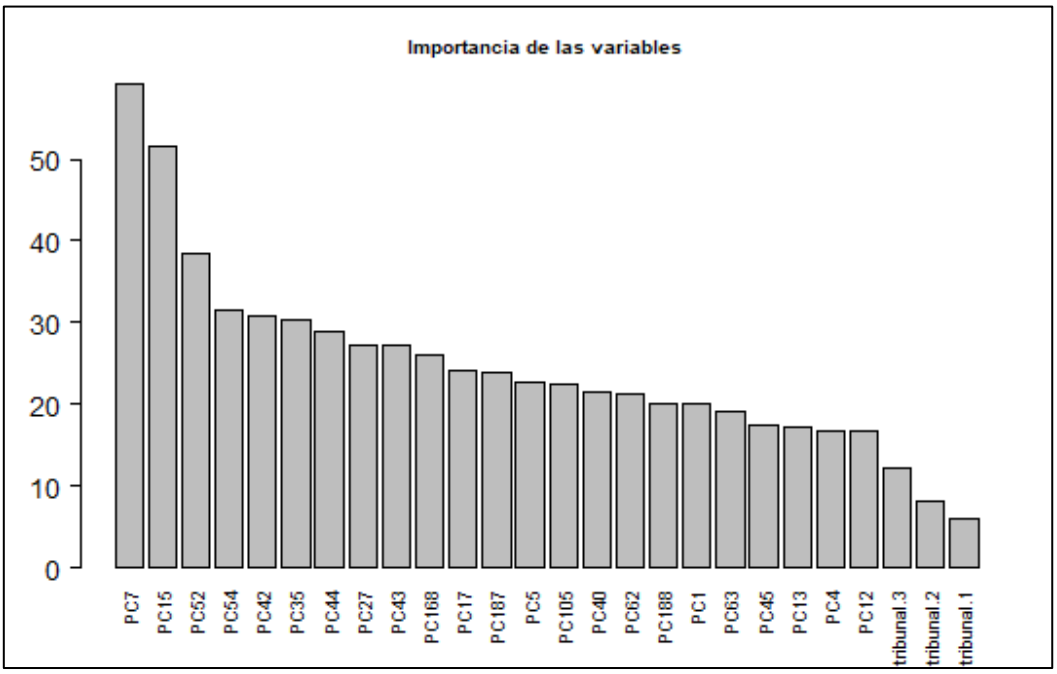


Figura 46: Importancia de las variables – Random Forest set 3

La variable PC7 ha resultado ser para todos los modelos y conjuntos de variables la más importante, por lo que puede resultar interesante recordar las diez primeras palabras que la componían, así como observar también para nuestro mejor modelo las diez primeras palabras de PC15 y PC52.

Tabla 5: Palabras más relevantes para las componentes 7, 15 y 52

CP7	**CP15**	**CP52**
homenaj	vend	semilla
recaudatori	tradicion	cannab
tenenci	automovil	asociacion
armas	compravent	compravent
port	refir	responsabiliz
metr	espir	preci
trabaj	extraterrit	vent
comun	oprim	indebid
gravesolicit	extravag	invers
peset	air	hierb

Como podemos observar en la Tabla 5, la componente 7 parece que se compone de palabras sobre diferentes temas, pero destacan la tenencia y porte de armas y puede que también el homicidio. También parece que se tratan de sentencias anteriores al año 2000 en su mayoría las que se explicarán por esta componente al aparecer la palabra "peset".

En el caso de la CP15 sí se ve que está más relacionado con robos o estafas de automóviles, al menos observando las 10 primeras palabras que lo componen. Lo mismo ocurre con la CP52, si atendemos a las palabras con las que contamos, es evidente que esta componente recoge elementos relacionados con la tenencia y tráfico de drogas.

De este modo parece que los delitos relacionados con armas, automóviles y drogas son los que pueden tener un mayor peso en el modelo.

Como ya vimos en el apartado de *Random Forest*, cuando se realiza un remuestreo con reemplazamiento, las observaciones que se quedan fuera del árbol y no son utilizadas para crear el mismo se predicen y se calcula su error. Se hace en cada iteración teniendo como resultado lo que se conoce como OOB error o *out of bag error.*

```
        OOB estimate of  error rate: 12.76%
Confusion matrix:
     No  Yes class.error
No  270  219  0.44785276
Yes  95 1877  0.04817444
```

Figura 47: Matriz de Confusión OOB error modelo rf_set 3

Podemos ver cómo el error medio es de 12.76%. Si observamos la matriz de confusión (Figura 47), podemos observar cómo el modelo predice muy bien los fallos condenatorios, algo que parece lógico dada la muestra desbalanceada con la que contamos, teniendo 80% de eventos condenatorios frente a tan solo 20% absolutorios. Esto provoca que los eventos de fallos absolutorios se predigan bastante peor, obteniendo un error del 44% aproximadamente, mientras que el fallo condenatorio posee un error aproximado de 4.8%.

Tabla 6: Estudio de la matriz de confusión del modelo ganador

	Predicción = No	**Predicción = Sí**	**Class.Error**
Realidad = No	270	219	0.44785276
Realidad = Sí	95	1877	0.04817444
Tasa de acierto	0.87240959		
Tasa de fallo	0.12759041		
Sensibilidad /Tasa de VP	0.951825558		
Especificidad/ Tasa de VN	0.552147239		
Valor predictivo positivo	0.895515267		
Valor predictivo negativo	0.739726027		

En la Tabla 6 se ha puesto de forma más clara la matriz de confusión obtenida en el *OOB error*, además de calcular algunos indicadores que pueden resultar relevantes para el análisis.

Así, observamos que contamos con un total de 270 verdaderos negativos (VN), 95 falsos negativos (FN), 1877 verdaderos positivos (VP) y 219 falsos positivos (FP).

Con estos datos calculamos los estadísticos que encontramos bajo la matriz de confusión, el error estimado en la salida de R es la tasa de fallo, y es aproximadamente un 12.7%. Por el contrario, la tasa de acierto es de un 87.3% aproximadamente.

La sensibilidad mide qué porcentaje de eventos está capturando el modelo. En este caso la sensibilidad del modelo es de un 95%, pero si observamos detenidamente vemos que es porque la mayoría de los eventos son condenatorios, que son los que predice bien. En el caso de la especificidad lo utilizamos para conocer la capacidad que tiene el modelo para capturar errores y su valor es de un 55%.

Al tener una muestra desbalanceada los errores de la clase minoritaria, en este caso, el fallo absolutorio, son más grandes. Esto se traduce en que la sensibilidad del modelo es alta y la especificidad es pequeña si la comparamos con la sensibilidad.

En el caso del valor predictivo positivo (VPP) se calcula dividiendo los verdaderos positivos entre todos los positivos reales del modelo. En este caso es un buen valor ya que predice casi el 90% de los positivos del modelo, por lo que este valor es fiable.

Por lo que se refiere al valor predictivo negativo (VPN) se calcula de la misma manera, dividiendo el verdadero negativo entre los negativos reales que hay en el modelo y el resultado no es tan bueno como para los positivos, algo que ya venimos observando a lo largo de todo el estudio y que se debe en parte a que contamos con pocos datos que cumplan con este evento.

La sensibilidad es muy alta pero la especificidad no, por lo que, resulta interesante cambiar el punto de corte para conseguir unos mejores resultados de especificidad aunque se reduzca el valor de la sensibilidad empleando para ello el índice de Youden[18].

En este caso no es posible buscar el punto de corte con los datos del error OOB, por lo que se hará con validación cruzada repetida para nuestro modelo ganador. Evidentemente, los resultados serán más optimistas pero, dado que lo que nos interesa es el mejor punto de corte, aunque los valores de especificidad y sensibilidad no tienen por qué ser exactamente esos - porque el modelo sea demasiado optimista-, el mejor punto de corte seguirá siendo el mismo.

Para ello aplicamos la función *sensEspCorte* (Calviño, A., 2018) al modelo ganador.

[18] El índice de Youden intenta equilibrar la sensibilidad y la especificidad. Lo útil sería maximizarlo, viniendo dado este índice por la sensibilidad + especificidad -1. El valor ideal para este índice sería uno (está en tanto por uno) y lo malo sería 0. Conforme movemos los datos de una columna a otra de la matriz tenemos sensibilidad de 1 y especificidad de 0 o al revés y al restar 1 lo mínimo que tendré es cero.

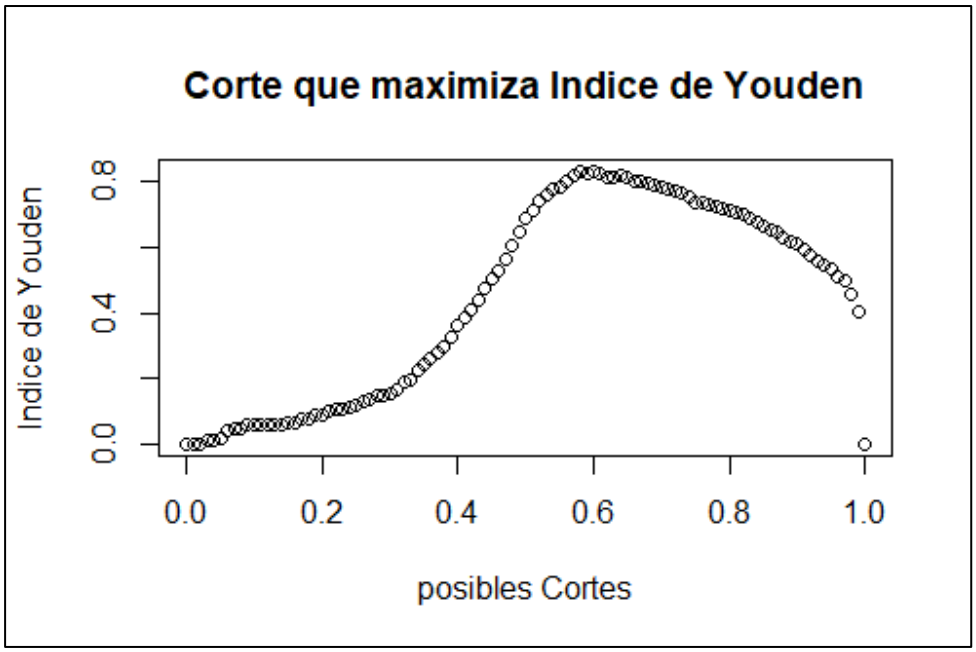


Figura 48: Punto de corte en función del Índice de Youden

Parece que el punto que maximiza el Índice de Youden está aproximadamente en 0.6 (Figura 48), en concreto con la función que acabamos de indicar sabemos que el valor exacto del punto de corte es de 0.58. Atendiendo a este punto de corte, aplicamos *sensEspCorte* para este valor concreto y obtenemos la especificidad y la sensibilidad.

```
> sensEspCorte(medias6,tfmbis,"sentido_fallo",0.58,"1")

Accuracy Sensitivity Specificity
  0.9459569   0.9655172   0.8670757
```

Como ya hemos indicado, son valores demasiado optimistas, pero lo relevante es que el punto de corte óptimo sí que es el resultante y nos sirve para poder realizar análisis futuros con mayor precisión.

6. Conclusiones y trabajo futuro

El objetivo principal de este Trabajo de Fin de Máster era tratar de predecir el resultado de una sentencia. Para ello tuvimos que crear nuestra propia base de datos, al no existir ninguna con la estructura necesaria para realizar dicho análisis. Por ello, en primer lugar, se debieron extraer los datos utilizando técnicas de *web scrapping* y recogiendo más variables de las que después serían utilizadas al comprobar la excesiva correlación entre algunas de ellas o la escasa utilidad de otras. Al tratarse de texto era necesario procesar el lenguaje natural, para lo cual se normalizó el texto eliminando signos de puntuación, diferencias entre mayúsculas y minúsculas y reduciendo las palabras a su raíz o base según el algoritmo empleado. Esto se realizó para el resumen de la sentencia, por lo que se descartó realizar el mismo paso con el fallo de la sentencia por obtener más información con el primero y porque dicho resumen contenía parte del fallo. Además, las variables voces procesales y sustantivas también tenían una gran correlación con el resumen que las hacían poco útiles para aportar información

adicional. Una vez realizada la normalización sobre el resumen, se aplicó la función TF*IDF, consistente en dar una puntuación a cada una de las palabras en función de su aparición, para descartar aquellas que aparecían en numerosas ocasiones y las otras que aparecían demasiado poco, quedándonos con una nueva tabla de palabras útiles a la que aplicaríamos un análisis de componentes principales para reducir su amplia dimensión. Una vez obtenidas las componentes que estimamos necesarias se completó la base de datos con otras variables que ya habíamos extraído o creado a base de otras extraídas y que creíamos relevantes para el estudio.

Con la base de datos depurada y apta para el análisis se realizó una pequeña exploración en la que no se encontró ningún error, por lo que se realizaron diferentes técnicas de *Machine Learning*, descartando el uso de la regresión logística al tratarse la mayoría de nuestras variables de componentes principales, las cuales no se podían interpretar, siendo esta la principal ventaja del análisis clásico. Por ello se optaron por el resto de vías a nuestra disposición y se obtuvo que el mejor modelo era aquel en el que se aplicaba *Random Forest* al set3 (compuesto por las variables PC7, PC15, PC27, PC54, PC35, PC44, PC52, PC43, PC42, PC17, PC62, PC12, PC4, PC1, PC168, PC45, PC40, PC5, PC13, PC105, PC63, PC188, PC187 y tribunal transformada en *dummy*). Para ello se tuvieron en cuenta como criterios la tasa de fallos, el principio de parsimonia y, en especial, el área bajo la curva ROC, debido a que nuestra muestra estaba desbalanceada y el AUC tenía en cuenta todos los posibles puntos de corte.

En ninguno de los conjuntos de variables a los que se aplicaron diferentes criterios de selección resultó relevante la variable género, por lo que parece que no es relevante que un juez sea hombre o mujer a la hora de enjuiciar un determinado delito. Lo mismo ocurrió con la variable “fecha_reforma”, que utilizamos para estudiar si eran relevantes las reformas que se habían realizado en el Código Penal a lo largo del tiempo, siendo también irrelevante para el estudio cuándo se dictó la sentencia para conseguir un resultado condenatorio o absolutorio.

Con nuestro mejor modelo se observaron cuáles eran las variables importantes y con la matriz de confusión el valor predictivo del modelo.

La conclusión es que, debido a la muestra desbalanceada, predecía muy bien los eventos de fallo condenatorio con un error del 5% aproximadamente, mientras que los fallos absolutorios no corrían la misma suerte, pues el error era de más del 40%. Pese a ello, parece mejor que realizar predicciones sin modelo alguno, ya que la tasa de error a nivel general está en torno al 12%. Por todo esto se intentó encontrar el punto de corte que maximizara el Índice de Youden y que pueda servir para futuros análisis sobre esta muestra tan dispar.

Como trabajo futuro sería interesante trasladar este tipo de análisis a otros órdenes jurisdiccionales, donde sea posible obtener muestras menos desbalanceadas y donde se pudiera filtrar por el lugar en el que se ha cometido el delito, algo que se intentó realizar en este trabajo, pero resultó imposible por no existir ningún identificador en la base de datos jurídica empleada que permitiera extraer dicho campo. Con muestras más equilibradas y variables como la localización para mejorar el análisis que se ha realizado en el trabajo se podría conocer cómo funciona la justicia en función del lugar en el que

se dicta, ya que salvo el Tribunal Superior de Justicia el resto de los tribunales pueden resolver de manera diferente un mismo caso. Con una base de datos jurídica accesible y limpia se podrían analizar casos que permitieran a las personas estudiar sus probabilidades de éxito y que ayudaran a decidir si optar por la vía judicial o por otros métodos de resolución de conflictos de ser éstos posibles.

En el caso del orden penal, son escasas las sentencias que encontramos para Juzgados de Instrucción y de lo Penal, por lo que de nuevo la clave es conseguir bases de datos más completas y accesibles para el mundo tecnológico. También el análisis que se realiza con el resultado de la sentencia podría trasladarse al análisis del éxito o fracaso de un abogado, si está disponible en la base de datos, o las tendencias de los jueces ante un determinado asunto. Es un campo en el que aún queda mucho por explorar.

7. Bibliografía

- Aletras, N., Tsarapatsanis, D., Preoţiuc-Pietro, D., Lampos, V., 2016. *Predicting judicial decisions of the European Court of Human Rights: a Natural Language Processing perspective*. San Diego. https://peerj.com/articles/cs-93/
- Alonso Revenga, J., 2018. *Apuntes sobre Complementos en Minería de Datos*, Facultad de Estudios Estadísticos, Universidad Complutensde de Madrid.
- Armonas Colombo, B., Buck, P., Miana Bezerra, V., 2017. *Challenges When Using Jurimetrics in Brazil—A Survey of Courts*. Future Internet 9, 68. https://www.mdpi.com/1999-5903/9/4/68
- Calviño Martínez, A., 2018. *Apuntes en Técnicas y Metodología de la Minería de Datos,* Facultad de Estudios Estadísticos, Universidad Complutensde de Madrid.
- *How to Scrape Web using Python, Selenium and Beautiful Soup* · Swetha's Blog, https://swethatanamala.github.io/2018/09/01/web-scraping-using-python-selenium-and-beautiful-soup/
- International Association for Artificial Intelligence and Law: *LeDAM 2018 - Workshop on Legal Data Analytics and Mining* http://www.iaail.org/?q=article/ledam-2018-workshop-legal-data-analytics-and-mining.
- Katz, D.M., Ii, M.J.B., Blackman, J., 2017. *A general approach for predicting the behavior of the Supreme Court of the United States*. PLOS ONE 12. https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0174698
- Kluwer, W., *Jurimetria* https://jurimetria.wolterskluwer.es/).
- Librería nltk: https://www.nltk.org/
- Librería sklearn: https://scikit-learn.org/stable/
- Librería Selenium: https://selenium-python.readthedocs.io/
- Librería SQLite: https://docs.python.org/2/library/sqlite3.html
- Librería Urllib: https://docs.python.org/2/library/urllib.html
- Loevinger, L., 1971. *Jurimetrics: the next step forward*. J. 12, 3–41.
- Martin, A.D., Quinn, K.M., Ruger, T.W., Kim, P.T., 2004. *Competing approaches to predicting supreme court decision making. Perspect. Polit.* 2, 761–767.
- Portela, J., 2019. *Apuntes sobre Técnicas de Machine Learning*, Facultad de Estudios Estadísticos, Universidad Complutensde de Madrid.
- The TF*IDF Algorithm Explained: https://www.onely.com/blog/what-is-tf-idf/

- Ulmer, S.S., 1963. *Quantitative Analysis of Judicial Processes: Some Practical and Theoretical Applications*. Probl. 28, 164. https://www.jstor.org/stable/1190728?origin=crossref&seq=1#page_scan_tab_contents

8. Anexos

8.1 Creación de la base de datos: SQL vs SQLite

En primer lugar, se creó una base de datos SQL en MySQL pero posteriormente se decidió cambiar dicha base de datos por una SQLite.

Para ello se tuvieron en cuenta las ventajas y desventajas que ambas proporcionaban. En el caso de MySQL, es un sistema de gestión de bases de datos relacionales de código abierto con un modelo cliente-servidor. Al ser un servidor, no se puede integrar directamente con su aplicación, a diferencia de SQLite, si no que requiere de instalación y conexión con la aplicación.

Las ventajas de MySQL son la seguridad que proporciona como sistema de administración de bases de datos, así como el soporte para procesamiento. Además, es muy fácil garantizar un buen rendimiento con MySQL, proporcionando también una escalabilidad inigualable para facilitar la administración de aplicaciones que manejan una gran cantidad de datos.

Alguna de sus desventajas es que, pese a ser muy útil para grandes cantidades de datos, pueden existir problemas de velocidad y rendimiento cuando se manejan tablas de datos muy pesadas. También hay que destacar que es un sistema dependiente de los recursos que el servidor tenga para proporcionar un correcto funcionamiento de nuestros sitios y aplicaciones.

En la otra cara de la moneda tenemos SQLite, el cual también es un sistema de gestión de bases de datos relacional, aunque muchos lo consideran como un fichero de datos por su portabilidad. En este caso, SQLite no se trata de un sistema cliente-servidor, si no que este sistema se enlaza con el programa pasando a ser parte integral del mismo, no tiene que instalarse en algún espacio ni conectar posteriormente con la aplicación. La base de datos se almacena en un único fichero y emplea registros de diferentes tamaños en función de sus necesidades, lo cual es muy útil ya que utiliza el espacio en disco que requiere en cada momento.

Como ventajas, si lo comparamos con MySQL, SQLite es un sistema ligero, no requiere instalación y el rendimiento es mayor, pues como ya hemos indicado, sólo carga los datos que necesita. Pero las principales ventajas que han hecho que SQLite sea elegido para realizar este proyecto es su portabilidad y su accesibilidad. Es posible guardar el fichero y posteriormente visualizarlo y gestionarlo a través de programas como *DB Browser*.

Como desventajas hay que destacar que no permite la concurrencia de conexiones (es mono usuario) lo que supone que, si un usuario está modificando datos, otro no podrá realizarlo hasta que éste termine. También hay que advertir que SQLite está enfocado para aplicaciones con tráfico medio o bajo y con un bajo tamaño de la base de datos.

En conclusión, MySQL es una buena opción al manejar gran cantidad de información o, si se trata de aplicaciones o páginas webs, aquellas con gran cantidad de tráfico, mientras que SQLite estaría enfocado a aplicaciones más sencillas y pequeñas con poco tráfico y/o a bases de datos ligeras.

En este caso, primero se optó por realizar el proyecto en MySQL pero se dio con la imposibilidad de exportar la base de datos, por lo que ante el temor de que pudiera surgir algún problema con el equipo utilizado se optó por emplear, dado que no es una base de datos muy pesada, SQLite para poder contar con una versión portátil del proyecto y accesible desde cualquier ordenador con DB Browser, y realizar las copias de seguridad oportunas sobre el mismo.

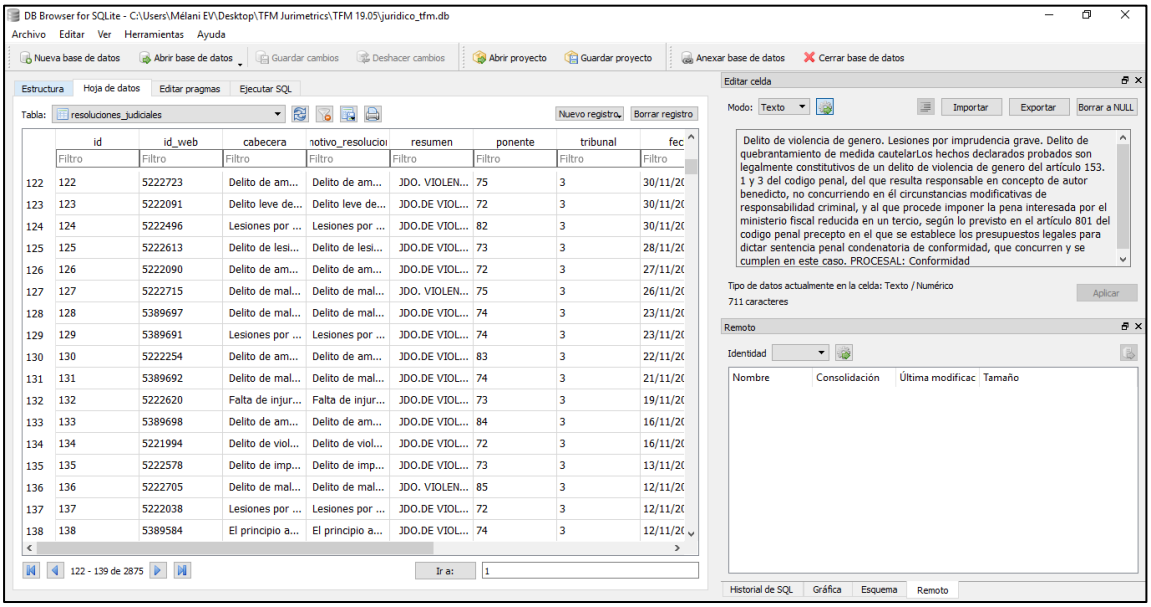


Figura 1: Ejemplo de visualización de base de datos en DB Browser

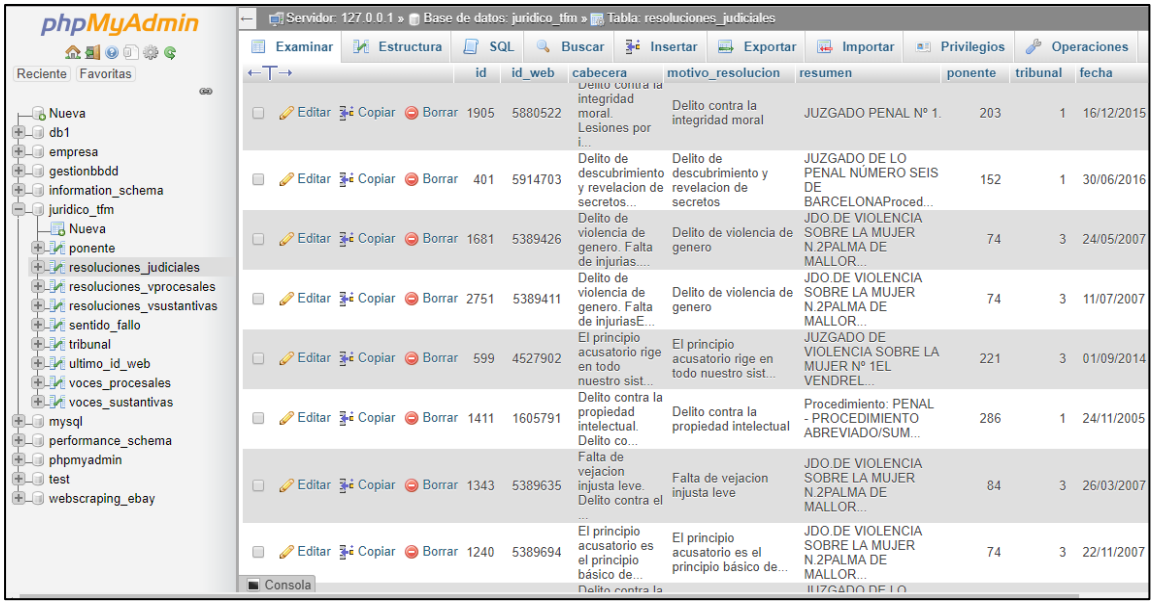


Figura 2: Ejemplo de visualización de base de datos en MySQL – php MyAdmin

DB Browser es una aplicación que, como ya se ha expuesto anteriormente, facilita la creación y administración de bases de datos con SQLite, con una interfaz muy sencilla de utilizar, similar a una hoja de cálculo, lo que permite un trabajo cómodo sobre las bases de datos. Además, como podemos observar en la Figura 1, posee controles y asistentes para realizar consultas e inspeccionar resultados, así como importar y exportar archivos en diferentes formatos

Vemos así cómo parece sencillo de usar pese a tener diferencias con MySQL, el cual permite realizar las mismas funciones, pero con las diferencias ya mencionadas y un aspecto diferente (Figura 2).

8.2 Scripts Python

8.2.1Creación de SQLite vacío

```
1. import sqlite3
2.
3. conn = sqlite3.connect('juridico_tfm.db')
4.
5. c = conn.cursor()
6.
7. with open('database_vacia.sql') as fp:
8.     lines = fp.readlines()
9.
10.        for l in lines:
11.            c.execute(l)
12.
13.    conn.commit()
14.    conn.close()
```

8.2.2 Extracción de datos e importación de la base de datos

```
1. import re
2. import urllib3
3. import time
4. from bs4 import BeautifulSoup
5. import pymysql
6. import re
7. import unicodedata
8. import sqlite3
9. import sys
10.    #connection = pymysql.connect(host='localhost',
11.    #                             user='root',
12.    #                             password='',
13.    #                             db='juridico_tfm',
14.    #                             charset='utf8mb4',
15.    #                             cursorclass=pymysql.cursors.DictCu
   rsor)
16.    #
17.
18.    #conexion a la base de datos
19.    connection = sqlite3.connect('juridico_tfm.db')
20.    cur = connection.cursor()
21.
22.    urllib3.disable_warnings()
23.    http = urllib3.PoolManager()
24.    r = http.request('GET', 'https://www.tirantonline.com/tol/login.
   do?user=complutense&password=madrid', redirect=False)
25.    headers = {'cookie': r.getheader('set-cookie')}
26.    #Jurisprudecnia -> Civil -> Buscar
27.
28.    juzgados = ['Juzgado de Instrucción', 'Juzgado de Violencia
   sobre la Mujer', 'Juzgado de Primera Instancia e
   Instrucción', 'Juzgado de lo Penal']
29.
30.    with open('lista_ids.txt') as f:
31.            lineas = f.readlines()
```

```
32.           for id in lineas:
33.                   id_web = id.replace('\n', '')
34.
35.                   try:
36.                           r = http.request('GET', "http://www.tira
   ntonline.com/tol/documento/show/" + str(id_web), headers=headers)
37.                           pet1 = "http://www.tirantonline.com/tol/
   documento/show/" + str(id_web)
38.
39.                           html = r.data
40.                           soup = BeautifulSoup(html, 'html.parser'
   )
41.
42.                           try:
43.
44.                                   content = soup.find("div", {"id"
   : "jurisdiccion"}).getText()
45.                                   jurisdiccion = content.replace('
   Jurisdicción:', '').replace('\t', '').replace('\n', '').replace('"'
   , "")
46.
47.                                   content = soup.find("div", {"id"
   : "tiporesolucion"}).getText()
48.                                   tipo_resolucion = content.replac
   e('Tipo resolución:
   ', '').replace('\t', '').replace('\n', '') .replace('"', "")
49.
50.                                   if jurisdiccion == 'Penal' and '
   Sentencia' in tipo_resolucion:
51.                                           content = soup.find("div
   ", {"id": "origen"}).getText()
52.                                           origen = content.replace
   ('Origen:', '').replace('\t', '').replace('\n', '').replace('"', ""
   )
53.                                           if origen in juzgados:
54.                                                   print('---Nueva
   resolucion con id: ' + str(id_web))
55.
56.                                                   content = soup.f
   ind("div", {"id": "parrafo_1"}).getText()
57.                                                   resumen = conten
   t.replace('\t', '').replace('\n', ' ').replace('  ', '
   ').replace('"', "")
58.                                                   resumen = resume
   n.replace('[siguiente]', '').replace('[anterior]', '').replace('[Co
   ntextualizar]', '')
59.
60.                                                   content = soup.f
   ind("div", {"id": "cabecera"}).getText()
61.                                                   cabecera = conte
   nt.replace('Cabecera:', '').replace('\t', '').replace('\n', '
   ').replace('  ', ' ').replace('"', "")
62.
63.                                                   motivo_resolucio
   n = cabecera.split('.')
64.                                                   motivo_resolucio
   n = motivo_resolucion[0]
65.
66.                                                   content = soup.f
   ind("div", {"id": "ponentenombre"}).getText()
```

```
67.                                                     nombre = content
   .replace('Ponente:', '').replace('\t', '').replace('\n', '').replac
   e('"', "")
68.
69.                                                     content = soup.f
   ind("div", {"id": "origen"}).getText()
70.                                                     tribunal = conte
   nt.replace('Origen:', '').replace('\t', '').replace('\n', '').repla
   ce('"', "")
71.
72.                                                     content = soup.f
   ind("div", {"id": "fecha"}).getText()
73.                                                     fecha = content.
   replace('Fecha:', '').replace('\t', '').replace('\n', '').replace('
   "', "")
74.
75.                                                     #obtenemos las
   voces procesales y sustantivas
76.                                                     content = soup.f
   ind("div", {"class": "docEsquema"})
77.                                                     divs = content.f
   ind_all('div')
78.
79.                                                     for div in divs:
80.                                                             if 'sust
   antivas' in str(div):
81.
   voces_sustantivas = div.getText().replace('Voces sustantivas:', '')
82.
   voces_sustantivas = voces_sustantivas.split(',')
83.                                                             if 'proc
   esales' in str(div):
84.
   voces_procesales = div.getText().replace('Voces procesales:', '')
85.
   voces_procesales = voces_procesales.split(',')
86.
87.
88.                                                     #sentido del
   fallo
89.                                                     content = soup.f
   ind("div", {"id": "themine_document_detail_sentido-del-fallo_mb"})
90.                                                     ul = content.fin
   d('ul')
91.                                                     li = ul.find('li
   ')
92.                                                     sentido_fallo =
   li.getText().replace('\n', '')
93.
94.
95.                                                     todas_las_divs =
    soup.find_all('div')
96.                                                     for div in todas
   _las_divs:
97.                                                             if 'FALL
   O' in str(div):
98.
   fallo = div.getText()
99.
   fallo = fallo.replace('[siguiente]', ' ').replace('[anterior]', '
   ').replace('[Contextualizar]', ' ')
```

```
100.
   fallo = fallo.replace('\t', ' ').replace('\n', ' ')
101.
   fallo2 = ""
102.
   for c in fallo:
103.
            if c >= 'A' or c == ' ':
104.
                    fallo2 += c
105.
   #fallo = div.getText().replace('\t', ' ').replace('\n', '
   ').replace('"', "")
106.
   for x in range(1,10):
107.
            fallo2 = fallo2.replace('  ', ' ')
108.                                                    fallo = fallo2
109.                                                    #**************
   *******
110.                                                    #BASE DE DATOS--
   -------
111.                                                    #**************
   *******
112.
113.                                                    #prmero,
   buscamos si el ponente ya esta en la base de datos
114.                                                    sql = "SELECT *
   FROM ponente WHERE ponente = '" + nombre + "'"
115.                                                    cur.execute(sql)
116.                                                    busqueda = cur.f
   etchone()
117.
118.                                                    if busqueda:
119.                                                            #si
   esta, aumentamos el contador en 1
120.                                                            id_ponen
   te = busqueda[0]
121.                                                            sql = "U
   PDATE ponente SET contador = " + str(busqueda[3] + 1) + " WHERE id
   = " + str(id_ponente)
122.                                                            cur.exec
   ute(sql)
123.                                                    else:
124.                                                            #si no
   esta, hay que anadirlo
125.                                                            sql = "I
   NSERT INTO `ponente`(`id`, `ponente`, `contador`) VALUES (NULL,'" +
   nombre + "',0)"
126.                                                            cur.exec
   ute(sql)
127.
128.                                                            #Una vez
   lo hemos anadido a la BD, obtenemos su ID
129.                                                            sql = "S
   ELECT id FROM ponente ORDER BY id DESC LIMIT 1"
130.                                                            cur.exec
   ute(sql)
131.                                                            busqueda
    = cur.fetchone()
```

```
132.                                                    id_ponen
   te = busqueda[0]
133.
134.                                                #--------------
   ------------------------------------------
135.                                                #ahora, hacemos
   lo mismo con el tribunal
136.                                                sql = "SELECT *
   FROM tribunal WHERE tribunal = '" + tribunal + "'"
137.                                                cur.execute(sql)
138.                                                busqueda = cur.f
   etchone()
139.
140.                                                if busqueda:
141.                                                        #si
   esta, aumentamos el contador en 1
142.                                                        id_tribu
   nal = busqueda[0]
143.                                                        sql = "U
   PDATE tribunal SET contador = " + str(busqueda[2] + 1) + " WHERE id
   = " + str(id_tribunal)
144.                                                        cur.exec
   ute(sql)
145.                                                else:
146.                                                        #si no
   esta, hay que anadirlo
147.                                                        sql = "I
   NSERT INTO `tribunal`(`id`, `tribunal`, `contador`) VALUES
   (NULL,'" + tribunal + "',0)"
148.                                                        cur.exec
   ute(sql)
149.
150.                                                        #Una vez
   lo hemos anadido a la BD, obtenemos su ID
151.                                                        sql = "S
   ELECT id FROM tribunal ORDER BY id DESC LIMIT 1"
152.                                                        cur.exec
   ute(sql)
153.                                                        busqueda
    = cur.fetchone()
154.                                                        id_tribu
   nal = busqueda[0]
155.
156.                                                #--------------
   ------------------------------------------
157.
158.                                                #ahora, hacemos
   lo mismo con el tribunal
159.                                                sql = "SELECT *
   FROM sentido_fallo WHERE sentido_fallo = '" + sentido_fallo + "'"
160.                                                cur.execute(sql)
161.                                                busqueda = cur.f
   etchone()
162.
163.                                                if busqueda:
164.                                                        #si
   esta, aumentamos el contador en 1
165.                                                        id_senti
   do_fallo = busqueda[0]
```

```
166.                                                    sql = "U
    PDATE sentido_fallo SET contador = " + str(busqueda[2] + 1) + "
    WHERE id = " + str(id_sentido_fallo)
167.                                                    cur.exec
    ute(sql)
168.                                                else:
169.                                                    #si no
    esta, hay que anadirlo
170.                                                    sql = "I
    NSERT INTO `sentido_fallo`(`id`, `sentido_fallo`, `contador`)
    VALUES (NULL,'" + sentido_fallo + "',0)"
171.                                                    cur.exec
    ute(sql)
172.
173.                                                    #Una vez
    lo hemos anadido a la BD, obtenemos su ID
174.                                                    sql = "S
    ELECT id FROM sentido_fallo ORDER BY id DESC LIMIT 1"
175.                                                    cur.exec
    ute(sql)
176.                                                    busqueda
     = cur.fetchone()
177.                                                    id_senti
    do_fallo = busqueda[0]
178.
179.                                                #---------------
    -------------------------------------------
180.
181.                                                #anadimos la
    linea con los datos generales
182.                                                sql = "INSERT
    INTO `resoluciones_judiciales`(`id`, `id_web`, `cabecera`,
    `motivo_resolucion`, `juzgado`, `ponente`, `tribunal`, `fecha`,
    `sentido_fallo`, `fallo`) VALUES (NULL, "+ str(int(id_web)) +", '"+
    cabecera +"', '"+ motivo_resolucion +"', '"+ resumen +"',
    "+ str(int(id_ponente)) +", "+ str(int(id_tribunal)) +", '"+ fecha
    +"', "+ str(int(id_sentido_fallo)) +", '"+ fallo +"')"
183.                                                cur.execute(sql)
184.
185.                                                #---------------
    -------------------------------------------
186.
187.
188.                                                sql = "SELECT id
    FROM resoluciones_judiciales ORDER BY id DESC LIMIT 1"
189.                                                cur.execute(sql)
190.                                                busqueda = cur.f
    etchone()
191.                                                id_resolucion_bd
     = busqueda[0]
192.
193.
194.                                                #VOCES
    SUSTANTIVAS -------------
195.                                                for v in voces_s
    ustantivas:
196.                                                    voces_su
    stantivas = v
197.
```

```
198.                                                                #Ahora,
   vamos a comprobar cada una de las voces con el mismo proceso que
   anteriormente
199.                                                                sql = "S
   ELECT * FROM voces_sustantivas WHERE voces_sustantivas = '" +
   voces_sustantivas + "'"
200.                                                                cur.exec
   ute(sql)
201.                                                                busqueda
    = cur.fetchone()
202.
203.                                                                if busqu
   eda:
204.
   #si esta, aumentamos el contador en 1
205.
   id_voz_sustantiva = busqueda[0]
206.
   sql = "UPDATE voces_sustantivas SET contador =
   " + str(busqueda[2] + 1) + " WHERE id = " + str(id_voz_sustantiva)
207.
   cur.execute(sql)
208.
209.
   #ahora, lo tenemos que anadir a la tabla resoluciones_vsusantivas
210.
   sql = "INSERT INTO
   `resoluciones_vsustantivas`(`id_resoluciones_judiciales`,
   `id_voces_sustantivas`) VALUES
   (" + str(id_resolucion_bd) + "," + str(id_voz_sustantiva) + ")"
211.
   cur.execute(sql)
212.
213.                                                                else:
214.
   #si no esta, hay que anadirlo
215.
   sql = "INSERT INTO `voces_sustantivas`(`id`, `voces_sustantivas`,
   `contador`) VALUES (NULL,'" + voces_sustantivas + "',0)"
216.
   cur.execute(sql)
217.
218.
   #Una vez lo hemos anadido a la BD, obtenemos su ID
219.
   sql = "SELECT id FROM voces_sustantivas ORDER BY id DESC LIMIT 1"
220.
   cur.execute(sql)
221.
   busqueda = cur.fetchone()
222.
   id_voz_sustantiva = busqueda[0]
223.
224.
   #ahora, lo tenemos que anadir a la tabla resoluciones_vsusantivas
225.
   sql = "INSERT INTO
   `resoluciones_vsustantivas`(`id_resoluciones_judiciales`,
   `id_voces_sustantivas`) VALUES
   (" + str(id_resolucion_bd) + "," + str(id_voz_sustantiva) + ")"
```

```
226.
    cur.execute(sql)
227.
228.
229.                                                    #VOCES
    PROCESALES -------------
230.                                                    for v in voces_p
    rocesales:
231.                                                            voces_pr
    ocesales = v
232.
233.                                                            #Ahora,
    vamos a comprobar cada una de las voces con el mismo proceso que
    anteriormente
234.                                                            sql = "S
    ELECT * FROM voces_procesales WHERE voces_procesales = '" +
    voces_procesales + "'"
235.                                                            cur.exec
    ute(sql)
236.                                                            busqueda
     = cur.fetchone()
237.
238.                                                            if busqu
    eda:
239.
    #si esta, aumentamos el contador en 1
240.
    id_voz_procesal = busqueda[0]
241.
    sql = "UPDATE voces_procesales SET contador =
    " + str(busqueda[2] + 1) + " WHERE id = " + str(id_voz_procesal)
242.
    cur.execute(sql)
243.
244.
    #ahora, lo tenemos que anadir a la tabla resoluciones_vsusantivas
245.
    sql = "INSERT INTO
    `resoluciones_vprocesales`(`id_resoluciones_judiciales`,
    `id_voces_procesales`) VALUES
    (" + str(id_resolucion_bd) + "," + str(id_voz_procesal) + ")"
246.
    cur.execute(sql)
247.
248.                                                            else:
249.
    #si no esta, hay que anadirlo
250.
    sql = "INSERT INTO `voces_procesales`(`id`, `voces_procesales`,
    `contador`) VALUES (NULL,'" + voces_procesales + "',0)"
251.
    cur.execute(sql)
252.
253.
    #Una vez lo hemos anadido a la BD, obtenemos su ID
254.
    sql = "SELECT id FROM voces_procesales ORDER BY id DESC LIMIT 1"
255.
    cur.execute(sql)
256.
    busqueda = cur.fetchone()
```

```
257.
   id_voz_procesal = busqueda[0]
258.
259.
   #ahora, lo tenemos que anadir a la tabla resoluciones_vsusantivas
260.
   sql = "INSERT INTO
   `resoluciones_vprocesales`(`id_resoluciones_judiciales`,
   `id_voces_procesales`) VALUES
   (" + str(id_resolucion_bd) + "," + str(id_voz_procesal) + ")"
261.
   cur.execute(sql)
262.
263.                                                    print('+++Insert
   ada resolucion con id: ' + str(id_web))
264.                            except Exception as e:
265.                                    print('1' + str(e))
266.                    except Exception as e:
267.                            print('2' + str(e))
268.
269.                    sql = "UPDATE `ultimo_id_web` SET
   `id_web`=" + str(id_web)
270.                    cur.execute(sql)
271.                    connection.commit()
272.
273.                    print('.', end="", flush=True)
```

8.2.3 Creación de la variable “fecha_reforma”

```
1. import sqlite3
2. import pymysql
3.
4. #connection = pymysql.connect(host='localhost',
5. #                             user='root',
6. #                             password='',
7. #                             db='juridico_tfm',
8. #                             charset='utf8mb4',
9. #                             cursorclass=pymysql.cursors.DictCurso
   r)
10.   #
11.
12.
13.   connection = sqlite3.connect('juridico_tfm.db')
14.   cur = connection.cursor()
15.
16.   sql = "SELECT id, fecha FROM resoluciones_judiciales"
17.   cur.execute(sql)
18.   busqueda = cur.fetchall()
19.
20.
21.   for resolucion in busqueda:
22.           id = resolucion[0]
23.           fecha = resolucion[1]
24.
25.           mes = fecha[3:5]
26.           anio = fecha[6:11]
27.
28.           if anio < '2010' or (anio == '2010' and mes < '5'):
29.                   revision = 1
30.
31.           elif (anio < '2011') or (anio == '2011' and mes < '3'):
```

```
                    revision = 2

            elif (anio < '2015') or (anio == '2015' and mes < '4'):
                    revision = 3
            else:
                    revision = 4

            sql = "UPDATE resoluciones_judiciales SET fecha_reforma
   = '" + str(revision) + "' WHERE id = " + str(id)
            cur.execute(sql)

    connection.commit()
```

8.2.4 Creación de la variable “genero_juez”

8.2.4.1 Lista nombres_mujer

nombres_mujer = ['MARIA CARMEN','MARIA','CARMEN','JOSEFA','ANA MARIA','ISABEL','MARIA PILAR','MARIA DOLORES','LAURA','MARIA TERESA','ANA','CRISTINA','MARIA ANGELES','MARTA','FRANCISCA','ANTONIA','MARIA ISABEL','MARIA JOSE','DOLORES','LUCIA','SARA','PAULA','MARIA LUISA','ELENA','PILAR','CONCEPCION','RAQUEL','ROSA MARIA','MANUELA','MERCEDES','MARIA JESUS','ROSARIO','BEATRIZ','JUANA','TERESA','JULIA','NURIA','SILVIA','ENCARNACION','IRENE','ALBA','PATRICIA','MONTSERRAT','ANDREA','ROSA','ROCIO','MONICA','MARIA MAR','ALICIA','ANGELA','SONIA','SANDRA','MARINA','SUSANA','YOLANDA','NATALIA','MARGARITA','MARIA JOSEFA','MARIA ROSARIO','EVA','INMACULADA','CLAUDIA','MARIA MERCEDES','ANA ISABEL','ESTHER','NOELIA','CARLA','VERONICA','SOFIA','ANGELES','CAROLINA','NEREA','MARIA VICTORIA','MARIA ROSA','EVA MARIA','AMPARO','MIRIAM','LORENA','INES','MARIA CONCEPCION','ANA BELEN','MARIA ELENA','VICTORIA','MARIA ANTONIA','DANIELA','CATALINA','CONSUELO','LIDIA','MARIA NIEVES','CELIA','ALEJANDRA','OLGA','EMILIA','GLORIA','LUISA','AINHOA','AURORA','MARIA SOLEDAD','MARTINA','FATIMA','MARIA CRISTINA','ESPERANZA','CLARA','MILAGROS','JOSEFINA','VIRGINIA','VANESA','MARIA LUZ','ANNA','PURIFICACION','LOURDES','ADRIANA','BLANCA','BEGOÑA','MARIA BELEN','ISABEL MARIA','ESTEFANIA','ELISA','MARIA BEGOÑA','MAGDALENA','MARIA ASUNCION','GEMA','MARIA PAZ','ARACELI','BELEN','MATILDE','ASUNCION','MARIA LOURDES','VICENTA','MARIA ESTHER','REMEDIOS','TAMARA','ELVIRA','TRINIDAD','LAIA','SOLEDAD','NATIVIDAD','PALOMA','GEMMA','REBECA','VANESSA','ALMUDENA','MIREIA','MARIA CRUZ','EMMA','ASCENSION','MARIA INMACULADA','VALERIA','NOEMI','ARIADNA','MARIA AMPARO','FELISA','NIEVES','RAFAELA','TANIA','MARIA EUGENIA','ADELA','NOA','AITANA','CARLOTA','JESSICA','DIANA','AMALIA','AMELIA','JUANA MARIA','VALENTINA','LEIRE','MARIA ROCIO','CARMEN MARIA','RAMONA','AINARA','LEONOR','JOAQUINA','ELISABET','LARA','LETICIA','BARBARA','MARIANA','GUADALUPE','CECILIA','PETRA','JUDITH','JULIANA','ESTER','MARIA MAGDALENA','AROA','AGUSTINA','ROSALIA','MARIA ENCARNACION','SHEILA','CANDELA','JENNIFER','JUDIT','GABRIELA','BERTA','ESTRELLA','MARIA FRANCISCA','ADORACION','ALEXANDRA','SORAYA','HELENA','RUTH','MARIA MONTSERRAT','MARIA YOLANDA','MARIA GLORIA','MARIA MILAGROS','MARIA CONSUELO','NORA','NADIA','ERIKA','NAIARA','ELSA','AIDA','LOLA','EUGENIA','JIMENA','PAOLA','EULALIA','HERMINIA','MARIA

REMEDIOS','ESTELA','MAR','AINA','ELISABETH','MACARENA','MARIA REYES','ENRIQUETA','YAIZA','TATIANA','LEYRE','MARIA ARANZAZU','AMAIA','AFRICA','SARAY','MARIA TRINIDAD','PIEDAD','ARANZAZU','MARIA ESPERANZA','ITZIAR','AMANDA','FLORENTINA','GREGORIA','JOANA','OLIVIA','MAITE','MIRIAN','AURELIA','IRIA','CANDIDA','IRIS','KHADIJA','ANE','ANA ROSA','MERITXELL','ANTONIA MARIA','MERCE','DESIREE','SAGRARIO','RITA','CARIDAD','MARIA SOL','NEUS','MARIA JULIA','MARIA FERNANDA','MARIA INES','AGUEDA','VIOLETA','ELIZABETH','LYDIA','MARIA MANUELA','ROSER','ALBA MARIA','VERA','ESMERALDA','CONSOLACION','ADELINA','AMAYA','MARIA GRACIA','MARIA ASCENSION','TOMASA','FELICIDAD','MARIA CANDELARIA','ABRIL','TEODORA','ANA CRISTINA','BENITA','CARME','MARIA DOLORS','HORTENSIA','MARIA PURIFICACION','ELIA','FILOMENA','ELOISA','CANDELARIA','NAYARA','ADELAIDA','GISELA','IRINA','MARIA CARME','VEGA','SALMA','MARIA SAGRARIO','ALMA','ANNA MARIA','IRATI','AZUCENA','ANGELA MARIA','ANABEL','MARIA AURORA','JENIFER','MARCELINA','JANA','ELENA MARIA','NAZARET','FLORA','MARIA PINO','ESTIBALIZ','ZAIRA','MICAELA','MARIA BLANCA','YASMINA','MONSERRAT','EDURNE','AICHA','ALEXIA','VISITACION','FERNANDA','MARIA AUXILIADORA','NAIMA','LUNA','ROSA ANA','LUISA MARIA','REGINA','MARIA PALOMA','DELIA','FUENSANTA','AYA','MARTA MARIA','MARIA VALLE','PAULINA','MARIA GUADALUPE','MARIA LUCIA','MARIA SOCORRO','SAMIRA','INMACULADA CONCEPCION','MARIAM','MARIONA','ARANTXA','AINOA','MARIA GEMA','AVELINA','LAURA MARIA','YANIRA','MARIA ESTRELLA','VIRTUDES','MARIA DESAMPARADOS','FLORENCIA','FABIOLA','JESICA','SALVADORA','MARA','MARIA SONIA','ANGELICA','MARISOL','CORAL','MARIA ELISA','DELFINA','INES MARIA','SEBASTIANA','LUZ MARIA','DOLORS','GENOVEVA','INGRID','MARIA ANGELS','MARIA OLGA','MIREYA','IDOIA','MARIA CINTA','ANGELINA','MARIA VICENTA','NAHIA','PATROCINIO','FELIPA','JUSTA','SABINA','NAGORE','MARIA NATIVIDAD','NAIA','ALFONSA','AZAHARA','MARIA ANGELA','LILIANA','IRATXE','AUREA','MAIDER','AMINA','JESUSA','MALIKA','IZASKUN','ROSANA','ISIDORA','LORENZA','PALMIRA','MARIA BEATRIZ','CARMEN ROSA','ZOE','FATIHA','MARIA ARACELI','OBDULIA','MARIA ANGUSTIAS','MARIA FATIMA','REYES','DESAMPARADOS','DEBORA','GRACIA','MAIALEN','GEORGINA','TRIANA','MARIA ALEJANDRA','BRIGIDA','MIHAELA','JUNE','TERESA JESUS','ONA','MARIA JUANA','JACINTA','EVANGELINA','DIONISIA','MARIA PIEDAD','DAVINIA','RACHIDA','SONIA MARIA','UXUE','ANASTASIA','NAROA','MODESTA','ANTIA','KARIMA','BIENVENIDA','ANAIS','ANGUSTIAS','URSULA','RUFINA','JOSEFA MARIA','CAYETANA','MARIA NURIA','OLGA MARIA','LINA','CLOTILDE','MARIA AZUCENA','OTILIA','SOCORRO','LAILA','YASMIN','GERTRUDIS','SARAI','EMILIANA','MALAK','DOMINGA','BALBINA','ERICA','MYRIAM','CRISTINA MARIA','MARIA RAQUEL','SAIDA','MARIA CONSOLACION','FRANCESCA','MARIBEL','EIDER','MARIA ELVIRA','GLORIA MARIA','MARCELA','MINERVA','CHLOE','NICOLE','COVADONGA','HANANE','MARIA LAURA','MARIA EVA','MELANIA','UXIA','BOUCHRA','MIREN','MARIA PRADO','MARIA SALUD','FELICIANA','SILVIA MARIA','MARIA ALICIA','OLATZ','ILUMINADA','SAMARA','MARIA ALMUDENA','IGNACIA','EUSEBIA','SANDRA MARIA','FELICITAS','MAITANE','LUZ MARINA','SARA MARIA','CINTIA','GUILLERMINA','MARIA MERCE','LATIFA','JULIA MARIA','CAMILA','GEMA MARIA','VICTORINA','ENGRACIA','LIA','INOCENCIA','VICTORIA EUGENIA','MARIA FE','PRESENTACION','MARIA CELIA','SALUD','RUT','NICOLASA','PAULA MARIA','ICIAR','FRANCISCA MARIA','ALINA','OLIVA','MARIA EMILIA','ANA CARMEN','ELISA

ISABEL','XENIA','FAUSTINA','SAIOA','MARIA MARGARITA','LORETO','MIA','HAIZEA','CLARA ISABEL','FERMINA','SATURNINA','SOUAD','SACRAMENTO','CELESTINA','SERAFINA','SABRINA','PASTORA','GRACIELA','AMAL','THAIS','PAZ','MARIA FELISA','OIHANE','NAYRA','HAJAR','ZAIDA','SARAH','CONCEPCIO','HAFIDA','MARIA SANDRA','LUZ','BIBIANA','SALOME','BASILISA','MARAVILLAS','NAJAT','PASCUALA','EVELYN','JACQUELINE','MARIA SUSANA','MARIA ADELA','MARIA VANESA','MARIA EULALIA','MARWA','DESIRE','IRUNE','IKRAM','BENEDICTA','DINA','CASILDA','JANIRE','CLARA MARIA','ZOHRA','IZARO','FATIMA ZAHRA','GALA','NICOLETA','SABELA','IMAN','ARIANA','ADA','MARIA CECILIA','SIMONA','ZAHRA','JERONIMA','CYNTHIA','ELISENDA','MARIA ESTER','OLAYA','CLEMENTINA','LUCIANA','SIHAM','NINA','CELSA','IVET','KAREN','SIRA','IRYNA','AITZIBER','ISABELLA','MARIA LORENA','HAYAT','IOANA','YESICA','OFELIA','HENAR','DAMARIS','JULIETA','JAMILA','ROSA ISABEL','MARIA VIRGINIA','ENCARNACIO','MARIA AMOR','ANUNCIACION','IMANE','YURENA','GARAZI','MARIA CAMINO','MARIA CLARA','MARIA ESTELA','TERESA MARIA','DUNIA','VICTORIANA','NEKANE','NELIDA','MARIA OLIVA','OLALLA','BLANCA MARIA','MARIYA','BERNARDA','DULCE MARIA','SVETLANA','MINA','FATNA','ELOINA','MELANIE','HANAN','SUSANA MARIA','CARMELA','JONE','NATALIYA','OKSANA','MARIA GEMMA','FLORINDA','ALICIA MARIA','HALIMA','ANA TERESA','BELINDA','PAMELA','MELISA','ARANCHA','SONSOLES','ELISA MARIA','MARIA AGUSTINA','LEILA','NAIRA','FATIMA ZOHRA','BRENDA','MARIA VISITACION','LUISA FERNANDA','MARIA GABRIELA','MARIA CATALINA','ANNE','ARANTZA','MARIA ANGELICA','PRIMITIVA','LUCRECIA','MARIA LEONOR','ISOLINA','MARIA LORETO','MARIA AMELIA','JARA','YESSICA','CARMEN DOLORES','OLENA','MARIA VIRTUDES','ESTHER MARIA','MAXIMA','MARIA ADORACION','ABIGAIL','MARIA LLANOS','RAHMA','FIDELA','MICHELLE','JUSTINA','CLOE','MAXIMINA','MARIA SALOME','MARIA O','ANA LUCIA','RAQUEL MARIA','DIANA MARIA','CARMEN DELIA','ARANTZAZU','RITA MARIA','CATERINA','CRISTOBALINA','ARLET','ANA LUISA','MARIA IRENE','MADDI','MARIA SANTOS','NAOMI','OLAIA','ELADIA','BRUNA','KARINA','EDELMIRA','MELISSA','MARIA SIERRA','MALENA','BASILIA','ALAZNE','ANA PILAR','ENARA','MARIA FLOR','PATRICIA MARIA','MARIA COVADONGA','MARIA FUENSANTA','VIORICA','ISIDRA','IRAIA','AMIRA','APOLONIA','MARIA MONSERRAT','BIANCA','DANIELLA','SVITLANA','LOUBNA','MIGUELA','PRISCILA','FELICIA','HABIBA','ARGENTINA','ANA VICTORIA','ANGELITA','BERNARDINA','DOLORES MARIA','MARIA ANA','LUCILA','MARIELA','MARIA ALBA','MARIA PAULA','ROSARIO MARIA','OLIMPIA','NANCY','PRUDENCIA','DOROTEA','MILAGROSA','MAYA','SELENE','MARIA VANESSA','MARIA LLUISA','YLENIA','NARCISA','MARIA RITA','MARIA CARIDAD','QUERALT','ZURIÑE','HIBA','LIBERTAD','TEOFILA','KATIA','LUZ DIVINA','MENCIA','MARIA CELESTE','AZIZA','MIRANDA','MARIA HENAR','MARIA MATILDE','FARAH','MARYAM','LOREA','AMADA','IDAIRA','ILHAM','YAMINA','EDUARDA','SEGUNDA','CARMEN PILAR','ITXASO','LIDIA MARIA','LUCIA MARIA','EMERITA','MARISA','RODICA','MARIA NEUS','NURIA MARIA','ANGELS','TETYANA','BENIGNA','DORINDA','ZINEB','MARIAN','ROXANA','IMMACULADA','MERIEM','MARIA VERONICA','GINA','MARGARIDA','SELENA','GEORGETA','MARIA CAROLINA','DAMIANA','EKATERINA','BLANCA NIEVES','SANAE','RESURRECCION','MARIA NOELIA','MIRELLA','IRMA','MARIA SONSOLES','ZORAIDA','DARIA','ERNESTINA','MARIA CANDELAS','ROSALINA','ANITA','MIRELA','MARIA JOAQUINA','MARGALIDA','ROSAURA','MARIA

SORAYA','OIHANA','MAIA','SAMANTHA','ISAURA','ANA BEATRIZ','SAADIA','CATALINA MARIA','MARIOLA','NOURA','ANA VANESA','CELESTE','MARIA AFRICA','IONELA','ASMAE','ROMINA','SANTIAGA','DEBORAH','JAIONE','MARIA RAMONA','MIRYAM','CELIA MARIA','KAOUTAR','MARIA ENGRACIA','DIGNA','LUZDIVINA','RANIA','MARLENE','PAULA ANDREA','MONICA MARIA','CARINA','PETRONILA','CINTA','NORMA','ALBINA','PLACIDA','RABIA','GEMMA MARIA','GLADYS','ANA DOLORES','EDUVIGIS','GUMERSINDA','ENCARNACION MARIA','IRENE MARIA','ANA PAULA','BLANCA ESTHER','ELIANA','MARIA ESMERALDA','CONSTANZA','JANET','EUFEMIA','MARTHA','MARIA JOSEFINA','HILARIA','VALERIANA','YARA','ANA JOSEFA','FE','GENEROSA','ANGELINES','IVANA','YOUSRA','DAFNE','CLAUDIA MARIA','DIVINA','MARIA CORO','HOUDA','CASIMIRA','AGNES','MARIA AMALIA','INDIRA','GRETA','ROSA ELENA','MISERICORDIA','FARIDA','CONSTANTINA','ANTONINA','ALDARA','GINESA','SILVANA','AINTZANE','EULOGIA','ENEDINA','MARIA ANUNCIACION','ROGELIA','HAKIMA','MAYRA','AISHA','ESPERANZA MACARENA','MARIA JULIANA','MARIA MARTA','DANA','ENERITZ','FLOR MARIA','RAIMUNDA','CRESCENCIA','YOLANDA MARIA','MARIA EMMA','MARGARITA MARIA','BEATRIZ MARIA','CHAIMAE','MARIA ROSER','CLAUDIA PATRICIA','NABILA','ALAITZ','JULITA','PAOLA ANDREA','DOMITILA','KARLA','ILEANA','FOUZIA','IDOYA','YOANA','YASMINE','ARIANE','CORNELIA','CAMELIA','JOSUNE','ALESSANDRA','MARIA ICIAR','MONIKA','ARIANNA','BLASA','CIPRIANA','GALINA','DALIA','MARIA MACARENA','SANDRA MILENA','CARLA MARIA','MIREN EDURNE','SEVERINA','ANDREA MARIA','NEREIDA','DORA','ANDREEA','GARBIÑE','OLVIDO','NOUR','ERUNDINA','PURA','BLANCA ROSA','OLHA','MARIA MILAGRO','CORINA','MERYEM','THALIA','ZULEMA','MILENA','MALEN','MARIA MILAGROSA','ALEJANDRA MARIA','NOELIA MARIA','ASTRID','ARRATE','MARTHA CECILIA','MARIA LIDIA','SUSANNA','LUCINDA','DIANA CAROLINA','JOVITA','ELBA','SAMANTA','MARIA PATRICIA','HIND','LIDIA ESTHER','AINOHA','MARIA FELICIDAD','CANDELAS','LORENA MARIA','MELANI','CUSTODIA','MARIA PETRA','SOUMIA','CLAUDINA','LOURDES MARIA','SHEYLA','JOANNA','ANA JESUS','JOSEPA','CESAREA','IULIANA','MARIA SILVIA','FLORICA','VIVIANA','LENUTA','SAMIA','FATOUMATA','MARIA SOFIA','ROSA DELIA','AMARA','ANGELICA MARIA','CATHERINE','LEOCADIA','BENILDE','HALYNA','ALMA MARIA','AURORA MARIA','LUA','TOURIA','MARTA ISABEL','DOMINICA','MARIA CORAL','SANDRA PATRICIA','AGRIPINA','CARMEN GLORIA','ETELVINA','MARIA ANDREA','MARIA TRANSITO','ANA PATRICIA','SOUKAINA','CLEMENCIA','LAYLA','KRISTINA','LIBRADA','FADMA','ADOLFINA','EDITH','HANNA','AMERICA','HOURIA','NATALIA MARIA','CELINA','DAYANA','XIANA','NAOUAL','MARIA CRISTO','MANAL','MARIA ENRIQUETA','CORAIMA','SARA ISABEL','AQUILINA','CARMEN ROCIO','MARIA DELIA','DEMETRIA','MARIA SARA','ZAHIRA','SANTA','EVELIA','MARIA ARANTZAZU','HILDA','ALEJANDRINA','MARIA GORETTI','SALIMA','ESTEL','TANIA MARIA','ASSIA','GHIZLANE','HAFSA','RUTH MARIA','AMADORA','GRACIA MARIA','MARIA ALEXANDRA','MARIA LETICIA','PILAR MARIA','GARA','GIULIA','KATHERINE','FABIANA','PINO','CEFERINA','CHIARA','MARIA PATROCINIO','JOHANNA','SEILA','MARIA FLORA','VALLE','PENELOPE','ASMAA','YULIA','DOUNIA','SOL','DORINA','LARISA','MARIA ELOISA','ANA MERCEDES','GUACIMARA','ADRIANA

MARIA','ELEUTERIA','GRISELDA','IBTISSAM','BASMA','MARIA PUERTO','ELSA MARIA','DENISE','LINDA','MARIA CABEZA','LINA MARIA','LIUDMILA','MARIA HORTENSIA','ANA CAROLINA','KIARA','SANAA','LYUDMYLA','VASILICA','MARIA FLORENCIA','SYLVIA','ANA LAURA','AGATA','MARIA VEGA','XIOMARA','ENMA','MARIA PRESENTACION','MARIA NAZARET','RENATA','MARIA CAMILA','BELLA','STEFANIA','CHRISTINE','IRAIDE','ELENE','ARES','ANA ROCIO','RICARDA','MARIA JESUSA','MARINELA','MARTA ELENA','ASMA','ISABEL CRISTINA','MIREN JOSUNE','HERMELINDA','MARIA HELENA','FLORINA','IVETTE','SABAH','MARIA OLVIDO','LIVIA','MARCIANA','MARIA AMAYA','LUMINITA','CRUZ','FADOUA','FELICITA','MARIA IGNACIA','SAFAE','IRANTZU','WIAM','ANA DELIA','MARIA PUY','ADELA MARIA','ANA ESTHER','FLAVIA','HELIODORA','LEANDRA','ALEGRIA','MARIA VIOLETA','HASNA','JUANA ANTONIA','KAWTAR','MARIA MONTAÑA','FATMA','LILIA','MARITZA','GIOVANNA','MARIA MONICA','NELLY','CARMINA','DULCENOMBRE','JOANA MARIA','LLUISA','ISABELA','CARMEN TERESA','OUMAIMA','SUSAN','MIREN NEKANE','MARIA ELSA','ALBERTA','ELIZABET','ARIADNE','JORDINA','ANIA','MARIA ADELAIDA','MARIEM','ITSASO','MARIA FLORENTINA','YULIYA','BONIFACIA','EMILY','MARIA CASTILLO','ERMITAS','LEA','ROSA CARMEN','JUANA ROSA','MARIA NATALIA','SILVINA','LAUREANA','EPIFANIA','NADIYA','MANAR','ANA VANESSA','MARIANELA','CARMEN NIEVES','MARIA IZASKUN','YADIRA','JUANITA','AHLAM','REMEI','FATOU','AINHARA','INNA','AURA','LENA','MARI A ARGENTINA','LEONARDA','NADINE','FUENCISLA','AMOR','MARIA HERMINIA','CASANDRA','STEPHANIE','JACOBA','MARIA DESIREE','ANA SOFIA','STELLA','CELEDONIA','NATHALIE','RIM','NEMESIA','LAURA ISABEL','YUMARA','ANGELES MARIA','MAGALI','MARIAMA','FATIMA MARIA','MARIA FUENCISLA','MARIA ESTIBALIZ','ALICE','CLARA EUGENIA','MARIA SEÑOR','PEREGRINA','LEONIDES','NOOR','MARIA ASSUMPCIO','NIEVES MARIA','ANTOLINA','DALILA','LILIAN','MARIIA','MARTHA LUCIA','REGLA','MARIA ESTEFANIA','MARIA JUNCAL','MARIA ELIZABETH','DOSINDA','TETIANA','ESTRELLA MARIA','YESENIA','AGURTZANE','CINTHIA','AMINATA','ELIDA','DOINA','MIREN KARMELE','ALLA','CRISTINA ISABEL','INAS','MELODY','MARY','JUNCAL','SALIHA','INDIA','EVARISTA','HELEN','AUXILIADORA',' MARIA AGUILA','JESSICA MARIA','MARIA ADELINA','LIBE','NAZIHA','VANESA MARIA','MARIOARA','VIRGINIA MARIA','JANIRA','NATALIIA','EVELIN','ALAIA','KATERYNA','MAYTE','ANTONELLA','AINHIZE','DIAN A MARCELA','FULGENCIA','NURIA ESTHER','SOPHIE','GEORGIANA','NOAH','TELMA','VALENTYNA','COLOMA','DULCE NOMBRE','CARMEN ISABEL','HIPOLITA','NAZARETH','MELANY','MONTSE','ALEXANDRA MARIA','MARIA NELIDA','MILAGRO','MABEL','MARIA BERTA','MAREN','MARIA REGINA','SHAILA','ANISOARA','LYUBOV','RKIA','TALIA','JOHANA','LUZ ADRIANA','EDITA','LUDIVINA','PEPITA','GABINA','GUIOMAR','LARA MARIA','OXANA','VICTORIA MARIA','LAUDELINA','ANA JULIA','DORIS','VERONIKA','GALYNA','NEZHA','MARIA AURELIA','JOAQUIMA','MARGARET','PRAXEDES','MIMOUNT','ASHLEY','INDARA','VICENTA MARIA','NAWAL','FANNY','ROSALBA','ATENEA','JAQUELINE','NAHIKARI','SANA','HIGINIA','VERO NICA MARIA','CARMEN ELENA','DOLORES CARMEN','GLORIA ESTHER','MOUNIA','LUCIA CARMEN','SOFIA MARIA','OMAIMA','MARIA ANTONIETA','YVONNE','ANA GLORIA','CATHAYSA','FELICISIMA','SOPHIA','MARIA BENITA','QUITERIA','ALEKSANDRA','PROVIDENCIA','VIKTORIYA','CAMINO','EMILIA MARIA','IXONE','ARMINDA','AROHA','CARMEN

LUISA','ESCOLASTICA','LLANOS','LOIDA','SAFAA','ANA JOSE','JOSEFA ANTONIA','LLUNA','ANA ELENA','UDANE','JUANA TERESA','ASIA','ESTELA MARIA','GALLA','TERESITA','AIDA MARIA','YAMNA','LAURENTINA','MARIA DELFINA','MIREN ITZIAR','PAQUITA','VANESSA MARIA','TRANSITO','CLARISA','DUNA','ANA MARGARITA','DANAE','MAURA','ZAINAB','CONCHA','MARIA REGLA','MARIA RAFAELA','MARIA ISOLINA','ROBERTA','GADEA','CAROLINE','ANE MIREN','ROCIO CARMEN','OLGA LUCIA','DIANA PATRICIA','ZENAIDA','FATIMA EZZAHRA','HELGA','CHANTAL','ATIKA','CARMEN BELEN','HASNAE','HONORINA','SALOUA','AIXA','CAROLA','BELEN MARIA','ISABEL CARMEN','KARMELE','MARIA NOEMI','SAFIA','CORA','NAILA','ROSA ELVIRA','BRIGITTE','JADE','ZIORTZA','ADELIA','JULENE','ALIA','WENDY','ZAKIA','ECATERINA','MA RIA PEÑA','YAMILA','MARIA LIDON','AGAPITA','REBECCA','ANAHI','SOUMAYA','JEZABEL','ELVA','CARMEN LUCIA','ELENA ISABEL','ANA CECILIA','HERMENEGILDA','JENNY','LUANA','MELINA','CONCEPCION MARIA','CONSOL','MERCEDES MARIA','CHAYMAE','JOSEFA CARMEN','ELEONORA','FEDERICA','MARIA ASSUMPTA','MARIA OFELIA','MARIA FILOMENA','ZOILA','SETEFILLA','LEONCIA','HUMILDAD','LISA','XIMENA','KIRA','NAYELI','SHEILA MARIA','IVONNE','URBANA','JUANA ISABEL','ANA GABRIELA','MAGDA','ILDEFONSA','FLOAREA','ZARA','LOREDANA','LIUBOV','MIMOUNA','NADA',' NATASHA','SEVERIANA','FRANCINA','HARIDIAN','DOUAA','NATACHA','NIDIA','HADDA','LINA MARCELA','CAROL','OUAFAE','MARIA GREGORIA','HANAE','NORAH','ROCIO PILAR','YAIZA MARIA','DULCE','EMMA MARIA','NICULINA','JANNAT','AIORA','JUANA ANA','ROMANA','ROSA PILAR','YANET','ALTAGRACIA','GUILLERMA','MAYRA ALEJANDRA','LETICIA MARIA','MOUNA','YENIFER','AIALA','ROSINA','CHAIMAA','DULCE NOMBRE MARIA','CLEMENTA','GENARA','MARIA VALENTINA','MARYNA','GLORIA PATRICIA','MARIA DANIELA','DACIL','MARIA RESURRECCION','CAROLINA MARIA','MAIRA','ANA FRANCISCA','LUZ ELENA','KARIN','KHADDOUJ','ELISABETA','MAROUA','TAMAR','BAHIJA','CONCHITA','KENIA','JUA NA DOLORES','MARIA GRACIELA','LAIDA','IGONE','ALESSIA','ELODIA','LUZ DARY','ANSELMA','LARRAITZ','STELA','MARIA CANDIDA','ALBA LUCIA','NAJIA','OLEKSANDRA','LARYSA','CHAIMA','NIHAD','TEODOSIA','VENANCIA','VILMA','IRA CHE','MARIA AVELINA','LUZ STELLA','BEATRIZ ELENA','VINYET','CONSTANCIA','BRAULIA','FRANCISCA ISABEL','LEONOR MARIA','EVELINA','MARIA ILUMINADA','MONICA PATRICIA','MARIA OTILIA','MARIE','MARCIA','WISSAL','ALBA ROCIO','DOUAE','MARIA SACRAMENTO','MARIA EVANGELINA','SECUNDINA','YANIRE','AVRIL','FLOR','IARA','MARIA ELIA','MIGUELINA','ARSENIA','FIONA','MARIA ITZIAR','EUNICE','ISIS','MELCHORA','EFIGENIA','DANIA','NASSIRA','ROSA BLANCA','MARIA CLAUDIA','MARIA FELIX','MILOUDA','ROSARIO CARMEN','FAIZA','GISELLE','ANAMARIA','RAJAE','IMELDA','JOY','MYRIAN','SERENA','CARMEN ROSARIO','MARIA CONCEPCIO','ANA ROSARIO','MARIA GADOR','MARIA TISCAR','AURICA','AMY','ISRAA','VIVIAN','AHINOA','SELMA','NISRINE','IZAR','DELIA MARIA','ANA ALICIA','JING','MIOARA','ISONA','ILENIA','IONE','ZOULIKHA','BERTA MARIA','EUFRASIA','NISRIN','BARBARA MARIA','MARIA ENCINA','SIMONE','JASMINA','MARIA GERTRUDIS','VALERIE','YING','YANA','GLORIA INES','SHIRLEY','MAJDA','DANIELA MARIA','EUSTAQUIA','ANA RAQUEL','GERMANA','NAHIARA','ISRAE','MARIA DIVINA','MARIANA JESUS','AYLA','ZULEIMA','ANASTASIIA','LILIYA','MARIA GENOVEVA','MANUELA MARIA','ANISSA','POLINA','RADIA','ANALIA','MARIA TERESA JESUS','IVANNA','IRAIDA','ASSUMPCIO','GLORIA ISABEL','OMAIRA','MADALINA','ANASTASIYA','DEVA','VANIA','JULIE','LEONISA','YAN','REBECA

MARIA','NIRA','ALENA','ANA LUZ','IZARBE','ROSANNA','MARIA ROSALIA','MARIA OBDULIA','ALISON','MARIA FABIOLA','AGATHA','AWA','JUANA JOSEFA','LAURA CRISTINA','FRANCISCA ROSA','TERESA CARMEN','ARIANNE','HAWA','AHINARA','ELENA CRISTINA','MARIA REBECA','SULAMITA','HANNAH','MARIA DIGNA','CHRISTINA','BELARMINA','MARIA CORTES','ELVIRA MARIA','JOSEFA DOLORES','IRANZU','MARIA CELINA','GERALDINE','MARIA PIA','GANNA','ANA MILENA','LIDE','MARIA AMERICA','SHEREZADE','MARIA IDOYA','HERMOSINDA','MARIA BELLA','MARIA MONTE','MARIÑA','NAIALA','NAJOUA','SALADINA','SHARON','MARIA HUERTAS','AGUEDA MARIA','DOHA','AMALIA MARIA','ANDREA CAROLINA','KAROLINA','LLUCIA','MARIA TAMARA','ANA MARTA','BADIA','NATALYA','PALOMA MARIA','IVANKA','KEILA','MIREN ITXASO','ANCA','NARA','ROSMERY','MARIA CRUCES','ARAITZ','CARMEN VICTORIA','FATIMA ZAHRAE','CONSTANTA','MARIA JOSEP','AISSATOU','MARIA ALTAGRACIA','NELA','TASNIM','CARMEN JULIA','DESIDERIA','MARIA MICAELA','VIRGILIA','ALAE','ALBERTINA','ANICETA','ANNE MARIE','SOUHAILA','NATHALIA','OMAYMA','SHAKIRA','ELAIA','MAHJOUBA','MIRARI','STELLA MARIS','ZOUBIDA','RABAB','VENTURA','MARIETA','NADEZDA','AGNIESZKA','PETRA MARIA','RAISA','HELENE','BALTASARA','ELIXABETE','ADARA','ANA CELIA','IOLANDA','CHARLOTTE','HERMINDA','KATERINA','LILI','SANDRA ELIZABETH','ANUNCIA','ANA BELLA','MARIA RUTH','PAULA CRISTINA','TANYA','KELLY','MANOLI','MARIA LINA','SEFORA','CIRILA','JOSEFA ISABEL','OROSIA','JARE','JUANA FRANCISCA','MARY LUZ','NATALIE','SAFA','EMERENCIANA','ROSALINDA','CECILIA MARIA','ELAINE','LAMIA','GABRIELLA','MARIA TOMASA','ORIANA','KIMBERLY','MARLENY','ROSENDA','PAULE','ANN','ELIANE','SANDRA ISABEL','CRISTINA ELENA','FIDELINA','MARIA ANGELINA','CAPILLA','MIREN AGURTZANE','FRANCISCA PAULA','LAURA CARMEN','JANETH','KENZA','MARIA CONCEIÇAO','ROSA NIEVES','IULIA','JOSEFA PILAR','MARIANNE','YOHANA','JAZMIN','RABHA','BETTY','MARIA JACINTA','MARGOT','PERFECTA','YOSRA','QUINTINA','EUGENIA MARIA','ILONA','KHAOULA','EDNA','YOUSSRA','ARWA','ESPERANZA MARIA','LEYLA','SABINE','BEATRICE','EMPAR','OVIDIA','RHIMOU','ALINE','ANA EVA','GORETTI','MAIKA','RUBI','ELNA','INOCENTA','MARIA AGUEDA','BEGOÑA MARIA','FINA','NELIA','CRISTINA CARMEN','ELDA','PEPA','MARIA AIDA','EVA CRISTINA','MARIA CELESTINA','MARILUZ','YAZMINA','BALDOMERA','LIERNI','LURDES','RIHAB','AMELIA MARIA','PERPETUA','ASSUMPTA','IONICA','MARIA ANGEL','MARIANNA','ROSA BELEN','TAMARA MARIA','ARGELIA','OUMAYMA','AMAGOIA','MARIA LILIANA','AYELEN','JEMAA','SALWA','ZAYRA','DEVORA','CONXITA','HAYDEE','MILA','SILVERIA','LEONILA','LIANA','VIKTORIIA','AINA MARIA','CARLOTA MARIA','ISABEL DOLORES','MARTIRIO','NAYLA','KATARZYNA','MARILYN','UMA','ITAHISA','NUR','ROSARIO PILAR','EVA ISABEL','LI','MBARKA','MIREN ARANTZAZU','ANTONIA ISABEL','CRISTINA VICTORIA','DESEADA','INGE','MARIA CARLOTA','MIREN GARBIÑE','INGA','JANNA','MARIA ISIDORA','ERIKA MARIA','IULIIA','ZHOR','ICIA','ANTONIA JESUS','ARAN','DARA','LUDMILA','MAURICIA','SAKINA','NOUHAILA','ELENA CARMEN','LUCICA','MARIA FELIPA','MELINDA','DENISA','MARGARETA','ANA ANTONIA','INSAF','MARIA AMADA','TINA','SILVIA PATRICIA','MARIA CLEOFE','NICASIA','DANIELA ALEJANDRA','MARINA CARMEN','EVA GLORIA','MONTEMAYOR','CRISANTA','LUR','MARIA JUSTA','ANA ELISA','LUZ MARY','NAZHA','ROSA AMELIA','ELIZAVETA','LEIA','MARIA AUREA','PATRICIA ISABEL','ATHENEA','LEONTINA','MALGORZATA','MARIA

LURDES','NORBERTA','RIANSARES','GRANADA','SUSANNE','ANETA','MARIA PASTORA','HANA','DEYANIRA','IBONE','MIREN BEGOÑA','TABITA','TANIT','WILMA','ELIA MARIA','IMMA','JUANA CARMEN','LAMIAE','MAGALY','VALERICA','XINYI','ADINA','FADILA','GABRIELA MARIA','NAYADE','WAFAE','ALCIRA','MARIA CELSA','JOSEPHINE','MARIA OLIVIA','AMBROSIA','JEAN','LACRAMIOARA','LIBIA','NOHA','OUARDA','MARLEN','ROSARIO FATIMA','LUCIA ISABEL','PATRICIA CARMEN','FADWA','MARION','MATEA','SADIA','ANA LOURDES','ANA RITA','MARIA JESSICA','PAULINE','CARMEN JESUS','MARTA PILAR','GHEORGHITA','MONTAÑA','RACHEL','JANE','LIUDMYLA','LUCY','MARIA ELISABET','MARILENA','MARIA ALEGRIA','CORONADA','JACKELINE','MARICELA','MURIEL','DEMELSA','NINO','GHITA','IONA','MARIA CLOTILDE','MARTA BEATRIZ','PORFIRIA','HELENA MARIA','ILDA','MARIA PALMIRA','VIKTORIA','EIRE','LILA','PRECIOSA','BLANCA ISABEL','ILIANA','LAVINIA','ANGIE','MARIA LORENZA','SUSANA BEATRIZ','ZULIMA','CLAUDIA ANDREA','MARIA BELINDA','TATYANA','ERIN','FRANCISCA DOLORES','LIRIA','NADYA','PATRICIA ANN','ISABEL PILAR','MARAVILLA','MARGARITA ROSA','MARIA TEODORA','CARMEN ALICIA','JOSEFA LUISA','JULIA ISABEL','PETRONELA','VIRA','IANIRE','JEANETTE','TAMIMOUNT','CIRA','MARINA ISABEL','AYNARA','ESPERANÇA','HANAA','GADOR','MELODIA','MICHELE','ALTEA','BERTHA','CARMEN LAURA','ISABEL ROSA','LLARA','MARIA ELISABETH','VENERANDA','CAROL ANN','DANNA','MERIAM','MARIA APARECIDA','MELODI','SIXTA','MARICICA','MAUREEN','CARMEN MERCEDES','ANABELLA','HEIDI','KATYA','MIRTA','ROMAISA','ZHANNA','ALEIDA','JAVIERA','MARIA DIANA','MARIA PALMA','MARICEL','ARABIA','MARIA NEREA','CARMEN CRISTINA','ANTONIA ROSA','CARMEN BEATRIZ','CINTHYA','CLARA INES','MARIA ARRATE','DAIRA','IRENA','LETIZIA','YAMILE','ANTONIA PILAR','MARIA VAL','PILAR CARMEN','PAOLA MARIA','ANI','EMANUELA','RAJAA','SINFOROSA','XANA','MARIA PAULINA','AGUAS SANTAS','ESTELA CARMEN','NADEZHDA','NICOL','LILIANA MARIA','MARIA IDOIA','MARIA LUNA','TRINITAT','MARIA FLORES','MARTHA ISABEL','MARUJA','ZALOA','LIGIA','SUHAILA','JANINA','MARIA VILLAR','PASION','ROSA MARGARITA','DESISLAVA','EUDOSIA','MAGNOLIA','CLAUDIA MARCELA','JASONE','WIDAD','ANA SILVIA','BLESSING','CELINE','MARIA NELLY','SACRAMENTOS','AMABLE','MARIA DESEADA','ABILIA','FAUSTA','FRANCISCA CARMEN','ISABELLE','MARIA LIBERTAD','ROSA ESTHER','MARIA FELICITAS','MARIA MARTINA','YOMARA','CIARA','DANIELLE','MARIA DULCE','MARIA LARA','SANDRA LILIANA','LAURA PATRICIA','JIHANE','MARGARITA ISABEL','FAOUZIA','FRANCISCA ANTONIA','KHRYSTYNA','MAIMOUNA','ROSA MERCEDES','SIGRID','ANA KARINA','EMELINA','JASMIN','RAKEL','ROSA MARINA','BALMA','ORFELINA','ALISA','AMELIE','ANA GEMA','TELESFORA','MERY','MARIA GRANADA','GLORIA AMPARO','GLORIA ELENA','NICANORA','PIA','CATARINA','MARIA IMMACULADA','MONIQUE','RIHAM','XIAOYAN','ELINA','ELMA','ANA MARINA','ELIDIA','ERMELINDA','MARIA SALVADORA','PARASCHIVA','TERESITA JESUS','AMPARO MARIA','AOUATIF','FRANCISCA PILAR','GRACE','HIDAYA','LAURA ELENA','LOUIZA','ANA INES','ANA MARIA CARMEN','GLORIA CARMEN','MARIA BERNARDA','TERESA PILAR','ANTONIA CARMEN','JENNIFER MARIA','LUZ AMPARO','LUZ MERY','MARIA CIELO','MARIA FLORINDA','MARY CARMEN','MATILDA','ANDREIA','INTZA','MICHAELA','JIE','KEYLA','AYESHA','GABRIELE','GENMA','NORMA BEATRIZ','DIKRA','GISELE','GAIA','MARIA LUCILA','MONICA ISABEL','ROBLEDO','MARIA OLIMPIA','ALINA MARIA','ENYA','LARISSA','MARIA

GLADYS','MEGAN','CLAIRE','MARYEM','ESTHER LIDIA','LAURA BEATRIZ','ROMUALDA','YOVANA','ALAA','LAURA PILAR','MONICA ALEXANDRA','ROCIO MARIA','SOLANGE','ANA BEGOÑA','MARIA CANDELA','MARTA LUCIA','NILDA','ASUNCION MARIA','CARMEN CECILIA','CLAUDIA LORENA','ESTER MARIA','KHEIRA','ROSANGELA','ANA LORENA','TERESA ISABEL','ANA CONCEPCION','MAXIMILIANA','ROSARIO ISABEL','WAFAA','YULIANA','EILEEN','FLORA MARIA','ANA CLARA','ANA ESMERALDA','ANDREA CARMEN','DOAA','FAYNA','INMACULADA MARIA','SILVIYA','LLEDO','MARIA MARAVILLAS','MONICA BEATRIZ','MARIA MERCED','MARAM','MARTA CRISTINA','ROSEMARY','LUCINIA','HASNAA','JOSEFA ROSA','TARSILA','AUDREY','AZAHARA MARIA','CARMEN YOLANDA','NAJWA','CRISTINA PILAR','DENISSE','KARINE','MARIA SAMPEDRO','SAIRA','BERENICE','EVA BELEN','MARIA REMEI','ROSA LUZ','AMAIUR','INDHIRA','NELI','DAISY','GERONIMA','NEREA MARIA','SILVIA CRISTINA','WAFA','BEATRIZ EUGENIA','GABRIELA ALEJANDRA','MARIA JUDITH','ANA CLAUDIA','ANA MANUELA','GOIZANE','JENNYFER','ALEXIA MARIA','ARGIMIRA','KATRIN','MARIA BRIGIDA','SABRIN','AIMARA','ELOISA MARIA','JOANE','JOYCE','ROSA MAR','TATIANA MARIA','VICTORIA ISABEL','JULIET','LUCIA PILAR','IDURRE','NAJIMA','ALICIA CARMEN','CONSUELO MARIA','GLADIS','LAMYAE','LESYA','SERGIA','YUSRA','ANTONIA JOSEFA','FIRDAOUS','LAZARA','MARIA EDUARDA','LUCITA','MANOLITA','PETRUTA','ALANA','PATRIZIA','MARIA BALBINA','ROCIO MAR','VERA LUCIA','CINDY','CIRIACA','MARIA DOMINGA','NAIR','CARMEN PATRICIA','LUCRETIA','PERLA','YORDANKA','ILARGI','MARIA LUCRECIA','TARA','DOLORES PILAR','HERMITAS','NIEVES ROSA','MARIA TANIA','OANA','OMAYRA','RENATE','ROSA MARY','SOLEDAD MARIA','ANNIA','CLAUDIA MILENA','MARIA SALES','NATALY','BEATRIZ CARMEN','ITSASNE','NIKOLE','DIANA ELIZABETH','HASSNA','BIANCA MARIA','FIORELLA','FRANÇOISE','JAEL','MARIA VILLA','ANNABEL','IOSUNE','JOSEFA TERESA','MARIA ELVA','RANYA','CARLOTTA','CARMELINA','ELLA','EZEQUIELA','MARIA MONSERRATE','JEANNETTE','KATALIN','ROSA INES','CARMEN LUZ','ELISABETE','MARIA ETELVINA','ROSA ANGELICA','KHADY','AINHOA MARIA','ANNIE','ATTENERI','CARMEN ESTHER','CLAUDIA ALEJANDRA','MARTA LUISA','YANETH','AFRICA MARIA','AMANCIA','CHRISTIANE','DAIANA','MARIA EUSEBIA','MARIA YURENA','NARCISA JESUS','OUAFAA','STEFANIE','ZAHIA','ESTIBALITZ','EUNATE','MACRINA','MARIA INOCENCIA','ALSIRA','CRISTIANE','KSENIA','LIDIYA','YAMILETH','ALODIA','ANGELA CARMEN','MARIA AROA','ROSA DOLORES','ANTONIA DOLORES','AURELIANA','IRIS MARIA','MARIA BARBARA','YENNY','ALEXANDRINA','DOMINIQUE','MARIA RAMOS','MENNANA','OLESYA','OUASSIMA','BENJAMINA','ELISEA','HASSANA','JONE MIREN','MADELEINE','MARIA LUJAN','DIANA MILENA','FRIDA','ISABEL TERESA','SASKIA','ANA LIDIA','ANGELA PATRICIA','AYMARA','DULCENOMBRE MARIA','DUNIA ESTHER','IZADI','MARIA LUZDIVINA','MIRYAN','MONICA CARMEN','MARIA ELBA','MARINE','NOHEMI','YAEL','ANJA','MARIA ADRIANA','MIREN JAIONE','YI','DEOLINDA','DOLORES ISABEL','NATIVITAT','NAYA','PAULA ISABEL','ARIA','CRISTIANA','BLANCA ELENA','CRUZ MARIA','JIAYI','MARIA AZAHARA','NADIIA','ADRIANA PATRICIA','LUJAN','STELUTA','TINGTING','CHADIA','JASMINE','LILIANA PATRICIA','LUNA MARIA','MARIA ELOINA','SONIA ISABEL','CARMEN ANGELES','DOAE','MARTA CARMEN','ALEXA','ANA ESPERANZA','JENIFFER','LAURA SOFIA','RALUCA','BUENAVENTURA','GIORGIA','MIREN GURUTZE','NATALI','PATRICIA ALEJANDRA','PATRICIA ELENA','ROSSANA','FELISA MARIA','FORTUNATA','RAJA','ANA VIRGINIA','ANABELA','ARCADIA','BLANCA FLOR','CARMEN MARINA','DORLETA','MARIA

MISERICORDIA','RITAJ','TIFFANY','ANA ANGELES','ESTHER GLORIA','LISBETH','SALUSTIANA','CONRADA','CORONA','JANICE','KEIRA','MARIA ISAURA','MARIA RUFINA','MARIA SABINA','RAFAELA MARIA','WISAL','ADAMA','ANDREEA MARIA','INA','JUANI','MARTA TERESA','SOFIYA','ANA YOLANDA','FRUCTUOSA','JULIETTE','MARIA PURA','SARA CARMEN','SILVIA BEATRIZ','VALVANERA','ANDRESA','KATHLEEN','MARIA MARINA','MARIA NANCY','NATALIA ANDREA','PAULA CARMEN','SUSANA ISABEL','TRINIDAD MARIA','VIDALINA','ELENA DANIELA','LUBNA','LUISA ISABEL','MARIA MARCELINA','NASSIMA','XANDRA','ANGELIKA','MIRTHA','ONINTZA','GETA','SANDA','DIAMANTI NA','EWA','LIDYA','LLUM','ANDREA ISABEL','ORENCIA','RABIAA','SILVESTRA','BLANDINA','OLGA LIDIA','WEI','ZHOUR','ALICIA ISABEL','EREA','FLORES','INMA','MARIA ZORAIDA','GIULIANA','MARIA ANTIGUA','MARIA CAMPO','MARIA ELIDA','MARIEL','SOUKAYNA','DIANA ISABEL','FRANCISCA TERESA','MARIA MINERVA','BEATRIU','ELIXABET','GUAYARMINA','JORGINA','MARIA JIMENA','MARIA MAGNOLIA','MEI','NIHAL','PETYA','ROSA EMILIA','ANICA','FRANCISCA JOSEFA','HLIMA','MARIA HILDA','MIKAELA','VERONICA CARMEN','BEATA','JELENA','JINGJING','BINTOU','CLEOFE','DIANE','EVA PILAR','HODA','MARISELA','VERENA','BETSABE','MARIA PAOLA','PATRICIA ELIZABETH','AMANDA MARIA','ANGELA ISABEL','ROSALIA MARIA','ZOYA','MIREN AMAIA','NUBIA','YULIIA','ALMUDENA MARIA','ARA','CARMEN JOSEFA','ELENA PILAR','MARTHA LILIANA','CALIXTA','IRENE CARMEN','NURIA ISABEL','OLGA BEATRIZ','VALERIYA','BERNADETTE','ELPIDIA','EUKENE','ISABEL VICTORIA','LAURA VICTORIA','MARIA NELA','MIREN MAITE','SABRINE','YESSENIA','ZULMA','CARMEN AMELIA','ELENA VICTORIA','IDA','KATTALIN','MARGARITA CARMEN','MARIA FERMINA','ISABEL ROCIO','IZIAR','MARIA LUCINDA','NOELA','REINA','ANNA ISABEL','FREDESVINDA','GEORGIA','ILARIA','JULIA CARMEN','LONGINA','LUIZA','OLEGARIA','PATRICIA PILAR','RUBIELA','ANXELA','IHSSAN','LIDUVINA','MARIA AINHOA','OUAFA','ZAYNAB','ANISA','CARMEN LOURDES','EIRA','ENEIDA','GLORIA ESPERANZA','ROSA ANGELES','TIZIANA','ANZHELA','JOSEFA ANGELES','LAURA ESTHER','MARIA ALFONSA','MERCEDES CARMEN','TEREZA','MARIA FELICIANA','ELISABEL','FANTA','HAMIDA','IRAYA','JESICA MARIA','LEAH','MARIA DULCE NOMBRE','WARDA','ARACELI MARIA','GILLIAN','MARJORIE','NOHAILA','ROSA EVA','YVETTE','JOANNE','LIS','MARIA ERMITAS','MARIA QUERALT','MARIA TATIANA','ANGELA CRUZ','BERNABELA','MANUELA CARMEN','MARIA PERPETUO SOCORRO','SOFIA ISABEL','MARIA CONSTANZA','RAMONA MARIA','SANDRA CRISTINA','CLAUDIA ISABEL','MARIA DORINDA','TAIS','UBALDINA','ALIDA','ANA MARIA PILAR','KAMILA','MARTHA ELENA','MIRIAM CARMEN','ZAHARA','ZELTIA','ALBA LUZ','ALOÑA','MARIA BENIGNA','MATILDE MARIA','NIURKA','ALINA ELENA','ANA PAOLA','CASSANDRA','FIRDAWS','LAURA DANIELA','MARIA AUGUSTA','MARIA CLEMENTINA','SANDRA LORENA','SILVIA ELENA','BETTINA','BINTA','LIN','MARIA GUILLERMINA','ITZEL','MARIA DAVINIA','MARIA VALVANERA','MICHELA','XIAOMEI','YASSMIN','YENNIFER','VIVIANE','AINIZE','ALBA MARINA','FAITH','JIHAN','REMIGIA','ROCIO BELEN','ANCUTA','CLARIBEL','CORALIA','HUERTAS','ISAMAR','NOELIA CARMEN','SYLVIE','AITANA MARIA','ANA INMACULADA','ATANASIA','MACARIA','MARIA CASILDA','DENIA','GEMA ISABEL','JOSEFA FRANCISCA','JUANA PILAR','LUCIE','MONICA ELIZABETH','ANDREA ESTEFANIA','ANDREINA','AYLEN','GHIZLAN','LAURA ALEJANDRA','MARIA JOSEPA','PATRICIA ALEXANDRA','SORAYA MARIA','ANA CATALINA','ANNETTE','CLAUDIA

ELENA','VALME','ALAINE','DELMIRA','IRIA MARIA','MARIA DESIRE','MARTA SUSANA','MIREN JASONE','SELINA','VICTORIA CARMEN','AMEL','BETH','CASTORA','FLORENCE','ISLAM','LORENA CARMEN','MARIA HUMILDAD','MIREN GOTZONE','ROSE','SONIA ELIZABETH','ANIANA','CARLA PATRICIA','GURUTZE','NAJLAE','ROSA AURORA','SOAD','CARMEN ELISA','EDUVIGES','LYDIA MARIA','ROSA VICTORIA','YU','ALICIA ESTHER','ANA ELIZABETH','MARIA MARCELA','MERCEDES PILAR','ANA RUTH','HUI','JOLANTA','KATARINA','CRISTETA','FATUMATA','JULIA ROSA','MARIA ZULEMA','MIROSLAVA','NADEJDA','SILVIA CARMEN','AFRIKA','CARMEN CONCEPCION','DRIFA','EMILIYA','JANETTE','JOCELYN','NANA','OLGA PATRICIA','URSULA MARIA','ANA ELVIRA','BEATRIZ ISABEL','JANINE','LUZ ANGELA','MARIA CORONADA','NOR','ROSEMARIE','ROXANA MARIA','CELERINA','EVGENIYA','IOANA MARIA','MARIA GUIA','NAZARIA','NEIVA','SERVANDA','SIERRA','SONIA CARMEN','ANA VERONICA','DIANA ALEXANDRA','ELISABETTA','IKHLAS','ISABEL CLARA','MARIA DORA','NADIA MARIA','YARITZA','ALBA
CARMEN','ARACELY','DAPHNE','LIDON','LOUISE','NAYMA','RUPERTA','SONIA ESTHER','ANA MATILDE','CARMEN LIDIA','CONCHI','CRISPINA','EXPECTACION','HELIA','ISABEL ANA','KATHARINA','MARIYANA','MIREN IDOIA','NIKOL','AURITA','KARLA PATRICIA','MONICA PILAR','XULIA','YESICA MARIA','ZUZANA','ELIZABETH MARIA','IBTISAM','MARIA OLATZ','MARIA SHEILA','MIRNA','ODETTE','YAMILET','ZUNILDA','BTISSAM','DAKOTA','FERDAOUS','ROSITA','ANA REYES','GLADYS MARIA','HALA','HILDA MARIA','ISABEL ANTONIA','MARIA ALICE','MARIA JENNIFER','CRISTINA BEATRIZ','LEIDY JOHANNA','MAHA','OHIANE','ALICIA BEATRIZ','ARGIÑE','BIRGIT','ISABEL ANGELES','MARIA ERNESTINA','PLACERES','STEFANY','ABIR','GORANE','RUSLANA','SONJA','VICTORIA
ANGELES','YAMILEY','ARIADNA MARIA','ARYA','FATIMA CARMEN','GERGANA','HADA','MARIA MODESTA','MARTA ROSA','GLORIA PILAR','LEXURI','RIMA','SOFIA ELENA','DIANA CRISTINA','DUMITRA','KAUTAR','ODALYS','ONELIA','REGINA MARIA','RITA CARMEN','CAROLINA CARMEN','ELIZA','KRASIMIRA','MARIA CARMO','ANA EMILIA','ANAHIT','ERLINDA','GEANINA','MAGDALENA MARIA','MARIA BENEDICTA','MARIA ELDA','MERCY','MONICA ALEJANDRA','MONICA ANDREA','OIANE','ELLEN','FACUNDA','GONZALA','MARIA CLEMENCIA','MARIA JUSTINA','PAULA ALEJANDRA','SIRIA','VERONICA ELIZABETH','YESIKA','ZAIDA MARIA','CORO','GENESIS','MARCELA ALEJANDRA','PRISCILLA','ROSITSA','SARA ELENA','YAJAIRA','AROA MARIA','AYLIN','DONATA','EUTIQUIA','FATIMA
ROSARIO','INDALECIA','IVONE','MARYANA','CARMEN MAR','ISABEL ROSARIO','JUANA MANUELA','KEREN','LINGLING','MADDALEN','MARIA ENEDINA','TSVETELINA','ANA JOAQUINA','DAIDA','MAFALDA','MAYRA ALEXANDRA','ESTEFANY','EVA CARMEN','MELODIE','POLONIA','TORIBIA','VERONICA ALEXANDRA','ANA ASUNCION','BLANCA CECILIA','ELKE','GALYA','MAYKA','REMEDIO','TANJA','ZAMARA','BLANCA
AZUCENA','CHENOA','EUTIMIA','LILIANE','LORE','ALIONA','ANA SOLEDAD','ELENA BEATRIZ','INASS','JOSEFA MANUELA','TXELL','AMETS','CARMEN MILAGROS','DRISSIA','LAURA ANDREA','YOLANDA ISABEL','YUMALAI','ADHARA','ADRIANA CAROLINA','AIDE','ANDREA PILAR','AYADA','EVA PATRICIA','GIUSEPPINA','ITXASNE','MARIA ROSARIO FATIMA','MIREN ARANTZA','ROSA ANTONIA','SILVIA ISABEL','WASSIMA','YUMALAY','ALEXANDRA ELENA','ANTONIA BELEN','CARMEN
FRANCISCA','DEISY','EGUZKIÑE','ERNESTA','MERYAM','OLGA ISABEL','OUIAM','MARIA ALBINA','MARIA EDELMIRA','MARIA EUFEMIA','MARTINE','SARAH JANE','BLANCA AMELIA','MARIA PASCUALA','NICOLA','CHAXIRAXI','CLAUDIA CRISTINA','JIAQI','KATERINE','MARIA EDITA','MARIA REMEDIO','RANDA','SAIMA','YUREMA','ZAIRA MARIA','ANDREA CRISTINA','CARLA

SOFIA','DIANA ELENA','EL BATOUL','ESTERA','JESSICA ALEXANDRA','JOSEFA ROSARIO','KATE','MARGARITA PILAR','NAHIR','ROSA TERESA','RUMYANA','RUTH ESTHER','DANIELA ALEXANDRA','ISABEL MERCEDES','MYROSLAVA','ORNELLA','ROZALIA','SALHA','SINAI','USUE','XIN','ESTILITA','EULALIA MARIA','IANA','MIREN IZASKUN','VALERIA MARIA','YANINA','ANA CRUZ','ANDREA ALEJANDRA','ESTELLA','HRISTINA','INES CARMEN','LUZIA','MAIRA ALEJANDRA','MARIA YASMINA','ANA MAR','ANELIYA','DANIELA CARMEN','CHAYMA','CORINNE','FATIMA SOHORA','GIMENA','JUANA DIOS','LUCINA','MARTA INES','MOIRA','AILEN','ANA EUGENIA','ANA GUADALUPE','ANDREA ELIZABETH','BUSHRA','CYNTIA','DOROTA','ELZBIETA','LUZ ANGELICA','NATHALY','OUAHIBA','SUSANA CARMEN','CINTA ROCIO','GLORIA ELIZABETH','MARIA EMERITA','MARIA IRMA','SOHORA','TATJANA','XIAOLI','ZOUHRA','BLANCA INES','GINEBRA','HONG','MARCELIANA','MERTXE','RUTH ELIZABETH','SONIA PATRICIA','SUKAINA','ANA MILAGROS','CHAYMAA','CORPUS','LEONIDA','WEIWEI','KASSANDRA','MAICA','SAMPEDRO','VERONIQUE','YAQUELIN','ALONA','EILA','HAIYAN','MARIA VALERIA','ALEJANDRA ISABEL','BLANCA PALOMA','CAROLINA ANDREA','MARIA DIONISIA','MARIA JESICA','PING','SONIA CRISTINA','YINGYING','YOSUNE','ZINAIDA','ALEYDA','ALLENDE','ARHANE','CARMEN ANTONIA','CAROLINA ISABEL','ELENA ALINA','ELICIA','ELISABET MARIA','FANG','MANUELA PILAR','MARIA GENEROSA','MAYELIN','PALOMA ROCIO','SHAZIA','VICTORIA ELENA','ANGELA ROSA','ARLETTE','CLAUDIA BEATRIZ','DEMELZA','JOSEFINA MARIA','KAOUTHAR','NURIA CARMEN','SARA CRISTINA','SOMAYA','VITORIA','YOSELIN','ANDERE','CARMEN MARGARITA','CINTIA MARIA','DULCINEA','FLORIANA','LAURA VALENTINA','LIHUA','LOBNA','MARIA ERIKA','RAQUEL CARMEN','SEVERA','SUZANNE','ANA AURORA','ANDREEA ELENA','IRENE ISABEL','JUDITH MARIA','KHOLOUD','LENKA','MARIA LUZ DIVINA','NATALIA CARMEN','TAMOU','THELMA','XIAOFEN','YSABEL','AINARA MARIA','ALEXANDRA ELIZABETH','ENCARNA','ETNA','EVERILDA','IKERNE','MARIEN','NADIRA','OUMOU','ROSA AMPARO','SELENIA','SILVIA PILAR','SORINA','WAHIBA','AMBAR','ANTONIA ANGELES','ANUTA','HONORIA','MARINA PILAR','RUT MARIA','SAMAR','XIAO','ANA BERTA','CAMILLA','CONCESA','EMMA LOUISE','FANICA','FRANCISCA JESUS','GIOVANA','MARIA ARGEME','MARIA ELADIA','MARIA JOANA','NATALIA ISABEL','SAMIHA','SONIA BEATRIZ','VITA','ANTONIETA','JUANA LUISA','LING','MARTA CECILIA','NERMIN','ROSA ANGELA','ROZA','YUE','DAGMAR','GERARDA','JUSTA MARIA','NOUR EL HOUDA','SEYNABOU','ANZHELIKA','ASELA','ELISABETH MARIA','LUISA CARMEN','MARIA GEORGINA','OLIVIA MARIA','RAMIRA','SAMRA','SERAPIA','SOFIA CARMEN','STEPHANY','ANDREEA CRISTINA','CANDIDA ROSA','FRANCISCA ANA','JULIANA MARIA','KATJA','LEIDY JOHANA','MACARENA ROCIO','MAMA','MARIA JERUSALEN','MATXALEN','SARA ESTHER','YOHANNA','ANA AMELIA','ANTONIA LUISA','DENISA MARIA','GASPARA','JESSICA PAOLA','MAEVA','MARIA JACQUELINE','NINFA','TEODOMIRA','AURA MARIA','BEATRIZ PILAR','BLAU','CARMEN ANA','CAROLE','CRISTINA MIHAELA','ELENA MIHAELA','GRACIELA MARIA','JENNY ELIZABETH','KHALIDA','LUTGARDA','MADIHA','MARIA FRANCESCA','MONSERRATE','TUDORITA','ALICIA PILAR','DAMASA','DIANA LORENA','GIADA','KHADRA','LYUDMILA','MARGARET ANN','MARIA JULIETA','MIN','NOEMI MARIA','OLGA MARINA','SANTAS','SILKE','CATALINA ANA','CLAUDIA LILIANA','HAFSSA','KRYSTYNA','LEOPOLDINA','OMARA','REMEDIOS MARIA','SALKA','ABIDA','EL GHALIA','LAURA MARCELA','LIZETH','MONICA ESTHER','RENEE','ROCIO ISABEL','SARA LUCIA','VENUS','VIDA','YASSMINE','AILEEN','CARMEN EMILIA','DJAMILA','IKRAME','LAURA

ELIZABETH','NEILA','SABA','STEFANA','SUSAN ELIZABETH','USOA','EMILIE','ENIA','FRANCISCA ASIS','GALIA','KAMAR','LAURA TERESA','LIFEN','MAITENA','MARIA NOEL','RODAINA','ROXANA ELENA','SANDRINE','VICTORITA','XIA','ANACLETA','ARINA','ARMONIA','KAREN ELIZABETH','MARIA BIENVENIDA','MARIA CUSTODIA','MARIA VALME','NOUZHA','SHARA','SHARAY','AFAF','ANDREA PAOLA','ARELIS','LUZMILA','MARIE THERESE','VANYA','CLARA ROSA','IZABELA','JULIE ANN','MARIA LUCIANA','NAYALA','NISAMAR','SARA PILAR','ULIANA','YAIMA','YAZMIN','YUDITH','ALEXANDRA IOANA','BET','CARLA ANDREA','DARYA','DOINITA','ELVIA','FIRDAUS','IHSAN','IVELINA','LUCIA TERESA','MARIA MARLENE','PRADO','SAFIRA','ANA IRENE','ANA REMEDIOS','ANJANA','CHRISTINE ANN','LIDIA ISABEL','MARIA BENILDE','OANA MARIA','SONIA PILAR','VIOLETA MARIA','XIAOWEI','YUXIN','ANTONIA FRANCISCA','ANTONIA TERESA','CARMENZA','CRISTINA ROCIO','GABRIELA ALEXANDRA','LAURA MERCEDES','LAURA ROCIO','LESLIE','LINAREJOS','NAZIA','ROSA BEATRIZ','RUBY','SASHA','SUSAN MARY','VALERY','WANDA','CARMELITA','COUMBA','GLORIA BEATRIZ','GLORIA MERCEDES','JALILA','JULIA ESTHER','MARIA ERUNDINA','MARIA FAUSTINA','MARTA EUGENIA','NAARA','PAULA ALEXANDRA','ZENEIDA','BLANCA AURORA','CHRISTEL','FIDELIA','KERSTIN','LAURA ROSA','LUZ CARMEN','MARUA','MONICA ROCIO','TERESITA NIÑO JESUS','VERONICA ISABEL','YOLANDA CARMEN','ANA OLGA','ANGELA CRISTINA','ANTOANETA','DELICIA','ELA','EMILIA CARMEN','GAYANE','JOHANNA ELIZABETH','JOSEFA CONCEPCION','KHADIDJA','MAME DIARRA','MARIA EVELIA','MARTHA PATRICIA','PAU','SALLY','TATSIANA','VIOLETTA','ANGELES CARMEN','BAYAN','EZTIZEN','GISELLA','KAYLA','MAJIDA','MANUELA ISABEL','MARIA NORMA','MARTHA MARIA','SIOMARA','ABRAR','AMARANTA','ARENE','AUREA MARIA','CARMEN RAQUEL','DOLORES ROCIO','ELIZABETH ANN','FATIMA ZAHRAA','MARIA AMABLE','MARIA CORINA','NAYAT','RADKA','SUMAYA','TIMOTEA','ULRIKE','AMANDA JANE','AURORA CARMEN','CANDIDA MARIA','CARMEN MANUELA','CECILIA CARMEN','EXPIRACION','AURIA','BELKIS','CARMEN INMACULADA','CARMEN REYES','GISELA MARIA','GUADALUPE MARIA','IOANA ALEXANDRA','JAMAA','JULIA ELENA','MARIA MIHAELA','ROSA CRISTINA','SIHAME','ALLISON','CARLA ISABEL','CLARA LUZ','CRISTINA EUGENIA','DORITA','ENRICA','JANAINA','KALTOUM','LAURA GABRIELA','LORETA','MARIA PAU','MARIA SILVINA','MARITA','NILA','PATRICIA ANNE','PATRICIA CAROLINA','SARA PATRICIA','ALIZE','ANOUK','ARMANDA','AZAHAR','CANDELA MARIA','DARINA','EDEN','ELENA PATRICIA','ELEONOR','EVELYNE','GAUDENCIA','KLARA','LILIIA','ROMAYSA','SARAH LOUISE','YEDRA','CATIA','DOMNICA','HADHOUM','KHADOUJ','MARIA FELICIA','MARIA NARCISA','SUSANA PILAR','VICTORIA ALEJANDRA','XOANA','CARMEN SOLEDAD','DIANA PAOLA','EMETERIA','FELICITACION','GOTZONE','ISABEL MARGARITA','JANAT','LILY','ROSA JULIA','SEREZADE','SHANSHAN','TRACY','ANA KAREN','BOUTAINA','CLARA ELENA','ELENA GABRIELA','EMA','ENOLA','JIAHUI','KARINA ELIZABETH','LAURINDA','MARIA INMACULADA CONCEPCION','PALMA','PAULA VICTORIA','PENKA','AFNAN','BLANCA ESTELA','CARMEN INES','CATALINA ISABEL','JOAQUINA MARIA','ROSA FRANCISCA','ROSE MARIE','SELVA','TECLA','YASIRA','ANNALISA','ANTONIA ROCIO','ASCENSION MARIA','INTISSAR','KIM','MARIA ORETO','NATALIA SOLEDAD','NORMA ELIZABETH','ROSARIO ANA','SARA BEATRIZ','VERONICA ANDREA','ANA AMPARO','ANA BLANCA','ASYA','BRENDA MARIA','DONIA','FRANCIA ELENA','JOSEFA MERCEDES','JULIA PILAR','MIREN IGONE','NAYRA MARIA','ROSA LAURA','SALETA','CARMEN FATIMA','DONATILA','HEIKE','JENNIFFER','KAWTHAR','LIDIA CARMEN','LORENA ISABEL','LORRAINE','MARIA ROSAURA','MARIA SEBASTIANA','NURIA PILAR','PAOLA ALEJANDRA','SOFIIA','SVETLA','YRAYA','ALEXANDRA GABRIELA','ASSIYA','DARIA

MARIA','DOROTHY','FLOARE','KRISTINE','MARIA IMELDA','MARIA URSULA','MARIA XIMENA','MARILIN','NADIN','ONINTZE','PAVLINA','VARVARA','ANA MAGDALENA','ANDREA VICTORIA','IDALIA','ISATOU','LLUVIA','LUZ ESTELLA','MAJA','MARCELLA','MARIA AMAIA','MARIA ANITA','MARIA REINA','MILAGROS MARIA','NANCY ELIZABETH','NEREA CARMEN','PEÑA','PILAR ISABEL','VICTORIA PILAR','ZINA','ADRIANA ISABEL','BERNABEA','DAVINIA MARIA','FATTOUMA','GOIZALDE','GRACIANA','LIPING','MARIA SARAY','MARILU','NADINA','NATIA','ROCIO MACARENA','XIAOHONG','ANA ADELA','ANA ALEJANDRA','ANA CONSUELO','ANGELA ROCIO','BELLA MARIA','CINZIA','EVA ROSA','GHERGHINA','JORDANA','JOSEFA ENCARNACION','LISANDRA','LUCERO','MARIA LILIA','MAY','MIREN LOREA','MONICA CRISTINA','SALUD MARIA','SARA VICTORIA','SARE','SONYA','YUANYUAN','ADA MARIA','AGUSTINA MARIA','CHARO','CLAUDIA CAROLINA','CONCEPCION CARMEN','CRISTINA ALEXANDRA','ELODIE','ESTEBANA','FERNANDA MARIA','ITHAISA','JORGELINA','LEONORA','LUCIA BELEN','LUCIA VICTORIA','MARIA CARMEN PILAR','MARIA CORPUS','ADONINA','AUGUSTA','CARMEN LORENA','CARMEN RITA','ESTITXU','ILHAME','JENIFER MARIA','JULIA TERESA','MARIA CARLA','MARIA ROXANA','TEA','ULYANA','YASSMINA','YURENA MARIA','ADRIANA CARMEN','ALBA ROSA','ANDREEA ALEXANDRA','CARMEN EUGENIA','DIGNA MARIA','FATHIA','FETTOUMA','GLENDA','INGEBORG','KLAUDIA','LAURA VANESSA','MANUELA JOSEFA','MARIA IRIS','MARIA LLUC','SILVIA ESTHER','VITALINA','YUDIT','CARMEN SUSANA','DAN','DANIELA ELENA','ELIZABETH ANNE','ERICA MARIA','EVA LUCIA','FRANCISCA ANGELES','KADIATOU','LAURA INES','LIDIA ROSA','LOLITA','MARIA CAPILLA','MARIA SETEFILLA','MARIA ZAIDA','MARTA BELEN','OCTAVIA','SOTERA','VIVIANA ANDREA','AIARA','ELENA ALEXANDRA','HALINA','HONORATA','IRENE PILAR','KORO','LAMIAA','MARIA NEREIDA','MARIA NICOLASA','NERE','PILAR ANGELES','RAFIKA','SONIA RAQUEL','ANDREA VERONICA','ARACELIS','ELISANGELA','EVGENIA','HANNELORE','INMACULADA CARMEN','IVA','LOURDES CARMEN','LYNN','MARIA ANGELITA','MARIA BLASA','MARIA GISELA','MARIA OLALLA','MARUXA','REINA ISABEL','SUSAN ANN','ACACIA','ALBA ISABEL','ALINA MIHAELA','CRISTEL','DEBORA MARIA','DESSIRE','HERMESINDA','LAURA CAROLINA','MANUELA ROCIO','MARIA YAIZA','MILANA','MIREN AINTZANE','NATALLIA','NONA','PIEDAD MARIA','AGUASANTA','ALVARINA','ARIAN','BIANCA ELENA','EKIÑE','JENNY ALEXANDRA','MARIA EDUVIGIS','MARIA ESPINO','MIQUELA','ROSA MATILDE','SAMYA','SORKUNDE','ANA LETICIA','ANA MARIA ISABEL','ARGELINA','CATUXA','DAURA','JOSEFA VICENTA','KENYA','LILIANA BEATRIZ','LORNA','MAIRENA','MANANA','MARIA YESICA','MARIA YSABEL','MARIE LOUISE','NASIRA','SIMONA MARIA','VOLHA','ANA CARLOTA','CHRISTELLE','GRACIELA BEATRIZ','IDALINA','MANUELA JESUS','MARTA PATRICIA','RUTH NOEMI','YURENA CARMEN','CARMEN AURORA','FIDENCIA','MAIER','MARIA BIBIANA','MARIA REPOSO','MARTA ROCIO','MIREN ALAZNE','MONICA LILIANA','NURI','ROCIO ALBA','ROMAISSA','SHAMIRA','ANGELA PILAR','AUDELINA','BOCHRA','ESCARLATA','ISABEL EUGENIA','JESUS MARIA','MARIA ISIDRA','MARIA VICTORINA','MONA','NISSRINE','SANDRA BEATRIZ','YAROSLAVA','ANDREA PATRICIA','ARTEMIA','BENICIA','CARMEN ELIZABETH','DEYSI','DORA MARIA','ISABEL FRANCISCA','JIAXIN','JOSEFA INES','JUANA VICTORIA','LUSINE','MARIA CISNE','MARIA LINAREJOS','MARIA MADALINA','MARIA MELANIA','MARIA ROGELIA','MONICA ELENA','NASTASIA','PRISCA','ROSABEL','SOFIA ALEJANDRA','AISSATA','GRIMANESA','LAURA CAMILA','MARIA ALINA','MARIA JOVITA','MARIA PENELOPE','NELLI','RAHIMA','RODAYNA','XIAOZHEN','ADRIANA ELIZABETH','ANA JUANA','ANA NIEVES','CRISTALINA','EUSTASIA','JAZMINA','KAUZAR','MARIA EULOGIA','MARIA SIMONA','RESTITUTA','SAMAH','AIZPEA','ANQI','AYSHA','ELBA

MARIA','ESTIVALIZ','IVON','JEANNE','JUANA JESUS','JUANA MERCEDES','MARIA ANASTASIA','MARIA EDITH','MARIA JUDIT','MARILIA','MAXIMIANA','NAJATE','RAYHANA','SANDRA CARMEN','AURELIA MARIA','ESPERANZA ROCIO','ISABEL JOSEFA','MANOLA','MARY ANN','OUIDAD','TERESA DOLORES','AISA','ALA','AMALUR','CRINA','INGRID MARIA','IRINA MARIA','MARIA CARMELA','MARIA GORETI','MARIA PEREGRINA','STEFKA','XINYU','ANDREEA MIHAELA','CLAUDIA SOFIA','HERMINIA MARIA','ISOLDA','LILIANA ELIZABETH','MAGHNIA','MANUELA DOLORES','MARIA JERONIMA','ROSA LIDIA','TERESA ANGELES','YESSICA MARIA','ALBENA','CARMEN ELVIRA','DANUTA','ELIECER','GEMA CARMEN','GEMA PILAR','JENNY PATRICIA','MARIA ROSANA','MARIA SERAFINA','MARY ELIZABETH','PAULA PILAR','SAHAR','SARA ELIZABETH','SOFIA VICTORIA','VERONICA PATRICIA','VERONICA ROCIO','YUN','BARTOLINA','CARMEN SOFIA','JUANA ROSARIO','MARIA LLUCH','MIGLENA','MINIA','NATIVIDAD MARIA','ODALIS','RABEA','SUSAN MARGARET','VALENTINA MARIA','ANDREA FABIANA','CLAUDIA ELIZABETH','EMILIA ISABEL','ESTRELLA CARMEN','JESSICA ALEJANDRA','JUTTA','LIZBETH','MARIA CEU','MARIA STELLA','MARIJA','MARTHA ELIZABETH','NESRINE','YANELI','YANG','ZITA','BLANCA LUZ','CARMEN GEMA','CECILE','CLAUDIA CARMEN','CONCEPCION PILAR','DANIELA ANDREA','DIANA LUCIA','EMPERATRIZ','ERHIMO','FATIMETOU','GABRIELA ELIZABETH','GOIURI','KHOULOUD','LEOVIGILDA','MARIA DOSINDA','MIHAELA CRISTINA','MONICA RAQUEL','SEHILA','SIENA','TRIANA MARIA','UTE','ALDA','AMNA','EVA MARINA','FRANCISCA ROSARIO','LUZ ALBA','NAHID','NAJMA','ODILE','RAWAN','SIMONA ELENA','SNEZHANA','VIKTORIJA','DOLORES ANGELES','EKHIÑE','ILDIKO','JILL','JOHANNA MARIA','KATIXA','KATRINA','MARIA ROBLEDO','MARIE CHRISTINE','PATIENCE','RITA ISABEL','XIAOYING','ADRIANA LUCIA','CLORINDA','DOLLY','DOLORES JOSEFA','IQRA','MARIA PRECIOSA','MERCEDES ISABEL','RAMONA ELENA','ROCIO ANGELES','ROEYA','SARRA','TLAITMAS','ELISA CARMEN','JANIS','MARGHERITA','MARIA ARA','MARIA GRISELDA','MARTINA MARIA','PATRICIA BEATRIZ','ROSA ALBA','ROSA LIMA','SUSANA CRISTINA','AMANI','CARMEN JOSE','CASTA','CLAIRE LOUISE','DALAL','DANIELA CRISTINA','FARZANA','GLORIA CECILIA','JOSEFA ROCIO','JULIA VICTORIA','LILIT','MARIA ALCIRA','MARIA COROMOTO','MARIA NAZARETH','MARTA VICTORIA','NELY','QI','ROSA ELIZABETH','ROSARIO ANGELES','SILVIA SUSANA','SONIA BELEN','AMAYA MARIA','ANA FE','ANGELES PILAR','BARBARA ANN','CARMEN IRENE','CHRISTA','CRISTINA TERESA','IREA','JUANA MARGARITA','KARLA MARIA','MARIA CASTAÑAR','MARIA ILDA','MIREN IRUNE','MONTIEL','OLESIA','ROCIO CINTA','ROWENA','SANDRA VIVIANA','SOFIA BELEN','TOURIYA','VALERIA ALEJANDRA','ABUNDIA','ADELINA MARIA','ALICIA CRISTINA','AURORA PILAR','DIANA GABRIELA','FRANCISCA JOSE','IHSSANE','JUANA BELEN','KANZA','LETICIA CARMEN','LYNDA','MARIA OLAYA','MARIA VIÑAS','NANCY BEATRIZ','NARJIS','OLINDA','PABLA','QIAN','REPOSO','SILVIA EUGENIA','ALEJANDRA CARMEN','ALISSON','ALLEGRA','ALOMA','AMPARO PILAR','ANA LEONOR','CANDELARIA MARIA','CARMEN ESPERANZA','CARMEN NURIA','DOLORES CONCEPCION','ELENA MADALINA','GERVASIA','JIANFEN','KELTOUM','LESLEY','LUISA PILAR','MARIA LYDIA','MARIA MIRABELA','MARIA SEGUNDA','MARTA SOFIA','SANTOS','SCHEREZADE','SILVIA TERESA','YELENA','ADRIANA CRISTINA','ANA FELISA','ANTONIA ANA','ASHA','CARMEN AMPARO','CECILIA ISABEL','CYNTHIA MARIA','ERKUDEN','GABRIELLE','KHAWLA','MANPREET','MARIA LUIZA','MAYRA ELIZABETH','SHANDRA','SUCESO','ARLENE','CLAUDIA CECILIA','LUISA ANTONIA','MARIA CONSOL','MARIA ERICA','MARIA GUACIMARA','MARIFE','MARTHA BEATRIZ','MELIDA','MONTSERRAT MARIA','NICOLA JANE','PASTORA MARIA','PINO

MARIA','REGLA MARIA','SANDY','SARA BELEN','SIDRA','AIDEE','BLASINA','CELIA CARMEN','CRISTINA ELIZABETH','ESTEFANI','EVANGELISTA','GILDA','GRECIA','ILSE','ITTO','JULIA MERCEDES','JUSTINIANA','LORENA ELIZABETH','MARIA FANNY','MARIA SANTA','MARITZA ELIZABETH','MIREN BEGOÑE','MIREN SORKUNDE','PATRICIA CRISTINA','TAHIRA','VERONICA ALEJANDRA','YOANNA','ZAINEB','ADRIANA BEATRIZ','ANA ENCARNACION','ANCA MARIA','CAMILA ANDREA','CARMEN VANESA','CAROLINA ELIZABETH','CELTIA','CRISTINA ALEJANDRA','CRISTINA GABRIELA','DANIELE','EDILIA','ELENA RAMONA','EROLA','HASSANIA','IWONA','JOSIANE','KATHERIN','LAURA LUCIA','MARIA ALBERTA','MARIA APOLONIA','MERIEME','MIREN JOSEBE','NANCY MARIA','ODILA','PAULA SOFIA','PEDRONA','ROSA ESPERANZA','SOODIA','SUSAGNA','VANESSA CAROLINA','ZOILA ROSA','ALBA CRISTINA','ANUSKA','ELENA ROCIO','EXPEDITA','FLORENTA','ISABEL LUCIA','IXEIA','KATHERINE ELIZABETH','LAMYA','MARIA SIRA','MARTA ESTHER','MERI','MILAGROS CARMEN','MOULOUDA','MYRNA','NURA','NURIA BELEN','OHIANA','OLGA CRISTINA','STELIANA','SYLWIA','VALERIIA','ZULAY','ADORACION REYES','AFRA','ALFONSINA','ANA MIRIAM','ANA MONTSERRAT','ARAIA','CARMEN SONIA','ELISABEHT','ESTRELLA MAR','GABRIELA FERNANDA','JENICA','JERUSALEN','JUSTYNA','KSENIIA','LOLI','LUZ ESTELA','MARIA DALIA','MARIA PLACERES','MARIA SALADINA','NOEMY','PURIFICACION MARIA','QIN','RAZAN','ROSA AMALIA','SALSABIL','SCARLETT','SOFIA VALENTINA','XIAOJING','AMANE','BRIANDA','CATALINA CARMEN','CINTYA','FRANCISCA ROCIO','GENTZANE','HEBA','LEIDY','MARIA DUNIA','MARIEME','MERCEDES ROCIO','PATRICIA MARY','PETRONA','SIRIN','SOFIA PILAR','SOHAILA','STANISLAVA','STEPHANIA','SUZANA','TERESA ROSARIO','VAITIARE','YANELY','ANDRA','ANNAMARIA','CHEN','CONSUELO PILAR','DOREEN','ELORA','EVA DOLORES','FRANCA','FRANCISCA VICTORIA','JUANA CRUZ','LICINIA','MARIA CINTIA','MARIA CLAUSTRE','MARIA RENE','MARIA VENTURA','MARIAME','ORLANDA','PACITA','QIANQIAN','ROSA MARTA','SANDRA CAROLINA','SOKAINA','VIRGINIA CARMEN','ANATOLIA','BLANCA DELIA','ESTELITA','LAMYAA','LAURA VANESA','MARIA AMANDA','MARIA CLAUDINA','MARIA VINYET','MIRA','MIREN BAKARNE','NOARA','PURIFICACIO','SANDRA PAOLA','XANTAL','XIAOLING','ZULEICA','ADAYA','AKANE','ANGELA VICTORIA','DAHIANA','DIANDRA','FATIMETU','IBALLA','IOANA CRISTINA','KATY','MARIA JOAO','NIOBE','ROSA IRENE','RUKHSANA','SILVIA RAQUEL','SORAIDA','SUSANA PATRICIA','WISSAM','ANA SONIA','ANJARA','CARLA CRISTINA','CHAHRAZAD','CRIPTANA','FATINE','JAQUELIN','LUCIA ANGELES','LYNNE','MARIA ROSALBA','MARINA MARIA','MERCHE','ROSA INMACULADA','SATURIA','SILA','SILVIA ALEJANDRA','TAYRI','VALVANUZ','XIAOXIAO','YANYAN','ADOSINDA','AMANDEEP','COLETTE','EM MA ROSA','EXALTACION','FATIMA EZZAHRAE','IRADI','JEMIMA','JESSICA CARMEN','JIMENA MARIA','JOSEFA JESUS','MAI','MORAIMA','NICOLETTA','NIKA','PATRICIA ROCIO','THERESA','ULPIANA','AINE','BRISA','CARMEN ENCARNACION','CARMEN VERONICA','ESTEFANA','FRANCISCA LUISA','IHSANE','JENNIFER ANN','JIHAD','JUANA ROCIO','LISSETTE','LUCIA ELENA','LUISA MERCEDES','MARIA ARGELIA','MARIA GINESA','MARIE FRANCE','MONICA CECILIA','NANCY ROCIO','SAMINA','STEFANI','CASIANA','CRISTAL','ELENA CLAUDIA','ELMIRA','FLORENTINA MARIA','FRANCINE','GEMA ROCIO','GRETEL','HAZEL','HEIDY','INMACULADA ROCIO','LIQIN','MARIA ASCENSIO','MARIA CANTO','MARIA LIBRADA','MARIA PROVIDENCIA','MARY CRUZ','MELBA','OLGA ESTHER','RALITSA','REYNA','ANA IRIS','ANA MONICA','ANGUSTIA','ARMINE','BAKARNE','CLAUDIA LUCIA','DOMICIANA','ESTEFANIA MARIA','EVA ANGELINA','KETEVAN','LISETTE','MARGARET MARY','MARIA CAMELIA','MARIA

CAYETANA','MARIA MONTIEL','MARIA SATURNINA','MIRIAM ELIZABETH','NORMA MARIA','PAQUI','PUREZA','ROSE MARY','SANDRA MARCELA','UZMA','VALERIE ANN','ANDREA GABRIELA','ANDREEA IOANA','ATILANA','BLANCA ALICIA','FRANCISCA MERCEDES','HORIA','JOSEFA AMPARO','LEYRE MARIA','MARIA ALEXIA','MARIA ASTRID','MARIA LEOCADIA','MARIA PRUDENCIA','MARIANA ISABEL','MARIELLA','PAULA DANIELA','ROKHAYA','TESSA','XEILA','YANELIS','ANTONIA MERCEDES','BLANCA MARGARITA','CRISTINA MERCEDES','GRAZIELLA','IDANIA','IRAITZ','IUNE','JUANA ANGELES','LEOPOLDA','LIDA','LUISA ELENA','MAGDALENA SOFIA','MARIA BASILISA','MARIA EDURNE','MARIA GRAZIA','MARISSA','MARTA ALICIA','MIGDALIA','MIMUNT','NDEYE','OUISSAL','PATROCINIA','SINFORIANA','ZAHIDA','AMPARO CARMEN','ARBIA','ARIANA MARIA','ASTOU','AURORA ISABEL','ELENA LOREDANA','ELZA','HEATHER','HUDA','ISABEL CONCEPCION','LIYAN','LIZHEN','LORENA PATRICIA','MARIA LEYRE','NIKOLINA','OUARDIA','TANTA','XUE','YAISA','YUNE','ZORIONE','ANA DANIELA','ANA PAZ','BRISEIDA','CARLA DANIELA','CESARINA','DEOGRACIAS','GLORIA NANCY','HAIZENE','IVETA','JOANA AINA','LUCELLY','NATALIA PILAR','NIRMIN','PAULA LUCIA','SAHARA','SOFYA','UMAIMA','VESELINA','ADELAIDA MARIA','ALEXANDRA CRISTINA','ANGELA LUCIA','CLAUDIA GABRIELA','CLAUDINE','FATOUMATA BINTA','GIANINA','GICA','IHINTZA','IXEYA','LAURA CECILIA','LISA MARIE','LIYING','LUCIA CRISTINA','LYA','MARIA GLADIS','QING','THERESE','TIBURCIA','VERONICA PILAR','YOLANDA PILAR','ADELA CARMEN','ADELE','AILA','AINET','ANTONIA ROSARIO','CARLA ALEJANDRA','CARLA LORENA','CARMEN ANDREA','CAROLINA ALEJANDRA','CRISTINA ANA','CRISTINA BELEN','CRISTINA DANIELA','DIANA ALEJANDRA','GOIATZ','IRMA MARIA','KATHRIN','MARIA EMILIANA','MARIA HERMELINDA','MARIA LUDIVINA','MARIANGELA','MARICARMEN','MARINA VICTORIA','MIREN ARANTXA','MIREN ELIXABETE','SONIA MARGARITA','TING','VIOLA','XIAOHUA','AILIN','ANA ARACELI','ANDREA VALENTINA','AUBA','CARMEN CONSUELO','CAROLINA PILAR','DENIS','ELENA MERCEDES','EMAN','IERA','JANE ELIZABETH','JUDIT MARIA','JURGITA','KHATUNA','LALI','MARIA FIDELA','MARIA LEONIDES','MIKELE','MIREN ZURIÑE','NACIRA','NATALIA CAROLINA','NORA MARIA','NOUHA','ROSA MILAGROS','ROSA VIRGINIA','SANDRA ELENA','XI','YOVANNA','YULISA','ADIELA','BRIANA','DOLORES ENCARNACION','DOLORES FRANCISCA','DOMINIKA','ELENA ROXANA','ELENA TERESA','ELIONOR','ERMINDA','GLORIA TERESA','GRACIA PATRICIA','HEGOA','JOSEFA ANA','KAREN ANDREA','LUCIA MAR','LUCIA MERCEDES','MARIANA CARMEN','MARLENI','MELITONA','MERIDA','NOURIA','ORETO','RAQUEL PILAR','ROMAYSAE','ROSA ALICIA','VANESSA CARMEN','YASHIRA','YUMEI','ADRIANNA','ANA ESTER','ANA RAMONA','CORAL MARIA','ELIGIA','ESPERANZA CARMEN','ESPIRITU SANTO','ESTEFANIA CARMEN','FARNERS','GOIZEDER','JOSEFA ASUNCION','LAURA MARGARITA','MIHAELA GABRIELA','ROSA JOSEFA','BLANCA NUBIA','ELIANA MARIA','HASMIK','IA','IBANA','ISABEL ELENA','ISABEL PATRICIA','JOSEFA JUANA','KEXIN','LUZ AMERICA','MARIA YANIRA','MARTA INMACULADA','NATALIA CRISTINA','RAFFAELLA','SOLEDAD CARMEN','SUFEN','SUSAN JANE','VASILKA','XIAOQIN','XIUQIN','YAHAIRA','ADRIANA ELENA','ALESANDRA','BADIAA','BATOUL','CARMEN JUANA','DANIELA CAROLINA','DIDINA','DOLORES ROSA','DOLORES ROSARIO','ELENA ANDREEA','FRANCISCA ELENA','JOELLE','KYRA','LIZA','LUCIENE','MARIA DORIS','OLYMPIA','RAFIA','RAMATOULAYE','REYNA ISABEL','ROSA ENCARNACION','SAJIDA','SARA INES','TABATA','TATIANE','VARINIA','WEIFEN','XINXIN','ACORAIDA','CARLA ALEXANDRA','CLAUDIA ALEXANDRA','CLAUDIA VERONICA','DARIFA','HOURIYA','LAURA BELEN','MARIA ABEL','MARIA ALBERTINA','MARIA HERMITAS','MONICA

SUSANA','ONEKA','PAOLA ALEXANDRA','SANDRA ROCIO','SOMIA','XELA','XUEMEI','YOBANA','ALINA CRISTINA','ANA FATIMA','ANDRA MARIA','ARAI','AURINA','BASSMA','CELSA MARIA','CHAMA','CRISTINA ANGELES','ELENA DIANA','EMILIA PILAR','ERENIA','FABIOLA MARIA','HILARY','JESSICA ELIZABETH','JIAJIA','LEIRE MARIA','LESIA','LU','MANUELA ANGELES','MARIA DOROTEA','MARIA HILARIA','MARIA PASION','MARIA ROMINA','MIAOMIAO','MILICA','NACERA','NASREEN','NORALBA','PAULA ELENA','YANELA','ADOLFA','AMPARO ISABEL','ANGELA TERESA','AOUICHA','BEATRIZ CRISTINA','BENILDA','ELENA MARGARITA','EVA VICTORIA','FRANCISCA BELEN','JIA','JOSEFA LUCIA','MARIA ALCOR','MARIA ENMA','RITA CASSIA','ROSA ADELA','RUDESINDA','SARAYMA','SILVIA LORENA','VELICHKA','AMALIA ISABEL','ANA ABEL','ARGEME','BLANCA PILAR','BRIANNA','CAÑOS SANTOS','EDELIA','ETHEL','EVA ANGELES','FAVOUR','GLORIA CRISTINA','HILDEGARD','LAURA RAQUEL','MARIA BLANCA NIEVES','MARIA SALOBRAR','MILKA','NORHAN','NURIA MONTSERRAT','ROSA MAGDALENA','SILVIA CAROLINA','SOUHAYLA','STANKA','TISCAR','VIRGINIE','ANA AMALIA','ARANZAZU MARIA','BEATRIZ TERESA','BLANCA ROCIO','CAMILLE','CARMEN GABRIELA','CARMEN NOELIA','CONSUELO CARMEN','DOLORES MERCEDES','HERMELINA','ISABEL BEATRIZ','ISABEL LUISA','JENARA','JULIA CRISTINA','KIRSTEN','MARIA ANNA','MARIA TURA','MIREN MIRARI','NORMA ISABEL','ROSA LUCIA','VICENTA ISABEL','VIOLANTE','YADHIRA','YARISA','ALEX','ANA MARIA ANGELES','DOLORES TERESA','GLORIA STELLA','HUA','KENDRA','MARIA ADELIA','MARIA DORES','NATALIA LORENA','PATRICIA ANDREA','PATRICIA EUGENIA','PILAR TERESA','RONG','ROSARIO DOLORES','SARAH ELIZABETH','SHAYLA','SILVIA ELIZABETH','SUNITA','TERESA NIÑO JESUS','YARELI','ALBA PILAR','ANA VICENTA','ANDREA ALEXANDRA','BAHIA','CARLINA','CARMEN VANESSA','CELIA ISABEL','CHEYENNE','CHRISTINE ANNE','ELENA SIMONA','ERIKA PATRICIA','EVA RAQUEL','EVA ROCIO','GLADYS BEATRIZ','GRISEL','JOSEFA MARGARITA','JULIE ANNE','LAUDINA','LAURENCE','LEKBIRA','MARGA','MARIA ALCAZAR','MARIA CORONA','MIREILLE','MONICA LUCIA','NAIHARA','NESRIN','PATRICIA LORENA','PAULA TERESA','PENDA','RASA','SANDRA PILAR','SANDRA VERONICA','SOUMIYA','TANIA ISABEL','ANA VALERIA','ANNELIESE','BOZENA','CATHAISA','CRISTINA DOLORES','DAMARI','EDILMA','ELISAVETA','EVA TERESA','JUN','KHADIDIATOU','LARA ISABEL','LEI','LIUBA','LUCIA ESTHER','LUISA MARGARITA','MANDEEP','MARIA CLARISA','MARIA MARTHA','MARY JANE','MIHAELA ELENA','MIREN LOURDES','NATALIA ALEJANDRA','RAMONA CARMEN','ROCIO VALLE','SAINZA','SANDRA LUCIA','SILVIA NOEMI','ZORYANA','ALINA GABRIELA','ANA MARY','ANA TRINIDAD','ANDREA FERNANDA','ANDREEA GEORGIANA','ANTONIETTA','BATIRTZE','CARMEN GUADALUPE','CLAUDIA XIMENA','CRISTINA LUCIA','CRISTINA PATRICIA','EKRAM','FRANCESCA MARIA','GLORIA MARINA','GOHAR','HAYATE','INSSAF','JANET ELIZABETH','JIAYING','JULE','LEIDY TATIANA','LOVETH','LUCIA ROSARIO','MARGARET ELIZABETH','MARIA MARTIRIO','MIRIAM ROCIO','NA','NARINE','NOELIA ISABEL','PATRICIA MONICA','ROSA CLARA','ROSA VICENTA','RUI','SARA, SOFIA','SFIA','SIENNA','SUELI','SUSANA ELENA','ALEXANDRA PATRICIA','ANA ROSALIA','ANTONIA JOSE','ARIEL','ARWEN']

8.2.4.2 Lista nombres_hombre

nombres_hombre = ['ANTONIO','JOSE','MANUEL','FRANCISCO','DAVID','JUAN','JOSE ANTONIO','JAVIER','JOSE LUIS','DANIEL','FRANCISCO JAVIER','JESUS','CARLOS','ALEJANDRO','MIGUEL','JOSE MANUEL','RAFAEL','PEDRO','MIGUEL

ANGEL','ANGEL','PABLO','JOSE MARIA','FERNANDO','SERGIO','LUIS','JORGE','ALBERTO','JUAN CARLOS','ALVARO','JUAN JOSE','DIEGO','ADRIAN','RAUL','JUAN ANTONIO','IVAN','ENRIQUE','RUBEN','RAMON','VICENTE','OSCAR','ANDRES','JOAQUIN','JUAN MANUEL','SANTIAGO','EDUARDO','VICTOR','MARIO','ROBERTO','JAIME','FRANCISCO JOSE','IGNACIO','MARCOS','ALFONSO','JORDI','SALVADOR','RICARDO','EMILIO','HUGO','GUILLERMO','GABRIEL','JULIAN','JULIO','MARC','TOMAS','JOSE MIGUEL','GONZALO','AGUSTIN','MOHAMED','JOSE RAMON','FELIX','NICOLAS','JOAN','MARTIN','ISMAEL','CRISTIAN','SAMUEL','AITOR','JUAN FRANCISCO','JOSEP','HECTOR','MARIANO','DOMINGO','JOSE CARLOS','ALFREDO','SEBASTIAN','IKER','CESAR','FELIPE','ALEX','LUCAS','JOSE ANGEL','JOSE IGNACIO','VICTOR MANUEL','LUIS MIGUEL','RODRIGO','GREGORIO','JOSE FRANCISCO','JUAN LUIS','XAVIER','ALBERT','LORENZO','ESTEBAN','CRISTOBAL','ANTONIO JOSE','PAU','BORJA','ARTURO','MATEO','EUGENIO','AARON','JOSE JAVIER','JUAN MIGUEL','ANTONIO JESUS','JESUS MARIA','ISAAC','FRANCISCO MANUEL','JAUME','ERIC','GERMAN','JOEL','ASIER','VALENTIN','MOHAMMED','PEDRO JOSE','JONATHAN','ABEL','JOSE VICENTE','IZAN','MIKEL','MOISES','SERGI','JUAN RAMON','CHRISTIAN','UNAI','DARIO','ADOLFO','AHMED','JUAN PEDRO','IÑIGO','MANUEL JESUS','ISIDRO','BENITO','GERARD','ERNESTO','JON','BRUNO','BERNARDO','POL','MIQUEL','GERARDO','ISRAEL','OMAR','ANTONIO MANUEL','CARMELO','ORIOL','ARNAU','FEDERICO','PASCUAL','ELOY','FRANCESC','JONATAN','MARCO','JESUS MANUEL','JOSE ALBERTO','JUAN JESUS','BARTOLOME','JOSEP MARIA','LUIS ALBERTO','MARCELINO','ADRIA','IÑAKI','ROGER','FERMIN','PERE','BENJAMIN','ADAM','ELIAS','KEVIN','CARLES','ALONSO','LLUIS','ANTONI','PEDRO ANTONIO','AURELIO','MATIAS','JOSE ENRIQUE','JUAN PABLO','MARCO ANTONIO','GUILLEM','ANGEL LUIS','ANDER','MARTI','LEO','JUAN IGNACIO','CARLOS ALBERTO','SAUL','JACINTO','EUSEBIO','ALEXANDER','FRANCISCO JESUS','JERONIMO','OLIVER','VICTORIANO','ABRAHAM','XABIER','ERIK','ROMAN','GORKA','JOSE JUAN','GUSTAVO','MANUEL ANGEL','FERRAN','YOUSSEF','SAID','ISIDORO','LUIS FERNANDO','DAMIAN','ALEIX','LUIS MANUEL','CARLOS JAVIER','TEODORO','BLAS','JUAN BAUTISTA','DIONISIO','ENRIC','ARMANDO','PEDRO LUIS','CANDIDO','JULEN','JUSTO','SANTOS','FLORENCIO','LEONARDO','YERAY','GINES','RACHID','JOSE DAVID','JOSE JOAQUIN','JUAN MARIA','EDUARD','EMILIANO','JAN','SERAFIN','FRANCISCO ANTONIO','JAIRO','FAUSTINO','ENEKO','ALI','MAXIMO','EMILIO JOSE','EDGAR','JOSE ANDRES','JESUS ANGEL','LUIS ANTONIO','FIDEL','LUIS MARIA','JULIO CESAR','MUSTAPHA','RAMIRO','AMADOR','LUIS ANGEL','SIMON','CLAUDIO','ROGELIO','IGOR','JACOBO','ALEXIS','MANUEL ANTONIO','FRANCISCO MIGUEL','LUCIANO','ION','ALEXANDRE','EZEQUIEL','BIEL','PEDRO MANUEL','IAN','MODESTO','ILDEFONSO','ANDREU','JOAQUIM','JOSE FERNANDO','FLORENTINO','JUAN DIOS','NIL','JUAN VICENTE','HASSAN','JUAN DIEGO','JOSE DANIEL','MANUEL JOSE','PAULINO','YAGO','ANTONIO JAVIER','CARLOS MANUEL','LEANDRO','ANTONIO MIGUEL','CONSTANTINO','JUAN ANDRES','CELESTINO','AVELINO','ENZO','ELADIO','ANTONIO LUIS','LUIS JAVIER','CAYETANO','SATURNINO','JORGE LUIS','PEDRO MARIA','JOSUE','PEDRO JESUS','GHEORGHE','VASILE','LUIS CARLOS','IBAI','RUFINO','FABIAN','MAURO','IAGO','NESTOR','MANEL','JUAN ANGEL','FRANCESC XAVIER','ARITZ','MUHAMMAD','ANGEL MANUEL','ROBERT','GASPAR','KHALID','AIMAR','IMANOL','CLEMENTE','IOAN','BASILIO','RICARD','FELICIANO','MARCELO','RAIMUNDO','ABDELLAH','RAYAN','ANDONI','LUCA','HILARIO','MARKEL

','OIER','JOSU','EVARISTO','ABDELKADER','JUAN ALBERTO','JESUS ANTONIO','BRAIS','ANGEL MARIA','FERNANDO
JOSE','MARCIAL','BERNAT','JOSEBA','LAUREANO','HICHAM','ELISEO','BILAL','ROBERTO CARLOS','NARCISO','BENIGNO','FABIO','LUIS
ENRIQUE','ALAN','PATRICIO','BERNARDINO','CECILIO','MARCEL','AYOUB','DIDAC','HAMZA','SEGUNDO','CONSTANTIN','PEDRO PABLO','GAEL','BRAHIM','ANTONIO FRANCISCO','DYLAN','LEOPOLDO','FRANCISCO
LUIS','VALERIANO','ABDELAZIZ','CAMILO','DEMETRIO','MARIAN','MAURICIO','NOEL','ANTON','CESAREO','PEDRO
JAVIER','BRIAN','ADAN','ANASTASIO','MICHAEL','CIPRIANO','FLORIN','OCTAVIO','RODOLFO','ROQUE','CARLOS JOSE','PEDRO MIGUEL','LUCIO','NICOLAE','MANUEL ALEJANDRO','BIENVENIDO','BERNABE','FRANCISCO
BORJA','ABDERRAHIM','ALAIN','ANSELMO','JOSE FELIX','JUAN DAVID','ANAS','IBRAHIM','KARIM','JOSE GABRIEL','HAMID','BALTASAR','LAZARO','JOSE PEDRO','AMADEO','JESUS MIGUEL','PELAYO','FRANCISCO ASIS','MANUEL FRANCISCO','EULOGIO','JOSE RAFAEL','PLACIDO','LUIS FRANCISCO','TEOFILO','CASIMIRO','FULGENCIO','CELSO','IGNASI','ABDESLAM','ELOI','JUAN ENRIQUE','BALDOMERO','JOSE DOMINGO','JESUS JAVIER','PAUL','HERMINIO','ALBERTO JOSE','DENIS','MAXIMINO','YASSINE','JORGE JUAN','DRISS','PABLO JOSE','BEÑAT','MAX','JAMAL','VICTOR JOSE','VICENTE JOSE','ANIBAL','JOSE ALEJANDRO','VICENT','AQUILINO','MAMADOU','ALEXANDRU','JUAN
GABRIEL','ORLANDO','GUMERSINDO','NORBERTO','LLUC','LUIS ALFONSO','THIAGO','AXEL','JOSE PABLO','MARINO','IBON','BONIFACIO','JUAN FERNANDO','CARLOS ENRIQUE','VIDAL','MARCOS ANTONIO','VICTORINO','AZIZ','IBAN','ANTONIO MARIA','MELCHOR','HIPOLITO','JOSE TOMAS','NOE','MARIUS','MIHAI','SAMIR','ROSENDO','STEFAN','JOSE
JESUS','ELEUTERIO','ARSENIO','GUSTAVO ADOLFO','GAIZKA','ZAKARIA','ISMAIL','CARLOS ANTONIO','VIRGILIO','EDUARDO JOSE','HIGINIO','JOSE ALFONSO','MOUSSA','JOSE EMILIO','ALEJANDRO JOSE','NABIL','ANTONIO DAVID','UNAX','AYMAN','PEDRO ANGEL','URBANO','MIGUEL ANTONIO','MAXIMILIANO','ALESSANDRO','JOSE EDUARDO','YASSIN','LUIS EDUARDO','AUGUSTO','FRANCISCO RAMON','ISAIAS','SEVERINO','ENRIQUE JOSE','ANGEL JOSE','CEFERINO','VENANCIO','PEDRO JUAN','ABELARDO','ADIL','JESUS ALBERTO','ALEJO','JOSE JULIAN','KILIAN','NOAH','JOSE RAUL','NOUREDDINE','INOCENCIO','JOSE BENITO','DUMITRU','MANUEL MARIA','JORGE MANUEL','ARKAITZ','JUAN RAFAEL','JOSE AGUSTIN','DIEGO JOSE','LANDER','MIMOUN','TEO','YOUNES','ANDREI','ANXO','PRIMITIVO','JOSE
RICARDO','ENDIKA','JON ANDER','LUIS FELIPE','YOEL','FERNANDO JAVIER','JORGE ANTONIO','BAUTISTA','IONEL','ANGEL ANTONIO','GENARO','CARLOS JESUS','CARLOS ANDRES','EDER','BRAULIO','CONRADO','RICHARD','LUIS
JOSE','BRYAN','MOHAMMAD','NEMESIO','URKO','PRUDENCIO','VICTOR
HUGO','DELFIN','GENIS','AGAPITO','JUAN CRUZ','NACHO','LEONCIO','BELTRAN','JOSE JULIO','AGUSTI','EFREN','JACOB','SECUNDINO','ADAY','FAUSTO','HUMBERTO','INDALECIO','GIOVANNI','MOSTAFA','JULIO ALBERTO','JUAN SEBASTIAN','OLEKSANDR','ANTONIO ANGEL','PABLO ANTONIO','JOHN','OVIDIO','CARLOS
MIGUEL','ESTEVE','ABDELLATIF','LEON','ROC','ABDERRAHMAN','VLADIMIR','VOLODYMYR','AMBROSIO','WALID','YERAI','IBRAHIMA','PEDRO FRANCISCO','TONI','CESAR AUGUSTO','LLORENÇ','GABINO','LUIS MARIANO','ANTONIO RAMON','EKAITZ','ARTUR','ANICETO','YAHYA','ABDELILAH','ANGEL JESUS','VIOREL','JUAN ALFONSO','IONUT','RUBEN DARIO','PETER','FRANCISCO

DAVID','XAVI','SILVESTRE','MOURAD','NICANOR','ANDRES FELIPE','SALAH','JONAS','HERNAN','LINO','KAMAL','ADRIANO','RAFAEL ANTONIO','HELIODORO','DESIDERIO','CARLOS LUIS','SABINO','FRANCISCO PAULA','VICENTE JAVIER','LAHCEN','LUIS IGNACIO','JOSE GREGORIO','JULIO JOSE','SIXTO','MHAMED','HERMENEGILDO','SEVERIANO','HORACIO','ERNEST','JESUS DAVID','TARIK','VICENÇ','ANGEL FRANCISCO','JUAN DANIEL','JOSE MARTIN','AMIN','ESTANISLAO','ABDELKARIM','ACHRAF','HONORIO','CARLOS EDUARDO','DAMASO','JOSE ALFREDO','IMRAN','AMIR','ANTOLIN','RAFAEL ANGEL','NELSON','JUAN SALVADOR','VENTURA','EMMANUEL','RENE','ULISES','DIEGO JESUS','MIGUEL JOSE','OIHAN','RAFAEL JESUS','IGNACIO JAVIER','VICENTE MANUEL','TELMO','FRANCESCO','ILYAS','BENEDICTO','FOUAD','JORGE ALBERTO','JONAY','ROI','ALBINO','EUSTAQUIO','GEORGE','MANUEL LUIS','JESUS FRANCISCO','ABRAHAN','CELEDONIO','WENCESLAO','ANDREA','MARK','FERNANDO MANUEL','JUAN JAVIER','JOKIN','EL HASSAN','OSCAR LUIS','JOSEP LLUIS','SEBASTIA','EGOITZ','FARID','THOMAS','NARCIS','SILVERIO','CHRISTOPHER','FRANCISCO GABRIEL','OSCAR MANUEL','OUSSAMA','JOSE SALVADOR','SERVANDO','MANUEL ENRIQUE','RICARDO JOSE','MANUEL RAMON','OLEGARIO','WILLIAM','ARCADIO','ERICK','JOSE PASCUAL','GILBERTO','LUIS JESUS','MOUNIR','NICASIO','RAFAEL JOSE','AMADO','CARLOS MARIA','IGNACIO JOSE','JULIO ANTONIO','QUIM','ARGIMIRO','MUSTAFA','IMAD','VICTOR JAVIER','CARLOS DANIEL','EPIFANIO','FLORIAN','GIUSEPPE','DEREK','DIEGO FERNANDO','LIAM','ANGEL DAVID','ABDERRAHMANE','ANGEL MIGUEL','ANTONIO CARLOS','ANTONIO JUAN','DANEL','MOHAMED AMIN','AMINE','CARLOS FRANCISCO','VASYL','ABDELKRIM','JUSTINO','JOSE ROBERTO','PABLO JESUS','ILIAS','ABDOULAYE','ALFONSO JOSE','AMANCIO','ZACARIAS','BALBINO','JAOUAD','ABILIO','ABDELALI','AIRAM','ABDELGHANI','JORGE ENRIQUE','DONATO','ABDELHAK','FRANCISCO VICENTE','HODEI','PABLO MANUEL','CARLOS DAVID','FERNANDO JESUS','ALVARO JOSE','DIMAS','DIEGO MANUEL','ABDELMAJID','DANIEL ALEJANDRO','DANIEL JESUS','COSTEL','JUAN DOMINGO','ALEJANDRO MANUEL','ANTONIO VICENTE','JESUS RAMON','CRESCENCIO','ANDRE','REMIGIO','SOUFIANE','VICTORIO','MANUEL VICENTE','JESUS JOSE','ENRIQUE MANUEL','LUIS DAVID','GIL','BUENAVENTURA','ETHAN','ANDRIY','JUAN ESTEBAN','ARIEL','LUIS ALFREDO','LUIS ALEJANDRO','GUZMAN','JESUS IGNACIO','AURELIANO','ANGEL JAVIER','FRANCISCO RAFAEL','JOSE SANTIAGO','MARIN','ANTONINO','BARTOMEU','PERFECTO','PETRU','PATRICK','FRANCISCO JUAN','ROMUALDO','KOLDO','MAHAMADOU','JESUS CARLOS','SANDRO','ANTONIO RAFAEL','LUIS RAMON','EKAIN','XOEL','MATTEO','MIQUEL ANGEL','SAAD','IULIAN','MANUEL FERNANDO','JOSE EUGENIO','ANTHONY','FRANCISCO DANIEL','ANDRES JESUS','COSME','JOSE IVAN','AKRAM','JOSE JORGE','SANTIAGO JOSE','ARAN','SAMI','FERNANDO LUIS','JUAN AGUSTIN','TIMOTEO','JUAN FELIPE','ALFONSO CARLOS','ABDELHAMID','ANER','DAN','LAURENTINO','OUSMANE','ABDESSAMAD','ALBERTO MANUEL','MANUEL ALBERTO','STEFANO','ANIOL','JOAN CARLES','ENRIQUE JAVIER','JUAN MARCOS','WALTER','VIKTOR','ANOUAR','LUIS VICENTE','RAFEL','JOSE GUILLERMO','MIGUEL MARIA','JOSHUA','CIRILO','MATEU','FRANCISCO ANGEL','PABLO JAVIER','MARÇAL','PAOLO','JOSE MARIO','MANUEL CARLOS','AMALIO','JESUS LUIS','OLEG','SORIN','FRANK','MIGUEL JESUS','ROLANDO','JOAN JOSEP','REDA','OSCAR JAVIER','ROMEO','BLAI','DOROTEO','JUAN MARTIN','EMETERIO','EMIL','MICHEL','PAULO','MOUSTAPHA','KEPA','SERGIO MANUEL','JOSE FELIPE','YOUNESS','BRAYAN','REYES','DAVID JOSE','RAMON JOSE','JUAN

CAMILO','GEORGI','JOSE DIEGO','JUAN TOMAS','JOSEP ANTONI','AMABLE','JOSE DAMIAN','LISARDO','NIZAR','CRISTOFER','DANIEL JOSE','ALBERTO JESUS','JUAN EMILIO','DIEGO ANTONIO','DENNIS','HERIBERTO','FERNANDO ANTONIO','AUREL','EDELMIRO','TRISTAN','PAVEL','ABDELHADI','JULIO MANUEL','ONOFRE','MYKOLA','ARON','AMADOU','ALBERTO JAVIER','NIKOLAY','JOSE JAIME','DIEGO ALEJANDRO','AMAR','GUILLERMO JOSE','MANUEL DAVID','FACUNDO','JOSEP ORIOL','IGNACIO MARIA','NATALIO','JOSE MARIANO','ENOL','FRUCTUOSO','NEIZAN','DAVID JESUS','EL MOSTAFA','SENEN','RAYCO','MARCIANO','EVELIO','SERGIO JOSE','LUIS DANIEL','ANDRES MANUEL','FREDERIC','ILIE','JESUS FERNANDO','MOHAMED AMINE','CHEIKH','JOSE ESTEBAN','ROBERTO JOSE','PEPE','MACARIO','SERGIO ANTONIO','CIRIACO','JUAN FELIX','FRANCISCO TOMAS','MAXIM','FRANCISCO ALBERTO','JORDAN','TIAGO','LUKA','MIGUEL FRANCISCO','JOSE LORENZO','CALIXTO','EMANUEL','ANGELO','JAVIER ANTONIO','JESUS ENRIQUE','UBALDO','RAMON ANTONIO','YURIY','PEIO','IOSU','AMANDO','FRANCISCO ANDRES','PATXI','JAMES','CARLOS FERNANDO','EDUARDO MANUEL','JESUS SALVADOR','JOAQUIN MANUEL','JOSE VICTOR','LARBI','SERGEY','ANTONIO JOAQUIN','NICO','CATALIN','SALUSTIANO','URIEL','ALEJANDRO JESUS','ALFONSO JAVIER','BORIS','PEDRO VICENTE','PEDRO ENRIQUE','REGINO','JOAN ANTONI','JOAQUIN JOSE','JORGE JAVIER','CRISTHIAN','HARITZ','MANUEL IGNACIO','ANGEL RAMON','ANASS','STEVEN','AMETS','MANUEL MIGUEL','DAVID MANUEL','RAFAEL MANUEL','JOSE RUBEN','VICENTE RAMON','ADAMA','JOSE BERNARDO','ALEJANDRO JAVIER','MANUEL JAVIER','BOGDAN','EMILI','ISIDRE','ANDRES JOSE','LADISLAO','YASSIR','JOSE MARCOS','JAVIER JESUS','NAIM','EUDALD','JESUS VICENTE','ANTONIO ENRIQUE','GARIKOITZ','OTMAN','DAVID ALEJANDRO','JOSEPH','JUAN ALEJANDRO','ANGEL GABRIEL','EDWIN','REDOUANE','VICTOR JESUS','EMILIO JESUS','JOSE ADRIAN','DOMINGO JOSE','RUPERTO','ATANASIO','FORTUNATO','PORFIRIO','GERVASIO','NAZARIO','CASTO','CRISTIN O','DMYTRO','ADNAN','JAVIER MARIA','LUIS ANDRES','PEDRO IGNACIO','ABDELHAFID','EULALIO','RAIMON','HENRY','MARIO ALBERTO','JORGE ANDRES','SOUFIAN','CUSTODIO','AYOZE','PEDRO RAMON','ISSAM','REINALDO','CORNEL','IHOR','JAVIER IGNACIO','JORGE MIGUEL','RAFAEL FRANCISCO','CASTOR','EFRAIN','ALFONS','ABDELHAKIM','TORCUATO','ANDREY','MASSIMO','HE CTOR MANUEL','ABDUL','ANTONIO FERNANDO','SALIM','INOCENTE','NAUZET','ASENSIO','BELARMINO','MOUAD','MYKHAYLO','ME HDI','VICENTE ANTONIO','FELICISIMO','PEDRO ALBERTO','PABLO LUIS','VICTOR MIGUEL','CIRO','ALFRED','YASSER','FRANCISCO JOAQUIN','JAVIER JOSE','OSCAR JOSE','DAVID ANTONIO','JEAN PIERRE','RAMON MARIA','EMILIO MANUEL','JOAQUIN ANTONIO','MARIO JOSE','MUSA','ALLAL','DAVIDE','PEDRO DAVID','JORGE JOSE','JUAN LORENZO','VICTOR ANTONIO','TELESFORO','FRANCISCO CARLOS','MAGIN','RICARDO MANUEL','URTZI','ARATZ','SIMEON','MIGUEL ALEJANDRO','PABLO ANDRES','REDOUAN','BADR','FELIX MANUEL','ARES','MAIKEL','BOSCO','YASIN','ALFONSO MANUEL','DANIEL ANTONIO','VICENTE JESUS','DANIELE','LUCIAN','TORIBIO','CRUZ','EDMUNDO','JOFRE','ENAITZ','FRANCISCO ALEJANDRO','JOSEBA ANDONI','MILOUD','VICENTE LUIS','SILVINO','JOSE CARMELO','WILSON','CARLOS GUSTAVO','OUMAR','RAFAEL CARLOS','ALVARO JESUS','MICHELE','RONALD','JORGE EDUARDO','JOSE OSCAR','MIHAIL','MIRCEA','ENRIQUE JESUS','ALVAR','MANUEL ANDRES','ZEUS','ELIO','HAKIM','JAIME JOSE','IVAN JOSE','RUSLAN','EDUARDO ANTONIO','HAFID','ATILANO','JORGE DAVID','NOUR EDDINE','MODOU','ALFREDO JOSE','MANEX','KHALIL','DOMINGO ANTONIO','OSCAR

DAVID','DANUT','ERLANTZ','JESUS DANIEL','ANDREAS','JAVIER ALEJANDRO','JOSE ARTURO','LUKAS','MILLAN','TAREK','LOIS','JEREMY','AHMAD','FRANCISCO ENRIQUE','LAMINE','JOSE AURELIO','FERNANDO MARIA','JUAN CRISTOBAL','JUAN EDUARDO','MORAD','NAHUEL','CARLOS ALFONSO','DAMIA','MOHAMED ALI','YONATAN','EDUARDO JAVIER','VICENTE MIGUEL','VALERIO','ANTONIO ALBERTO','DIMITAR','NADIR','XIAN','FELIX JOSE','RAFAEL LUIS','FELIX ANGEL','JHONATAN','HASSANE','LEOCADIO','AIMAN','CARLO','VICTOR DANIEL','ABDERRAZAK','ANDREW','ABDON','ANGEL ALBERTO','SAMBA','ABDOU','ALEXANDRO','OSKAR','LIVIU','FRANCO','JHON JAIRO','JORGE FRANCISCO','JOAQUIN MARIA','DANI','RAMON JESUS','SALVATORE','BAKARY','CASIANO','ANWAR','JUAN SANTIAGO','JENARO','JOSEP RAMON','TIRSO','RICARDO ANTONIO','QUINTIN','SERHIY','KAI','LUIS GABRIEL','LUKEN','EL MUSTAPHA','OLEH','ELIAN','EUGEN','SOULEYMANE','CRISTIAN DAVID','CONSTANCIO','EDUARDO JESUS','JORGE CARLOS','JOAN MANUEL','ALVARO MANUEL','FERNANDO MIGUEL','JEAN CARLOS','CESAR MANUEL','JORGE DANIEL','FRANCISCO MARIA','MARWAN','ENRIQUE ANTONIO','FRANCISCO FERNANDO','RAFAEL MARIA','MIGUEL LUIS','ANTONIO DANIEL','PEDRO ANDRES','JOAN RAMON','MBAREK','MOHAMED REDA','EL BACHIR','JUSTINIANO','BOUBACAR','JORGE ALEJANDRO','JOSE NICOLAS','MARCOS JOSE','CRISTOPHER','ORENCIO','AYTHAMI','LUIS GUILLERMO','MAGI','SANTIAGO MANUEL','MANUEL ALFONSO','RAMON MANUEL','IVAN JESUS','RUYMAN','AINGERU','DANTE','EÑAUT','ALBERTO CARLOS','CARLOS ARTURO','CARLOS ALEJANDRO','MHAMMED','GRACIANO','JAVIER FRANCISCO','OSCAR ANTONIO','JUAN JULIAN','RENATO','RICARDO JAVIER','JEREMIAS','LUIGI','NICOLA','ALEXEY','ANGEL DANIEL','CESAR ANTONIO','MARCELIANO','BACHIR','FILIBERTO','JUAN BOSCO','JUAN GUILLERMO','FRANCISCO JULIAN','PETRE','FELIX ANTONIO','RAMON LUIS','AITZOL','FAISAL','JAIME ANTONIO','FRAN','ELEAZAR','MIGUEL MANUEL','ENMANUEL','PABLO MIGUEL','MICHAEL JOHN','ALEJANDRO ANTONIO','KENNETH','DIEGO MIGUEL','VICENTE JUAN','VITALIY','ANGEL CARLOS','ALFONSO JESUS','JOSE ARMANDO','JOSE FERMIN','PEDRO JOAQUIN','GABRIEL ANGEL','GERONIMO','MOSTAPHA','FROILAN','JESUS ANDRES','RESTITUTO','SEVERO','IGNACIO MANUEL','JAGOBA','OSCAR JESUS','SILVIO','ADRIEL','CIPRIAN','VICENTE FRANCISCO','OSAMA','ANDY','JULIA','STEPHEN','YAROSLAV','ANGEL VICENTE','MARIANO JOSE','NEFTALI','PIO','CARLOS VICENTE','ARTEM','ISSA','MANUEL JOAQUIN','JOAN MARC','JOHN JAIRO','SIMONE','MIKHAIL','NATANAEL','ANGEL FERNANDO','MANUEL REYES','POLICARPO','ANTONIO GABRIEL','JOSE ALVARO','EUTIMIO','GALDER','LIONEL','YOUSEF','JOSE ERNESTO','TEODOR','VALENTINO','JAWAD','AMARO','JOSEBA IÑAKI','GRIGORE','ADEL','ARMAND','JOAN MIQUEL','JULIAN ANDRES','SERGIO DAVID','LUIS SANTIAGO','MANUEL JUAN','ADELINO','ANTONIO ANDRES','MAHMOUD','PEDRO ALFONSO','AISSA','CARLOS ALFREDO','IGNACIO JESUS','JAVIER ANGEL','ADONAY','GABRIEL ALEJANDRO','JOSE CRISTOBAL','TAOUFIK','ALDO','JOAN FRANCESC','ZAID','MAXIMIANO','VALENTI','NATHAN','RIDA','JOSE AUGUSTO','SALVADOR JOSE','JOAN MARIA','YUSEF','CARLOS IGNACIO','MANOLO','ZIGOR','DANIEL ALBERTO','LUIS GONZALO','GENEROSO','ALBERTO ANTONIO','EUGENI','DOMINGO JESUS','IRAITZ','VINCENZO','JOSE ROMAN','MAROUANE','SERGIY','ALIOU','DIEGO FRANCISCO','PEDRO CARLOS','PERU','MARTIN JOSE','NICOMEDES','RAMSES','STANISLAV','CESC','DAVID JOHN','PEDRO DANIEL','DANILO','JOSE ADOLFO','PABLO DANIEL','EL MEHDI','ANGEL ANDRES','ANTONIO PEDRO','EL

HASSANE','HECTOR JOSE','ANTONIO ALEJANDRO','RADOUANE','ABDELMALEK','SERGIO JESUS','CRISANTO','DOMENEC','KHALED','LEONEL','GALO','GIANLUCA','JOAO','EBRIMA','CARLOS RAMON','LUIS EMILIO','BABACAR','EDWARD','EUFRASIO','JOSE ISMAEL','ROBERTO ANTONIO','JOSE ELIAS','JOSE ORIOL','OVIDIU','SEGISMUNDO','PABLO FRANCISCO','GUIDO','JOSE ABEL','JUAN RICARDO','NIKITA','PEP','ABDALLAH','DIEGO ANDRES','DIEGO ARMANDO','FLAVIO','IORITZ','CARLOS ANGEL','BREIXO','ANDRES FRANCISCO','JAVIER MANUEL','SALAH EDDINE','EL MILOUD','JESUS ALEJANDRO','JORGE JESUS','OSWALDO','IVO','RADU','MARCOS JESUS','ANGEL CUSTODIO','PETRO','WEI','LOPE','ADRIAN JOSE','APOLINAR','OLIVIER','JESUS RAFAEL','CARLOS RAFAEL','JAIME MANUEL','TEODOMIRO','FELIPE JOSE','LUIS ROBERTO','MARCOS MANUEL','YASIR','ABDELOUAHID','EL HABIB','LAURENTIU','FRANCISCO IGNACIO','DIEGO JAVIER','EDUARDO LUIS','RAMON ANGEL','PETRICA','ABDELOUAHED','ANCOR','ELVIS','ANGEL RAFAEL','ENRIQUE LUIS','FABRIZIO','FRANCIS','PABLO ALBERTO','ALBERTO LUIS','SANTIAGO JESUS','PEDRO ALEJANDRO','EMILIO JAVIER','SERGIU','ELIGIO','FREDDY','YURI','DOREL','MANUEL EDUARDO','UGAITZ','JOSE GERMAN','MARIO ANTONIO','CLAUDIU','JEAN PAUL','LUIS PEDRO','MANUEL SANTIAGO','VICTOR LUIS','JORGE IVAN','ILYASS','MARIO JESUS','CARLOS ROBERTO','FRANCISCO PEDRO','JON ANDONI','ANTONIO TOMAS','TEODOSIO','IZEI','LUCINIO','TANAUSU','FRANCISCO SALVADOR','ALEJANDRO MIGUEL','MOKHTAR','OUALID','BOUJEMAA','EROS','LUIS RAFAEL','OSCAR ALBERTO','LIBERTO','MBAYE','RICCARDO','ARTURO JOSE','DAYRON','ARNALDO','MOR','ANDERSON','RICARDO JESUS','ALEJANDRO DAVID','OT','ADRIAN JESUS','LUIS MARIO','SANTI','BRANDON','MIGUEL RAMON','JUAN GREGORIO','KRASIMIR','MAROUAN','TARIQ','NICUSOR','ANDRES MIGUEL','JOSE MATIAS','LUIS JORGE','CARLOS AUGUSTO','ZOILO','WASSIM','HONORATO','IOSIF','NAJIM','SOTERO','CRISTO MANUEL','ALEKSANDAR','BENAISSA','ULPIANO','ALEKSANDR','DIEGO LUIS','GOTZON','MIGUEL ENRIQUE','CARLOS IVAN','MAKSYM','MELQUIADES','YAEL','DAOUDA','IVAN MANUEL','ISABELO','MANUEL RAFAEL','SALOMON','MIGUEL CARLOS','MISAEL','PEDRO RAFAEL','NEL','TOMAS JOSE','PEDRO FERNANDO','ROBERTO JAVIER','WILFREDO','GABRIEL ANTONIO','JOSE CRUZ','RAMON FRANCISCO','VLADISLAV','NAJIB','MARCO VINICIO','ODEI','EUTIQUIO','GIORGIO','LUIS ERNESTO','JOAQUIN JESUS','RAFAEL JUAN','YONATHAN','ZOUHAIR','ABDERRAZZAK','RAFAEL ALBERTO','LAHBIB','ANDRII','JULIO ENRIQUE','LIHER','MANUEL PEDRO','RAFAEL MIGUEL','ROBERTO MANUEL','DEMBA','GABRIEL JOSE','MIROSLAV','YON','DANIEL EDUARDO','JALAL','JORGE IGNACIO','DANIEL ANGEL','DMITRY','MOHAMED SAID','OSVALDO','FEDERICO JOSE','JOSE GERARDO','LAMIN','TXOMIN','CRISTIAN JOSE','DANIEL FRANCISCO','EMILIO ANTONIO','GONZALO JOSE','LUIS GONZAGA','MAURIZIO','SERGIO LUIS','DOMINGO JAVIER','ALFONSO LUIS','ANGEL ENRIQUE','DORIAN','PETAR','SUFIAN','RAUL ANTONIO','ANTONIO DOMINGO','BOUCHAIB','EKHI','CARLOS JUAN','GIORGI','MIGUEL JUAN','CHARLES','JOSEP MIQUEL','RABAH','SILVANO','JULIO ANGEL','MANUEL SALVADOR','ENRIQUE FRANCISCO','GUIM','JOSE BLAS','PABLO ENRIQUE','TAHA','ARTEMIO','JOAN MANEL','MOUHAMED','CRISTIAN JESUS','ENRICO','RAFAEL ENRIQUE','SHAHID','ELISARDO','JULIO MIGUEL','SANTIAGO JAVIER','TIBURCIO','VICTOR ANDRES','DANIEL ENRIQUE','OLEGUER','DEOGRACIAS','DIEGO ALBERTO','JOSE VALENTIN','VICTOR ALFONSO','PLAMEN','ROBERTO JESUS','DOMINGO MANUEL','JUAN NICOLAS','EIDAN','OTHMAN','PABLO ALEJANDRO','ABIAN','CARLOS GABRIEL','PABLO CESAR','JULIAN JOSE','VALERO','ANTONIO IGNACIO','EUSTASIO','LUIS PABLO','MOULOUD','PATROCINIO','FRANCISCO EDUARDO','JOAQUIN FRANCISCO','APOLONIO','CHERKAOUI','HAO','JUAN ROBERTO','SERAPIO','AMADEU','JIE','JUAN

JORGE','MARCOS JAVIER','PEDRO TOMAS','JOSE SEBASTIAN','MANUEL EMILIO','DELMIRO','CESAR LUIS','ERNESTO JOSE','MIGUEL ANDRES','RAYANE','STEPAN','OINATZ','CRISTOBAL JESUS','PIETRO','ALVARO ANTONIO','ANTONIO SALVADOR','BERNARDO JOSE','JAVIER ENRIQUE','RUBEN JESUS','GABRIELE','NILO','RUBEN JOSE','HRISTO','NECULAI','ANDRES DAVID','MANUEL AGUSTIN','EDUARDO FRANCISCO','BOHDAN','PAULO JORGE','ANDRES ANTONIO','ARIDANE','JAVIER LUIS','TARAS','JULI','OSCAR EDUARDO','ADOLFO JOSE','EDORTA','JOSE GONZALO','ARISTIDES','PEDRO JULIAN','SMAIL','GABRIEL JESUS','JAIME LUIS','JORGE RAMON','SERGIO ANDRES','ABDENBI','EL MOKHTAR','ABDELJALIL','HATIM','COSMIN','JESUS ALFONSO','JESUS JUAN','JESUS PEDRO','NICOLAU','ALVARO JAVIER','OSCAR DANIEL','XOAN','JESUS EDUARDO','RIGOBERTO','LUIS RICARDO','ANTONIO EDUARDO','ENRIQUE MIGUEL','FELIX JAVIER','ALIN','CRISTIAN ANDRES','FRANCISCO AGUSTIN','LUIS ARMANDO','MIGUEL ALBERTO','LEOVIGILDO','SERGIO JAVIER','ADUR','ALFONSO MARIA','CURRO','GUILLERMO ANTONIO','HERNANDO','JACINT','MAJID','SIRO','TONY','ANDRES AVELINO','ANTONIO ALFONSO','EUGENIO JOSE','FELIP','JOSE ISRAEL','JUAN BERNARDO','SERGIO DANIEL','HERMOGENES','VICTOR ALEJANDRO','YORDAN','CARLOS HUMBERTO','FRANCISCO RAUL','JASON','VALERIY','VASIL','FLORENTIN','KEVIN ALEXANDER','ANTONIO AGUSTIN','JULIAN ANTONIO','PABLO DAVID','GIANCARLO','JOSE ALEXIS','DANIEL ANDRES','IVAYLO','MASSIMILIANO','BREOGAN','FRANCISCO XAVIER','MARKO','ABDELKHALEK','FRANCESC JOSEP','JESUS JOAQUIN','TAHAR','ANUAR','AZEDDINE','JAVIER ANDRES','RAYMOND','TAYEB','FRANCISCO ALFONSO','TOMAS JESUS','ANGEL EDUARDO','FRUTOS','OLIVERIO','VLADYSLAV','ASER','ROLAND','ELIEZER','JULIO FRANCISCO','PABLO NICOLAS','DORIN','EDUARDO MIGUEL','HARON','FELIX JESUS','JON IÑAKI','JULIO JAVIER','LORIEN','RAUL JOSE','AIDAN','TOBIAS','TOMEU','JOAQUIN MIGUEL','EIDEN','JUN','RAFAEL VICENTE','EVAN','TRINITARIO','VICTOR FRANCISCO','ZAKARIAE','ALESSIO','BRIGIDO','AGUSTIN MANUEL','ANATOLIY','JULIO JESUS','JUAN CARMELO','KADDOUR','DOMICIANO','KERMAN','MIGUEL JAVIER','NEIL','ANDRES JAVIER','AYMANE','HEGOI','ROY','ALADINO','BIBIANO','BOUAZZA','ADALBERTO','CRISTIAN MANUEL','DOMENICO','KHALIFA','ABUNDIO','BAUDILIO','CRISTO','EL HOUSSINE','PAVLO','ALBERTO MIGUEL','IU','JUAN ALONSO','LIER','JOSE RAMIRO','JUAN VICTOR','ARIAN','JUAN BENITO','SERGIO MIGUEL','MANUEL DOMINGO','PIERRE','ALEXEI','JAIME ALBERTO','MIGUEL VICENTE','ASTERIO','CARLOS JULIO','RAFAEL DAVID','SERGIO ALEJANDRO','VALERIU','CAYO','FERNANDO DAVID','LUIS GERARDO','JOSE SANTOS','NUNO','RAFAEL ALEJANDRO','JORGE FERNANDO','ASSANE','HABIB','JOSEP MANEL','KHADIM','NELU','TODOR','ANGEL IGNACIO','MIGUEL IGNACIO','JUAN RAUL','VICTOR DAVID','CARLOS MARIO','ALEJANDRO DANIEL','CRISTOBAL JOSE','JOAQUIN VICENTE','LUIS ARTURO','ROBIN','HOUSSAM','MELITON','JON MIKEL','TRAIAN','YI','JOAQUIN ANGEL','VICTOR ANGEL','MOHAMAD','NICU','ANACLETO','ARAM','EL MAHDI','JOSE LEONARDO','ABDELMALIK','ANTONIO JORGE','CRISTIAN CAMILO','DANIEL FERNANDO','LAKHDAR','ALEJANDRO LUIS','EUFEMIO','FERNANDO FRANCISCO','ONESIMO','ABDELOUAHAB','DAVID FRANCISCO','KRISTIAN','PEDRO AGUSTIN','XURXO','YOUCEF','CARMELO JESUS','DAVID MIGUEL','YAYA','ELIECER','FRANCISCO PABLO','IGNACIO ANTONIO','JESUS GABRIEL','KOLDOBIKA','LUIS HUMBERTO','OTHMANE','PABLO IGNACIO','ROBUSTIANO','NIKOLA','SENDOA','USMAN','ABDULLAH','JESUS PABLO','XABI','HECTOR FABIO','ISHAK','BELKACEM','JUAN ALFREDO','LEONARD','PABLO

ANGEL','ANTONIO PABLO','OBDULIO','JOSE AMADOR','JUAN ISIDRO','MALICK','OSCAR MIGUEL','PABLO MARIA','SANDALIO','AMJAD','GHULAM','RICARDO LUIS','SINFORIANO','JULIAN MANUEL','NASSIM','HAMED','OCTAVI','RAUL ALBERTO','ANGEL PEDRO','JESUS MARIANO','KEVIN ANDRES','XOSE','ALIPIO','GUILLERMO MANUEL','OLMO','EL HOUCINE','FELIPE ANTONIO','FRANKLIN','THEO','TRINIDAD','JUAN EUGENIO','AIRAN','CESAR JOSE','DOSITEO','PHILIPPE','ANGEL TOMAS','OSCAR ANDRES','CRISTIAN DANIEL','MANUEL TOMAS','REYNALDO','ANTONIO JULIAN','ISLAM','ABDELFATTAH','ANDRES EDUARDO','ANDRES MARIA','CARLOS FELIPE','ESAU','KONSTANTIN','MATHIAS','MATTHEW','AGUSTIN JOSE','JOSE GINES','JUAN MARIANO','PATXI XABIER','SEKOU','SOFIAN','SOHAIB','ANGEL JUAN','JOSE ISIDRO','NORDIN','PERE
JOAN','SALMAN','VADIM','ABID','LAUDELINO','MAHER','BERNARD','PACO','BOUSSELHAM','COS TICA','MARIO LUIS','PHILIP','CRISTINEL','DENYS','JAIME JESUS','JOSE ELOY','NORMAN','RUBEN ANTONIO','EMILIO FRANCISCO','MARCO AURELIO','FERNANDO CARLOS','FRANCISCO SANTIAGO','ALVARO LUIS','XUBAN','AIUR','GHEORGHITA','JERONI','MOHAMMED AMINE','BABA','EDUARDO ENRIQUE','JEAN','JOHN ALEXANDER','NATAN','SLIMANE','TUDOR','ESTEBAN JOSE','GAUDENCIO','JOSE AVELINO','XABAT','DANIEL
ALEXANDER','LEONIDES','MILAN','ALBANO','ANARTZ','BELISARIO','TOMAS
MANUEL','RYAN','ANDRES LUIS','MANSOUR','VICTOR RAMON','BENET','JAIME ANDRES','WAEL','ARTHUR','JEFFERSON','KAMEL','TOMAS ANTONIO','IYAD','JUNIOR','MANUEL FELIPE','ADEI','GUIU','IYAN','TITO','CARLOS GUILLERMO','JAVIER CARLOS','AXIER','CARLOS EMILIO','DAVID FERNANDO','SALVADOR JESUS','VINCENT','ZENON','JESUS EMILIO','LUCCA','RICARDO ANDRES','ABDESSELAM','DIEGO MARTIN','ISIDRO JOSE','CARLOS HUGO','CRISPIN','CESAR JAVIER','JAIME FRANCISCO','CARLOS RAUL','DJAMEL','FRANCISCO DOMINGO','JESUS SANTIAGO','JUAN LUCAS','MIGUEL EDUARDO','NIKOLAS','ALEN','JAVIER ALBERTO','DAILOS','JOSE RODRIGO','ALEJANDRO FRANCISCO','AYOSE','DARIUS','OTMANE','ALFONSO MIGUEL','FILIP','MANUEL RICARDO','ANDRES MAURICIO','BOUZEKRI','ENRIQUE CARLOS','GUILLERMO JESUS','ALFREDO MANUEL','ISABELINO','SULAYMAN','DJIBRIL','GREGORIO
JOSE','HAROLD','AGER','JARED','CARMELO JOSE','FABRICIANO','FERNANDO DANIEL','KARAM','VADYM','WASIM','ANTONIO SANTIAGO','NORBERT','WOLFGANG','CARLOS JOAQUIN','LEONID','NASIR','SOFIANE','FELIPE MANUEL','GERSON','JOHAN','LUIS JUAN','PETER JOHN','SATURIO','STELIAN','CHRISTIAN DAVID','MIGUEL FERNANDO','SALIOU','FILIPPO','SANCHO','JOSE FIDEL','JOSEP FRANCESC','VICENS','JESUS TOMAS','JULIO LUIS','MARKUS','JOSE ALEXANDER','KIMETZ','GASTON','TOUFIK','LUIS TOMAS','OTTO','ANGEL PABLO','ANGEL SANTIAGO','DANNY','ECHEDEY','JOAN LLUIS','RAFA','SANTIAGO ANTONIO','FELIX MARIA','FRANCISCO SOLANO','MANUEL PABLO','RAFIK','AZZEDDINE','JORGE VICENTE','HERACLIO','HERMES','LORENZO JOSE','FERNANDO ENRIQUE','JORGE ELIECER','NADAL','RUFO','JON JOSEBA','XIMO','ELPIDIO','FENG','JAVIER EDUARDO','JONATHAN DAVID','LUIS JOAQUIN','OTILIO','ANDRES FERNANDO','IDELFONSO','CONSTANTE','JUAN MATIAS','LAMBERTO','MATTIA','ALBERTO FRANCISCO','BENITO JOSE','LUIS DOMINGO','SIMION','YACINE','ARMANDO JOSE','DAVID ALEXANDER','EVGENY','FERNANDO ALBERTO','JESUS DOMINGO','KRZYSZTOF','LEKBIR','ROBERTO LUIS','ADRIAN MANUEL','ENRIQUE JUAN','SAULO','ILUMINADO','KEITH','LONGINOS','YUNES','HECTOR JAVIER','LUIS ESTEBAN','MAEL','MARIO ALEJANDRO','SAJID','UNAY','FELIPE JESUS','HAITAM','LUIS MARTIN','ANTXON','CRISTIAN JAVIER','JOSE MATEO','ORESTES','JOANES','NICHOLAS','DAME','DAVID

GABRIEL','ODILO','ZIYAD','EMANUELE','FRANÇOIS','GONZALO JAVIER','MARLON','CARLOS SANTIAGO','IVAN ANTONIO','JORGE ANGEL','OSSAMA','AMERICO','OCTAVIAN','RAFAEL EDUARDO','SANTIAGO MIGUEL','CESAR ALBERTO','FRANCISCO EMILIO','GERARDO JOSE','IDRISSA','MAREK','SILVIU','DANIEL DAVID','DANIIL','GUSTAVO JAVIER','MOHCINE','SEYDOU','VALERI','ALFONSO ANTONIO','GIACOMO','JIAN','LEVI','MOHSIN','JUAN MARIO','MANUEL ESTEBAN','PIOTR','TOMAS FRANCISCO','VICENTE SALVADOR','VIRGIL','AGONEY','GONZALO MANUEL','JHONNY','JOSE ELADIO','JOSEP MANUEL','JULIAN DAVID','LUIS CESAR','NASSER','PAULO CESAR','RUBEN DAVID','JORGE RAFAEL','MIRKO','JOAO PAULO','LAHOUCINE','PELLO','YAHIA','YONG','CARLOS ADRIAN','IDRISS','ILLAN','VIRGINIO','AMOS','JUAN ELIAS','RAMDANE','ADNANE','JOSE BENJAMIN','JOSE CESAR','KIRIAN','LIBORIO','JHON ALEXANDER','GUIFRE','JULIO ANDRES','KIRIL','MIGUEL DAVID','CRISTIANO','JESUS FELIX','MUHAMADOU','ALEXANDRU GABRIEL','BASSIROU','DAVID ALBERTO','ENRIQUE VICENTE','FERNANDO GABRIEL','FERNANDO JUAN','JOAO PEDRO','NUNO MIGUEL','RAUL MANUEL','SALVADOR ANTONIO','RICARDO ALBERTO','SABIN','ANIANO','BRU','GARY','HANS','PEDRO EMILIO','EDILBERTO','FELIX ALBERTO','FRANCISCO RUBEN','IGNACIO LUIS','JOAN BAPTISTA','NADEEM','RAMON MIGUEL','ROBINSON','SALVADOR MANUEL','ALY','LAID','MARIO JAVIER','YASER','ZIAD','HELIOS','JOAQUIN LUIS','JOHNNY','FRANCISCO IVAN','JIAHAO','JILALI','LUIS OSCAR','MAKSIM','SAMUEL JESUS','AZAEL','GUILLERMO JAVIER','JAVIER FERNANDO','JORGE MARIO','KLAUS','MOHAMED SALEM','RAFAEL IGNACIO','EDISON','MOAD','RAHAL','DELIO','OSCAR FRANCISCO','RAUL JESUS','RUMEN','SALEM','EL ARBI','JACK','LUIS ADRIAN','MICHAL','MIGUEL ALFONSO','JOSE ALONSO','TADEO','VITALII','YURY','ALEXANDRU IONUT','ATANAS','MIHAITA','PAPA','PEDRO GABRIEL','DAVID ANDRES','ESCOLASTICO','JUAN VALENTIN','RAFAEL JAVIER','ANDRES ALBERTO','DIEGO ALEXANDER','VICENTE ENRIQUE','ANTONIO EMILIO','ANTONIO FELIX','FRANCISCO MARTIN','JOAQUIN RAMON','MANUEL ALFREDO','VICENTE RAFAEL','CARLOS ERNESTO','JOSEP VICENT','BORISLAV','CHERKI','GUSTAVO ALBERTO','IGNACIO FRANCISCO','JESUS MARTIN','CARLOS TOMAS','CHRISTOPHER JOHN','EUGENIO MANUEL','FELIX MIGUEL','FERNANDO ANDRES','PEDRO SANTIAGO','ROMULO','UMAR','ANDRZEJ','JOSE VIDAL','JUAN CLAUDIO','MOHAMED RIDA','VICTOR GABRIEL','ZOUHIR','ADONAI','ALVARO FRANCISCO','ASIF','DANIEL FELIPE','EL MAHJOUB','LUIS GUSTAVO','TAHIR','ALEXANDR','CRISPULO','JOSE ORLANDO','STEPHANE','DILAN','JUAN PASCUAL','NICOLAS JOSE','OWEN','RAFAEL FERNANDO','YANNICK','ANTONIO RAUL','CAMILO ANDRES','PASTOR','RUI','SAM','GREGORI','HUGO JOSE','JOAN ALBERT','JOHN MICHAEL','JULIAN ANGEL','LASSANA','MAXIMILIAN','MOHAMED LARBI','VICTOR RAUL','JULIO FERNANDO','MARIO MANUEL','SANTIAGO ANDRES','ILYA','JIMMY','MANFRED','RODRIGO JOSE','CESAR DAVID','DAVID ANGEL','GILBERT','NAEL','OLEKSIY','ALASSANE','ANGEL SALVADOR','MOHAND','OSCAR FERNANDO','YOUSSOUF','DALMACIO','FERNANDO RAMON','HAROUN','HECTOR DANIEL','MANUEL JORGE','YU','HOSSAIN','HUGO ALBERTO','JOSE LEANDRO','NIKO','PABLO SEBASTIAN','SEBASTIAN JOSE','SOULAIMAN','ANTONIO RICARDO','CESAR ENRIQUE','EDDY','FELISINDO','MARTIÑO','RICARDO FRANCISCO','VICENTE CARLOS','ANTONIO FELIPE','DIDIER','FERNANDO ANGEL','HAMMADI','NAOUFAL','PANTALEON','RAMON VICENTE','SOUHAIL','ANDRES ALEJANDRO','FREDY','JORGE ARMANDO','LUIS ADOLFO','MANUEL EUGENIO','PEDRO DOMINGO','SALAHEDDINE','SANTIAGO LUIS','IRENEO','LEONARDO JOSE','RUBEN MANUEL','EVENCIO','MARCOS DAVID','KEVIN JESUS','ALEJANDRO GABRIEL','DAVID JAVIER','JOSE FELICIANO','ANTOINE','GIANFRANCO','JOSE MAURICIO','LUIS ALONSO','RAMON

ALBERTO','TOMASZ','FERNANDO RAFAEL','JULIAN FRANCISCO','WAIL','ANTONIO NICOLAS','DAVID DANIEL','ERWIN','GONTZAL','HECTOR LUIS','JESUS DAMIAN','JOSE CANDIDO','JUAN FERMIN','MANUEL LORENZO','ALLAN','CLAUDI','DAVID ANDREI','FRANCISCO ESTEBAN','JEFFREY','JOAO CARLOS','ARMEN','MARIO ANDRES','SELLAM','ZAHID','DAVID ENRIQUE','DIEGO DAVID','JURGEN','MAMADOU LAMINE','MANU','NAZAR','RAFAEL ANDRES','WERNER','ANGEL DOMINGO','ARCADI','HONORINO','HORTENSIO','KIRILL','PABLO JUAN','RICARDO MIGUEL','ANTERO','BADREDDINE','DANIEL MANUEL','GRACILIANO','IRFAN','MARIANO JESUS','VICTOR ENRIQUE','COLIN','DANIEL LUIS','DORU','INHAR','JULIAN ALBERTO','MONIR','RADOUAN','SHAHZAD','AURELIAN','BRYAN ALEXANDER','FABRICIO','FAHD','LUIS EUGENIO','RIDOUAN','UXIO','FRANCISCO DAMIAN','CHRISTIAN JOSE','FLOREA','JESUS CARMELO','JOAO MANUEL','JOHN WILLIAM','JUAN MATEO','NUÑO','ROSARIO','WALDO','CLAUDINO','EINAR','FRANCISCO JORGE','HECTOR JESUS','ISIDRO MANUEL','JUAN ERNESTO','ZEBENZUI','JOSE HUMBERTO','JUSTIN','MARCOS DANIEL','AYAD','CARLOS JORGE','GURPREET','JAIME MIGUEL','JOAQUIN ALBERTO','JOHANNES','MOUHCINE','NEO','PABLO FERNANDO','GUILHERME','JESUS ESTEBAN','JOSE GUSTAVO','JUVENTINO','NAVEED','DIEGO ENRIQUE','EDUARDO ALBERTO','SERGIO FRANCISCO','VITOR MANUEL','EDUARDO ANDRES','JAVIER VICENTE','JUAN GONZALO','KEVIN JOSE','LUIS GERMAN','PONCIANO','RAPHAEL','SANDU','SEBASTIAN JESUS','TOM','WAQAS','ACOIDAN','JAVIER MIGUEL','JUAN SIMON','PEDRO JORGE','SOHAIL','VICTOR EDUARDO','ANTONIO MARTIN','GERARDO ANTONIO','JORGE RAUL','JUAN ADOLFO','ENOC','JAIME ENRIQUE','OUSMAN','SIDI','VESELIN','ANTONIO MARCOS','JOSE LUCAS','JUSTO MANUEL','LEI','RAZVAN','VICENTE ANGEL','ABDESELAM','ALVARO MIGUEL','BIN','GONZALO JESUS','IONUT ALEXANDRU','JOSE CLAUDIO','MAURICI','MODIBO','MYKHAILO','RICARDO DANIEL','ALFREDO JESUS','FRANCISCO NICOLAS','JAIME JAVIER','JESUS JULIAN','RECAREDO','VASILICA','MAHDI','SERGEI','ADRIAN ANTONIO','CHRISTOPHE','CONRAD','JUAN ARTURO','MARIO FERNANDO','MIMOUNE','CARLOS RUBEN','JEROME','JORDI JOAN','OSCAR ALEJANDRO','RAMI','RAMON JAVIER','ROMEN','STEPHEN
JOHN','ASEN','JOSEF','LAURO','SABAS','ABDELHALIM','ACISCLO','EMILIO
LUIS','GUERAU','GUSTAVO JOSE','IGNACIO MIGUEL','PABLO MARTIN','PEDRO JULIO','ANTONIO ABAD','DOUGLAS','HENRIQUE','JESUS ALFREDO','JOSE DARIO','SANTIAGO DAVID','SERGII','DIEGO ANGEL','EMILIO ANGEL','FRANCISCO LORENZO','JOSE SAMUEL','REMUS','ABDELMOULA','CRISTO JESUS','JULIO RAMON','RIAD','MARTINIANO','OSCAR ENRIQUE','PASCAL','STOYAN','BOUCHTA','EDUARDO DAVID','GUSTAVO DANIEL','HELIO','JESUS MARIO','JONATHAN JESUS','LORENZO JAVIER','MOHAMED YASSIN','ABDESSLAM','ARMAN','CORNELIU','HASAN','JAVIER ALFONSO','MIKEL AINGERU','PIERO','SERGIO ENRIQUE','ADONIS','MANUEL JULIAN','MIKEL GOTZON','ROBERTO MIGUEL','RUI MANUEL','XIN','ALIOUNE','ISMAILA','JEAN CLAUDE','JOAQUIN JAVIER','JUAN BLAS','LORENZO MANUEL','SALVADOR FRANCISCO','ANGEL MARIO','CESAR JESUS','DAVID NICOLAS','ELLIOT','ILIASS','KEVIN DAVID','LUIS JULIAN','MONSERRATE','NICULAE','PASQUALE','SERIGNE','CARLOS
CESAR','ELICIO','MATHEUS','OUSSEYNOU','VICENTE JOAQUIN','VICTOR FERNANDO','AIERT','ARAY','CARLOS MARTIN','ILIA','JAIR','JOHAN SEBASTIAN','RABIE','SACHA','ANTONIO CESAR','CORNELIO','EKI','FRANCISCO PASCUAL','HIMAR','JOSE MARCELINO','JUAN MATA','LUIS SALVADOR','ALFREDO ANTONIO','ANGEL JOAQUIN','JESUS FELIPE','JOHN EDWARD','JOSE ANIBAL','LARO','LAURENT','RAFFAELE','SAMUEL DAVID','SEBASTIAN ANDRES','SIDI MOHAMED','ABDESSALAM','CASILDO','DAVID EDUARDO','JOAQUIN

CARLOS','NACOR','CRISTIAN ALEJANDRO','FRANCISCO FELIX','GONZALO ANTONIO','GUILLERMO LUIS','HUGO MANUEL','JACQUES','LUIS BERNARDO','MARCUS','SANTIAGO FRANCISCO','SERHII','TARSICIO','XOSE MANUEL','ALEJANDRO ENRIQUE','CHAIB','DRAGOS','FERNANDO VICENTE','GREGORIO JESUS','HECTOR ANTONIO','IVAN ALEJANDRO','JOSE EMILIANO','JOSE HILARIO','JOSE ROGELIO','MILTON','PEDRO DAMIAN','SADIK','FRANCISCO ADRIAN','FRANCISCO GUILLERMO','JESUS IVAN','JUAN ELOY','LUIS MARCELO','MAKAN','OTGER','PAWEL','RADOSLAV','ROMA','ALEJANDRO NICOLAS','DARWIN','DELFINO','ENRIQUE ALBERTO','FELIPE JAVIER','FELIPE SANTIAGO','FRANCISCO GINES','GUILLERMO ALEJANDRO','KEVIN DANIEL','NAHUM','PEDRO MARTIN','TOMAS ANGEL','CESAR ALEJANDRO','FELIX FRANCISCO','GRAHAM','JONATHAN ALEXANDER','JOSE JACINTO','JOSEP ANTON','JUAN ADRIAN','JUAN ALVARO','JUNJIE','LOGAN','SHAHBAZ','BRUNO MIGUEL','JULIAN JESUS','MANUEL IVAN','TUDOREL','CLEMENTINO','EMILIO MIGUEL','JUAN MARCELO','VICTOR ALBERTO','ANGEL EMILIO','BARRY','CAMILO JOSE','ESTEBAN MANUEL','JOSE BENIGNO','MANUEL GUILLERMO','MARIO ENRIQUE','MATTIN','TON','ALESANDER','ANTONIO LORENZO','ANTTON','FRANCISCO RICARDO','GODOFREDO','JORGE HUMBERTO','NOUREDDIN','RIZWAN','CARLOS ESTEBAN','ELIES','FRANCISCO EUGENIO','HAMMOU','JESUS GREGORIO','NICOLAS ALEJANDRO','ANTONIO PADUA','CESAR EDUARDO','FRANCISCO FELIPE','GREGORIO MANUEL','JOHN FREDY','JOSE FLORENCIO','JOSE GASPAR','NIKOLAI','RAMON CARLOS','ANTONIO DAMIAN','ALBERTO JUAN','CARLOS JULIAN','GUSTAVO ANTONIO','JORGE GABRIEL','JUAN JAIME','LISANDRO','LOUIS','LUIS GREGORIO','MARZOUK','ABDELAAZIZ','AGUSTIN FRANCISCO','CLIMENT','ELIOT','HAMADI','MARTIN JESUS','MUNIR','SABER','TIBERIU','ZAYD','CRISTIAN GABRIEL','DAVIT','DIEGO SEBASTIAN','ESTEBAN JESUS','HECTOR DAVID','JHON FREDY','KARLOS','MALIK','MARCELLO','SAMUEL JOSE','YEREMI','BALLA','BENITO MANUEL','DAREK','KARL HEINZ','MANUEL BENITO','PETR','SAJJAD','ALBERTO DAVID','BOUALEM','CATALINO','ENRIQUE RAMON','FALLOU','IVAN DAVID','JOVINO','LHOUSSAINE','LUIS ANIBAL','RAUL ALEJANDRO','SALEH','YACOUBA','YOSU','ACORAN','CESAR ANDRES','FODE','GABRIEL FRANCISCO','GIULIO','IONUT DANIEL','IVAN DARIO','JOSE FEDERICO','JULIO ALEJANDRO','MARTIN ALEJANDRO','NICOLAS ANTONIO','ROBERTO DANIEL','SANTIAGO RAMON','JOSE CLEMENTE','JOSE DOLORES','LUIS CARMELO','MICAEL','VICTOR MARIA','ALEJANDRO RAFAEL','DOMINGOS','JOSE YERAY','PEDRO EDUARDO','ABDELILLAH','AKETZA','FERNANDO JORGE','FILOMENO','JORGE ARTURO','JULIEN','LAUTARO','MARUAN','PASCASIO','PEDRO FELIPE','QIANG','ADRIAN ALEJANDRO','CALIN','JOAN PAU','JOSE OMAR','LUIS RODRIGO','MATTHIAS','SERGIO ALBERTO','ENRIQUE ANGEL','FABIAN ANDRES','IGNACIO LOYOLA','IRIOME','JAVIER DAVID','JOAN PERE','LUIS ALEXANDER','PRIMO','RAFAEL RAMON','ABDELGHAFOUR','ANDRES CARLOS','ANGEL ROBERTO','ENRIQUE FERNANDO','GUILLEN','JAVIER ALEXANDER','MAZHAR','SULAIMAN','UBAY','AGUSTIN JESUS','CARLOS MAURICIO','FAISSAL','MARC ANTONI','MOHAMMED SAID','RICARDO JORGE','VITALIE','ZUHAITZ','ALFREDO JAVIER','AURELIO JOSE','DRIS','FLOREAL','HARRY','JAIME JUAN','JOSU MIRENA','JULIO VICENTE','BITTOR','CRISTOBAL MANUEL','GABRIEL MARIA','IFTIKHAR','JESUS RICARDO','JORITZ','TAO','VICTOR JUAN','CEDRIC','FAROUK','GIANNI','JOSE MODESTO','JOSE PATRICIO','JOSEBA MIKEL','THIERNO','ANGEL ALFONSO','ANGEL FELIX','DIRK','GARI','HAYTAM','ION ANDER','JESUS ROBERTO','JOSE VENTURA','LORENZO JESUS','LUIS IVAN','THIERRY','YANG','BENIAMIN','PABLO

RAFAEL','SIDY','ANTONIO PASCUAL','DIMITRI','ERASMO','ANTONIO ESTEBAN','AQUILES','CRHISTIAN','FRANCISCO ROBERTO','GEOFFREY','JAVED','JOAN ENRIC','LUIS RAUL','ZAFAR','AGUSTIN ANTONIO','BRAIAN','JERAY','JOSE FABIAN','JOSE REYES','KAIET','KENAI','VYACHESLAV','AYMEN','CESAR MIGUEL','DIETER','FELIU','JOSE ISAAC','NOURDINE','PABLO EMILIO','SIRAJ','ALVARO DANIEL','ALVARO MARIA','CLAUDIO JOSE','ELEDER','JOSEBA KOLDO','SINESIO','TOADER','ACAYMO','ALBERTO RAMON','ALEXANDRU DANIEL','FRANCISCO JULIO','JACINTO JOSE','JEAN FRANÇOIS','JORGE ALFREDO','MICHAEL JAMES','SUKHWINDER','LUIS FELIX','LUIS JAIME','MARCELO ALEJANDRO','MARINEL','MARIO DAVID','RIYAD','RONALDO','ALBERTO DANIEL','ALFREDO LUIS','DANIEL ESTEBAN','GABRIEL EDUARDO','JAVIER RAMON','LUIS ALVARO','MANUEL FELIX','MARTIN ANTONIO','RICARDO ENRIQUE','VICENTE ANDRES','ABDELATIF','ABOUBACAR','CARLOS ARMANDO','CHRISTIAN ANDRES','DIEGO HERNAN','GELU','JESUALDO','JULIAN CARLOS','MANUEL AUGUSTO','MANUEL MARTIN','MARIANO ANTONIO','NICOLAS MANUEL','RASHID','VIACHESLAV','WILMER','XIANG','EDUARDO RAMON','GRZEGORZ','ISAI','JOSE RAIMUNDO','MOHAMADOU','PABLO VICENTE','ABBAS','EGOI','JON IMANOL','JOSE MARCIAL','ACACIO','ASAD','BOUABID','CALEB','ETIENNE','IVAN FRANCISCO','JOSE LEON','MOHAMMADI','PEDRO PASCUAL','SID AHMED','ZOLTAN','ARSHAD','CRISTIAN FERNANDO','GUSTAVO ENRIQUE','JONATHAN JOSE','JORGE PABLO','JULIO IGNACIO','STEPHAN','ALEJANDRO JUAN','AUGUSTIN','DAVID WILLIAM','DONOVAN','JOSE EUSEBIO','JOSEBA KOLDOBIKA','MARCELLI','MOHAMMED AMIN','OSCAR IVAN','ALICIO','ANGEL FELIPE','EDUARDO DANIEL','FRANCISCO GERMAN','JESUS LORENZO','KACEM','MARCIN','SERGHEI','VICENTE DAVID','YANN','ANGEL MARIANO','ANTONIO SEBASTIAN','DARIO JOSE','HOCINE','JOSEBA IMANOL','KEVIN ALEJANDRO','LUIS ORLANDO','MARIUS DANIEL','OLAF','RAMON ENRIQUE','SIGFRIDO','ALEJANDRO ALBERTO','ALFONSO FRANCISCO','ARTURO MANUEL','EMIGDIO','FRANCISCO JAIME','PEDRO FELIX','RAFAEL PEDRO','BOUBKER','DAVID SANTIAGO','EL MOULOUDI','ERNESTO JAVIER','GERMAN ANTONIO','JOSE ABRAHAM','JOSEBA MIRENA','JULIO CARLOS','MIGUEL SANTIAGO','PABLO RAMON','PAULO ALEXANDRE','PEDRO EUGENIO','VITALY','ZACARIA','ANDRES RAMON','BESAY','CESAR CARLOS','GUSTAVO ANDRES','JUSTO JOSE','MIGUEL RAFAEL','MIGUEL SANTOS','RUDOLF','ANDRES ANGEL','BO','DANIEL VICENTE','JOHN DAVID','MARIO CESAR','SEBASTIAN ALEJANDRO','ABDELKHALAK','ANGEL RAUL','DAVID JUAN','DAVID RAFAEL','FIKRI','GAVRILA','JOSE CRISTIAN','MANUEL CARMELO','MANUEL GABRIEL','MATIAS NICOLAS','NAEEM','ODIN','ROBERT JOHN','RUBENS','SEAN','SEGUNDO MANUEL','TOMAS MIGUEL','TOMMASO','YAN','ANTONIO DIEGO','JORGE MARIA','JOSE LINO','REDUAN','ROSALINO','ABDELHAY','ANGEL RICARDO','ANTONIO CARMELO','CRISTIAN ANTONIO','DIEGO ALFONSO','EDUARDO JUAN','FERNANDO MARTIN','ISMAEL JESUS','JOSE JERONIMO','MADY','MARVIN','PEDRO ESTEBAN','PERE ANTONI','PRAXEDES','RENZO','SAMUEL ALEJANDRO','ALAI','ALBERTO ENRIQUE','ALCIDES','ALEXANDRU MARIAN','ALFONSO ANGEL','ALVARO DAVID','DEMBO','DIEGO MAURICIO','FRANC','HOSSAM','JORGE RICARDO','LLUIS MARIA','UMBERTO','VICTOR RAFAEL','ARTAI','ATTILA','DIEGO RAMON','JAVI','JUAN DAMIAN','LAHSEN','NDIAGA','RAUL FRANCISCO','VICENTE TOMAS','ABDOULIE','AGRIPINO','ANTHONY JOHN','CHARAF','FLORINEL','MAYKEL','PEDRO IVAN','ROBERTO ALEJANDRO','ABDENNOUR','AITOR JOSE','ALEKSEI','ALEXANDRU CRISTIAN','AMILCAR','ANTONIO EUGENIO','ARTURO JAVIER','AZHAR','BASTIAN','DARIO JAVIER','EIZAN','GAVRIL','IURII','JESUS ADRIAN','ROCCO','SEBASTIEN','TIM','AMMAR','DJILALI','EL HADJI','JAIME

MARIA','JOB','MOUHAMADOU','RANDY','VITTORIO','AGUSTIN JAVIER','ALEJANDRO CARLOS','ANGEL ALEJANDRO','BORJA JOSE','CANDELARIO','DUARTE','EDSON','FRANCISCO DIEGO','HAITZ','HUSSAIN','JAIME FERNANDO','JORGE HERNAN','RICARDO DAVID','VICENTE PASCUAL','ANDRES VICENTE','ANGHEL','ANTONIO ROBERTO','ARSEN','GORA','JAIME RAMON','JESUS AGUSTIN','MARCOS LUIS','SEBASTIAN MANUEL','SERGIO IVAN','TERENCE','TOMAS JAVIER','ALEKSANDER','ANGEL JULIAN','CARLOS DOMINGO','GIULIANO','GREGORY','JESUS RAUL','JUAN AURELIO','JUAN GINES','JUAN ROMAN','KAMRAN','KAREL','M HAMED','ADELARDO','ANDREW JOHN','BAGHDAD','BELAID','EKAI','ELMER','EMILIAN','ERNESTO MANUEL','GERALD','GUILLERMO FRANCISCO','ISAAC MANUEL','LEONIDAS','MARCELO FABIAN','SIMO','CARLOS PEDRO','CARLOS SEBASTIAN','CHRISTIAN JAVIER','CLEMENCIO','DAOUD','EGOR','ELHADJI','GEORGEL','HELMUT','JORGE EMILIO','JOSE EULOGIO','MANUEL ARTURO','MARIO DANIEL','MARK ANTHONY','PATRIK','ANTOLIANO','ANTONIO GUILLERMO','ANTONIO JULIO','JOAQUIN IGNACIO','JONATHAN JAVIER','JOSE LUCIANO','JUAN RUBEN','MARIUS IONUT','MIKE','SVEN','ACHOUR','ANTONIO IVAN','BLAS JOSE','EMILIO JUAN','ENRIQUE MARIA','SIGIFREDO','ZEBENSUI','DIEGO MARIA','EL HADI','EMERITO','ENRIQUE ALEJANDRO','INAR','MANUEL ADOLFO','MARTIN JAVIER','ROLF','ARITZA','BENJAMI','DIOGENES','EDUARDO RAFAEL','ENRIQUE DAVID','HUGO DANIEL','IOSEBA','JUAN JULIO','LUIS BENITO','MARCOS ANDRES','MICHAEL ALEXANDER','ROBERTO ANDRES','VITO','ANTONIO RUBEN','DANIEL ADRIAN','DOMINGO GUZMAN','FEDERICO MANUEL','FELIX RAMON','GUSTAVO ALEJANDRO','JULIO DAVID','LUIS AURELIO','RUBEN FRANCISCO','RUDY','USAMA','ABDELOUAHAD','EL MOSTAPHA','HUG','ISAURO','JOSE CAMILO','JUAN ALEXIS','JUNHAO','MANUEL SEBASTIAN','MIGUEL PEDRO','MILEN','PABLO GABRIEL','WILLY','AYUB','BENICIO','CARLOS EUGENIO','CARLOS FABIAN','CHERIF','HAROUNA','JAKES','JOSUE DAVID','JUAN ISMAEL','LICINIO','LOTFI','MACIA','MANUEL GONZALO','PEDRO SALVADOR','RUDESINDO','SVETOSLAV','VLAD','ABDELFETTAH','ALVARO ANDRES','ANDREI ALEXANDRU','BLAS ANTONIO','FIDENCIO','IVAN JAVIER','JAIME EDUARDO','JOAN SEBASTIAN','JOSEP JOAN','KAREN','MANUEL NICOLAS','OSCAR RAUL','PERE JOSEP','RALF','SALIF','ZEESHAN','CLODOALDO','CRISTIAN ALEXANDER','IACOB','IBRA','JOSE OCTAVIO','LEONARDO DAVID','MICHAEL DAVID','MUDASSAR','RAUL DAVID','SAIKOU','ANDRES ENRIQUE','ANDRES JUAN','ANTONIO AUGUSTO','BOUJAMAA','CARLOS AGUSTIN','CESAR FRANCISCO','FANEL','FRANZ','JOSE EVARISTO','JUAN JERONIMO','MAMOUDOU','NICOLAS ANDRES','PEDRO NICOLAS','SADIO','SERGIO RAMON','TIZIANO','ANIS','BERND','BOUZIANE','DAVID CARLOS','EUSEBI','FELIPE ANDRES','GERARDO MANUEL','IURIE','JAD','JOAQUIN ANDRES','JOSE FAUSTINO','MIGUEL SALVADOR','NASREDDINE','SALVADOR VICENTE','CLAUDIO ANTONIO','EMILIO RAMON','FAOUZI','FEDERICO JAVIER','GERHARD','HUGO ANTONIO','JACINTO JAVIER','JOSEP ENRIC','KOUIDER','MARCOS MIGUEL','MARIUS CRISTIAN','THAMI','XIAOFENG','ALVARO ENRIQUE','ARNOLD','CHAKIR','DIOGO','EUDALDO','EUTIQUIANO','FLORENCI','FLORINDO','HANS PETER','ILIYA','IVAN CARLOS','JIHAD','JOAQUIN ENRIQUE','JONATHAN ANDRES','KELVIN','LUCAS DANIEL','MANUEL ROBERTO','TASIO','ABOU','ADELAIDO','CHAKIB','CHEN','DIOSDADO','DOMINIK','GUILLERMO ENRIQUE','HECTOR EDUARDO','JORGE ROBERTO','NICOLAS JESUS','OUSAMA','RAFAEL SANTIAGO','RAMON IGNACIO','ADOLFO JESUS','ADOLFO MANUEL','ALFONSO JUAN','BOURAMA','DAVID JAMES','DIEGO ALONSO','EMILIO VICENTE','EUGENIO JESUS','FELIX LUIS','GAETANO','HERMAN','JORGE ALEXANDER','JOSE AITOR','JOSE MARCELO','JUAN

GERARDO','KASHIF','MAHFOUD','MIHAI ALEXANDRU','MIKAEL','OSCAR RAMON','RAUL FERNANDO','RUBEN ALEJANDRO','CRISTIAN ADRIAN','EL AID','FLORIN DANIEL','MANUEL VICTOR','MIGUEL JOAQUIN','NICOLAI','NICOLAIE','PENG','ROBERTO FRANCISCO','RODRIGO JAVIER','WILLIAMS','ABDESLEM','AMARA','BUBACARR','HECTOR MIGUEL','JEAN CARLO','JOAN IGNASI','LEONARDO JAVIER','MAMERTO','MANUEL DANIEL','MBARK','WILLIAN','ALBERTO ANGEL','ALEXANDRU MIHAI','ALFONSO DAVID','ANTONIO JAIME','AUSIAS','FERNANDO ALFONSO','HOUCINE','HUGO MIGUEL','IGNACIO RAMON','ISTVAN','JOSE PAULINO','LAYACHI','MARIUS ADRIAN','PABLO TOMAS','PASCUAL JOSE','DAVID MICHAEL','ESTEFANO','HAMIDOU','ISMAEL JOSE','NOURDIN','QAISAR','ROBERTO ENRIQUE','SAMUEL ANTONIO','YAMAL','YURII','ABDELMOUNAIM','AGUSTIN ANGEL','AMER','CHRISTIAN MANUEL','CRISTOFOL','DIEGO RAFAEL','FELIPE CARLOS','FRANCISCO CARMELO','FRANCISCO GREGORIO','HECTOR ALEJANDRO','ISOLINO','IZHAN','JOSE JUSTO','JOSEP XAVIER','JULIAN MARIA','LUIS AUGUSTO','SAFWAN','YANIS','EDUARDO ALEJANDRO','HERBERT','ISIDRO JAVIER','JULIAN MIGUEL','LUIS DIEGO','RAMON FERNANDO','RODRIGO ANTONIO','BRYAN STEVEN','CHRISTIAN JESUS','DANIEL MIGUEL','EL HOUSSAINE','FARES','HOUARI','JIANFENG','JOHN HENRY','JOSE SIMON','KAMIL','LLIBERT','LUCAS MANUEL','LUIS AGUSTIN','LUIS FILIPE','MATIAS EZEQUIEL','MICHAEL ANTHONY','RICARDO JUAN','SERGIO JUAN','ALBERTO ALEJANDRO','BRYAN ANDRES','CARLOS ALEXANDER','CARLOS NICOLAS','GABRIEL ANDRES','GUILLERMO ANDRES','JOAQUIN FERNANDO','JURGI','XACOBE','ALEJANDRO ANDRES','DAVID LUIS','ELIAS JOSE','ELOY JOSE','GERMAN JOSE','IOAN DANIEL','JOHANN','JUAN OSCAR','LASZLO','LUIS LORENZO','MAGDALENO','MARIO FRANCISCO','MARTIN MANUEL','NASIM','ANATOLI','ANGEL GUILLERMO','BELAL','DANIEL CARLOS','EDUARDO VICENTE','JAKUB','JORGE FELIX','JOSE SERGIO','JUSTO ANTONIO','MARIO MIGUEL','ROSSEND','UWE','ABDELKABIR','BERNARDO JESUS','DARIUSZ','EDINSON','ETELVINO','IVAN LUIS','JAMIL','LUCAS JOSE','PABLO ALFONSO','PEDRO JAIME','ROBERTO ANGEL','RODRIGO ANDRES','TSVETAN','ADOLF','DIEGO NICOLAS','FAYSAL','FRANCISCO SEBASTIAN','GABRIEL ENRIQUE','HECTOR HUGO','LYUBOMIR','MARIUS GABRIEL','MITKO','ODON','RAFAEL JOAQUIN','RAHHAL','ABEL JESUS','ANGEL ESTEBAN','AUREO','CHRISTIAN ALEXANDER','DJIBY','DOMINIC','GEORGES','HARKAITZ','JOSE JONATHAN','LARS','OSCAR RUBEN','PASQUAL','RAFAEL PABLO','RUBEN ALBERTO','RUBEN DANIEL','TANVEER','TIMUR','YUSUF','ABDENNABI','AITOR MANUEL','ALVARO JUAN','ANGEL RUBEN','BINGEN','CARLOS ALEXIS','IOAN ADRIAN','IONE','JORGE GUILLERMO','LAHOUARI','LEANDRO JOSE','LUIS HERNAN','MANUEL GREGORIO','MIMON','XESUS','ANGEL MARTIN','CARLOS FEDERICO','CHAFIK','CRISTIAN ALBERTO','DARIEL','EDUARDO ANGEL','EL HOUSSAIN','JASWINDER','JOHN ANTHONY','MARIANO JAVIER','MOHA','PEDRO CESAR','RAUL EDUARDO','XIAO','ZULFIQAR','ADRIAN FRANCISCO','ALEJANDRO RAMON','ATAULFO','BORJA MANUEL','CRISTHIAN DAVID','DIEGO IGNACIO','GIOVANNY','HUGO ALEJANDRO','JESUS OSCAR','JOSE ABELARDO','MOHAMED RAYAN','NICETO','OMAR JESUS','RUBEN ANGEL','SEBASTIAN ANTONIO','VICENTE ALBERTO','AFRODISIO','CRISANTOS','ERNESTO JESUS','GONZALO MARIA','HAMDI','IMTIAZ','JORGE WASHINGTON','JUAN CESAR','MARCOS ALBERTO','NICOLAS FRANCISCO','OLEKSII','RICHARD JOHN','WIFREDO','ZOSIMO','ADRIAN GABRIEL','DANIEL JUAN','FRANCISCO ISMAEL','GUILLERMO JUAN','IELTXU','JORGE ADRIAN','JORGE OMAR','LUIS SEBASTIAN','MANUEL JAIME','MARICEL','MICHAEL STEVEN','PEDRO HENRIQUE','SANTIAGO ALEJANDRO','SERGIO FERNANDO','AGUSTIN MIGUEL','AIDEN','ANTONIO ISMAEL','ATIF','CARLOS ADOLFO','DOMINIQUE','EDUARDO CARLOS','JOSE DIONISIO','JOSE TEODORO','JOSUA','RAFAEL ALFONSO','RODRIGO

ALEJANDRO','RUI MIGUEL','ANGEL IVAN','DRISSA','IMRANE','JESUS GUILLERMO','JOHN PAUL','JOSE LAUREANO','KURT','LIANG','SANTIAGO JUAN','SAQIB','WASEEM','YUSSEF','ADRIAN IONUT','ANATOLIE','ANTONY','EDGARD','ESSAID','IONUT ADRIAN','IONUT CATALIN','JACINTO MANUEL','JEREMI','JOAQUIN RAFAEL','JUAN RODRIGO','MARCELO DANIEL','MARCO TULIO','RAMON JUAN','ALBERTO MARIA','ALEXANDRU FLORIN','ANTONIO FERMIN','DAGOBERTO','DANIEL SANTIAGO','EL HOSSAIN','FERNANDO TOMAS','IRAI','JORGE PEDRO','MALCOLM','MARIUS FLORIN','PABLO EDUARDO','ROGERIO','VICTOR JOAQUIN','VICTOR VICENTE','ZAKARIYA','ALAE','ANGEL AGUSTIN','CARLOS RODRIGO','DOMINGO LUIS','FERNANDO JOAQUIN','GEORGIOS','GORAN','HECTOR ENRIQUE','HOUSSAIN','HUSSEIN','JONHATAN','JORGE ERNESTO','LAKBIR','LUCAS GABRIEL','PEDRO VICTOR','ROMICA','TIJANI','TOMAS ENRIQUE','ADRIAN DAVID','AOMAR','BENITO JESUS','DANIEL RAMON','EDUARDO MARIA','HUI','JAVIER JUAN','JIALE','JORGE JOAQUIN','JUSTO JAVIER','KEVIN MANUEL','LORENZO ANTONIO','MARCIO','MARTXEL','MOHAMMED LARBI','RAINER','XAN','ABDELHAQ','AIMAD','DANIEL ALFONSO','FRANCISCO MARIANO','GUSTAVO MANUEL','JAIME VICENTE','JESUS NICOLAS','JOAN VICENT','OLIMPIO','PETRONILO','RANJIT','SALEK','SALVADOR MIGUEL','ALEXANDRU IOAN','AMEUR','BYRON','CARLOS MARCELO','DANIEL IGNACIO','DANIL','DAVID ALEXANDRU','DIEGO GABRIEL','EMERSON','FADI','JOAN ANDREU','LUKASZ','MANUEL ADRIAN','MATAR','MATEO JOSE','OSMAN','PARIS','PETER JAMES','RAFAEL EMILIO','RAUL ANDRES','SIAKA','TIHOMIR','ASIM','DAVID ISRAEL','ENRIQUE RAFAEL','FERNANDO PEDRO','GREGORIO ANTONIO','JORGE SANTIAGO','LUIS FABIAN','LUPICINIO','MANUEL MESIAS','MOHAMED LAMINE','NOREDDINE','SANTIAGO MARIA','TOMAS ALBERTO','ADRIAN DANIEL','DYLAN ALEXANDER','FLAVIANO','GRAU','HARPREET','IVAN ANDRES','JOSE BAUTISTA','JUAN EUSEBIO','MARCOS GABRIEL','MARIUSZ','MING','MOHAMED FADEL','PABLO SANTIAGO','ROBERTO JUAN','RONNY','SIGFREDO','ALEXANDER JOSE','ANDREI CRISTIAN','BALWINDER','CARMELO JAVIER','CESAR DANIEL','ISAM','ISHAQ','JESUS ALVARO','JESUS EUGENIO','JOSE MOISES','MANUEL RAUL','MANUEL SANTOS','MAURICIO ALEJANDRO','PAUL ANTHONY','ROBERT JAMES','VICENTE FERNANDO','AZZEDINE','BONAVENTURA','CARLOS HERNAN','EXPEDITO','FELIX CARLOS','GABRIEL MANUEL','HARALD','JAIME ALEJANDRO','JORGE ANIBAL','LOUAY','OSCAR VICENTE','PABLO JOAQUIN','QUIQUE','RAFAEL JORGE','ROBERTO DAVID','RUBEN CARLOS','CRISTIAN EDUARDO','DAVID VICENTE','EL AMIN','ENRIQUE DANIEL','JOHN JAMES','JORGE LEONARDO','JUANJO','JUVENAL','MARCOS FRANCISCO','MAREN','MARIA','MARIO ANGEL','MARIUS CATALIN','SOULAIMANE','VICTOR JULIAN','ALBERTO RAFAEL','ALFREDO MIGUEL','AZZOUZ','CARLOS MARIANO','DANIEL RICARDO','DMITRII','DUMITRU DANIEL','EDOARDO','ELOY MANUEL','GINO','IJAZ','JABIER','JOSE RODOLFO','JULIO RAFAEL','LOAY','NEYZAN','OSCAR JUAN','PABLO ADRIAN','QAMAR','RAUL JAVIER','SANDEEP','YEVHEN','ANATOLY','ASCENSIO','BRUNO JOSE','ELYAS','FIDEL ANGEL','JIAJUN','JUAN JOAQUIN','JULIO ALFONSO','LEOPOLDO JOSE','MARTIN FRANCISCO','MAURICIO ANDRES','RODRIGO MANUEL','ROSEN','SOCRATES','VICTORIANO MANUEL','YERAY JESUS','ANTONIO BENITO','DAVID RAMON','EDEN','GEORGY','GUNTER','HADJ','IVAN DANIEL','JEAN LOUIS','JORGE ALFONSO','LI','MARCOS ALEJANDRO','MARIUS ALEXANDRU','PATRICIO JOSE','PEDRO RAUL','SOULIMAN','TOMAS LUIS','AHSAN','ANDRES DANIEL','ANSAR','ANTONIO ADRIAN','ARGELIO','AYYOUB','CRISTOBAL JAVIER','DANIEL JAVIER','EDUARDO GABRIEL','ELIAS MANUEL','FELIX ENRIQUE','HUGO FERNANDO','JUAN SERGIO','JULIO ERNESTO','MATEI','MOUSTAFA','PEDRO GINES','ANDRES IGNACIO','BLAS JESUS','CARLOS SALVADOR','CHEIKH TIDIANE','CHENG','DANIEL MARIA','EUSEBIO

JOSE','FAYCAL','GEORGIAN','JEAN MICHEL','JENS','MANUEL ISIDRO','MIGUEL AGUSTIN','NESTOR JOSE','PEDRO RICARDO','POMPEYO','PRINCE','RAUL ENRIQUE','SANDOR','SERGIO GABRIEL','VALERICA','ALEX DAVID','ANDREI DANIEL','ANTONIO VICTOR','BLAS MANUEL','DANIEL FLORIN','DANIEL SEBASTIAN','DATIVO','EL KHALIL','ENRIQUE EDUARDO','GUILLERMO MIGUEL','HAGIE','JESUS JORGE','JUAN SANTOS','MANUEL OSCAR','MIGUEL PABLO','MILO','NECO','STANLEY','VEACESLAV','VINICIUS','YULEN','AISSAM','ANTONIO MARIANO','CHRISTIAN DANIEL','EL HAJ','FERMI','FRANCISCO ADOLFO','HECTOR FRANCISCO','IGNACIO ANDRES','ITZAN','JAAFAR','JEAN MARC','JULIO JUAN','MARCELO JAVIER','RAMIRO JOSE','SASHA','VICENTE PEDRO','ADRIEN','ARMANDO JESUS','BERNARDO ANTONIO','CESAR FERNANDO','DAVID ALFONSO','FELIPE LUIS','GABRIEL FERNANDO','HUGO JAVIER','JALIL','JOSE LEOPOLDO','JULIO EDUARDO','KENAY','MANRIQUE','MANUEL DAMIAN','MATIES','PABLO AGUSTIN','SABIR','STIVEN','VENTSISLAV','VICTOR EMILIO','ZIHAO','ABDELKEBIR','ALBERTO ANDRES','ANGELINO','ANTONIO ISIDRO','DALMIRO','FERNANDO PABLO','FRANCISCO ISRAEL','KHURRAM','LUIS MARCOS','RAUL MIGUEL','RICARDO ANGEL','TALLA','UGO','ARSLAN','CARLOS FELIX','FRANCISCO ALEXIS','FRANCISCO XABIER','HENRI','IGNACIO JUAN','ILIAN','IONUT CRISTIAN','JIANWEI','JOAO MIGUEL','JUAN PATRICIO','LUCKY','MANUEL JULIO','ORLANDO JOSE','PETRISOR','SAMER','SERGIO EDUARDO','TEODULO','VALERY','ALFONSO RAMON','ANATOLII','ANGEL NICOLAS','CARLOS PATRICIO','EDGARDO','JIANPING','LIN','MANUEL CRISTOBAL','MITICA','RICARDO ALEJANDRO','SANTIAGO ALBERTO','TAIB','YVES','ABDOURAHMANE','ADRIAN VASILE','ARIDANI','CARLOS RICARDO','CHRISTIAN ALEJANDRO','DAVID GEORGE','FLORIN ADRIAN','ILIYAN','ISMAEL ANTONIO','JORGE NICOLAS','KARL','KINGSLEY','PASCUAL ANTONIO','ROSTYSLAV','ANDRES SEBASTIAN','ANTONIO GERMAN','BLADIMIR','IOAN ALEXANDRU','ISIDRO ANTONIO','JAIME DAVID','LUIS RUBEN','MATIAS JOSE','MIGUEL ESTEBAN','OSCAR MAURICIO','OUTMAN','WILDER','ABDEL LAH','ADEEL','ALPHA','ALVARO RAFAEL','ARTURO JESUS','BEN','BENITO JAVIER','FEDERICO ANTONIO','FRANCISCO BERNARDO','HELDER','JELLOUL','LUCRECIO','MARIO GABRIEL','MARWANE','MAURICE','MIGUEL JORGE','MIGUEL OSCAR','MIGUEL TOMAS','RAYHAN','ROBERT ANDREI','VALER','VICENTE ALEJANDRO','AARON JOSE','AUGUSTO JOSE','BALDUINO','DAVID ADRIAN','GENNARO','GUILLAUME','IOAN FLORIN','IONUT GABRIEL','JABER','JOSE HIGINIO','JOSE RUFINO','LIAN','LORETO','MANUEL GERARDO','NADER','NATHANIEL','OSCAR ALFREDO','ROBERT GABRIEL','SAMY','TXABER','VALENTYN','ANTONIO VALENTIN','BORYS','DOMINGO FRANCISCO','EUGENIO ANTONIO','FERNANDO IGNACIO','FERNANDO NICOLAS','FREDERICK','GABOR','JACEK','JESUS VALENTIN','JIANJUN','JOACHIM','KARMELO','KOSTYANTYN','LEONARDO ANTONIO','MOHAMED YASSINE','PEDRO GUILLERMO','QUITERIO','RAMON ANDRES','RAMON EMILIO','ROBERT WILLIAM','SANTIAGO ANGEL','SANTIAGO ENRIQUE','SANTIAGO NICOLAS','SERGIO RAFAEL','SHENG','ADRIAN FLORIN','ALBERTO TOMAS','ALEX DANIEL','BADER','CRISTIAN IONUT','DAIRON','DIEGO JUAN','FRANCISCO ALONSO','HANS JOACHIM','JAIME CARLOS','JORGE TOMAS','LEONARDO ANDRES','MANUEL ERNESTO','NAWFAL','PABLO PEDRO','ABSELAM','ANGEL ALFREDO','ARMINDO','BARA','CESAR MARIA','DANY','DAVID ESTEBAN','ERNESTO ANTONIO','FELIX JUAN','FERMIN JOSE','GUILLERMO DANIEL','ISKANDER','JESUS PASCUAL','JORDI JOSEP','LUKE','MONSEF','MOUHSSINE','ORLANDO JESUS','SOLIMAN','ALEXANDRU CONSTANTIN','ARCANGEL','ARMICHE','BOGDAN IONUT','BRYAN DAVID','CRISTIAN MIGUEL','DAVID SEBASTIAN','DEMIAN','ESTEBAN FRANCISCO','ESTEBAN JAVIER','FRANCISCO ROMAN','JOEL ALEXANDER','JOSE ANSELMO','JOSE HERMINIO','LESMES','LUIS FERMIN','MARIYAN','SANTIAGO

DANIEL','VALERII','VENERANDO','YARED','ACHER','ALEJANDRO VICENTE','ALEXIS JESUS','ALFONSO ENRIQUE','CARLOS DAMIAN','CARLOS JAIME','CHAO','DOMINGO JUAN','EUGENIO FRANCISCO','FERNANDO AGUSTIN','FRANCISCO MARIO','IGNACIO CARLOS','JORGE MARTIN','JUAN ABEL','JUAN ARMANDO','JUAN LEON','LUIS OSWALDO','MAMADY','MODEST','NICOLAS JAVIER','NOURDDINE','PEDRO LORENZO','RICARDO VICENTE','SERGIO CARLOS','ALEJANDRO MARTIN','ALVARO FERNANDO','ANGEL GREGORIO','AUGUST','CONSUELO','DRAMANE','EDISON JAVIER','GEORGE ALEXANDRU','JESUS ADOLFO','JOSE CELESTINO','MAMADOU SALIOU','MARIANO LUIS','PAULO SERGIO','QI','SALVADOR DAVID','SALVADOR JUAN','SERGIO RAUL','AHARON','ALAN JOHN','ANTONI JOSEP','ANTONIO ALFREDO','ASIS','BABAR','BENYOUNES','EHSAN','ELADIO JOSE','EMILIO CARLOS','ERICK ALEXANDER','JOAN ANTON','QASIM','SOUHAIB','TIGRAN','TOMAS VICENTE','VICTORIANO JOSE','WAQAR','ALEJANDRO TOMAS','ANDREI IONUT','ANDREW JAMES','ANGEL JULIO','ARYAN','DIEGO DANIEL','DOMINGO MIGUEL','DOMINGO RAMON','GERMAN LUIS','GUY','JAVIER TOMAS','JOAQUIN PEDRO','JULIO MARIA','KEVIN JAVIER','LUIS FEDERICO','LUIS NICOLAS','MANUEL ALVARO','MAXIME','NIDAL','PAUL DAVID','STUART','VASILE ADRIAN','VICTOR ADRIAN','ARIS','CHISTIAN','EUGENIU','GORGONIO','HERNAN DARIO','ISIDRO JESUS','JOAN DAVID','MAATI','MARTIN MIGUEL','RAMON ALEJANDRO','RAUL DANIEL','WILLIAM JOHN','XUAN','AAMIR','ALBERTO FERNANDO','ALBERTO VICENTE','ALEX FERNANDO','ANTONIO BLAS','DANIEL NICOLAS','EUGENIO JAVIER','FELIX FERNANDO','FRANCISCO ALFREDO','FRANCISCO CRISTOBAL','JAIME IGNACIO','JOSE FRANCISCO JAVIER','KOSTADIN','LAKHWINDER','LIBRADO','LONGINO','MAMADOU ALIOU','MELVIN','NICOLAS DANIEL','PEDRO NOLASCO','SANTIAGO RAFAEL','ALEXANDRU ANDREI','ANTONIO GREGORIO','ANTONIO MATIAS','DAVID MARIA','FRANCISC','GORDON','IVAN MIGUEL','MARCO ANDRES','MARCO JOSE','MARCOS AURELIO','MATEUSZ','OSCAR IGNACIO','RAMON EDUARDO','SERGE','XOSE ANTON','ZOHAIR','ZURAB','ANTONIO CRISTOBAL','CARLOS OMAR','CHADI','CHRISTIAN FERNANDO','DONALD','EUKEN','GERMAN ANDRES','JESUS ISRAEL','JHON','JON KEPA','LEONARDO FABIO','LOIC','MAURICIO JOSE','MOISES DAVID','NILS','TOUHAMI','ABLAYE','CARMINE','CSABA','DANIEL IVAN','EDUARD ANDREI','FERMIN JAVIER','GUILLERMO ALBERTO','JAVIER AGUSTIN','JOSE AQUILINO','JUAN CLEMENTE','LAUDINO','LEONARDO GABRIEL','MANUEL RUBEN','MONICO','NESTOR MANUEL','NEYMAR','SEBASTIAN FRANCISCO','SOULAYMAN','TREVOR','VICENTE DANIEL','ZAIN','ABDELTIF','ANGEL CRISTO','CANDIDO JOSE','DANAIL','DAVID CRISTIAN','DIEGO EDUARDO','FRANCISCO ELIAS','FRANCISCO SALES','GABRIEL ALBERTO','GABRIEL ALEXANDER','GAROE','GIGI','IDRIS','IQBAL','JAIRO JESUS','JERONIMO JOSE','JESUS RUBEN','JORGE RUBEN','JOSE FULGENCIO','JOSE LAZARO','JOSE MARINO','JOSE NORBERTO','JUAN GERMAN','JUAN REYES','LUIS PATRICIO','MANUEL ALEXANDER','MARIANO FRANCISCO','MATIAS ALEJANDRO','MUHAMMED','MYROSLAV','NICOLAS DAVID','PELEGRIN','SANTIAGO FERNANDO','VASILE CRISTIAN','VICTOR JULIO','WLADIMIRO','ABDEL ILAH','ABDELFATAH','ABDENOUR','ADELIO','ANDRES RAFAEL','CARLOS ANIBAL','ERIZ','FRANCISCO ABEL','FRANCK','GONZALO ANDRES','IGNACIO ANGEL','JAVIER RAFAEL','JESUS NAZARET','LUAN','MANUEL AURELIO','SALVI','SEBASTIAN GABRIEL','ZBIGNIEW','ADOLFO ANTONIO','ADRIAN ALBERTO','ALEJANDRO MARIA','ANDRES SANTIAGO','ANGEL SEBASTIAN','DAVID EMANUEL','ESTEBAN LUIS','FARHAN','GERMINAL','GONZALO LUIS','HECTOR ANDRES','JAIME ALFONSO','JOSE SEGUNDO','JOZEF','JULIAN JAVIER','PETKO','STEVE','VICTOR SANTIAGO','ALEX JAVIER','CLAUDIO ANDRES','DAVID ANTHONY','FILIPE','FRANCESC PAULA','FRANCISCO MARCOS','FRANCISCO SAMUEL','GHEORGHE DANIEL','GREGORIO JAVIER','HORST','JACOPO','JAVIER SANTIAGO','JOHN

ROBERT','JOSEP IGNASI','JULIO DANIEL','LAZAR','LUC','MATHIEU','MIGUEL ANGELO','QUIRINO','RICARDO RAMON','VITOR','XIAOJUN','YEVGEN','ADAI','ANTONIO BERNARDO','ARIF','BADR EDDINE','BENITO ANTONIO','CHAHID','DAVID RICARDO','EDESIO','EDUARDO JORGE','FELIPE ALBERTO','FERNANDO SANTIAGO','GABRIEL LUIS','GONZALO FRANCISCO','HANS JURGEN','HECTOR RAUL','HERVE','JAVIER MARTIN','JESUS SEBASTIAN','JOE','JUAN FEDERICO','KIM','LEE','MALANG','MANJIT','MANUEL VALENTIN','PABLO ESTEBAN','PABLO ROBERTO','TIMOTHY','VASILE DANIEL','XIAODONG','ADRIAN JAVIER','ALEIXANDRE','ALEXIS JOSE','CEZAR','DANIEL GABRIEL','ERITZ','GUSTAVO JESUS','IGNACIO ALBERTO','JORGE AGUSTIN','JOSE ADAN','JOSE ELISEO','LEVAN','MANUEL ARMANDO','MIREL','OCTAVIANO','ULRICH','XIKER','YOSEF','AARON JESUS','ASDRUBAL','BORJA JESUS','CHRISTOPH','DIMITRIOS','FLORIN CRISTIAN','GUILLERMO CARLOS','HEBER','HUGO ENRIQUE','JEAN JACQUES','JESUS JULIO','JOSE OVIDIO','KIKE','NICOLO','SANTOS MANUEL','SIMON PEDRO','VICENTE MARIA','VICTOR IGNACIO','ADOLFO JAVIER','ADRIAN IOAN','ALEKS','ALONSO JOSE','CIBRAN','DANIEL IONUT','DOMITILO','FRANCISCO MATIAS','HARJINDER','IZAM','JONATHAN MANUEL','JUAN BARTOLOME','NICOLAS ALBERTO','PROSPERO','ADOLFO LUIS','AMETZ','ANGEL CARMELO','AYMAR','CARLOS VICTOR','CLEMENT','CONSTANTIN CRISTIAN','DANIEL CRISTIAN','EMILIO ENRIQUE','ESTEFAN','FEDOR','FELIX DAVID','FERDINAND','ISAAC JOSE','JAIME RAFAEL','JOEL DAVID','JOSE EZEQUIEL','JOSE NARCISO','JOSE ROQUE','JULIAN LUIS','LUIS MATEO','MAURICIO JAVIER','MEDARDO','NACER','NICOLAE ADRIAN','SERGUEI','SHOAIB','TOMAS DAVID','WOJCIECH','CARMELO ANTONIO','CLARO','CRESCENCIANO','DAVID IGNACIO','EL MAMOUN','FABIANO','FARIS','FRANCISCO ARTURO','GEVORG','GINES JOSE','HOSSEIN','IGNACIO DAVID','ILLYA','JEAN MARIE','JESUS BENITO','JHOAN SEBASTIAN','JOEL ANTONIO','JOHN JOSEPH','JOHNATAN','JOSE BASILIO','JOSELITO','LESLIE','MANDEEP','MANUEL MARIANO','MIN','NICOLAS GABRIEL','OUASSIM','RAUL GABRIEL','SALVA','SERGIO ADRIAN','ABDELLAZIZ','AGUSTIN CARLOS','ALBERTO IGNACIO','ALBERTO JOAQUIN','ALPHA OUMAR','ANTONIO GINES','ASHRAF','BERNARDO MANUEL','CLETO','CONSTANTIN CATALIN','DIEGO CARLOS','FRANCISCO FERMIN','GEORGE ADRIAN','HECTOR FERNANDO','IOANNIS','IONUT FLORIN','JIANHUA','JOHN ALEJANDRO','JOSE GILBERTO','JUAN JACOBO','KALOYAN','RAYYAN','XERACH','YEISON','ZAHEER','AGUSTIN MARIA','AKHTAR','ALAN DAVID','ANGEL RODRIGO','BISER','CATALIN IONUT','CHRISTOPHER JAMES','DANIEL IOAN','EDUARD GABRIEL','FELIX ANDRES','FODAY','FRANCISCO GERARDO','HEINZ','ISRAEL DAVID','JORGE DOMINGO','JORGE OSCAR','JOSE VICTORIANO','JULIUS','KILLIAN','LEONARDO JESUS','LLUIS MIQUEL','NELSON JOSE','PABLO MARCELO','PEDRO ISMAEL','PEDRO ROBERTO','RAUL ANGEL','SALVIO','WILLIAM ALEXANDER','XIAOWEI','AARON DAVID','ANTHONY DAVID','BASILISO','BOUZIAN','DIEGO PABLO','EL ALAMI','ENRIQUE PEDRO','ES SAID','ESTEBAN ANTONIO','EUGENIO LUIS','FERNANDO AUGUSTO','FRANCISCO VALENTIN','GRACIAN','JOHN RICHARD','JOSE FLORENTINO','JOSEP CARLES','MARCELO ANTONIO','MARIO EDUARDO','MARTIN ANGEL','PAUL JOHN','RAFAEL TOMAS','SACRAMENTO','ABEL JOSE','ANDREI GABRIEL','ANTONI MARIA','ANTONIO MARIO','ASLAM','BJORN','CHRISTIAN EDUARDO','DARIUS ANDREI','EDWIN ANDRES','ELIER','ESSA','FRANCISCO MOISES','GIGEL','HAJI','JESUS VICTOR','JUAN MAURICIO','JULIAN FERNANDO','LEV','LIZER','MANUEL BERNARDO','MARCO PAULO','OSCAR RAFAEL','SAEED','SOLOMON','VALENTIN JOSE','ZUBAIR','ALFREDO ENRIQUE','ANGEL DAMIAN','ANGEL DIEGO','ASHOT','AURELI','BRUC','ERICH','FERNANDO RAUL','FILEMON','FLORIANO','GAGIK','GERARDO JESUS','IOAN CRISTIAN','JESUS ABEL','JESUS ERNESTO','JESUS JAIME','JORGE ESTEBAN','JOSE ISIDORO','LUCAS JAVIER','LUIS VICTOR','MATEUS','NESTOR JAVIER','PHILIP JOHN','RAFAL','SAIDOU','SALVADOR ANGEL','SALVADOR JAVIER','SERIGNE FALLOU','WEIFENG','WESLEY','YENEDEY','ANDRES

NICOLAS','BONOSO','DAVID CHARLES','DEEPAK','EDGAR JOSE','EDGAR MANUEL','FELIX IGNACIO','GABRIEL DAVID','GREGORIO FRANCISCO','IGNACIO RAFAEL','JADEN','JESUS MARCOS','JOAQUIN ALEJANDRO','JOSIAS','JUAN JACINTO','JUAN LEONARDO','KEVIN JOEL','LUDOVIC','LUIS DAMIAN','LUIS JULIO','MANASES','MARCO ALEJANDRO','NICOLAE DANIEL','PARAMJIT','RAMON PEDRO','SANTO','TRIFON','VICENTE GABRIEL','XIAOPING','YIFAN','ANGEL GUSTAVO','ANGEL MARCOS','ANTONIO GONZALO','ANTONIO REYES','CARLOS ARIEL','CLAUDIO MANUEL','CONSTANTIN DANIEL','ESMERALDO','FRANCISCO MATEO','GERMAN JESUS','HERMANN','IRAKLI','JONNATHAN','JOSE AMADEO','LUCAS DAVID','NAUFAL','OLEXANDR']

8.4.3 Creación variable "genero_juez"

```
1. from nombres_mujer import *
2. from nombres_hombre import *
3. import pymysql
4. import re
5. from unicodedata import normalize
6. import sqlite3
7.
8. connection = sqlite3.connect('juridico_tfm.db')
9. cur = connection.cursor()
10.
11.    sql = "SELECT id, ponente FROM ponente"
12.    cur.execute(sql)
13.    busqueda = cur.fetchall()
14.
15.    contador_mujer = 0
16.    contador_hombre = 0
17.    contador_mixto= 0
18.    contador_nada = 0
19.
20.    for nombre in busqueda:
21.            juez = nombre[1]
22.            id = nombre[0]
23.            juez_split = juez.split()
24.
25.            es_mujer = 0
26.            es_hombre = 0
27.
28.            for n in juez_split:
29.                    #quitamos los acentos
30.                    s = re.sub(
31.                            r"([^n\u0300-
   \u036f]|n(?!\u0303(?![\u0300-\u036f])))[\u0300-\u036f]+", r"\1",
32.                            normalize( "NFD", n), 0, re.I
33.                            )
34.
35.                    # -> NFC
36.                    s = normalize( 'NFC', s)
37.                    #ponemos todas las letras en mayuscula
38.                    palabra = s.upper()
39.
40.                    if palabra in nombres_mujer:
41.                            es_mujer = 1
```

```
                                sql = "UPDATE ponente set genero =
    'MUJER' WHERE id=" + str(id)
                                cur.execute(sql)
                                break
                        if palabra in nombres_hombre:
                                es_hombre = 1
                                sql = "UPDATE ponente set genero =
    'HOMBRE' WHERE id=" + str(id)
                                cur.execute(sql)
                                break

            if es_mujer:
                    contador_mujer = contador_mujer + 1
            elif es_hombre:
                    contador_hombre = contador_hombre + 1
            else:
                    contador_nada = contador_nada + 1
    print ("mujer:", str(contador_mujer))
    print ("hombre:", str(contador_hombre))
    print ("nada:", str(contador_nada))

    connection.commit()
```

8.2.5 Normalización – Stemming de palabras

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import RegexpTokenizer
from nltk.stem import SnowballStemmer
import re
from unicodedata import normalize
import pymysql
import sqlite3

    #ejecutar esto únicamente la primera vez
    #nltk.download('stopwords')
    #nltk.download('punkt')

    def resumir_texto(texto):
            texto = texto.upper()
            tokenizer = RegexpTokenizer(r'\w+')
            #pasamos el texto a tokens, es decir, una lista de todas
    las palabras
            tokens = tokenizer.tokenize(texto)

            #listado de palabras que no aportan nada de NLTK
            #primera importamos la lista de palabras que tiene la
    libreria NLTK
            palabras_a_quitar = set(stopwords.words('Spanish'))
            #Pasamos todas las palabras a mayusculas
            palabras_a_quitar = [x.upper() for x in palabras_a_quita
    r]
            palabras_a_quitar = palabras_a_quitar
    + ['FALLO', 'IR','ARRIBA']
            ############
            tokens_v2 = []

```

```
30.           #creamos una lista con las palabras importantes, es
   decir, aquellas no esten en palabras_a_quitar
31.           for palabra in tokens:
32.                   if palabra not in palabras_a_quitar:
33.                           tokens_v2.append(palabra)
34.
35.           #convertimos una lista en un set, que es una lista pero
   no tiene elementos duplicados
36.           tokens_v3 = set(tokens_v2)
37.
38.           #aplicamos un algoritmo de normalizacion de palabras.
   Esto reduce las palabras a su raiz
39.           stemmer = SnowballStemmer('spanish')
40.           #para cada token en tokens_v3, le pasamos la funcion
   steemer.stem que devuelve el token pero normalizado
41.           #esto lo vamos a introducir en la lista token_v4
42.           tokens_v4 = [stemmer.stem(i) for i in tokens_v3]
43.
44.           #volvemos a quitar duplicados
45.           tokens_v5 = set(tokens_v4)
46.
47.           #ahora quitamos todos los tokens que no contengan letras
48.           tokens_v6 = list(tokens_v5)
49.           for x in range(1,5):
50.                   for token in tokens_v6:
51.                           for c in token:
52.                                   if c < 'A' or c > 'z':
53.                                           tokens_v6.remove(token)
54.                                           break
55.
56.           #preparamos la lista final en tipo STRING para
   insertarla en la base de datos
57.           lista_final = ""
58.           for token in tokens_v6:
59.                   lista_final += token + ', '
60.
61.           lista_final  = lista_final [:-2]
62.
63.           return lista_final
64.
65.   connection = sqlite3.connect('juridico_tfm.db')
66.   #conexion a la base de datos
67.   cur = connection.cursor() #pone un cursor que apunta a la BD de
   tal manera que las operaciones que hagas vayan a la BD
68.
69.   sql = "SELECT * FROM `resoluciones_judiciales`"
70.   cur.execute(sql)
71.   busqueda = cur.fetchall()
72.
73.   for i in busqueda:
74.           id = i[0]
75.           cabecera = i[2]
76.           cabecera_normalizada = resumir_texto(cabecera)
77.
78.           sql = "UPDATE `resoluciones_judiciales` SET
   `cabecera_normaliz`='" + cabecera_normalizada + "' WHERE id =
   " + str(id)
79.           cur.execute(sql)
80.
81.           print(id)
82.
```

```
83.    connection.commit()
```

8.2.6 Term Frequency – Inverse Document Frequency

```
1. from sklearn.feature_extraction.text import TfidfVectorizer
2. import pymysql
3. import sqlite3
4.
5. connection = sqlite3.connect('juridico_tfm.db')
6. cur = connection.cursor()
7.
8. #buscamos todos los fallos normalizados
9. sql = "SELECT id, cabecera_normaliz FROM resoluciones_judiciales"
10.    cur.execute(sql)
11.    busqueda = cur.fetchall()
12.
13.    #metemos todos los fallos en la lista, porque queremos en una
   lista todos los fallos
14.    lista_fallos_nor = []
15.    for cabecera in busqueda:
16.            lista_fallos_nor.append(cabecera[1])
17.
18.    #creamos el vectorizer
19.    vectorizer = TfidfVectorizer() #funcion que inicializa el vector
20.    #pasamos el vectorizer a la lista de fallos
21.    vectorizer.fit(lista_fallos_nor)
22.    #lista de puntuaciones por cada palabra
23.    puntuaciones = vectorizer.idf_
24.    #diccionario de palabras
25.    vocabulario = vectorizer.vocabulary_
26.
27.    #creamos un diccionario con las palabras y la puntuacion de cada
   una
28.    vocabulario_punt = {}
29.    i = 0
30.    while i < len(puntuaciones):
31.            vocabulario_punt[str(list(vocabulario.keys())[i])] = pun
   tuaciones[i]
32.            i += 1
33.
34.    print (vocabulario_punt)
35.    #media de puntuaciones
36.    suma = 0
37.    for p in puntuaciones:
38.            suma += p
39.
40.    media = suma / len(puntuaciones)
41.    #definimos la nota de corte
42.    corte = media
43.    minimo = 4
44.    maximo = 6
45.
46.    print('Se van a eliminar las palabras con una puntuacion menor
   a: ' + str(minimo))
47.    print('Se van a eliminar las palabras con una puntuacion mayor
   a: ' + str(maximo))
48.
49.    #volvemos a buscar todos los fallos normalizados
50.    sql = "SELECT id, cabecera_normaliz FROM
   resoluciones_judiciales"
51.    cur.execute(sql)
```

```
52.    busqueda = cur.fetchall()
53.
54.    palabras_eliminadas = 0
55.    total_palabras = 0
56.    for resolucion in busqueda:
57.            lista_palabras = resolucion[1].split(', ')
58.            lista_palabras_texto = resolucion[1]
59.            id_resolucion = resolucion[0]
60.
61.            #en la lista, comparamos cada palabra con nuestra
   puntuacion de corte
62.            for palabra in lista_palabras:
63.                    #para cada palabra, la buscamos en el
   diccionario de vocabulario_punt
64.                    #si al buscar la palabra en el diccionario
   obtenemos un error, esa palabra la quitamos
65.                    #asi solventamos algunos errores
66.                    total_palabras += 1
67.
68.                    try:
69.                            if vocabulario_punt[str(palabra)] < mini
   mo or vocabulario_punt[str(palabra)] > maximo:
70.                                    #si al buscar la palabra en el
   diccionario obtenemos el valor y es menor que nuestra nota de
   corte,
71.                                    #la eliminamos de la lista
72.                                    lista_palabras_texto = lista_pal
   abras_texto.replace(' ' + str(palabra) + ',', '')
73.                                    palabras_eliminadas += 1
74.                    except:
75.                            lista_palabras_texto = lista_palabras_te
   xto.replace(' ' + str(palabra) + ', ', '')
76.                            palabras_eliminadas += 1
77.
78.                    sql = "UPDATE resoluciones_judiciales SET
   palabras_utiles = '" + lista_palabras_texto + "' WHERE id=
   " + str(id_resolucion)
79.                    cur.execute(sql)
80.
81.
82.    print('Palabras eliminadas: ' + str(palabras_eliminadas))
83.    print('Total palabras: ' + str(total_palabras))
84.
85.    connection.commit()
86.    connection.close()
```

8.2.7 Tabla de palabras útiles

```
1. from sklearn.feature_extraction.text import TfidfVectorizer
2. import pymysql
3. import sqlite3
4. import sys
5. import nltk
6. from nltk.corpus import stopwords
7. from nltk.tokenize import RegexpTokenizer
8. from nltk.stem import SnowballStemmer
9. import re
10.
11.    from unicodedata import normalize
12.    connection = sqlite3.connect('juridico_tfm.db')
13.    cur = connection.cursor()
```

```
14.
15.    #buscamos todos los fallos normalizados
16.    sql = "SELECT id, cabecera, palabras_utiles FROM
   resoluciones_judiciales"
17.    cur.execute(sql)
18.    busqueda = cur.fetchall()
19.
20.    for resolucion in busqueda:
21.            id_resolucion = resolucion[0]
22.            cabecera = resolucion[1]
23.            palabras = resolucion[2]
24.
25.            # Transformamos la cabecera en tokens (uno por palabra)
   reducidos a la raiz
26.            texto = cabecera.upper()
27.            tokenizer = RegexpTokenizer(r'\w+')
28.            cabecera_tokens = tokenizer.tokenize(texto)
29.
30.            stemmer = SnowballStemmer('spanish')
31.            cabecera_tokens = [stemmer.stem(i) for i in cabecera_tok
   ens]
32.            ##########
33.
34.            vectorizer = TfidfVectorizer()
35.            vectorizer.fit(cabecera_tokens)
36.            puntuaciones = vectorizer.idf_
37.            vocabulario = vectorizer.vocabulary_
38.            vocabulario_punt = {}
39.            i = 0
40.            while i < len(puntuaciones):
41.                    vocabulario_punt[str(list(vocabulario.keys())[i]
   )] = puntuaciones[i]
42.                    i += 1
43.
44.            palabras = palabras.replace(' ', '').split(',')
45.
46.            for palabra in palabras:
47.                    #print(palabra + ': ' +
   str(vocabulario_punt[palabra]))
48.                    try:
49.                            sql = "INSERT INTO
   `palabras_utiles`(`id`, `id_resolucion`, `palabra`, `puntuacion`)
   VALUES (NULL," + str(id_resolucion) + ",'" + palabra
   + "','" + str(vocabulario_punt[palabra]) + "')"
50.                            cur.execute(sql)
51.                    except Exception as e:
52.                            print(str(e))
53.
54.    connection.commit()
55.    connection.close()
```

8.2.8 Preparación de la base de datos y transformación en CSV

```
1. from sklearn.feature_extraction.text import TfidfVectorizer
2. import pymysql
3. import sqlite3
4. import sys
5. import re
6. from unicodedata import normalize
7. # lista de documentos de texto
8.
```

```
connection = sqlite3.connect('juridico_tfm.db')
    cur = connection.cursor()

    #seleccionamos todos los campos de palabras utiles
    sql = "SELECT id, id_resolucion, palabra, puntuacion FROM
palabras_utiles"
    cur.execute(sql)
    busqueda = cur.fetchall()

    #creamos una lista vacía en la que incluiremos las palabras que
queremos que sean campos (las palabras utiles)
    total_palabras = []
    #anadimos todas las palabras a la lista
    for palabra in busqueda:
            total_palabras.append(palabra[2])

    total_palabras_unicas = list(set(total_palabras))

    total_palabras_unicas = sorted(total_palabras_unicas, reverse =
True)[::-1]

    total_palabras_unicas = ['id_resolucion'] +
total_palabras_unicas

    sql = "SELECT id FROM resoluciones_judiciales"
    cur.execute(sql)
    busqueda2 = cur.fetchall()


    componentes = []
    for resolucion in busqueda2:
            #inicializamos una lista con todo 0 igual al numero de
palabras mas 1 (en al primera posicion metemos la ID de la
sentencia)
            lista = [0 for i in range(len(total_palabras_unicas))]
            lista[0] = resolucion[0]
            componentes.append(lista)
    #con esto hemos creado una lista llena de ceros teniendo en
cuenta como rango la longitud de la lista y hemos creado una lista
de

    for palabra in busqueda:
            id_resolucion = palabra[1]
            palab = palabra[2]
            puntuacion = palabra[3]

            for linea in componentes:
                    if linea[0] == id_resolucion:

                            indice = total_palabras_unicas.index(pal
ab)
                            linea[indice] = puntuacion
                            break

    f = open("componentes_principales.csv","w+")
    for p in total_palabras_unicas:
            f.write(str(p) + ',')
    f.write('\n')

    #recorremos toda la "tabla" y pintamos todas las lineas.
    for componente in componentes:
```

```
61.             for e in componente:
62.                     f.write(str(e) + ',')
63.             f.write('\n')
```

8.2.9 Unión de todas las variables y creación de dataset definitivo

```
1. from sklearn.feature_extraction.text import TfidfVectorizer
2. import pymysql
3. import sqlite3
4. import sys
5. import re
6. from unicodedata import normalize
7. # lista de documentos de texto
8.
9. connection = sqlite3.connect('juridico_tfm.db')
10.    cur = connection.cursor()
11.
12.
13.    sql = "SELECT rj.id, p.id as ponente, p.genero, tribunal,
   fecha_reforma, sentido_fallo FROM resoluciones_judiciales rj  LEFT
   JOIN ponente p ON rj.ponente = p.id"
14.
15.    cur.execute(sql)
16.    busqueda = cur.fetchall()
17.
18.    sentencias = []
19.
20.    #seleccionamos todas las sentencias y la gaurdamos en una lista
21.    for sentencia in busqueda:
22.            id_sentencia = sentencia[0]
23.            ponente = sentencia[1]
24.            genero = 1 if sentencia[2] == 'HOMBRE' else 0
25.            tribunal = sentencia[3]
26.            fecha_reforma = sentencia[4]
27.            sentido_fallo = sentencia[5]
28.
29.            sentencias.append([id_sentencia,ponente,genero,tribunal,
   fecha_reforma,sentido_fallo])
30.
31.    #abrimos un nuevo fichero donde insertaremos todos los datos
32.    f = open("BD_juridica.csv","w+")
33.    try:
34.            #abrimos el archivo Matriz_PCA y vamos leyendo linea por
   linea
35.            with open('matriz_PCA.csv') as fp:
36.
37.                    line = fp.readline()
38.                    lineaux = line
39.
40.                    for c in line:
41.                            if c != ',':
42.                                    lineaux = lineaux[1:]
43.                            else:
44.                                    break
45.
46.                    #escribimos el encabezado, que es la primera
   columna en nuestro nuevo fichero
47.                    f.write('"id","ponente","genero","tribunal","fec
   ha_reforma","sentido_fallo"' + lineaux)
48.
49.                    cont = 0
```

```
                    while line:
                            line = fp.readline()
                            lineaux = line
                            for c in line:
                                    if c != ',':
                                            lineaux = lineaux[1:]
                                    else:
                                            break

                            sentencia = sentencias[cont]

                            f.write(str(sentencia[0]) + ',' + str(sentencia[1]) + ',' + str(sentencia[2]) + ',' + str(sentencia[3]) + ',' + str(sentencia[4]) + ',' + str(sentencia[5]) + lineaux)
                            cont += 1

    except:
            pass
```

8.3 Análisis de Componentes Principales

8.3.1 Script R análisis de Componentes Principales

```
library(irlba)
library(car)
library(readxl)
library(factoextra)
library(FactoMineR)

tabla_para_componentes_principales_excel <- read_excel("C:/Users/Mélani EV/Desktop/TFM Jurimetrics/tabla para componentes principales excel.xlsx")
View(tabla_para_componentes_principales_excel)

   x<-tabla_para_componentes_principales_excel
   x<-x[-1] #quitamos la variable id

   options(max.print=999999)

   #n es el número de componentes que quieres obtener (500 está bien)
   p1 <- prcomp_irlba(x, n=500, scale. = T)
   summary(p1) # información sobre la importancia de las componentes

   #Nos vamos a quedar con 200 componentes porque explican el 50% de los datos, asi que lo guardamos para trabajar con ello
   #y hacer nuestros modelos

   p2 <- prcomp_irlba(x, n=200, scale. = T)

   # la propia función te devuelve las n componentes principales de las observaciones (así que no hay que multiplicar las matrices, ya lo hace la función)
   datos<-p2$x #estos datos ya servirían parra modelizar, aunque habría que añadirles cualquier otra variable que tengas, como el sexo del juez, aquello de la reforma, etc.

   write.csv(datos, "matriz_PCA.csv")
```

```
28.
29.    # matriz de rotación
30.    p1$rotation
31.
32.    #palabras mas importantes de cada componente
33.    i<-1 #indice del factor a interpretar
34.    nn<-10 #número de palabras más importantes
35.    bb<-
   sort(abs(p1$rotation[,i]),index.return=T,decreasing=T)$ix[1:nn]
36.    colnames(x)[bb] #x es la matriz original
37.
38.    i<-2 #indice del factor a interpretar
39.    nn<-10 #número de palabras más importantes
40.    bb<-
   sort(abs(p1$rotation[,i]),index.return=T,decreasing=T)$ix[1:nn]
41.    colnames(x)[bb] #x es la matriz original
42.
43.    i<-3 #indice del factor a interpretar
44.    nn<-10 #número de palabras más importantes
45.    bb<-
   sort(abs(p1$rotation[,i]),index.return=T,decreasing=T)$ix[1:nn]
46.    colnames(x)[bb] #x es la matriz original
47.
48.    i<-4 #indice del factor a interpretar
49.    nn<-10 #número de palabras más importantes
50.    bb<-
   sort(abs(p1$rotation[,i]),index.return=T,decreasing=T)$ix[1:nn]
51.    colnames(x)[bb] #x es la matriz original
52.
53.    i<-10 #indice del factor a interpretar
54.    nn<-10 #número de palabras más importantes
55.    bb<-
   sort(abs(p1$rotation[,i]),index.return=T,decreasing=T)$ix[1:nn]
56.    colnames(x)[bb] #x es la matriz original
57.
58.
59.    #Scree plot
60.    salida<-summary(p1)
61.    plot(salida$importance[1,], main="Scree plot",
   ylab = "autovalores", xlab = "n° componentes")
62.
63.    #library("factoextra") scree plot automatico
64.    fviz_eig(p1, addlabels=TRUE, hjust = -0.3,
65.             linecolor ="red") + theme_minimal()
66.    fviz_eig(p1, geom="line")
67.
68.    fviz_eig(p1, ncp = 500, geom = "line")
69.
70.
71.    #% varianza acumulada
72.    plot(salida$importance[3,],type="l",lty=4,main="Varianza
   acumulada", ylab = "porcentaje de varianza acumulada",
   xlab = "componentes")
73.    points(salida$importance[2,],col=2,type="l") #para añadir la
   proporción
74.    abline(h=0.5, col=c("green"))
75.    legend('topleft','groups',c("Var acumulada", "Var explicada
   indiv", "50% datos explicados"), lty=c(1,1),
76.           lwd=c(1,1,1), col=c("black","red","green"),cex=0.6)
77.
78.
```

```
#individuals (libreria factoMineR)
fviz_pca_ind(p1, geom.ind = "point",
             col.ind = "#FC4E07",
             axes = c(1, 2),
             pointsize = 1.5)

#Scree Plot (libreria factoextra)
fviz_screeplot(p1, addlabels = TRUE, ylim = c(0, 0.9))

# Top 10 variables que más contribuyen a PC1
fviz_contrib(p1, choice = "var", axes = 1, top = 10)

# Top 15 variables que más contribuyen a PC7
fviz_contrib(p1, choice = "var", axes = 7, top = 15)
```

8.3.2 Gráficos sobre análisis de componentes principales

A continuación se incluyen algunos gráficos que no son relevantes para estudiar en el cuerpo del trabajo por su escasa claridad pero que son utilizados en análisis clásico de componentes principales, por lo que se incluyen por si son de interés del lector.

Gráfico de individuos (sentencias) en las dimensiones 1 y 2

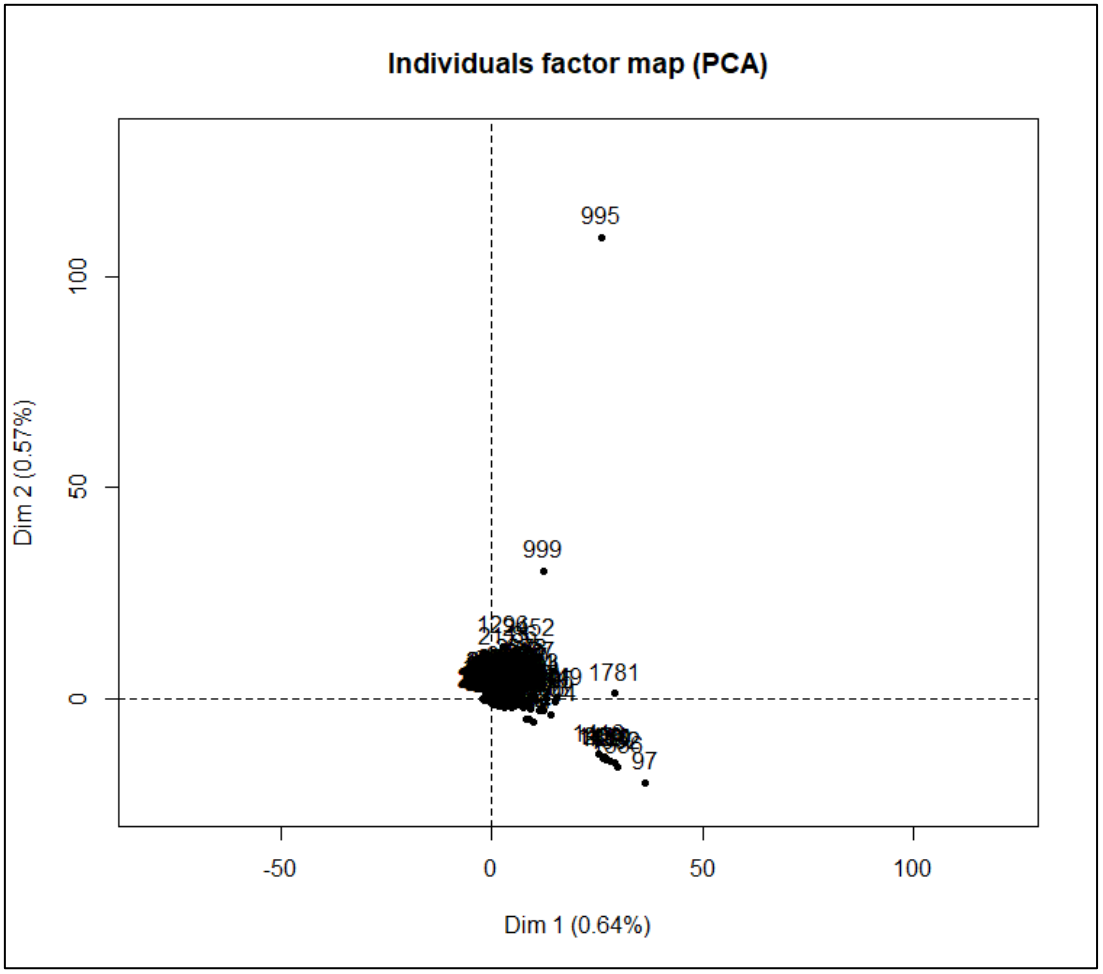
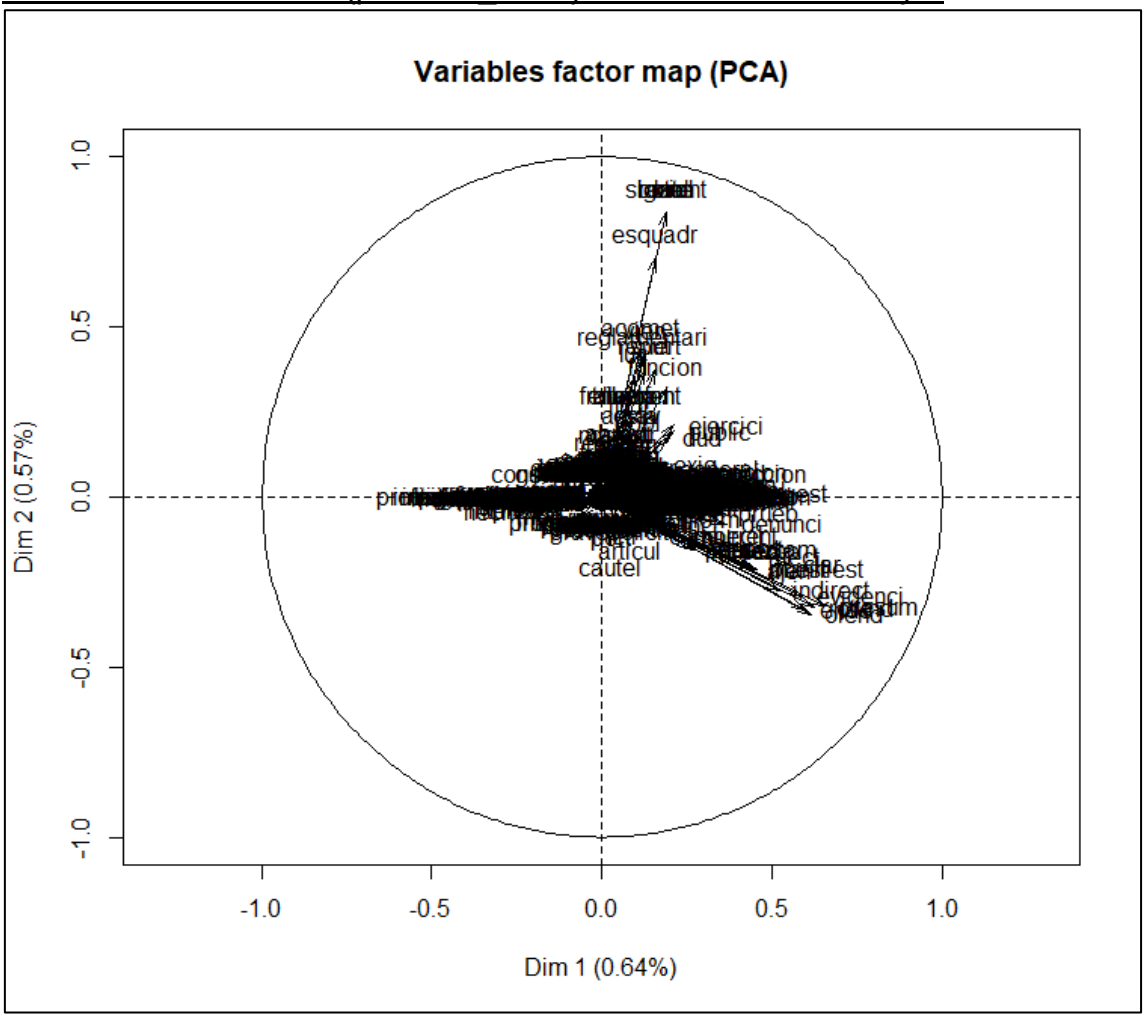


Gráfico de las variables (palabras_utiles) en las dimensiones 1 y 2



8.4 Modelización

8.4.1 Modelización en SAS

8.4.1.1 Script SAS

A continuación se proporciona el script del análisis y modelización realizado en SAS gracias a las macros proporcionadas por el profesor Portela en la asignatura “Técnicas de Machine Learning”.

```
libname discoc "C:\Users\Mélani EV\Desktop\TFM Jurimetrics"; run;
data ml;
set discoc.datos_pruebaml;
run;
proc freq data=ml;run;


proc contents data=ml out=sal;run;quit;
data;set sal;put name @@;run;

/* PRUEBAS CON LA RED*/
```

```
%macro
neuralbinariabasica(archivo=,listconti=,listclass=,vardep=,nodos=,cort
e=,semilla=,porcen=,algo=levmar);
title '';
data archivobase;set &archivo nobs=nume;ene=int(&porcen*nume);
call symput('ene',left(ene));
run;

proc sort data=archivobase;by &vardep;run;

proc surveyselect data=archivobase out=muestra outall N=&ene
seed=&semilla;
/*si se quiere estratificacion en el muestreo quitar los comentarios en
strata*/
/*strata &vardep /alloc=proportional;*/ run;
data train valida;set muestra;if selected=1 then output train;else
output valida;run;

PROC DMDB DATA=train dmdbcat=cataprueba;
target &vardep;
var &listconti;
class &listclass &vardep;
run;

%if &listclass ne %then %do;
proc neural data=train dmdbcat=cataprueba;
input  &listconti;
input &listclass /level=nominal;
target  &vardep /level=nominal;
hidden &nodos;
prelim 5;
train tech=&algo;
score data=valida out=salpredi outfit=salfit ;
run;
%end;

%else %do;
proc neural data=train dmdbcat=cataprueba;
input  &listconti;
target  &vardep /level=nominal;
hidden &nodos;
prelim 5;
train tech=&algo;
score data=valida out=salpredi outfit=salfit ;
run;
%end;

data salpredi;set salpredi;if p_&vardep.1>&corte/100 then predi1=1;else
predi1=0;run;
proc freq data=salpredi;tables predi1*&vardep/out=sal1;run;

/* Cálculo de estadísticos */

data estadisticos (drop=count percent predi1 &vardep);
retain vp vn fp fn suma 0;
set sal1 nobs=nume;
suma=suma+count;
if predi1=0 and &vardep=0 then vn=count;
if predi1=0 and &vardep=1 then fn=count;
if predi1=1 and &vardep=0 then fp=count;
if predi1=1 and &vardep=1 then vp=count;
```

```
if _n_=nume then do;
if vn=. then vn=0;if fn=. then fn=0;if vp=. then vp=0;if fp=. then fp=0;
porcenVN=vn/suma;
porcenFN=FN/suma;
porcenVP=VP/suma;
porcenFP=FP/suma;
sensi=vp/(vp+fn);
especif=vn/(vn+fp);
tasafallos=1-(vp+vn)/suma;
tasaciertos=1-tasafallos;
precision=vp/(vp+fp);
if vp=0 then precision=0;
if vp=0 then sensi=0;
if vn=0 then especif=0;
F_M=2*Sensi*Precision/(Sensi+Precision);
output;
end;
run;
proc print data=estadisticos;run;

%mend;
%neuralbinariabasica(archivo=ml,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
vardep=sentido_fallo,nodos=20,corte=50,semilla=12345,porcen=0.80);

%neuralbinariabasica(archivo=ml,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
vardep=sentido_fallo,nodos=20,corte=100,semilla=12345,porcen=0.80);

/* Con levmar */
%macro
variar(seminicio=,semifin=,inicionodos=,finalnodos=,increnodos=);
title '';
data union;run;
%do semilla=&seminicio %to &semifin;
%do nodos=&inicionodos %to &finalnodos %by &increnodos;
      %neuralbinariabasica(archivo=ml,
      listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43
      PC4   PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96
      PC59  PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
vardep=sentido_fallo,nodos=10,corte=50,semilla=12345,porcen=0.80,algo=
levmar);
   data                                                estadisticos;set
estadisticos;nodos=&nodos;semilla=&semilla;run;
   data union;set union estadisticos;run;
%end;
%end;
proc sort data=union;by nodos;run;
proc boxplot data=union;plot (porcenVN porcenFN porcenVP porcenFP
sensi especif tasafallos tasaciertos precision F_M)*nodos;run;
%mend;

%variar(seminicio=12345,semifin=12355,inicionodos=2,finalnodos=10,incr
enodos=2);
```

```
/* Con Bprop */
%macro
variar(seminicio=,semifin=,inicionodos=,finalnodos=,increnodos=);
title '';
data union;run;
%do semilla=&seminicio %to &semifin;
%do nodos=&inicionodos %to &finalnodos %by &increnodos;
   %neuralbinariabasica(archivo=ml,
      listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43
      PC4   PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96
      PC59  PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
vardep=sentido_fallo,nodos=&nodos,corte=50,semilla=&semilla,porcen=0.8
0,algo=bprop mom=0.2 learn=0.1);
   data                                                estadisticos;set
estadisticos;nodos=&nodos;semilla=&semilla;run;
   data union;set union estadisticos;run;
%end;
%end;
proc sort data=union;by nodos;run;
proc boxplot data=union;plot (porcenVN porcenFN porcenVP porcenFP
sensi especif tasafallos tasaciertos precision F_M)*nodos;run;
%mend;

%variar(seminicio=12345,semifin=12355,inicionodos=3,finalnodos=15,incr
enodos=3);
%macro
calcular(archivo=,test=0,archivotest=,vardepen=,conti=,categor=,corte=
,porcaptura=0);
%if      &test=1      %then      %do;data      union;set      &archivo
&archivotest(in=vie);vardep=&vardepen;if vie=1 then vardep=.;run;%end;
%else %do;data union;set &archivo;vardep=&vardepen;run;%end;

            ods output ROCAssociation=roca;

            proc logistic data=union;
            %if   (&categor   ne)   %then   %do;class   &categor;model
vardep=&conti &categor ;%end;
            %else %do;model vardep=&conti;%end;
            output out=sal p=predi;/*roc;*/run;
            data roca;set roca;if rocmodel='Model' then output;run;

            data  sal2;set  sal;pro=1-predi;if  pro>&corte  then  pre11=1;
else pre11=0;
             output;run;

            %if   &test=1   %then   %do;proc   freq   data=sal2;tables
pre11*&vardepen/out=sal3;
                                          where vardep=.;
                                          run;
                                          %end;
            %else        %do;proc        freq        data=sal2;tables
pre11*&vardepen/out=sal3;
                                          run;
                                          %end;
            data estadisticos (drop=count percent pre11 &vardepen);
            if _n_=1 then set roca;
            retain vp vn fp fn suma 0;
            set sal3 nobs=nume;
            suma=suma+count;
```

```
            if pre11=0 and &vardepen=0 then vn=count;
            if pre11=0 and &vardepen=1 then fn=count;
            if pre11=1 and &vardepen=0 then fp=count;
            if pre11=1 and &vardepen=1 then vp=count;
            if _n_=nume then do;
            porcenVN=vn/suma;
            porcenFN=FN/suma;
            porcenVP=VP/suma;
            porcenFP=FP/suma;
            sensi=vp/(vp+fn);
            especif=vn/(vn+fp);
            tasafallos=1-(vp+vn)/suma;
            tasaciertos=1-tasafallos;
            precision=vp/(vp+fp);
            F_M=2*Sensi*Precision/(Sensi+Precision);
            Mcc=VP*VN-
FP*FN;be=(VP+FP)*(VP+FN)*(VN+FP)*(VN+FN);be=sqrt(be);
            MCC=MCC/be;
            Youden=especif+sensi-1;
            AUC=Area;
            output;
            end;
            run;
            %if &test=1 %then %do;data sal2;set sal2;if vardep=.;run;
%end;
            %if &porcaptura ne 0 %then %do;
            proc sort data=sal2;by descending pre11;
            data sal4;retain suma1 0;set sal2 nobs=nume;
                  if &vardepen=1 then suma1=suma1+1;
                  if           _n_=int(&porcaptura*nume)           then
do;ncapturados=suma1;capturados=suma1/_n_;ntot=_n_;output;
                  stop;end;
            run;
            data  estadisticos;set  estadisticos;if  _n_=1  then  set
sal4;run;
            %end;
data estadisticos;set estadisticos;
keep AUC F_M Mcc Youden ncapturados ntot capturados especif fn fp
porcenFN porcenFP porcenVN porcenVP precision
sensi tasaciertos tasafallos vn vp;
run;
proc print data=estadisticos;run;
%mend;

%calcular(archivo=ml,
vardepen=sentido_fallo,
conti=PC7   PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4   PC111
      PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59  PC125
      PC199 PC95  PC176 PC198 PC71  PC74  PC78,
categor=tribunal,corte=0.5 ,porcaptura=0.3);

/*EARLY STOPPING*/

%macro
redneuronalbinaria(archivo=,listclass=,listconti=,vardep=,porcen=,semi
lla=,ocultos=,meto=levmar,acti=);

PROC DMDB DATA=&archivo dmdbcat=catauno;
target &vardep;
var &listconti ;
class &vardep &listclass;
```

```
run;

data ooo;set &archivo;run;
data       datos;set       ooo       nobs=nume;tr=int(&porcen*nume);call
symput('tr',left(tr));u=ranuni(&semilla);run;
proc sort data=datos;by u;run;
data datos valida;set datos;if _n_>tr then output valida;else output
datos;run;

proc neural data=datos dmdbcat=catauno validata=valida graph;
input &listconti / id=i;
input &listclass / level=nominal;
target &vardep / level=nominal id=o error=ENT;
hidden &ocultos / id=h act=&acti;
nloptions maxiter=10000;
netoptions randist=normal ranscale=0.1 random=15115;
prelim 0;
train maxiter=10000 outest=mlpest estiter=1 technique=&meto;
score data=datos out=mlpout outfit=mlpfit;
score data=valida out=mlpout2 outfit=mlpfit2 role=valid;
run;

data mlpest2 ;
k=3;
retain iterepocas 0;
set mlpest nobs=nume;
call symput('numeroit',left(nume));
eval=_VOBJERR_;
x3=lag3(eval);
x6=lag6(eval);
if _n_>6 and eval>x3 and eval>x6 then iterepocas=_n_;
run;

data;
set mlpest2 nobs=nume;
if iterepocas ne 0 then do;
call symput('earlystop',left(iterepocas));
stop;
end;
else if _n_=nume and iterepocas=0 then do;iterepocas=&numeroit;
call symput('earlystop',left(iterepocas));
stop;
end;
run;

data fin;j=&earlystop;set mlpest point=j;output;stop;run;

data mlpest;set mlpest nobs=nume; if _n_=&earlystop then do;
cosa1=put(_OBJERR_,20.6) ;
cosa2=put(_VOBJERR_,20.6) ;
end;
else do;cosa1=' ';cosa2=' ';end;
run;

title1
h=2 box=1 j=c c=red 'TRAIN' c=blue '  VALIDA'
h=1.5 j=c c=black "EARLY STOPPING=&earlystop " "semilla=&semilla"
h=1  j=c  c=green  "NODOS  OCULTOS:  &ocultos    "  "  METODO:  &meto  "
"ACTIVACIÓN: &acti"
h=1 j=c c=black "EL ERROR ES EL VALOR DE LA ENTROPÍA";
;
```

```
symbol1   c=red   v=circle   i=join   pointlabel=("#cosa1"   h=1   c=red
position=bottom  j=c);
symbol2   c=blue   v=circle   i=join   pointlabel=("#cosa2"   h=1   c=blue
position=top j=c);

axis1 label=none;
proc gplot data=mlpest;plot _OBJERR_ *_iter_=1 _VOBJERR_*_iter_=2
/overlay href=&earlystop vaxis=axis1 haxis=axis1 ;run;

proc print data=fin;
var  _iter_ _OBJERR_  _AVERR_  _VNOBJ_   _VOBJ_  _VOBJERR_  _VAVERR_
;run;

%mend;
%redneuronalbinaria(archivo=ml,
   listconti=PC7  PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
   listclass=tribunal,
vardep=sentido_fallo,porcen=0.80,semilla=442711,ocultos=6,meto=BPROP
mom=0.2 learn=0.1,acti=TANH);

%redneuronalbinaria(archivo=ml,
   listconti=PC7  PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
   listclass=tribunal,
vardep=sentido_fallo,porcen=0.80,semilla=442711,ocultos=6,meto=BPROP
mom=0.2 learn=0.1,acti=SIN);
/*estas dos ponen que pare en 13*/

/*acti= TANH LOG ARC LIN SIN SOF GAU*/

%redneuronalbinaria(archivo=ml,
   listconti=PC7  PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
   listclass=tribunal,
vardep=sentido_fallo,porcen=0.80,semilla=442711,ocultos=6,meto=BPROP
mom=0.2 learn=0.1,acti=GAU);

%redneuronalbinaria(archivo=ml,
   listconti=PC7  PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
   listclass=tribunal,
vardep=sentido_fallo,porcen=0.80,semilla=442711,ocultos=6,meto=BPROP
mom=0.2 learn=0.1,acti=SOF);

/*estas dos ponen que pare en 7*/

/* MACRO CRUZADA BINARIA NEURAL */


%macro
cruzadabinarianeural(archivo=,vardepen=,conti=,categor=,ngrupos=,sinic
io=,sfinal=,nodos=,algo=,early=300,
acti=tanh,basura=c:\basura.txt,objetivo=tasafallos);
title ' ';
data final;run;
```

```
proc printto print=&basura;

/* Bucle semillas */
%do semilla=&sinicio %to &sfinal;
      data dos;set &archivo;u=ranuni(&semilla);
      proc sort data=dos;by u;run;
      data dos (drop=nume);
      retain grupo 1;
      set dos nobs=nume;
      if _n_>grupo*nume/&ngrupos then grupo=grupo+1;
      run;
      data fantasma;run;
      %do exclu=1 %to &ngrupos;

      data trestr tresval;
            set dos;if grupo ne &exclu then output trestr;else output
tresval;
            PROC DMDB DATA=trestr dmdbcat=catatres;
            target &vardepen;
            var &conti;
            class &vardepen;
            %if &categor ne %then %do;class &categor &vardepen;%end;
            run;
            proc neural data=trestr dmdbcat=catatres random=789 ;
            input &conti;
            %if     &categor     ne     %then     %do;input     &categor
/level=nominal;%end;
            target &vardepen /level=nominal;
            hidden &nodos /acti=&acti;/*<<<<<******PARA DATOS LINEALES
ACT=LIN (funciónde activación lineal)
            NORMALMENTE PARA DATOS NO LINEALES MEJOR ACT=TANH */
            /* A PARTIR DE AQUI SON ESPECIFICACIONES DE LA RED, SE PUEDEN
CAMBIAR O AÑADIR COMO PARÁMETROS */

            /*nloptions maxiter=500*/;
            netoptions randist=normal ranscale=0.15 random=15459;
            /* Si se desea hacer early stopping se pone prelim 0 y se
marca como comentario
            la línea prelim 15...*/
            /*prelim 0 */
            prelim 15 preiter=10 pretech=&algo;
            train maxiter=&early outest=mlpest technique=&algo;
            score data=tresval role=valid out=sal ;
            run;
            data sal2;set sal;pro=1-%str(p_&vardepen)0;if pro>0.5 then
pre11=1; else pre11=0;run;
                        proc           freq           data=sal2;tables
pre11*&vardepen/out=sal3;run;

      data estadisticos (drop=count percent pre11 &vardepen);
            retain vp vn fp fn suma 0;
            set sal3 nobs=nume;
            suma=suma+count;
            if pre11=0 and &vardepen=0 then vn=count;
            if pre11=0 and &vardepen=1 then fn=count;
            if pre11=1 and &vardepen=0 then fp=count;
            if pre11=1 and &vardepen=1 then vp=count;
            if _n_=nume then do;
            porcenVN=vn/suma;
            porcenFN=FN/suma;
            porcenVP=VP/suma;
```

```
            porcenFP=FP/suma;
            sensi=vp/(vp+fn);
            especif=vn/(vn+fp);
            tasafallos=1-(vp+vn)/suma;
            tasaciertos=1-tasafallos;
            precision=vp/(vp+fp);
            F_M=2*Sensi*Precision/(Sensi+Precision);
            output;
            end;
            run;
      data fantasma;set fantasma estadisticos;run;
      %end;
      proc means data=fantasma sum noprint;var &objetivo;
      output out=sumaresi sum=suma mean=media;
      run;
      data sumaresi;set sumaresi;semilla=&semilla;
      data final (keep=suma media semilla);set final sumaresi;if suma=.
then delete;run;
%end;
proc printto ;
proc print data=final;run;
%mend;

%cruzadabinarianeural(archivo=ml,vardepen=sentido_fallo,conti=PC7
      PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4   PC111 PC13
      PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59  PC125 PC199
      PC95  PC176 PC198 PC71  PC74  PC78,

categor=tribunal,ngrupos=4,sinicio=12345,sfinal=12365,nodos=6,algo=qua
new,early=13,acti=tanh);
data final1;set final;modelo=1;

%cruzadabinarianeural(archivo=ml,vardepen=sentido_fallo,conti=PC7
      PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4   PC111 PC13
      PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59  PC125 PC199
      PC95  PC176 PC198 PC71  PC74  PC78,

categor=tribunal,ngrupos=10,sinicio=12345,sfinal=12365,nodos=6,algo=qu
anew,early=7,acti=gau);
data final2;set final;modelo=2;

%cruzadabinarianeural(archivo=ml,vardepen=sentido_fallo,conti=PC7
      PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4   PC111 PC13
      PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59  PC125 PC199
      PC95  PC176 PC198 PC71  PC74  PC78,

categor=tribunal,ngrupos=4,sinicio=12345,sfinal=12365,nodos=4,algo=qua
new,early=13,acti=tanh);
data final3;set final;modelo=3;

%cruzadabinarianeural(archivo=ml,vardepen=sentido_fallo,conti=PC7
      PC15  PC27  PC54  PC25  PC44  PC35  PC3   PC2   PC52  PC61  PC22
      PC9   PC55  PC42  PC36  PC166 PC43  PC64  PC40  PC4   PC32  PC105
      PC69  PC140 PC117 PC142 PC26  PC76  PC131 PC193 PC24  PC115 PC150
      PC56  PC134 PC125 PC112 PC63  PC46  PC176 PC16  PC104 PC20  PC14
      PC166 PC30  PC71,
categor=tribunal,ngrupos=4,sinicio=12345,sfinal=12365,nodos=6,algo=qua
new,early=13,acti=tanh);
data final4;set final;modelo=4;
```

```
%cruzadabinarianeural(archivo=ml,vardepen=sentido_fallo,conti=PC7
      PC15  PC27  PC54  PC25  PC44  PC35  PC3   PC2   PC52  PC61  PC22
      PC9   PC55  PC42  PC36  PC166 PC43  PC64  PC40  PC4   PC32  PC105
      PC69  PC140 PC117 PC142 PC26  PC76  PC131 PC193 PC24  PC115 PC150
      PC56  PC134 PC125 PC112 PC63  PC46  PC176 PC16  PC104 PC20  PC14
      PC166 PC30  PC71,
categor=,ngrupos=4,sinicio=12345,sfinal=12365,nodos=6,algo=quanew,earl
y=7,acti=gau);
data final5;set final;modelo=5;

%cruzadabinarianeural(archivo=ml,vardepen=sentido_fallo,conti=PC7
      PC15  PC27  PC54  PC25  PC44  PC35  PC3   PC2   PC52  PC61  PC22
      PC9   PC55  PC42  PC36  PC166 PC43  PC64  PC40  PC4   PC32  PC105
      PC69  PC140 PC117 PC142 PC26  PC76  PC131 PC193 PC24  PC115 PC150
      PC56  PC134 PC125 PC112 PC63  PC46  PC176 PC16  PC104 PC20  PC14
      PC166 PC30  PC71,

categor=,ngrupos=4,sinicio=12345,sfinal=12365,nodos=3,algo=bprop,early
=13,acti=tanh);
data final6;set final;modelo=6;

/* Representamos todas las redes*/
data union;set final1 final2 final3 final4 final5 final6;
proc boxplot data=union;plot media*modelo;run;


/*PROBANDO */

%macro
redneuronalbinaria(archivo=,listclass=,listconti=,vardep=,porcen=,semi
lla=,ocultos=,meto=quanew,acti=);

PROC DMDB DATA=&archivo dmdbcat=catauno;
target &vardep;
var &listconti ;
class &vardep &listclass;
run;

data ooo;set &archivo;run;
data      datos;set      ooo      nobs=nume;tr=int(&porcen*nume);call
symput('tr',left(tr));u=ranuni(&semilla);run;
proc sort data=datos;by u;run;
data datos valida;set datos;if _n_>tr then output valida;else output
datos;run;

proc neural data=datos dmdbcat=catauno validata=valida graph;
input &listconti / id=i;
input &listclass / level=nominal;
target &vardep / level=nominal id=o error=ENT;
hidden &ocultos / id=h act=&acti;
nloptions maxiter=10000;
netoptions randist=normal ranscale=0.1 random=15115;
prelim 0;
train maxiter=10000 outest=mlpest estiter=1 technique=&meto;
score data=datos out=mlpout outfit=mlpfit;
score data=valida out=mlpout2 outfit=mlpfit2 role=valid;
run;

data mlpest2 ;
k=3;
retain iterepocas 0;
```

```
set mlpest nobs=nume;
call symput('numeroit',left(nume));
eval=_VOBJERR_;
x3=lag3(eval);
x6=lag6(eval);
if _n_>6 and eval>x3 and eval>x6 then iterepocas=_n_;
run;

data;
set mlpest2 nobs=nume;
if iterepocas ne 0 then do;
call symput('earlystop',left(iterepocas));
stop;
end;
else if _n_=nume and iterepocas=0 then do;iterepocas=&numeroit;
call symput('earlystop',left(iterepocas));
stop;
end;
run;

data fin;j=&earlystop;set mlpest point=j;output;stop;run;

data mlpest;set mlpest nobs=nume; if _n_=&earlystop then do;
cosa1=put(_OBJERR_,20.6) ;
cosa2=put(_VOBJERR_,20.6) ;
end;
else do;cosa1=' ';cosa2=' ';end;
run;

title1
h=2 box=1 j=c c=red 'TRAIN' c=blue '  VALIDA'
h=1.5 j=c c=black "EARLY STOPPING=&earlystop " "semilla=&semilla"
h=1  j=c  c=green  "NODOS  OCULTOS:  &ocultos    "  "  METODO:  &meto  "
"ACTIVACIÓN: &acti"
h=1 j=c c=black "EL ERROR ES EL VALOR DE LA ENTROPÍA";
;

symbol1   c=red   v=circle   i=join   pointlabel=("#cosa1"   h=1   c=red
position=bottom  j=c);
symbol2   c=blue   v=circle   i=join   pointlabel=("#cosa2"   h=1   c=blue
position=top j=c);

axis1 label=none;
proc gplot data=mlpest;plot _OBJERR_ *_iter_=1 _VOBJERR_*_iter_=2
/overlay href=&earlystop vaxis=axis1 haxis=axis1 ;run;

proc print data=fin;
var  _iter_ _OBJERR_  _AVERR_  _VNOBJ_   _VOBJ_  _VOBJERR_  _VAVERR_
;run;

%mend;

/* EJEMPLO REDNEURONALBINARIA con el mejor set para red*/


%redneuronalbinaria(archivo=ml,listclass=tribunal,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
vardep=sentido_fallo,porcen=0.80,semilla=442711,ocultos=7,meto=bprop,a
cti=TANH);
```

```
%redneuronalbinaria(archivo=ml,listclass=tribunal,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
vardep=sentido_fallo,porcen=0.80,semilla=442711,ocultos=7,meto=quanew,
acti=TANH);


/* LA MACRO CRUZADARANDOMFORESTBIN REALIZA VALIDACIÓN CRUZADA REPETIDA
EN RANDOM FOREST PARA VARIABLE
DEPENDIENTE BINARIA */


%macro cruzadarandomforestbin(archivo=,vardep=,conti=,categor=,
maxtrees=100,variables=3,porcenbag=0.80,maxbranch=2,tamhoja=5,maxdepth
=10,pvalor=0.1,
ngrupos=4,sinicio=12345,sfinal=12355,objetivo=tasafallos);

data final;run;
/* Bucle semillas */
%do semilla=&sinicio %to &sfinal;

data dos;set &archivo;u=ranuni(&semilla);
      proc sort data=dos;by u;run;
      data dos ;
      retain grupo 1;
      set dos nobs=nume;
      if _n_>grupo*nume/&ngrupos then grupo=grupo+1;
      run;


data fantasma;run;

%do exclu=1 %to &ngrupos;
data tres;set dos;if grupo ne &exclu then vardep=&vardep;


ods listing close;
proc hpforest data=tres
maxtrees=&maxtrees
vars_to_try=&variables
trainfraction=&porcenbag
leafsize=&tamhoja
maxdepth=&maxdepth
alpha=&pvalor
exhaustive=5000
missing=useinsearch ;
target vardep/level=nominal;
input &conti/level=interval;
%if (&categor ne) %then %do;
      input &categor/level=nominal;
      %end;
score out=salo;
run;
ods listing ;

data salo;merge salo tres;
if p_vardep1>0.5 then pre11=1;else pre11=0;
if grupo=&exclu;
run;
```

```
      proc freq data=salo;tables pre11*&vardep/out=sal3;run;
            data estadisticos (drop=count percent pre11 &vardep);
            retain vp vn fp fn suma 0;
            set sal3 nobs=nume;
            suma=suma+count;
            if pre11=0 and &vardep=0 then vn=count;
            if pre11=0 and &vardep=1 then fn=count;
            if pre11=1 and &vardep=0 then fp=count;
            if pre11=1 and &vardep=1 then vp=count;
            if _n_=nume then do;
            porcenVN=vn/suma;
            porcenFN=FN/suma;
            porcenVP=VP/suma;
            porcenFP=FP/suma;
            sensi=vp/(vp+fn);
            especif=vn/(vn+fp);
            tasafallos=1-(vp+vn)/suma;
            tasaciertos=1-tasafallos;
            precision=vp/(vp+fp);
            F_M=2*Sensi*Precision/(Sensi+Precision);
            output;
            end;
            run;

data fantasma;set fantasma estadisticos;run;

%end;/* fin grupos */
      proc means data=fantasma sum noprint;var &objetivo;
      output out=sumaresi sum=suma mean=media;
      run;
      data sumaresi;set sumaresi;semilla=&semilla;
      data final (keep=suma media semilla);set final sumaresi;if suma=.
then delete;run;
%end;/* fin semillas validación cruzada repetida*/


proc print data=final;run;


%mend;

%cruzadarandomforestbin(
archivo=ml,vardep=sentido_fallo,
conti=PC7   PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4   PC111
      PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59  PC125
      PC199 PC95  PC176 PC198 PC71  PC74  PC78,
categor= tribunal,
maxtrees=100,variables=3,porcenbag=0.80,maxbranch=4,tamhoja=25,maxdept
h=5,pvalor=0.1,
ngrupos=4,sinicio=13345,sfinal=13355,objetivo=tasafallos);
data final7;set final;modelo=7;

%cruzadarandomforestbin(
archivo=ml,vardep=sentido_fallo,
conti=PC7   PC15  PC27  PC54  PC25  PC44  PC35  PC3   PC2   PC52  PC61
      PC22  PC9   PC55  PC42  PC36  PC166 PC43  PC64  PC40  PC4   PC32
      PC105 PC69  PC140 PC117 PC142 PC26  PC76  PC131 PC193 PC24  PC115
      PC150 PC56  PC134 PC125 PC112 PC63  PC46  PC176 PC16  PC104 PC20
      PC14  PC166 PC30  PC71,
categor=,
```

```
maxtrees=100,variables=3,porcenbag=0.80,maxbranch=4,tamhoja=25,maxdept
h=5,pvalor=0.1,
ngrupos=4,sinicio=13345,sfinal=13355,objetivo=tasafallos);
data final8;set final;modelo=8;

%cruzadarandomforestbin(
archivo=ml,vardep=sentido_fallo,
conti=PC7   PC15  PC27  PC54  PC35  PC44  PC35  PC52  PC43  PC42  PC17
      PC62  PC12  PC4   PC1   PC168 PC45  PC40  PC5   PC13  PC105 PC63
      PC188 PC187,
categor=tribunal,
maxtrees=100,variables=3,porcenbag=0.80,maxbranch=4,tamhoja=25,maxdept
h=5,pvalor=0.1,
ngrupos=4,sinicio=13345,sfinal=13355,objetivo=tasafallos);
data final9;set final;modelo=9;

%cruzadarandomforestbin(
archivo=ml,vardep=sentido_fallo,
conti=PC7   PC15  PC27  PC54  PC25  PC44  PC35  PC3   PC2   PC52  PC61
      PC22  PC9   PC55  PC42  PC36  PC166 PC43  PC64  PC40  PC4   PC32
      PC105 PC69  PC140 PC117 PC142 PC26  PC76  PC131 PC193 PC24  PC115
      PC150 PC56  PC134 PC125 PC112 PC63  PC46  PC176 PC16  PC104 PC20
      PC14  PC166 PC30  PC71,
categor=,
maxtrees=100,variables=3,porcenbag=0.80,maxbranch=6,tamhoja=20,maxdept
h=5,pvalor=0.1,
ngrupos=4,sinicio=13345,sfinal=13355,objetivo=tasafallos);
data final10;set final;modelo=10;

%cruzadarandomforestbin(
archivo=ml,vardep=sentido_fallo,
conti=PC7   PC15  PC27  PC54  PC25  PC44  PC35  PC3   PC2   PC52  PC61
      PC22  PC9   PC55  PC42  PC36  PC166 PC43  PC64  PC40  PC4   PC32
      PC105 PC69  PC140 PC117 PC142 PC26  PC76  PC131 PC193 PC24  PC115
      PC150 PC56  PC134 PC125 PC112 PC63  PC46  PC176 PC16  PC104 PC20
      PC14  PC166 PC30  PC71,
categor=,
maxtrees=100,variables=3,porcenbag=0.80,maxbranch=4,tamhoja=20,maxdept
h=4,pvalor=0.1,
ngrupos=4,sinicio=13345,sfinal=13355,objetivo=tasafallos);
data final11;set final;modelo=11;

data union;set final7 final8 final9 final10 final11;
proc boxplot data=union;plot media*modelo;run;


/* GRADIENT BOOSTING*/

%macro
cruzadatreeboostbin(archivo=,vardepen=,conti=,categor=,ngrupos=,sinici
o=,sfinal=,leafsize=5,
iteraciones=500,shrink=0.01,maxbranch=2,maxdepth=4,mincatsize=15,minob
s=20,objetivo=tasafallos);
data final;run;
/* Bucle semillas */
%do semilla=&sinicio %to &sfinal;
      data dos;set &archivo;u=ranuni(&semilla);
      proc sort data=dos;by u;run;
      data dos ;
      retain grupo 1;
      set dos nobs=nume;
```

```
      if _n_>grupo*nume/&ngrupos then grupo=grupo+1;
      run;
      data fantasma;run;
      %do exclu=1 %to &ngrupos;
            data tres;set dos;if grupo ne &exclu then vardep=&vardepen;

      proc treeboost data=tres
      exhaustive=1000 intervaldecimals=max
      leafsize=&leafsize iterations=&iteraciones maxbranch=&maxbranch
      maxdepth=&maxdepth   mincatsize=&mincatsize   missing=useinsearch
shrinkage=&shrink
      splitsize=&minobs;
      %if (&categor ne) %then %do;
      input &categor/level=nominal;
      %end;
      input &conti/level=interval;
      target vardep /level=binary;
      save fit=iteraciones importance=impor model=modelo rules=reglas;
      subseries largest;
      score out=sal;

      data sal2;set sal;pro=1-p_vardep0;if pro>0.5 then pre11=1; else
pre11=0;
            if grupo=&exclu then output;run;
            proc freq data=sal2;tables pre11*&vardepen/out=sal3;run;
            data estadisticos (drop=count percent pre11 &vardepen);
            retain vp vn fp fn suma 0;
            set sal3 nobs=nume;
            suma=suma+count;
            if pre11=0 and &vardepen=0 then vn=count;
            if pre11=0 and &vardepen=1 then fn=count;
            if pre11=1 and &vardepen=0 then fp=count;
            if pre11=1 and &vardepen=1 then vp=count;
            if _n_=nume then do;
            porcenVN=vn/suma;
            porcenFN=FN/suma;
            porcenVP=VP/suma;
            porcenFP=FP/suma;
            sensi=vp/(vp+fn);
            especif=vn/(vn+fp);
            tasafallos=1-(vp+vn)/suma;
            tasaciertos=1-tasafallos;
            precision=vp/(vp+fp);
            F_M=2*Sensi*Precision/(Sensi+Precision);
            output;
            end;
            run;

            data fantasma;set fantasma estadisticos;run;
      %end;
      proc means data=fantasma sum noprint;var &objetivo;
      output out=sumaresi sum=suma mean=media;
      run;
      data sumaresi;set sumaresi;semilla=&semilla;
      data final (keep=suma media semilla);set final sumaresi;if suma=.
then delete;run;
%end;
proc print data=final;run;
%mend;

%cruzadatreeboostbin(archivo=ml,vardepen=sentido_fallo,
```

```
conti=PC7   PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4   PC111
      PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59  PC125
      PC199 PC95  PC176 PC198 PC71  PC74  PC78,
categor=tribunal,
ngrupos=4,sinicio=12345,sfinal=12365,leafsize=5,
iteraciones=500,shrink=0.05,maxbranch=2,maxdepth=6,mincatsize=15,minob
s=20,objetivo=tasafallos);
data final12;set final;modelo=12;

%cruzadatreeboostbin(archivo=ml,vardepen=sentido_fallo,
conti=PC7   PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4   PC111
      PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59  PC125
      PC199 PC95  PC176 PC198 PC71  PC74  PC78,
categor=tribunal,
ngrupos=4,sinicio=12345,sfinal=12365,leafsize=5,
iteraciones=500,shrink=0.01,maxbranch=2,maxdepth=5,mincatsize=15,minob
s=20,objetivo=tasafallos);
data final13;set final;modelo=13;

%cruzadatreeboostbin(archivo=ml,vardepen=sentido_fallo,
conti=PC7   PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4   PC111
      PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59  PC125
      PC199 PC95  PC176 PC198 PC71  PC74  PC78,
categor=tribunal,
ngrupos=4,sinicio=12345,sfinal=12365,leafsize=5,
iteraciones=500,shrink=0.03,maxbranch=2,maxdepth=4,mincatsize=15,minob
s=20,objetivo=tasafallos);
data final14;set final;modelo=14;

data union;set final12 final13 final14;
proc boxplot data=union;plot media*modelo;run;


/* CRUZADASVM */

%macro
cruzadaSVMbin(archivo=,vardepen=,listclass=,listconti=,ngrupos=,sinici
o=,sfinal=,kernel=linear,c=10,directorio=c:);
data final;run;
*proc                printto                print="&directorio\ca.txt"
log="&directorio\loga.txt";run;
%do semilla=&sinicio %to &sfinal;/*<<<<<******AQUI SE PUEDEN CAMBIAR LAS
SEMILLAS */
data dos;set &archivo;u=ranuni(&semilla);
proc sort data=dos;by u;run;
data dos (drop=nume);
retain grupo 1;
set dos nobs=nume;
if _n_>grupo*nume/&ngrupos then grupo=grupo+1;
run;

data fantasma;run;

%do exclu=1 %to &ngrupos;

data    tres    valida;set    dos;if    grupo    ne    &exclu    then
do;vardep=&vardepen;output tres;end;else output valida;run;

/*************************************************************/
/* SVM */
/*************************************************************/
```

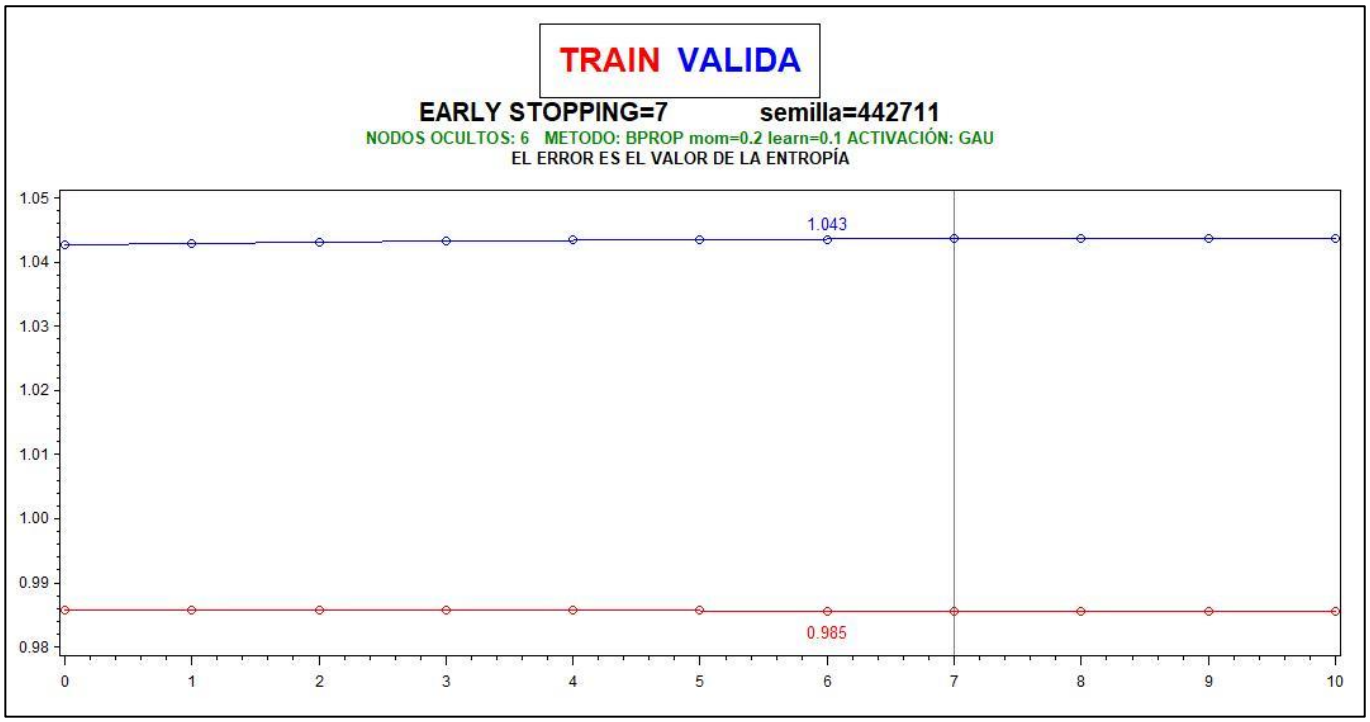
```
PROC DMDB DATA=tres dmdbcat=catatres out=cua;
target vardep ;
var &listconti;
class vardep &listclass;
;run;

proc  svm  data=cua  dmdbcat=catatres  testdata=valida  kernel=&kernel
testout=sal6 c=&c;
   var &listconti &listclass;
   target vardep;
run;
data sal1(keep=&vardepen predi1 grupo vardep);set sal6;predi1=_i_;run;

data salbis;
set sal1;if grupo=&exclu;
if predi1>0.5 then pre1=1;
if predi1<=0.5 then pre1=0;
run;

data salbos;run;
proc freq data=salbis noprint;tables pre1*&vardepen /out=salconfu;run;
data confu1 (keep=tasa1);retain buenos 0 malos 0;set salconfu nobs=nume;
if &vardepen=pre1 then buenos=buenos+count;
if &vardepen ne pre1 then malos=malos+count;
if _n_=nume then do;tasa1=malos/(malos+buenos);output;end;
run;
data salbos;merge salbos confu1;run;
;

data fantasma;set fantasma salbos;run;

%end;
/* FIN GRUPOS */
proc means data=fantasma noprint;var tasa1;
output out=mediaresi mean=media;
run;
data mediaresi;set mediaresi;semilla=&semilla;run;
data  final  (keep=media  semilla);set  final  mediaresi;if  media=.  then
delete;run;
%end;
proc printto; run;
proc print data=final;run;
%mend;

%cruzadaSVMbin
(archivo=ml,vardepen=sentido_fallo,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
ngrupos=4,sinicio=12345,sfinal=12365,kernel=rbf k_par=10,c=10);

data final15;set final;modelo='SVM-RBF';

%cruzadaSVMbin
(archivo=ml,vardepen=sentido_fallo,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
```

```
ngrupos=4,sinicio=12345,sfinal=12365,kernel=polynom k_par=2,c=10);

data final16;set final;modelo='SVM-poly';


%cruzadaSVMbin
(archivo=ml,vardepen=sentido_fallo,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
ngrupos=4,sinicio=12345,sfinal=12365,kernel=linear ,c=10);

data final17;set final;modelo='SVM1';

%cruzadaSVMbin
(archivo=ml,vardepen=sentido_fallo,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
ngrupos=4,sinicio=12345,sfinal=12365,kernel=rbf k_par=5,c=10);

data final18;set final;modelo='SVM-RBF2';

%cruzadaSVMbin
(archivo=ml,vardepen=sentido_fallo,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
ngrupos=4,sinicio=12345,sfinal=12365,kernel=polynom k_par=4,c=10);

data final19;set final;modelo='SVM-poly2';


%cruzadaSVMbin
(archivo=ml,vardepen=sentido_fallo,
listconti=PC7     PC15  PC27  PC54  PC25  PC52  PC22  PC42  PC43  PC4
      PC111 PC13  PC175 PC126 PC154 PC193 PC144 PC200 PC141 PC96  PC59
      PC125 PC199 PC95  PC176 PC198 PC71  PC74  PC78,
listclass=tribunal,
ngrupos=4,sinicio=12345,sfinal=12365,kernel=linear ,c=5);

data final20;set final;modelo='SVM2';

data union;length modelo $ 40;set final15 final16 final17 final18 final19
final20;

proc boxplot data=union;plot media*modelo;run;

data union;length modelo $40 ;set final1, final2, final3, final4, final5,
final6, final8, final9, final10, final11, final12, final13,final14;

proc boxplot data=union;plot media*modelo;run;
```

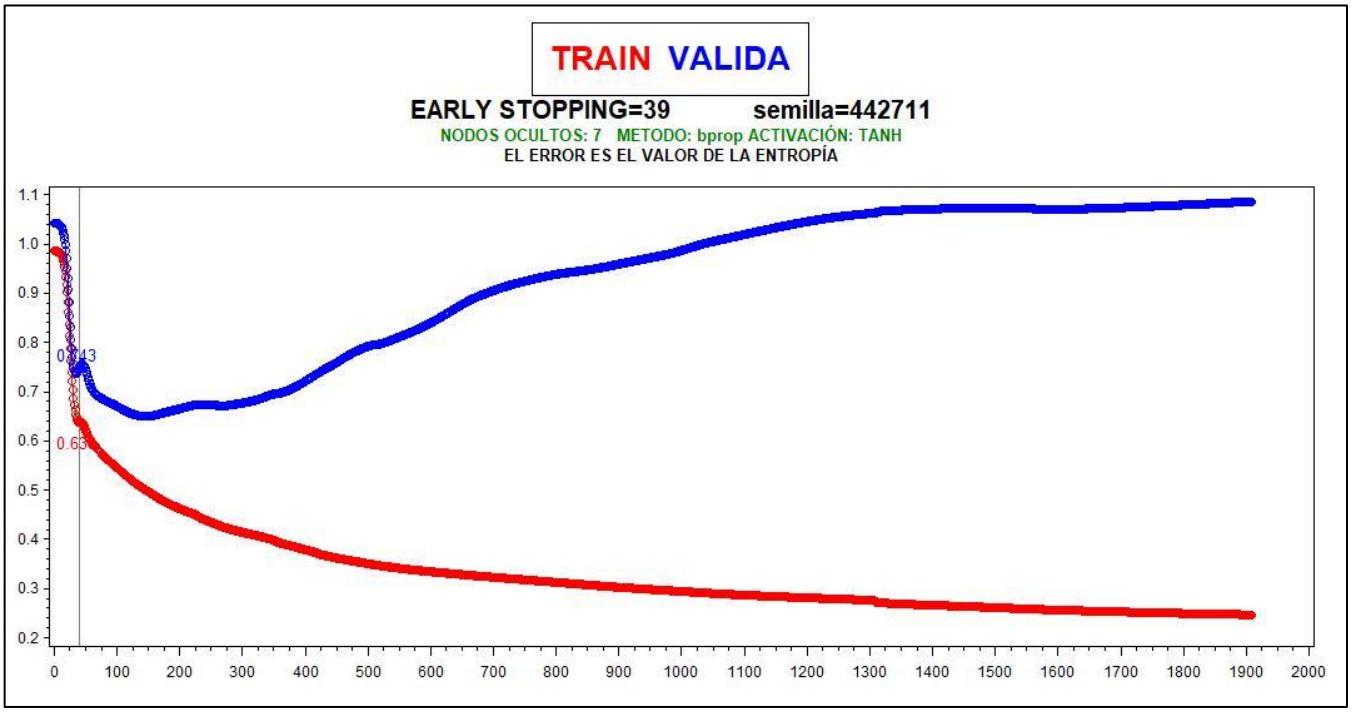
8.4.1.2 Principales salidas SAS

En este epígrafe incluimos algunos de los gráficos que hemos obtenido al realizar la modelización en SAS, al haber explicado el cuerpo del trabajo con el análisis realizado en R por ofrecer más detalles.

Early Stopping con back propagation y función de activación gausiana set 1



Podemos estudiar las posibilidades de emplear early-stopping en la red con la macro %redneuronalbinaria, para ello incluimos el o los sets que deseemos, la función de activación y el algoritmo de optimización, así como el número de nodos ocultos y la semilla. Estos son los gráficos para los sets 2 y 3 respectivamente:



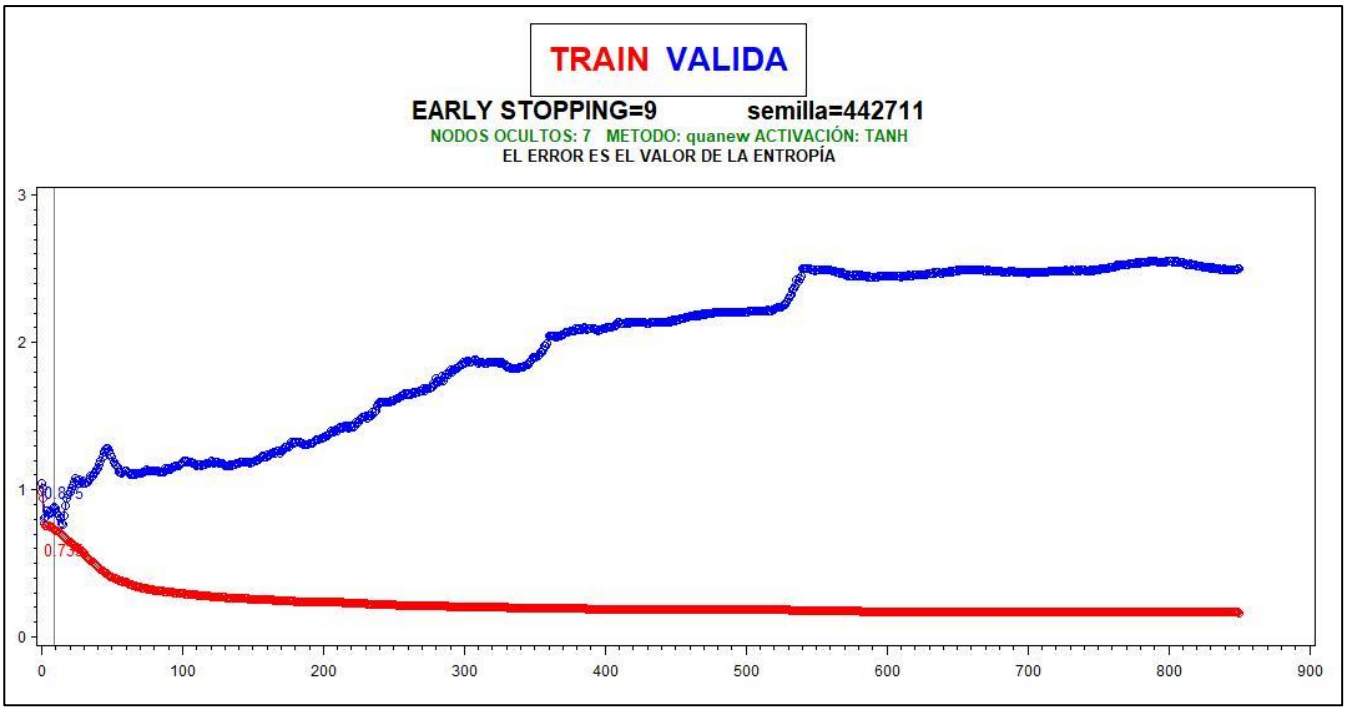


Gráfico de principales modelos con redes neuronales

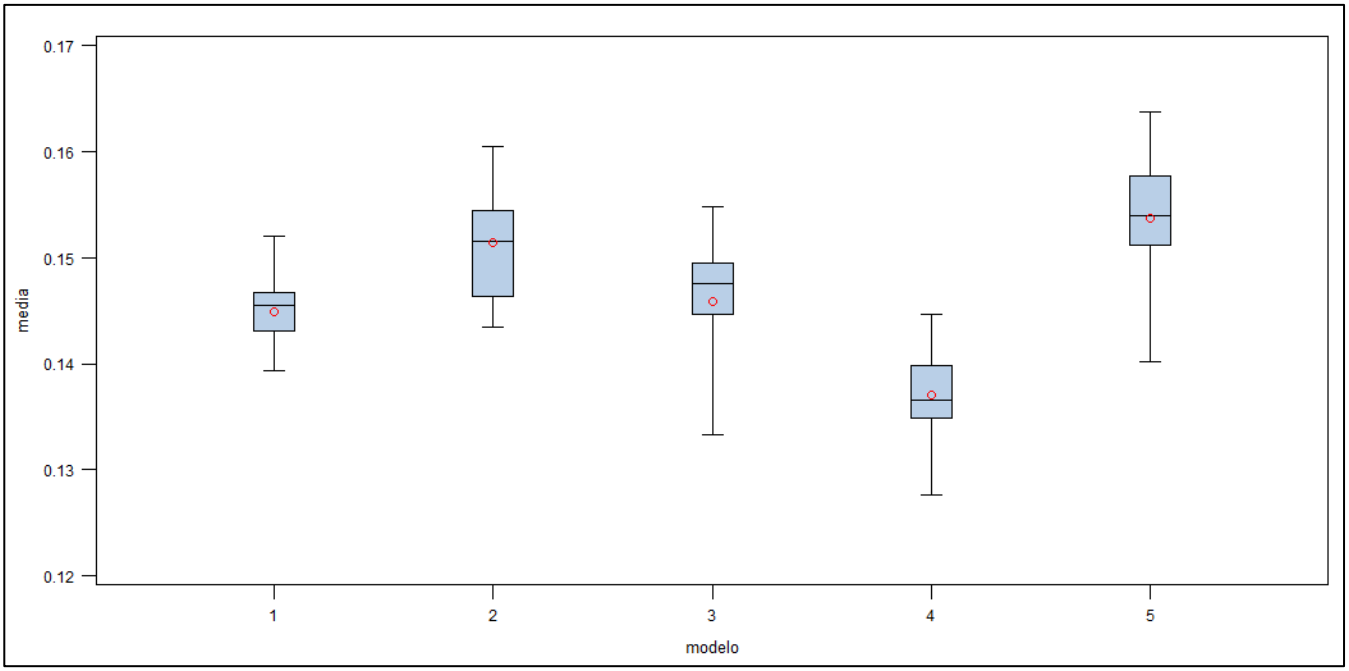


Gráfico de principales resultados Random Forest

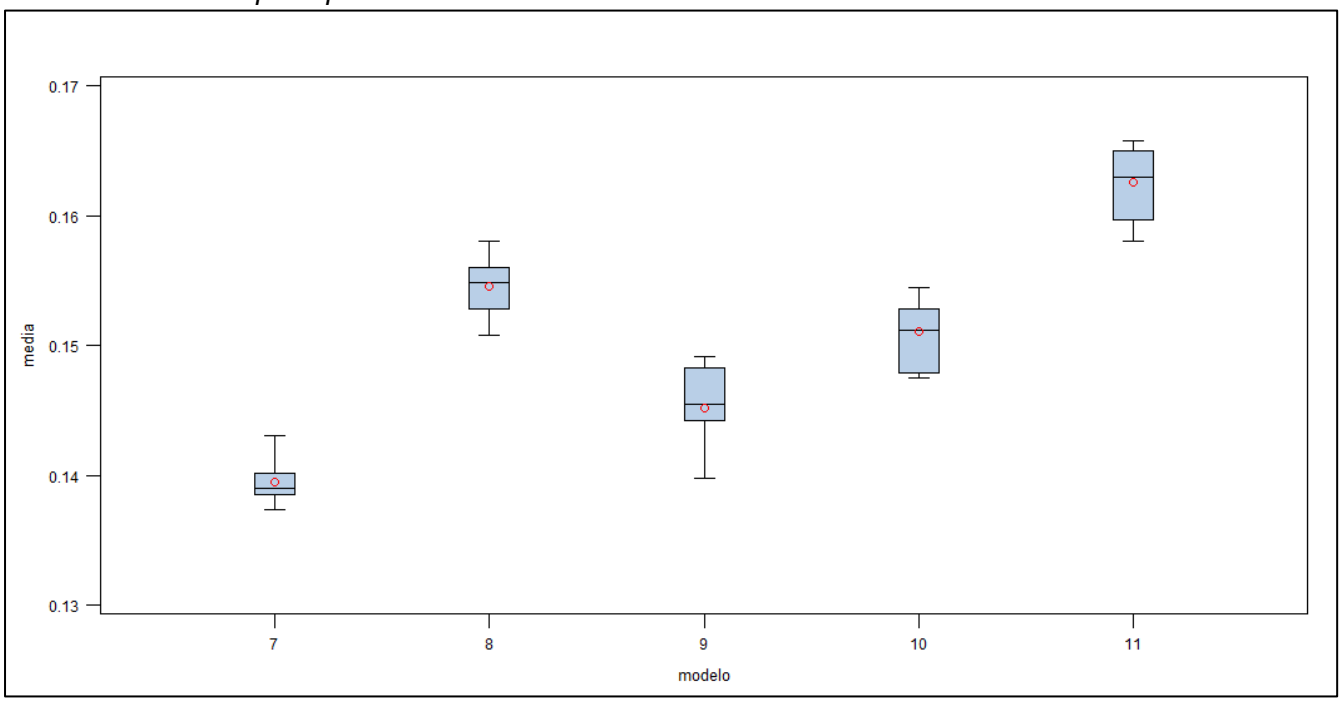
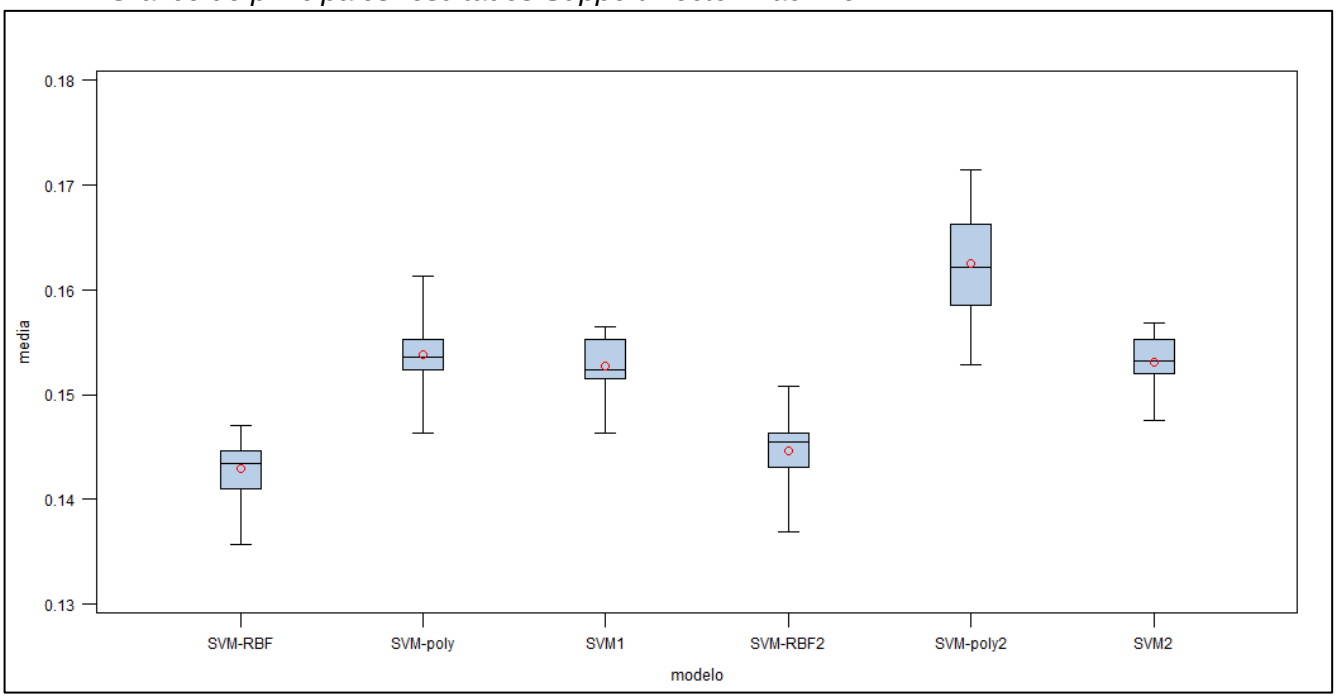


Gráfico de principales resultados Support Vector Machine



8.4.2 Modelización en R

```
library(sas7bdat)
library(nnet)
library(h2o)
library(dummies)
library(MASS)
library(reshape)
library(caret)
library(readr)
# Lectura y esquema de variables
  tfm<-read.csv("C:/Users/Mélani EV/Desktop/TFM Jurimetrics/BD_juridica_csv.csv")
  dput(names(tfm))

  continuas<-c("PC1", "PC2", "PC3", "PC4", "PC5", "PC6", "PC7", "PC8", "PC9",
               "PC10", "PC11", "PC12", "PC13", "PC14", "PC15", "PC16", "PC17",
               "PC18", "PC19", "PC20", "PC21", "PC22", "PC23", "PC24", "PC25",
               "PC26", "PC27", "PC28", "PC29", "PC30", "PC31", "PC32", "PC33",
               "PC34", "PC35", "PC36", "PC37", "PC38", "PC39", "PC40", "PC41",
               "PC42", "PC43", "PC44", "PC45", "PC46", "PC47", "PC48", "PC49",
               "PC50", "PC51", "PC52", "PC53", "PC54", "PC55", "PC56", "PC57",
               "PC58", "PC59", "PC60", "PC61", "PC62", "PC63", "PC64", "PC65",
               "PC66", "PC67", "PC68", "PC69", "PC70", "PC71", "PC72", "PC73",
               "PC74", "PC75", "PC76", "PC77", "PC78", "PC79", "PC80", "PC81",
               "PC82", "PC83", "PC84", "PC85", "PC86", "PC87", "PC88", "PC89",
               "PC90", "PC91", "PC92", "PC93", "PC94", "PC95", "PC96", "PC97",
               "PC98", "PC99", "PC100", "PC101", "PC102", "PC103", "PC104",
               "PC105", "PC106", "PC107", "PC108", "PC109", "PC110", "PC111",
               "PC112", "PC113", "PC114", "PC115", "PC116", "PC117", "PC118",
               "PC119", "PC120", "PC121", "PC122", "PC123", "PC124", "PC125",
               "PC126", "PC127", "PC128", "PC129", "PC130", "PC131", "PC132",
               "PC133", "PC134", "PC135", "PC136", "PC137", "PC138", "PC139",
               "PC140", "PC141", "PC142", "PC143", "PC144", "PC145", "PC146",
               "PC147", "PC148", "PC149", "PC150", "PC151", "PC152", "PC153",
               "PC154", "PC155", "PC156", "PC157", "PC158", "PC159", "PC160",
               "PC161", "PC162", "PC163", "PC164", "PC165", "PC166", "PC167",
               "PC168", "PC169", "PC170", "PC171", "PC172", "PC173", "PC174",
```

```
                "PC175", "PC176", "PC177", "PC178", "PC179", "PC180
  ", "PC181",
                "PC182", "PC183", "PC184", "PC185", "PC186", "PC187
  ", "PC188",
                "PC189", "PC190", "PC191", "PC192", "PC193", "PC194
  ", "PC195",
                "PC196", "PC197", "PC198", "PC199", "PC200")

  options(max.print=999999)

  tfm$tribunal<-factor(tfm$tribunal)
  tfm$fecha_reforma<-factor(tfm$fecha_reforma)
  tfm$sentido_fallo<-factor(tfm$sentido_fallo)
  tfm$genero<-factor(tfm$genero)

  summary(tfm)

  categoricas<-c("tribunal","fecha_reforma","genero")

  # a)Eliminar las observaciones con missing en alguna variable

  tfm2<-na.omit(tfm,(!is.na(tfm)))

  # b)pasar las categÃ³ricas a dummies

  tfm3<- dummy.data.frame(tfm2, categoricas, sep = ".")

  # c)estandarizar las variables continuas

  # Calculo medias y dtipica de datos y estandarizo (solo las
  continuas)

  means <-apply(tfm3[,continuas],2,mean)
  sds<-sapply(tfm3[,continuas],sd)

  tfmbis<-scale(tfm3[,continuas], center = means, scale = sds)
  numerocont<-which(colnames(tfm3)%in%continuas)
  tfmbis<-cbind(tfmbis,tfm3[,-numerocont])

  # El archivo tfmbis ya estÃ¡ preparado:no hay missing, las
  continuas salvo la dependiente
  # estÃ¡n estandarizadas y las categoricas pasadas a dummy

  dput(names(tfmbis))

  # EN LO SUCESIVO APLICAMOS VALIDACIÃ"N CRUZADA REPETIDA
  # Importante classProbs=TRUE para guardar las probabilidades
  # y definir la variable de salida con valores alfanumÃ©ricos
  Yes, No
  tfmbis$sentido_fallo<-ifelse(tfmbis$sentido_fallo==1,"Yes","No")

  set.seed(12346)
  # ValidaciÃ³n cruzada repetida
  control<-
  trainControl(method = "repeatedcv",number=4,repeats=5,savePredictio
  ns = "all")
  #
  ************************************************************
  # avNNet: parÃ¡metros
  # Number of Hidden Units (size, numeric)
  # Weight Decay (decay, numeric)
```

```
88.   # Bagging (bag, logical)
89.   #
   ************************************************************
90.   #set arbol
91.   avnnetgrid <-
   expand.grid(size=c(3,5,10,15),decay=c(0.01,0.1,0.001),bag=FALSE)
92.
93.   redavnnet1<- train(sentido_fallo~tribunal.1+tribunal.2+tribunal.
   3 + PC7 + PC15 +        PC27 + PC54     + PC25  + PC52 + PC22 + PC4
   2 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC154 + PC193 + PC1
   44+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+ PC176+ PC198+ PC
   71+ PC74+ PC78,
94.                     data=tfmbis,
95.                     method="avNNet",linout = TRUE,maxit=500,trCont
   rol=control,repeats=5,tuneGrid=avnnetgrid)
96.
97.   redavnnet1
98.
99.   #set Gradient Boosting
100.
101.  redavnnet2<- train(sentido_fallo~PC7+   PC15+   PC27+   PC54+
   PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC61+   PC22+   PC9
   +    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+   PC40+   PC4+
     PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  PC26+   PC76+   P
   C131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC134+  PC125+  PC11
   2+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+   PC14+   PC166+
   PC30+   PC71,
102.                    data=tfmbis,
103.                    method="avNNet",linout = TRUE,maxit=500,trCont
   rol=control,repeats=5,tuneGrid=avnnetgrid)
104.
105.  redavnnet2
106.
107.  #set seleccion
108.
109.  redavnnet3<- train(sentido_fallo~tribunal.1+tribunal.2+tribunal.
   3+PC7+  PC15+   PC27+   PC54+   PC35+   PC44+   PC35+   PC52+   PC4
   3+   PC42+   PC17+   PC62+   PC12+   PC4+    PC1+    PC168+  PC45+
     PC40+   PC5+    PC13+   PC105+  PC63+   PC188+  PC187,
110.                    data=tfmbis,
111.                    method="avNNet",linout = TRUE,maxit=500,trCon
   trol=control,repeats=5,tuneGrid=avnnetgrid)
112.
113.  redavnnet3
114.
115.  #con todo
116.
117.  avnnetgrid <-
   expand.grid(size=c(3,5,10,15),decay=c(0.01,0.1,0.001),bag=FALSE)
118.
119.
120.  redavnnet4<- train(sentido_fallo~.,
121.                    data=tfmbis,
122.                    method="avNNet",linout = TRUE,maxit=1500,trCo
   ntrol=control,repeats=5,tuneGrid=avnnetgrid)
123.
124.  redavnnet4
125.
126.
127.  red1<-redavnnet1$resample$Accuracy
128.  red2<-redavnnet2$resample$Accuracy
```

```
129.  red3<-redavnnet3$resample$Accuracy
130.  red4<-redavnnet4$resample$Accuracy
131.
132.
133.  union<-rbind(red1,red2,red3,red4)
134.  par(cex.axis=1)
135.  boxplot(red1,red2,red3,main="VC repetida de Redes con sets",
136.          col="lightblue",names = c("red1","red2","red3"))
137.
138.  library(sas7bdat)
139.  library(nnet)
140.  # library(h2o)
141.  library(dummies)
142.  library(MASS)
143.  library(reshape)
144.  library(caret)
145.  library(dplyr)
146.  library(pROC)
147.
148.  #REDES EN R
149.
150.  # *********************************
151.  # CRUZADA avNNet
152.  # **************
153.
154.
155.  cruzadaavnnetbin<-
156.    function(data=data,vardep="vardep",
157.             listconti="listconti",listclass="listclass",grupos=4,
     sinicio=1234,repe=5,
158.             size=c(5),decay=c(0.01),repeticiones=5,itera=100)
159.    {
160.
161.      # PreparaciÃ³n del archivo
162.
163.      # b)pasar las categÃ³ricas a dummies
164.
165.      if  (listclass!=c(""))
166.      {
167.        databis<-data[,c(vardep,listconti,listclass)]
168.        databis<- dummy.data.frame(databis, listclass, sep = ".")
169.      }  else   {
170.        databis<-data[,c(vardep,listconti)]
171.      }
172.
173.      # c)estandarizar las variables continuas
174.
175.      # Calculo medias y dtipica de datos y estandarizo (solo las
     continuas)
176.
177.      means <-apply(databis[,listconti],2,mean)
178.      sds<-sapply(databis[,listconti],sd)
179.
180.      # Estandarizo solo las continuas y uno con las categoricas
181.
182.      datacon<-scale(databis[,listconti],
     center = means, scale = sds)
183.      numerocont<-which(colnames(databis)%in%listconti)
184.      databis<-cbind(datacon,databis[,-numerocont,drop=FALSE ])
185.
186.      databis[,vardep]<-as.factor(databis[,vardep])
```

```
187.
188.       formu<-formula(paste(vardep,"~.",sep=""))
189.
190.       # Preparo caret
191.
192.       set.seed(sinicio)
193.       control<-
   trainControl(method = "repeatedcv",number=grupos,repeats=repe,
194.                               savePredictions = "all",classProbs=TRU
   E)
195.
196.       # Aplico caret y construyo modelo
197.
198.       avnnetgrid <-  expand.grid(size=size,decay=decay,bag=FALSE)
199.
200.       avnnet<- train(formu,data=databis,
201.                      method="avNNet",linout = FALSE,maxit=itera,re
   peats=repeticiones,
202.                      trControl=control,tuneGrid=avnnetgrid)
203.
204.       print(avnnet$results)
205.
206.       preditest<-avnnet$pred
207.
208.       preditest$prueba<-strsplit(preditest$Resample,"[.]")
209.       preditest$Fold <- sapply(preditest$prueba, "[", 1)
210.       preditest$Rep <- sapply(preditest$prueba, "[", 2)
211.       preditest$prueba<-NULL
212.
213.       tasafallos<-function(x,y) {
214.         confu<-confusionMatrix(x,y)
215.         tasa<-confu[[3]][1]
216.         return(tasa)
217.       }
218.
219.       # Aplicamos funciÃ³n sobre cada RepeticiÃ³n
220.
221.       medias<-preditest %>%
222.         group_by(Rep) %>%
223.         summarize(tasa=1-tasafallos(pred,obs))
224.
225.       # CalculamoS AUC  por cada RepeticiÃ³n de cv
226.       # Definimnos funciÃ³n
227.
228.       auc<-function(x,y) {
229.         curvaroc<-roc(response=x,predictor=y)
230.         auc<-curvaroc$auc
231.         return(auc)
232.       }
233.
234.       # Aplicamos funciÃ³n sobre cada RepeticiÃ³n
235.
236.       mediasbis<-preditest %>%
237.         group_by(Rep) %>%
238.         summarize(auc=auc(obs,Yes))
239.
240.       # Unimos la info de auc y de tasafallos
241.
242.       medias$auc<-mediasbis$auc
243.
244.       return(medias)
```

```
245.
246.     }
247.
248.
249.  medias1<-cruzadaavnnetbin(data=tfmbis,
250.                             vardep="sentido_fallo",listconti=c("PC
   7","PC15","PC27","PC54","PC25","PC52","PC22","PC42","PC43","PC4","P
   C111","PC13","PC175","PC126","PC154","PC193","PC144","PC200","PC141
   ","PC96","PC59","PC125","PC199","PC95","PC176","PC198","PC71","PC74
   ","PC78"),
251.                             listclass=c("tribunal.1","tribunal.2",
   "tribunal.3"),grupos=4,sinicio=1234,repe=5,
252.                             size=c(5),decay=c(0.01),repeticiones=5
   ,itera=500)
253.
254.  medias1$modelo="avnnet1"
255.  medias1$auc
256.  medias1$tasa
257.
258.  medias2<-cruzadaavnnetbin(data=tfmbis,
259.                             vardep="sentido_fallo",listconti=c("PC
   7","PC15","PC27","PC54","PC25","PC44","PC35","PC3","PC2","PC52","PC
   61","PC22","PC9","PC55","PC42","PC36","PC43","PC64","PC40","PC4","P
   C32","PC105","PC69","PC140","PC117","PC142","PC26","PC76","PC131","
   PC193","PC24","PC115","PC150","PC56","PC134","PC125","PC112","PC63"
   ,"PC46","PC176","PC16","PC104","PC20","PC14","PC166","PC30","PC71")
   ,
260.                             listclass=c(""),grupos=4,sinicio=1234,
   repe=5,
261.                             size=c(3),decay=c(0.1),repeticiones=5,
   itera=500)
262.
263.  medias2$modelo="avnnet2"
264.
265.  medias2$auc
266.  medias2$tasa
267.
268.  medias3<-cruzadaavnnetbin(data=tfmbis,
269.                              vardep="sentido_fallo",listconti=c("P
   C7","PC15","PC27","PC54","PC44","PC35","PC52","PC43","PC42","PC17",
   "PC62","PC12","PC4","PC1","PC168","PC45","PC40","PC5","PC13","PC105
   ","PC63","PC188","PC187"),
270.                              listclass=c("tribunal.1","tribunal.2"
   ,"tribunal.3"),grupos=4,sinicio=1234,repe=5,
271.                              size=c(5),decay=c(0.01),repeticiones=
   5,itera=500)
272.
273.  medias3$modelo="avnnet3"
274.
275.  medias3$auc
276.  medias3$tasa
277.
278.  medias4<-cruzadaavnnetbin(data=tfmbis,
279.                             vardep="sentido_fallo",listconti=c("PC
   1", "PC2", "PC3", "PC4", "PC5", "PC6", "PC7", "PC8", "PC9",
280.                                                                "PC
   10", "PC11", "PC12", "PC13", "PC14", "PC15", "PC16", "PC17",
281.                                                                "PC
   18", "PC19", "PC20", "PC21", "PC22", "PC23", "PC24", "PC25",
282.                                                                "PC
   26", "PC27", "PC28", "PC29", "PC30", "PC31", "PC32", "PC33",
```

```
283.                                                                    "PC
   34", "PC35", "PC36", "PC37", "PC38", "PC39", "PC40", "PC41",
284.                                                                    "PC
   42", "PC43", "PC44", "PC45", "PC46", "PC47", "PC48", "PC49",
285.                                                                    "PC
   50", "PC51", "PC52", "PC53", "PC54", "PC55", "PC56", "PC57",
286.                                                                    "PC
   58", "PC59", "PC60", "PC61", "PC62", "PC63", "PC64", "PC65",
287.                                                                    "PC
   66", "PC67", "PC68", "PC69", "PC70", "PC71", "PC72", "PC73",
288.                                                                    "PC
   74", "PC75", "PC76", "PC77", "PC78", "PC79", "PC80", "PC81",
289.                                                                    "PC
   82", "PC83", "PC84", "PC85", "PC86", "PC87", "PC88", "PC89",
290.                                                                    "PC
   90", "PC91", "PC92", "PC93", "PC94", "PC95", "PC96", "PC97",
291.                                                                    "PC
   98", "PC99", "PC100", "PC101", "PC102", "PC103", "PC104",
292.                                                                    "PC
   105", "PC106", "PC107", "PC108", "PC109", "PC110", "PC111",
293.                                                                    "PC
   112", "PC113", "PC114", "PC115", "PC116", "PC117", "PC118",
294.                                                                    "PC
   119", "PC120", "PC121", "PC122", "PC123", "PC124", "PC125",
295.                                                                    "PC
   126", "PC127", "PC128", "PC129", "PC130", "PC131", "PC132",
296.                                                                    "PC
   133", "PC134", "PC135", "PC136", "PC137", "PC138", "PC139",
297.                                                                    "PC
   140", "PC141", "PC142", "PC143", "PC144", "PC145", "PC146",
298.                                                                    "PC
   147", "PC148", "PC149", "PC150", "PC151", "PC152", "PC153",
299.                                                                    "PC
   154", "PC155", "PC156", "PC157", "PC158", "PC159", "PC160",
300.                                                                    "PC
   161", "PC162", "PC163", "PC164", "PC165", "PC166", "PC167",
301.                                                                    "PC
   168", "PC169", "PC170", "PC171", "PC172", "PC173", "PC174",
302.                                                                    "PC
   175", "PC176", "PC177", "PC178", "PC179", "PC180", "PC181",
303.                                                                    "PC
   182", "PC183", "PC184", "PC185", "PC186", "PC187", "PC188",
304.                                                                    "PC
   189", "PC190", "PC191", "PC192", "PC193", "PC194", "PC195",
305.                                                                    "PC
   196", "PC197", "PC198", "PC199", "PC200"),
306.                              listclass=c("genero.0", "genero.1", "t
   ribunal.1", "tribunal.2", "tribunal.3",
307.                                          "fecha_reforma.1", "fecha_
   reforma.2", "fecha_reforma.3", "fecha_reforma.4"),grupos=4,sinicio=
   1234,repe=5,
308.                              size=c(3),decay=c(0.001),repeticiones=
   5,itera=1500)
309.
310.  medias4$modelo="avnnet4"
311.
312.  medias4$auc
313.  medias4$tasa
314.
315.  union2<-rbind(medias1,medias2,medias3,medias4)
316.
```

```
317.  par(cex.axis=1.25)
318.  boxplot(data=union2,tasa~modelo,main="TASA FALLOS
   REDES",col="pink")
319.  boxplot(data=union2,auc~modelo,main="AUC
   REDES",col="lightgreen")
320.
321.
322.  #ARBOLES
323.  library(caret)
324.
325.  set.seed(12345)
326.  rfgrid<-expand.grid(mtry=c(3,6,9,12,15,18,21,24,28,32))
327.
328.  control<-
   trainControl(method = "cv",number=10,savePredictions = "all",
329.                        classProbs=TRUE)
330.
331.  rf1<- train(data=tfmbis,
332.              factor(sentido_fallo)~tribunal.1+ tribunal.2 + tribun
   al.3 + PC7 + PC15 +     PC27 + PC54     + PC25  + PC52 + PC22 + PC4
   2 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC154 + PC193 + PC1
   44+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+ PC176+ PC198+ PC
   71+ PC74+ PC78 ,
333.             method="rf",trControl=control,tuneGrid=rfgrid,
334.             linout = FALSE,ntree=5000,sampsize=200,nodesize=10,re
   place=TRUE, importance = TRUE)
335.
336.  rf1
337.  plot(rf1)
338.
339.  rf1b<- train(data=tfmbis,
340.              factor(sentido_fallo)~tribunal.1+ tribunal.2 + tribu
   nal.3 + PC7 + PC15 +    PC27 + PC54     + PC25  + PC52 + PC22 + PC4
   2 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC154 + PC193 + PC1
   44+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+ PC176+ PC198+ PC
   71+ PC74+ PC78 ,
341.              method="rf",trControl=control,tuneGrid=rfgrid,
342.              linout = FALSE,ntree=5000,sampsize=200,nodesize=15,r
   eplace=TRUE, importance = TRUE)
343.
344.  rf1b
345.
346.  rf1c<- train(data=tfmbis,
347.               factor(sentido_fallo)~tribunal.1+ tribunal.2 + trib
   unal.3 + PC7 + PC15 +   PC27 + PC54     + PC25  + PC52 + PC22 + PC4
   2 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC154 + PC193 + PC1
   44+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+ PC176+ PC198+ PC
   71+ PC74+ PC78 ,
348.               method="rf",trControl=control,tuneGrid=rfgrid,
349.               linout = FALSE,ntree=5000,sampsize=200,nodesize=6,r
   eplace=TRUE, importance = TRUE)
350.
351.  rf1c
352.
353.  set.seed(12345)
354.  rfgrid<-expand.grid(mtry=c(12,15,18,21,24,30,32,40,45,48))
355.
356.  rf2<- train(data=tfmbis,
357.              factor(sentido_fallo)~PC7+  PC15+   PC27+   PC54+
   PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC61+   PC22+   PC9
   +    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+   PC40+   PC4+
```

```
    PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  PC26+   PC76+   P
  C131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC134+  PC125+  PC11
  2+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+   PC14+   PC166+
  PC30+   PC71,
358.             method="rf",trControl=control,tuneGrid=rfgrid,
359.             linout = FALSE,ntree=5000,sampsize=200,nodesize=10,r
  eplace=TRUE, importance = TRUE)
360.
361.  rf2
362.  plot(rf2)
363.
364.
365.  rf2b<- train(data=tfmbis,
366.               factor(sentido_fallo)~PC7+  PC15+   PC27+   PC54+
  PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC61+   PC22+   PC9
  +    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+   PC40+   PC4+
    PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  PC26+   PC76+   P
  C131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC134+  PC125+  PC11
  2+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+   PC14+   PC166+
  PC30+   PC71,
367.             method="rf",trControl=control,tuneGrid=rfgrid,
368.             linout = FALSE,ntree=5000,sampsize=200,nodesize=15,r
  eplace=TRUE, importance = TRUE)
369.
370.  rf2b
371.
372.  rf2c<- train(data=tfmbis,
373.               factor(sentido_fallo)~PC7+ PC15+   PC27+   PC54+
  PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC61+   PC22+   PC9
  +    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+   PC40+   PC4+
    PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  PC26+   PC76+   P
  C131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC134+  PC125+  PC11
  2+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+   PC14+   PC166+
  PC30+   PC71,
374.              method="rf",trControl=control,tuneGrid=rfgrid,
375.              linout = FALSE,ntree=5000,sampsize=200,nodesize=6,r
  eplace=TRUE, importance = TRUE)
376.
377.  rf2c
378.
379.  rf3<- train(data=tfmbis,
380.             factor(sentido_fallo)~tribunal.1+ tribunal.2 + tribu
  nal.3+PC7+      PC15+   PC27+   PC54+   PC35+   PC44+   PC35+   PC5
  2+   PC43+   PC42+   PC17+   PC62+   PC12+   PC4+    PC1+    PC168+
    PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   PC188+  PC187,
381.             method="rf",trControl=control,tuneGrid=rfgrid,
382.             linout = FALSE,ntree=5000,sampsize=200,nodesize=10,r
  eplace=TRUE, importance = TRUE)
383.
384.  rf3
385.  plot(rf3)
386.
387.  rf3b<- train(data=tfmbis,
388.             factor(sentido_fallo)~tribunal.1+ tribunal.2 + tribu
  nal.3+PC7+      PC15+   PC27+   PC54+   PC35+   PC44+   PC35+   PC5
  2+   PC43+   PC42+   PC17+   PC62+   PC12+   PC4+    PC1+    PC168+
    PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   PC188+  PC187,
389.             method="rf",trControl=control,tuneGrid=rfgrid,
390.             linout = FALSE,ntree=5000,sampsize=200,nodesize=15,r
  eplace=TRUE, importance = TRUE)
391.
```

```
392.  rf3b
393.
394.  rf3c<- train(data=tfmbis,
395.               factor(sentido_fallo)~tribunal.1+ tribunal.2 + tribu
   nal.3+PC7+       PC15+    PC27+    PC54+    PC35+    PC44+    PC35+    PC5
   2+    PC43+    PC42+    PC17+    PC62+    PC12+    PC4+     PC1+     PC168+
     PC45+    PC40+    PC5+     PC13+    PC105+  PC63+    PC188+  PC187,
396.               method="rf",trControl=control,tuneGrid=rfgrid,
397.               linout = FALSE,ntree=5000,sampsize=200,nodesize=6,re
   place=TRUE, importance = TRUE)
398.
399.  rf3c
400.
401.  rf4<- train(data=tfmbis,
402.               factor(sentido_fallo)~genero.0+genero.1+tribunal.1+t
   ribunal.2+tribunal.3+fecha_reforma.1+fecha_reforma.2+fecha_reforma.
   3+fecha_reforma.4+PC1+PC2+PC3+PC4+PC5+PC6+PC7+PC8+PC9 +PC10+PC11+PC
   12+PC13+PC14+PC15+PC16+PC17+PC18+PC19+PC20+PC21+PC22+PC23+PC24+PC25
    +PC26+PC27+PC28+PC29+PC30+PC31+PC32+PC33 +PC34+PC35+PC36+PC37+PC38
   +PC39+PC40+PC41 +PC42+PC43+PC44+PC45+PC46+PC47+PC48+PC49 +PC50+PC51
   +PC52+PC53+PC54+PC55+PC56+PC57 +PC58+PC59+PC60+PC61+PC62+PC63+PC64+
   PC65 +PC66+PC67+PC68+PC69+PC70+PC71+PC72+PC73 +PC74+PC75+PC76+PC77+
   PC78+PC79+PC80+PC81 +PC82+PC83+PC84+PC85+PC86+PC87+PC88+PC89 +PC90+
   PC91+PC92+PC93+PC94+PC95+PC96+PC97 +PC98+PC99+PC100+PC101+PC102+PC1
   03+PC104 +PC105+PC106+PC107+PC108+PC109+PC110+PC111 +PC112+PC113+PC
   114+PC115+PC116+PC117+PC118 +PC119+PC120+PC121+PC122+PC123+PC124+PC
   125 +PC126+PC127+PC128+PC129+PC130+PC131+PC132 +PC133+PC134+PC135+P
   C136+PC137+PC138+PC139 +PC140+PC141+PC142+PC143+PC144+PC145+PC146 +
   PC147+PC148+PC149+PC150+PC151+PC152+PC153 +PC154+PC155+PC156+PC157+
   PC158+PC159+PC160 +PC161+PC162+PC163+PC164+PC165+PC166+PC167 +PC168
   +PC169+PC170+PC171+PC172+PC173+PC174 +PC175+PC176+PC177+PC178+PC179
   +PC180+PC181 +PC182+PC183+PC184+PC185+PC186+PC187+PC188 +PC189+PC19
   0+PC191+PC192+PC193+PC194+PC195 +PC196+PC197+PC198+PC199+PC200,
403.               method="rf",trControl=control,tuneGrid=rfgrid,
404.               linout = FALSE,ntree=5000,sampsize=200,nodesize=10,r
   eplace=TRUE, importance = TRUE)
405.
406.  rf4
407.  plot(rf4)
408.
409.  rf4b<- train(data=tfmbis,
410.               factor(sentido_fallo)~genero.0+genero.1+tribunal.1+t
   ribunal.2+tribunal.3+fecha_reforma.1+fecha_reforma.2+fecha_reforma.
   3+fecha_reforma.4+PC1+PC2+PC3+PC4+PC5+PC6+PC7+PC8+PC9 +PC10+PC11+PC
   12+PC13+PC14+PC15+PC16+PC17+PC18+PC19+PC20+PC21+PC22+PC23+PC24+PC25
    +PC26+PC27+PC28+PC29+PC30+PC31+PC32+PC33 +PC34+PC35+PC36+PC37+PC38
   +PC39+PC40+PC41 +PC42+PC43+PC44+PC45+PC46+PC47+PC48+PC49 +PC50+PC51
   +PC52+PC53+PC54+PC55+PC56+PC57 +PC58+PC59+PC60+PC61+PC62+PC63+PC64+
   PC65 +PC66+PC67+PC68+PC69+PC70+PC71+PC72+PC73 +PC74+PC75+PC76+PC77+
   PC78+PC79+PC80+PC81 +PC82+PC83+PC84+PC85+PC86+PC87+PC88+PC89 +PC90+
   PC91+PC92+PC93+PC94+PC95+PC96+PC97 +PC98+PC99+PC100+PC101+PC102+PC1
   03+PC104 +PC105+PC106+PC107+PC108+PC109+PC110+PC111 +PC112+PC113+PC
   114+PC115+PC116+PC117+PC118 +PC119+PC120+PC121+PC122+PC123+PC124+PC
   125 +PC126+PC127+PC128+PC129+PC130+PC131+PC132 +PC133+PC134+PC135+P
   C136+PC137+PC138+PC139 +PC140+PC141+PC142+PC143+PC144+PC145+PC146 +
   PC147+PC148+PC149+PC150+PC151+PC152+PC153 +PC154+PC155+PC156+PC157+
   PC158+PC159+PC160 +PC161+PC162+PC163+PC164+PC165+PC166+PC167 +PC168
   +PC169+PC170+PC171+PC172+PC173+PC174 +PC175+PC176+PC177+PC178+PC179
   +PC180+PC181 +PC182+PC183+PC184+PC185+PC186+PC187+PC188 +PC189+PC19
   0+PC191+PC192+PC193+PC194+PC195 +PC196+PC197+PC198+PC199+PC200,
411.               method="rf",trControl=control,tuneGrid=rfgrid,
```

```
412.              linout = FALSE,ntree=5000,sampsize=200,nodesize=15,r
   eplace=TRUE, importance = TRUE)
413.
414.  rf4b
415.
416.  rf4c<- train(data=tfmbis,
417.               factor(sentido_fallo)~genero.0+genero.1+tribunal.1+
   tribunal.2+tribunal.3+fecha_reforma.1+fecha_reforma.2+fecha_reforma
   .3+fecha_reforma.4+PC1+PC2+PC3+PC4+PC5+PC6+PC7+PC8+PC9 +PC10+PC11+P
   C12+PC13+PC14+PC15+PC16+PC17+PC18+PC19+PC20+PC21+PC22+PC23+PC24+PC2
   5 +PC26+PC27+PC28+PC29+PC30+PC31+PC32+PC33 +PC34+PC35+PC36+PC37+PC3
   8+PC39+PC40+PC41 +PC42+PC43+PC44+PC45+PC46+PC47+PC48+PC49 +PC50+PC5
   1+PC52+PC53+PC54+PC55+PC56+PC57 +PC58+PC59+PC60+PC61+PC62+PC63+PC64
   +PC65 +PC66+PC67+PC68+PC69+PC70+PC71+PC72+PC73 +PC74+PC75+PC76+PC77
   +PC78+PC79+PC80+PC81 +PC82+PC83+PC84+PC85+PC86+PC87+PC88+PC89 +PC90
   +PC91+PC92+PC93+PC94+PC95+PC96+PC97 +PC98+PC99+PC100+PC101+PC102+PC
   103+PC104 +PC105+PC106+PC107+PC108+PC109+PC110+PC111 +PC112+PC113+P
   C114+PC115+PC116+PC117+PC118 +PC119+PC120+PC121+PC122+PC123+PC124+P
   C125 +PC126+PC127+PC128+PC129+PC130+PC131+PC132 +PC133+PC134+PC135+
   PC136+PC137+PC138+PC139 +PC140+PC141+PC142+PC143+PC144+PC145+PC146
   +PC147+PC148+PC149+PC150+PC151+PC152+PC153 +PC154+PC155+PC156+PC157
   +PC158+PC159+PC160 +PC161+PC162+PC163+PC164+PC165+PC166+PC167 +PC16
   8+PC169+PC170+PC171+PC172+PC173+PC174 +PC175+PC176+PC177+PC178+PC17
   9+PC180+PC181 +PC182+PC183+PC184+PC185+PC186+PC187+PC188 +PC189+PC1
   90+PC191+PC192+PC193+PC194+PC195 +PC196+PC197+PC198+PC199+PC200,
418.               method="rf",trControl=control,tuneGrid=rfgrid,
419.               linout = FALSE,ntree=5000,sampsize=200,nodesize=6,r
   eplace=TRUE, importance = TRUE)
420.
421.  rf4c
422.
423.  #IMPORTANCIA DE LAS VARIABLES
424.
425.  final1<-rf1$finalModel
426.  final2<-rf2$finalModel
427.  final3<-rf3$finalModel
428.  final4<-rf4$finalModel
429.
430.  tabla1<-as.data.frame(importance(final1))
431.  tabla1<-tabla1[order(-tabla1$MeanDecreaseAccuracy),]
432.  tabla1
433.
434.  tabla2<-as.data.frame(importance(final2))
435.  tabla2<-tabla2[order(-tabla2$MeanDecreaseAccuracy),]
436.  tabla2
437.
438.  tabla3<-as.data.frame(importance(final3))
439.  tabla3<-tabla3[order(-tabla3$MeanDecreaseAccuracy),]
440.  tabla3
441.
442.  par(cex=0.5,las=2)
443.  barplot(tabla3$MeanDecreaseAccuracy,names.arg=row.names(tabla3),
   main="Importancia de las variables",cex.names = 1.12)
444.
445.
446.
447.  barplot(tabla01$rel.inf,names.arg=row.names(tabla01),main="Impor
   tancia de las variables")
448.  tabla4<-as.data.frame(importance(final4))
449.  tabla4<-tabla4[order(-tabla4$MeanDecreaseAccuracy),]
450.  tabla4
```

```
451.
452.
453.
454.  par(cex=1.5,las=2)
455.  barplot(tabla1$MeanDecreaseAccuracy,names.arg=rownames(tabla1),m
   ain="Importancia vbles rf1")
456.
457.  plot(rf1, main="Random Forest.Set 1")
458.
459.  par(cex=1.5,las=2)
460.  barplot(tabla2$MeanDecreaseAccuracy,names.arg=rownames(tabla2),m
   ain="Importancia vbles rf2")
461.
462.  plot(rf2, main="Random Forest.Set 2")
463.
464.  par(cex=1.5,las=2)
465.  barplot(tabla3$MeanDecreaseAccuracy,names.arg=rownames(tabla3),m
   ain="Importancia vbles rf3")
466.
467.  plot(rf3,main="Random Forest.Set 3")
468.
469.  par(cex=0.5,las=1.5)
470.  barplot(tabla4$MeanDecreaseAccuracy,names.arg=rownames(tabla4),m
   ain="Importancia vbles rf4")
471.
472.  plot(rf4,main="Random Forest.Set 4")
473.
474.
475.  # PARA PLOTEAR EL ERROR OOB A MEDIDA QUE AVANZAN LAS ITERACIONES
476.  # SE USA DIRECTAMENTE EL PAQUETE randomForest
477.
478.  library(randomForest)
479.  set.seed(12345)
480.
481.  rfbis<-
   randomForest(factor(sentido_fallo)~tribunal.1+ tribunal.2 + tribuna
   l.3+PC7+      PC15+   PC27+   PC54+   PC35+   PC44+   PC35+   PC52+
      PC43+   PC42+   PC17+   PC62+   PC12+   PC4+    PC1+    PC168+
   PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   PC188+  PC187,
482.                       data=tfmbis,
483.                       mtry=24,ntree=5000,nodesize=10,replace=TRUE)
484.  rf_set3<-rfbis
485.
486.  varImpPlot(rf_set3)
487.  importance(rf_set3)
488.  getTree(rfbis,labelVar=TRUE)
489.  zzzz<-getTree(rf_set3, k=12, labelVar=TRUE)
490.
491.  print(rfbis)
492.
493.  plot(rfbis$err.rate[,1], main="Error OOB set 3", ylab = "error",
   xlab = "n° árboles")
494.
495.  par(cex=0.5,las=1)
496.  plot(rfbis, main="Error OOB set 3")
497.  legend('center','groups',c("Error OOb", "Error en churn =
   no", "Error en churn = yes"), lty=c(1,1),
498.         lwd=c(0.5,0.5,0.5), col=c("black","red","green"),cex=1.5)
499.
500.  rfbis1<-randomForest(factor(sentido_fallo)~.,
501.                       data=tfmbis,
```

```
                      mtry=205,ntree=5000,nodesize=10,replace=TRUE
  )

 plot(rfbis1$err.rate[,1], main="Error OOB", ylab = "error",
  xlab = "n° árboles")

 par(cex=0.5,las=1)
 plot(rfbis1, main="Error OOB bagging")
 legend('center','groups',c("Error OOb", "Error en churn =
  no", "Error en churn = yes"), lty=c(1,1),
       lwd=c(0.5,0.5,0.5), col=c("black","red","green"),cex=1.5)


 #RANDOM FOREST CON VALIDACION CRUZADA
 cruzadarfbin<-
   function(data=data,vardep="vardep",
            listconti="listconti",listclass="listclass",
            grupos=4,sinicio=1234,repe=5,nodesize=20,
            mtry=2,ntree=50,replace=TRUE,sampsize=1)
   {
     # if  (sampsize==1)
     # {
     #  sampsize=floor(nrow(data)/(grupos-1))
     # }

     # PreparaciÃ³n del archivo

     # b)pasar las categÃ³ricas a dummies

     if  (listclass!=c(""))
     {
       databis<-data[,c(vardep,listconti,listclass)]
       databis<- dummy.data.frame(databis, listclass, sep = ".")
     }  else   {
       databis<-data[,c(vardep,listconti)]
     }

     # c)estandarizar las variables continuas

     # Calculo medias y dtipica de datos y estandarizo (solo las
  continuas)

     means <-apply(databis[,listconti],2,mean)
     sds<-sapply(databis[,listconti],sd)

     # Estandarizo solo las continuas y uno con las categoricas

     datacon<-scale(databis[,listconti],
  center = means, scale = sds)
     numerocont<-which(colnames(databis)%in%listconti)
     databis<-cbind(datacon,databis[,-numerocont,drop=FALSE ])

     databis[,vardep]<-as.factor(databis[,vardep])

     formu<-formula(paste("factor(",vardep,")~.",sep=""))

     # Preparo caret

     set.seed(sinicio)
```

```
      control<-
   trainControl(method = "repeatedcv",number=grupos,repeats=repe,
                              savePredictions = "all",classProbs=TRU
   E)

      # Aplico caret y construyo modelo

      rfgrid <-expand.grid(mtry=mtry)

      if  (sampsize==1)
      {
        rf<- train(formu,data=databis,
                   method="rf",trControl=control,
                   tuneGrid=rfgrid,nodesize=nodesize,replace=repla
   ce,ntree=ntree)
      }

      else  if  (sampsize!=1)
      {
        rf<- train(formu,data=databis,
                   method="rf",trControl=control,
                   tuneGrid=rfgrid,nodesize=nodesize,replace=repla
   ce,sampsize=sampsize,
                   ntree=ntree)
      }


      print(rf$results)

      preditest<-rf$pred

      preditest$prueba<-strsplit(preditest$Resample,"[.]")
      preditest$Fold <- sapply(preditest$prueba, "[", 1)
      preditest$Rep <- sapply(preditest$prueba, "[", 2)
      preditest$prueba<-NULL

      tasafallos<-function(x,y) {
        confu<-confusionMatrix(x,y)
        tasa<-confu[[3]][1]
        return(tasa)
      }

      # Aplicamos funciÃ³n sobre cada RepeticiÃ³n

      medias<-preditest %>%
        group_by(Rep) %>%
        summarize(tasa=1-tasafallos(pred,obs))

      # CalculamoS AUC  por cada RepeticiÃ³n de cv
      # Definimnos funciÃ³n

      auc<-function(x,y) {
        curvaroc<-roc(response=x,predictor=y)
        auc<-curvaroc$auc
        return(auc)
      }

      # Aplicamos funciÃ³n sobre cada RepeticiÃ³n

      mediasbis<-preditest %>%
        group_by(Rep) %>%
```

```
614.         summarize(auc=auc(obs,Yes))
615.
616.       # Unimos la info de auc y de tasafallos
617.
618.       medias$auc<-mediasbis$auc
619.
620.       return(medias)
621.
622.     }
623.
624.
625.
626.   medias04<-cruzadarfbin(data=tfmbis,
627.                          vardep="sentido_fallo",listconti=c("PC7",
    "PC15","PC27","PC54","PC25","PC52","PC22","PC42","PC43","PC4","PC11
    1","PC13","PC175","PC126","PC154","PC193","PC144","PC200","PC141","
    PC96","PC59","PC125","PC199","PC95","PC176","PC198","PC71","PC74","
    PC78"),
628.                          listclass=c("tribunal.1","tribunal.2","tr
    ibunal.3"),
629.                          grupos=4,sinicio=1234,repe=5,nodesize=10,
630.                          mtry=28,ntree=200,replace=TRUE,
    sampsize = 500)
631.
632.   medias04$modelo="rf_set1"
633.
634.   medias04$tasa
635.   medias04$auc
636.
637.   medias5<-cruzadarfbin(data=tfmbis,
638.                           vardep="sentido_fallo",listconti=c("PC7"
    ,"PC15","PC27","PC54","PC25","PC44","PC35","PC3","PC2","PC52","PC61
    ","PC22","PC9","PC55","PC42","PC36","PC43","PC64","PC40","PC4","PC3
    2","PC105","PC69","PC140","PC117","PC142","PC26","PC76","PC131","PC
    193","PC24","PC115","PC150","PC56","PC134","PC125","PC112","PC63","
    PC46","PC176","PC16","PC104","PC20","PC14","PC166","PC30","PC71"),
639.                           listclass=c(""),
640.                           grupos=4,sinicio=1234,repe=5,nodesize=10
    ,
641.                           mtry=40,ntree=200,replace=TRUE,
    sampsize = 500)
642.
643.   medias5$modelo="rf_set2"
644.
645.   medias5$tasa
646.   medias5$auc
647.
648.   medias6<-cruzadarfbin(data=tfmbis,
649.                           vardep="sentido_fallo",listconti=c("PC7"
    ,"PC15","PC27","PC54","PC44","PC35","PC52","PC43","PC42","PC17","PC
    62","PC12","PC4","PC1","PC168","PC45","PC40","PC5","PC13","PC105","
    PC63","PC188","PC187"),
650.                           listclass=c("tribunal.1","tribunal.2","t
    ribunal.3"),
651.                           grupos=4,sinicio=1234,repe=5,nodesize=10
    ,
652.                           mtry=24,ntree=200,replace=TRUE,
    sampsize = 500)
653.
654.   medias6$modelo="rf_set3"
655.
```

```
656.  medias6$tasa
657.  medias6$auc
658.
659.
660.  union3<-rbind(medias04,medias5, medias6)
661.  par(cex.axis=1.2)
662.  boxplot(data=union3,tasa~modelo,main="TASA FALLOS
   RF",col="lightblue")
663.  boxplot(data=union3,auc~modelo,main="AUC RF",col="lightgrey")
664.
665.
666.  medias7<-cruzadarfbin(data=tfmbis,
667.                        vardep="sentido_fallo",listconti=c("PC1",
   "PC2", "PC3", "PC4", "PC5", "PC6", "PC7", "PC8", "PC9",
668.                                                          "PC10",
    "PC11", "PC12", "PC13", "PC14", "PC15", "PC16", "PC17",
669.                                                          "PC18",
    "PC19", "PC20", "PC21", "PC22", "PC23", "PC24", "PC25",
670.                                                          "PC26",
    "PC27", "PC28", "PC29", "PC30", "PC31", "PC32", "PC33",
671.                                                          "PC34",
    "PC35", "PC36", "PC37", "PC38", "PC39", "PC40", "PC41",
672.                                                          "PC42",
    "PC43", "PC44", "PC45", "PC46", "PC47", "PC48", "PC49",
673.                                                          "PC50",
    "PC51", "PC52", "PC53", "PC54", "PC55", "PC56", "PC57",
674.                                                          "PC58",
    "PC59", "PC60", "PC61", "PC62", "PC63", "PC64", "PC65",
675.                                                          "PC66",
    "PC67", "PC68", "PC69", "PC70", "PC71", "PC72", "PC73",
676.                                                          "PC74",
    "PC75", "PC76", "PC77", "PC78", "PC79", "PC80", "PC81",
677.                                                          "PC82",
    "PC83", "PC84", "PC85", "PC86", "PC87", "PC88", "PC89",
678.                                                          "PC90",
    "PC91", "PC92", "PC93", "PC94", "PC95", "PC96", "PC97",
679.                                                          "PC98",
    "PC99", "PC100", "PC101", "PC102", "PC103", "PC104",
680.                                                          "PC105"
   , "PC106", "PC107", "PC108", "PC109", "PC110", "PC111",
681.                                                          "PC112"
   , "PC113", "PC114", "PC115", "PC116", "PC117", "PC118",
682.                                                          "PC119"
   , "PC120", "PC121", "PC122", "PC123", "PC124", "PC125",
683.                                                          "PC126"
   , "PC127", "PC128", "PC129", "PC130", "PC131", "PC132",
684.                                                          "PC133"
   , "PC134", "PC135", "PC136", "PC137", "PC138", "PC139",
685.                                                          "PC140"
   , "PC141", "PC142", "PC143", "PC144", "PC145", "PC146",
686.                                                          "PC147"
   , "PC148", "PC149", "PC150", "PC151", "PC152", "PC153",
687.                                                          "PC154"
   , "PC155", "PC156", "PC157", "PC158", "PC159", "PC160",
688.                                                          "PC161"
   , "PC162", "PC163", "PC164", "PC165", "PC166", "PC167",
689.                                                          "PC168"
   , "PC169", "PC170", "PC171", "PC172", "PC173", "PC174",
690.                                                          "PC175"
   , "PC176", "PC177", "PC178", "PC179", "PC180", "PC181",
```

```
691.                                                                  "PC182"
   , "PC183", "PC184", "PC185", "PC186", "PC187", "PC188",
692.                                                                  "PC189"
   , "PC190", "PC191", "PC192", "PC193", "PC194", "PC195",
693.                                                                  "PC196"
   , "PC197", "PC198", "PC199", "PC200"),
694.                          listclass=c("genero.0", "genero.1", "tribu
   nal.1", "tribunal.2", "tribunal.3",
695.                                      "fecha_reforma.1", "fecha_refo
   rma.2", "fecha_reforma.3", "fecha_reforma.4"),
696.                          grupos=4,sinicio=1234,repe=5,nodesize=10,
697.                          mtry=205,ntree=200,replace=TRUE,
   sampsize = 500)
698.
699.  medias7$modelo="rf_set4"
700.
701.  medias7$tasa
702.  medias7$auc
703.
704.  union3<-rbind(medias04,medias5, medias6,medias7)
705.  par(cex.axis=1.2)
706.  boxplot(data=union3,tasa~modelo,main="TASA FALLOS
   RF",col="lightblue")
707.  boxplot(data=union3,auc~modelo,main="AUC RF",col="lightgrey")
708.
709.  # EJEMPLOS GRADIENT BOOSTING
710.
711.  # TUNEADO DE GRADIENT BOOSTING CON CARET
712.
713.  # Caret permite tunear estos parÃ¡metros bÃ¡sicos:
714.  #
715.  #       shrinkage (parÃ¡metro v de regularizaciÃ³n, mide la
   velocidad de ajuste, a menor v, mÃ¡s lento y necesita mÃ¡s
   iteraciones, pero es mÃ¡s fino en el ajuste)
716.  #       n.minobsinnode: tamaÃ±o mÃ¡ximo de nodos finales (el
   principal parÃ¡metro que mide la complejidad)
717.  #       n.trees=el nÃºmero de iteraciones (Ã¡rboles)
718.  #       interaction.depth (2 para Ã¡rboles binarios)
719.
720.
721.  library(caret)
722.
723.  set.seed(12345)
724.
725.  gbmgrid<-expand.grid(shrinkage=c(0.1,0.05,0.03,0.01,0.001),
726.                       n.minobsinnode=c(5,10,20),
727.                       n.trees=c(100,500,1000,5000),
728.                       interaction.depth=c(2))
729.
730.  control<-
   trainControl(method = "cv",number=4,savePredictions = "all",
731.                       classProbs=TRUE)
732.
733.  gbm1<- train(factor(sentido_fallo)~tribunal.1 +tribunal.2 +tribu
   nal.3 + PC7 + PC15 +    PC27 + PC54     + PC25  + PC52 + PC22 + PC4
   2 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC154 + PC193 + PC1
   44+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+ PC176+ PC198+ PC
   71+ PC74+ PC78,
734.               data=tfmbis,
735.               method="gbm",trControl=control,tuneGrid=gbmgrid,
```

```
736.                distribution="bernoulli",
   bag.fraction=1,verbose=FALSE)
737.
738.  gbm1
739.
740.  plot(gbm1)
741.
742.
743.  gbm2<- train(factor(sentido_fallo)~PC7+ PC15+   PC27+   PC54+
   PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC61+   PC22+   PC9
   +    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+   PC40+   PC4+
     PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  PC26+   PC76+   P
   C131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC134+  PC125+  PC11
   2+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+   PC14+   PC166+
   PC30+   PC71,
744.                data=tfmbis,
745.                method="gbm",trControl=control,tuneGrid=gbmgrid,
746.                distribution="bernoulli",
   bag.fraction=1,verbose=FALSE)
747.
748.  gbm2
749.
750.  plot(gbm2)
751.
752.  gbm3<- train(factor(sentido_fallo)~tribunal.1 +tribunal.2 +tribu
   nal.3 +PC7+     PC15+   PC27+   PC54+   PC35+   PC44+   PC35+   PC5
   2+   PC43+   PC42+   PC17+   PC62+   PC12+   PC4+    PC1+    PC168+
     PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   PC188+  PC187,
753.                data=tfmbis,
754.                method="gbm",trControl=control,tuneGrid=gbmgrid,
755.                distribution="bernoulli",
   bag.fraction=1,verbose=FALSE)
756.
757.  gbm3
758.
759.  plot(gbm3)
760.
761.
762.  gbm4<- train(factor(sentido_fallo)~.,  data=tfmbis,
763.                method="gbm",trControl=control,tuneGrid=gbmgrid,
764.                distribution="bernoulli",
   bag.fraction=1,verbose=FALSE)
765.
766.  gbm4
767.
768.  plot(gbm4)
769.
770.
771.  # ESTUDIO DE EARLY STOPPING
772.  # Probamos a fijar algunos parÃ¡metros para ver como evoluciona
773.  # en funcion de las iteraciones
774.
775.
776.  gbmgrid<-expand.grid(shrinkage=c(0.001),
777.                       n.minobsinnode=c(5),
778.                       n.trees=c(100,300,500,800,1000,1200),
779.                       interaction.depth=c(2))
780.
781.  control<-
   trainControl(method = "cv",number=4,savePredictions = "all",
782.                        classProbs=TRUE)
```

```
783.
784.
785.  egbm1<- train(factor(sentido_fallo)~tribunal.1 +tribunal.2 +trib
   unal.3 + PC7 + PC15 +   PC27 + PC54     + PC25  + PC52 + PC22 + PC4
   2 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC154 + PC193 + PC1
   44+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+ PC176+ PC198+ PC
   71+ PC74+ PC78,
786.               data=tfmbis,
787.               method="gbm",trControl=control,tuneGrid=gbmgrid,
788.               distribution="bernoulli",
   bag.fraction=1,verbose=FALSE)
789.
790.  egbm1
791.
792.  plot(egbm1, main = "early stopping set 1")
793.
794.  egbm2<- train(factor(sentido_fallo)~PC7+        PC15+   PC27+
   PC54+   PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC61+   PC2
   2+   PC9+    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+   PC40+
     PC4+    PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  PC26+   P
   C76+   PC131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC134+  PC12
   5+  PC112+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+   PC14+
   PC166+  PC30+   PC71,
795.               data=tfmbis,
796.               method="gbm",trControl=control,tuneGrid=gbmgrid,
797.               distribution="bernoulli",
   bag.fraction=1,verbose=FALSE)
798.
799.  egbm2
800.
801.  plot(egbm2)
802.
803.  egbm3<- train(factor(sentido_fallo)~tribunal.1 +tribunal.2 +trib
   unal.3 +PC7+    PC15+   PC27+   PC54+   PC35+   PC44+   PC35+   PC5
   2+   PC43+   PC42+   PC17+   PC62+   PC12+   PC4+    PC1+    PC168+
     PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   PC188+  PC187,
804.               data=tfmbis,
805.               method="gbm",trControl=control,tuneGrid=gbmgrid,
806.               distribution="bernoulli",
   bag.fraction=1,verbose=FALSE)
807.
808.  egbm3
809.
810.  plot(egbm3)
811.
812.
813.  egbm4<- train(factor(sentido_fallo)~.,  data=tfmbis,
814.                method="gbm",trControl=control,tuneGrid=gbmgrid,
815.                distribution="bernoulli",
   bag.fraction=1,verbose=FALSE)
816.
817.  egbm4
818.
819.  plot(egbm4)
820.
821.
822.  # IMPORTANCIA DE VARIABLES
823.  par(cex=1.3)
824.  summary(egbm1)
825.
826.  tabla01<-summary(egbm1)
```

```
827.  par(cex=1.5,las=2)
828.  barplot(tabla01$rel.inf,names.arg=row.names(tabla01),main="Impor
   tancia de las variables")
829.
830.
831.  par(cex=1.3)
832.  summary(egbm2)
833.
834.  tabla02<-summary(egbm2)
835.  par(cex=1.5,las=2)
836.  barplot(tabla02$rel.inf,names.arg=row.names(tabla02),main="Impor
   tancia de las variables")
837.
838.  par(cex=1.3)
839.  summary(egbm3)
840.
841.  tabla03<-summary(egbm3)
842.  par(cex=1.5,las=2)
843.  barplot(tabla03$rel.inf,names.arg=row.names(tabla03),main="Impor
   tancia de las variables")
844.
845.  tabla04<-summary(egbm4)
846.  par(cex=0.5,las=2)
847.  barplot(tabla04$rel.inf,names.arg=row.names(tabla04),main="Impor
   tancia de las variables")
848.
849.
850.  #VALIDACION CRUZADA
851.  cruzadagbmbin<-
852.    function(data=data,vardep="vardep",
853.             listconti="listconti",listclass="listclass",
854.             grupos=4,sinicio=1234,repe=5,
855.             n.minobsinnode=20,shrinkage=0.1,n.trees=100,interacti
   on.depth=2)
856.    {
857.
858.      # PreparaciÃ³n del archivo
859.
860.      # b)pasar las categÃ³ricas a dummies
861.
862.      if  (listclass!=c(""))
863.      {
864.        databis<-data[,c(vardep,listconti,listclass)]
865.        databis<- dummy.data.frame(databis, listclass, sep = ".")
866.      }  else   {
867.        databis<-data[,c(vardep,listconti)]
868.      }
869.
870.      # c)estandarizar las variables continuas
871.
872.      # Calculo medias y dtipica de datos y estandarizo (solo las
   continuas)
873.
874.      means <-apply(databis[,listconti],2,mean)
875.      sds<-sapply(databis[,listconti],sd)
876.
877.      # Estandarizo solo las continuas y uno con las categoricas
878.
879.      datacon<-scale(databis[,listconti],
   center = means, scale = sds)
880.      numerocont<-which(colnames(databis)%in%listconti)
```

```
881.      databis<-cbind(datacon,databis[,-numerocont,drop=FALSE ])
882.
883.      databis[,vardep]<-as.factor(databis[,vardep])
884.
885.      formu<-formula(paste("factor(",vardep,")~.",sep=""))
886.
887.      # Preparo caret
888.
889.      set.seed(sinicio)
890.      control<-
   trainControl(method = "repeatedcv",number=grupos,repeats=repe,
891.                           savePredictions = "all",classProbs=TRU
   E)
892.
893.      # Aplico caret y construyo modelo
894.
895.
896.
897.      gbmgrid <-expand.grid(n.minobsinnode=n.minobsinnode,
898.                           shrinkage=shrinkage,n.trees=n.trees,
899.                           interaction.depth=interaction.depth)
900.
901.      gbm<- train(formu,data=databis,
902.                  method="gbm",trControl=control,
903.                  tuneGrid=gbmgrid,distribution="bernoulli",verbos
   e=FALSE)
904.
905.      print(gbm$results)
906.
907.      preditest<-gbm$pred
908.
909.
910.      preditest$prueba<-strsplit(preditest$Resample,"[.]")
911.      preditest$Fold <- sapply(preditest$prueba, "[", 1)
912.      preditest$Rep <- sapply(preditest$prueba, "[", 2)
913.      preditest$prueba<-NULL
914.
915.      tasafallos<-function(x,y) {
916.        confu<-confusionMatrix(x,y)
917.        tasa<-confu[[3]][1]
918.        return(tasa)
919.      }
920.
921.      # Aplicamos funciÃ³n sobre cada RepeticiÃ³n
922.
923.      medias<-preditest %>%
924.        group_by(Rep) %>%
925.        summarize(tasa=1-tasafallos(pred,obs))
926.
927.      # CalculamoS AUC  por cada RepeticiÃ³n de cv
928.      # Definimnos funciÃ³n
929.
930.      auc<-function(x,y) {
931.        curvaroc<-roc(response=x,predictor=y)
932.        auc<-curvaroc$auc
933.        return(auc)
934.      }
935.
936.      # Aplicamos funciÃ³n sobre cada RepeticiÃ³n
937.
938.      mediasbis<-preditest %>%
```

```
939.         group_by(Rep) %>%
940.         summarize(auc=auc(obs,Yes))
941.
942.       # Unimos la info de auc y de tasafallos
943.
944.       medias$auc<-mediasbis$auc
945.
946.       return(medias)
947.
948.     }
949.
950.
951.
952.   medias100<-cruzadagbmbin(data=tfmbis, vardep="sentido_fallo",
953.                            listconti=c("PC7","PC15","PC27","PC54",
    "PC25","PC52","PC22","PC42","PC43","PC4","PC111","PC13","PC175","PC
    126","PC154","PC193","PC144","PC200","PC141","PC96","PC59","PC125",
    "PC199","PC95","PC176","PC198","PC71","PC74","PC78"),
954.                            listclass=c("tribunal.1", "tribunal.2",
     "tribunal.3"),
955.                            grupos=4,sinicio=1234,repe=5,
956.                            n.minobsinnode=5,shrinkage=0.001,n.tree
    s=500,interaction.depth=2)
957.
958.   medias100$modelo="gbm1"
959.   medias100$tasa
960.   medias100$auc
961.
962.   medias101<-cruzadagbmbin(data=tfmbis, vardep="sentido_fallo",
963.                            listconti=c("PC7","PC15","PC27","PC54",
    "PC25","PC44","PC35","PC3","PC2","PC52","PC61","PC22","PC9","PC55",
    "PC42","PC36","PC43","PC64","PC40","PC4","PC32","PC105","PC69","PC1
    40","PC117","PC142","PC26","PC76","PC131","PC193","PC24","PC115","P
    C150","PC56","PC134","PC125","PC112","PC63","PC46","PC176","PC16","
    PC104","PC20","PC14","PC166","PC30","PC71"),
964.                            listclass=c(""),
965.                            grupos=4,sinicio=1234,repe=5,
966.                            n.minobsinnode=20,shrinkage=0.05,n.tree
    s=800,interaction.depth=2)
967.
968.   medias101$modelo="gbm2"
969.
970.   medias101$tasa
971.   medias101$auc
972.
973.   medias102<-cruzadagbmbin(data=tfmbis, vardep="sentido_fallo",
974.                            listconti=c("PC7","PC15","PC27","PC54",
    "PC44","PC35","PC52","PC43","PC42","PC17","PC62","PC12","PC4","PC1"
    ,"PC168","PC45","PC40","PC5","PC13","PC105","PC63","PC188","PC187")
    ,
975.                            listclass=c("tribunal.1", "tribunal.2",
     "tribunal.3"),
976.                            grupos=4,sinicio=1234,repe=5,
977.                            n.minobsinnode=20,shrinkage=0.05,n.tree
    s=1000,interaction.depth=2)
978.
979.   medias102$modelo="gbm3"
980.   medias102$tasa
981.   medias102$auc
982.
983.
```

```
984.  medias103<-cruzadagbmbin(data=tfmbis,
   vardep="sentido_fallo",listconti=c("PC1", "PC2", "PC3", "PC4", "PC5
   ", "PC6", "PC7", "PC8", "PC9",
985.                                                          "PC1
   0", "PC11", "PC12", "PC13", "PC14", "PC15", "PC16", "PC17",
986.                                                          "PC1
   8", "PC19", "PC20", "PC21", "PC22", "PC23", "PC24", "PC25",
987.                                                          "PC2
   6", "PC27", "PC28", "PC29", "PC30", "PC31", "PC32", "PC33",
988.                                                          "PC3
   4", "PC35", "PC36", "PC37", "PC38", "PC39", "PC40", "PC41",
989.                                                          "PC4
   2", "PC43", "PC44", "PC45", "PC46", "PC47", "PC48", "PC49",
990.                                                          "PC5
   0", "PC51", "PC52", "PC53", "PC54", "PC55", "PC56", "PC57",
991.                                                          "PC5
   8", "PC59", "PC60", "PC61", "PC62", "PC63", "PC64", "PC65",
992.                                                          "PC6
   6", "PC67", "PC68", "PC69", "PC70", "PC71", "PC72", "PC73",
993.                                                          "PC7
   4", "PC75", "PC76", "PC77", "PC78", "PC79", "PC80", "PC81",
994.                                                          "PC8
   2", "PC83", "PC84", "PC85", "PC86", "PC87", "PC88", "PC89",
995.                                                          "PC9
   0", "PC91", "PC92", "PC93", "PC94", "PC95", "PC96", "PC97",
996.                                                          "PC9
   8", "PC99", "PC100", "PC101", "PC102", "PC103", "PC104",
997.                                                          "PC1
   05", "PC106", "PC107", "PC108", "PC109", "PC110", "PC111",
998.                                                          "PC1
   12", "PC113", "PC114", "PC115", "PC116", "PC117", "PC118",
999.                                                          "PC1
   19", "PC120", "PC121", "PC122", "PC123", "PC124", "PC125",
1000.                                                         "PC1
   26", "PC127", "PC128", "PC129", "PC130", "PC131", "PC132",
1001.                                                         "PC1
   33", "PC134", "PC135", "PC136", "PC137", "PC138", "PC139",
1002.                                                         "PC1
   40", "PC141", "PC142", "PC143", "PC144", "PC145", "PC146",
1003.                                                         "PC1
   47", "PC148", "PC149", "PC150", "PC151", "PC152", "PC153",
1004.                                                         "PC1
   54", "PC155", "PC156", "PC157", "PC158", "PC159", "PC160",
1005.                                                         "PC1
   61", "PC162", "PC163", "PC164", "PC165", "PC166", "PC167",
1006.                                                         "PC1
   68", "PC169", "PC170", "PC171", "PC172", "PC173", "PC174",
1007.                                                         "PC1
   75", "PC176", "PC177", "PC178", "PC179", "PC180", "PC181",
1008.                                                         "PC1
   82", "PC183", "PC184", "PC185", "PC186", "PC187", "PC188",
1009.                                                         "PC1
   89", "PC190", "PC191", "PC192", "PC193", "PC194", "PC195",
1010.                                                         "PC1
   96", "PC197", "PC198", "PC199", "PC200"),
1011.                          listclass=c("genero.0", "genero.1", "tr
   ibunal.1", "tribunal.2", "tribunal.3",
1012.                                      "fecha_reforma.1", "fecha_r
   eforma.2", "fecha_reforma.3", "fecha_reforma.4"),
1013.                          grupos=4,sinicio=1234,repe=5,
```

```
                          n.minobsinnode=5,shrinkage=0.001,n.trees=1000,interaction.depth=2)

medias103$modelo="gbm4"
medias103$tasa
medias103$auc


union1000<-rbind(medias100,medias101,medias102,medias103)

par(cex.axis=0.8)
boxplot(data=union1000,tasa~modelo,main="TASA FALLOS GBM",col="pink")
boxplot(data=union1000,auc~modelo,main="AUC GBM", col="lightblue")



# EJEMPLOS XGBOOST

# TUNEADO DE XGBOOST CON CARET

# Caret permite tunear estos parÃ¡metros bÃ¡sicos:
#
# nrounds (# Boosting Iterations)
# max_depth (Max Tree Depth)
# eta (Shrinkage)
# gamma (Minimum Loss Reduction)
# colsample_bytree (Subsample Ratio of Columns)
# min_child_weight (Minimum Sum of Instance Weight)
# subsample (Subsample Percentage)
set.seed(12345)

xgbmgrid<-expand.grid(
  min_child_weight=c(5,10,20),
  eta=c(0.1,0.05,0.03,0.01,0.001),
  nrounds=c(100,500,1000,5000),
  max_depth=6,gamma=0,colsample_bytree=1,subsample=1)

control<-
  trainControl(method = "cv",number=4,savePredictions = "all",
                       classProbs=TRUE)

xgbm1<- train(factor(sentido_fallo)~tribunal.1 +tribunal.2 +tribunal.3 + PC7 + PC15 +   PC27 + PC54     + PC25  + PC52 + PC22 + PC42 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC154 + PC193 + PC144+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+ PC176+ PC198+ PC71+ PC74+ PC78,data=tfmbis,
              method="xgbTree",trControl=control,
              tuneGrid=xgbmgrid,verbose=FALSE)

xgbm1

plot(xgbm1)

xgbm2<- train(factor(sentido_fallo)~PC7+        PC15+   PC27+   PC54+   PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC61+   PC22+   PC9+    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+   PC40+   PC4+    PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  PC26+   PC76+   PC131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC134+  PC12
```

```
   5+  PC112+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+   PC14+
   PC166+  PC30+   PC71,
1062.               data=tfmbis,
1063.               method="xgbTree",trControl=control,
1064.               tuneGrid=xgbmgrid,verbose=FALSE)
1065.
1066. xgbm2
1067.
1068. plot(xgbm2, main= "XGBoost set 2")
1069.
1070. xgbm3<- train(factor(sentido_fallo)~tribunal.1 +tribunal.2 +trib
   unal.3 +PC7+    PC15+   PC27+   PC54+   PC35+   PC44+   PC35+   PC5
   2+   PC43+   PC42+   PC17+   PC62+   PC12+   PC4+    PC1+    PC168+
     PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   PC188+  PC187,
1071.               data=tfmbis,
1072.               method="xgbTree",trControl=control,
1073.               tuneGrid=xgbmgrid,verbose=FALSE)
1074.
1075. xgbm3
1076.
1077. plot(xgbm3)
1078.
1079. xgbm4<- train(factor(sentido_fallo)~.,data=tfmbis,
1080.               method="xgbTree",trControl=control,
1081.               tuneGrid=xgbmgrid,verbose=FALSE)
1082.
1083. xgbm4
1084.
1085. plot(xgbm4)
1086.
1087.
1088. # ESTUDIO DE EARLY STOPPING
1089. # Probamos a fijar algunos parÃ¡metros para ver como evoluciona
1090. # en funciÃ³n de las iteraciones
1091.
1092. xgbmgrid<-expand.grid(eta=c(0.1),
1093.                       min_child_weight=c(10),
1094.                       nrounds=c(50,100,150,200,250,300,500,1000,
   3000,5000),
1095.                       max_depth=6,gamma=0,colsample_bytree=1,sub
   sample=1)
1096.
1097. set.seed(12345)
1098. control<-
   trainControl(method = "cv",number=4,savePredictions = "all",
1099.                       classProbs=TRUE)
1100.
1101. exgbm1<- train(factor(sentido_fallo)~tribunal.1 +tribunal.2 +tri
   bunal.3 + PC7 + PC15 +  PC27 + PC54     + PC25  + PC52 + PC22 + PC4
   2 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC154 + PC193 + PC1
   44+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+ PC176+ PC198+ PC
   71+ PC74+ PC78,
1102.                data=tfmbis,
1103.               method="xgbTree",trControl=control,
1104.               tuneGrid=xgbmgrid,verbose=FALSE)
1105.
1106. plot(exgbm1,ylim=c(0.8,1))
1107.
1108. exgbm2<- train(factor(sentido_fallo)~PC7+       PC15+   PC27+
   PC54+   PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC61+   PC2
   2+   PC9+    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+   PC40+
```

```
     PC4+    PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  PC26+   P
   C76+   PC131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC134+  PC12
   5+  PC112+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+   PC14+
   PC166+  PC30+   PC71,
                 data=tfmbis,
                 method="xgbTree",trControl=control,
                 tuneGrid=xgbmgrid,verbose=FALSE)

plot(exgbm2,ylim=c(0.8,1), main="early stopping set 2")

exgbm3<- train(factor(sentido_fallo)~tribunal.1 +tribunal.2 +tri
   bunal.3 +PC7+   PC15+   PC27+   PC54+   PC35+   PC44+   PC35+   PC5
   2+   PC43+   PC42+   PC17+   PC62+   PC12+   PC4+    PC1+    PC168+
     PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   PC188+  PC187,
                 data=tfmbis,
                 method="xgbTree",trControl=control,
                 tuneGrid=xgbmgrid,verbose=FALSE)

plot(exgbm3,ylim=c(0.8,1), main="estudio early stopping - set
   3")

exgbm4<- train(factor(sentido_fallo)~., data=tfmbis,
                 method="xgbTree",trControl=control,
                 tuneGrid=xgbmgrid,verbose=FALSE)

plot(exgbm4,ylim=c(0.8,1))


# IMPORTANCIA DE VARIABLES

varImp(xgbm1)
plot(varImp(xgbm1), main = "Importancia vbles set 1")

varImp(xgbm2)
plot(varImp(xgbm2), main = "Importancia vbles set 2")

varImp(xgbm3)
plot(varImp(xgbm3), main = "Importancia vbles set 3")

varImp(xgbm4)
plot(varImp(xgbm4), main = "Importancia vbles set 4")



# UTILIZACIÃ"N DE LOS PARÃ□METROS  DE REGULARIZACIÃ"N

# SE CREAN DATA FRAMES VACÃ□OS Y SE VA METIENDO EL RESULTADO
# DE REALIZAR EL PROCESO CON CADA VALOR DEL PARÃ□METRO DE
# REGULARIZACIÃ"N

xgbmgrid<-expand.grid(eta=c(0.08),
                      min_child_weight=c(10),
                      nrounds=c(100),
                      max_depth=6,gamma=0,colsample_bytree=1,sub
   sample=1)


df<- data.frame(matrix(ncol = 2, nrow = 0))
x <- c("lambda", "Accuracy")
colnames(df) <- x
df2<- data.frame(matrix(ncol = 2, nrow = 0))
```

```
x <- c("lambda", "Accuracy")
colnames(df2) <- x

for (lambda in seq(0,20,1))
{
  xgbm<- train(factor(sentido_fallo)~.,data=tfmbis,
               method="xgbTree",trControl=control,
               tuneGrid=xgbmgrid,lambda=lambda,verbose=FALSE)
  cat(lambda,"\n")
  cat(xgbm$results$Accuracy,"\n")
  df[1,1]<-lambda
  df[1,2]<-xgbm$results$Accuracy
  df2<-rbind(df2,df)
}

plot(df2$lambda,df2$Accuracy)

xgbm$results$Accuracy

df<- data.frame(matrix(ncol = 2, nrow = 0))
x <- c("alpha", "Accuracy")
colnames(df) <- x
df2<- data.frame(matrix(ncol = 2, nrow = 0))
x <- c("alpha", "Accuracy")
colnames(df2) <- x

for (alpha in seq(0,20,1))
{
  xgbm<- train(factor(outcome)~.,data=monica1,
               method="xgbTree",trControl=control,
               tuneGrid=xgbmgrid,alpha=alpha,verbose=FALSE)
  cat(alpha,"\n")
  cat(xgbm$results$Accuracy,"\n")
  df[1,1]<-alpha
  df[1,2]<-xgbm$results$Accuracy
  df2<-rbind(df2,df)
}

plot(df2$alpha,df2$Accuracy)

xgbm$results$Accuracy

df<- data.frame(matrix(ncol = 2, nrow = 0))
x <- c("lambda_bias", "Accuracy")
colnames(df) <- x
df2<- data.frame(matrix(ncol = 2, nrow = 0))
x <- c("lambda_bias", "Accuracy")
colnames(df2) <- x


for (lambda_bias in seq(0,20,1))
{
  xgbm<- train(factor(outcome)~.,data=monica1,
               method="xgbTree",trControl=control,
               tuneGrid=xgbmgrid,lambda_bias=lambda_bias,verbose
  =FALSE)
  cat(lambda_bias,"\n")
  cat(xgbm$results$Accuracy,"\n")
  df[1,1]<-lambda_bias
  df[1,2]<-xgbm$results$Accuracy
  df2<-rbind(df2,df)
```

```
1221. }
1222.
1223. plot(df2$lambda_bias,df2$Accuracy)
1224.
1225. xgbm$results$Accuracy
1226.
1227. # La funciÃ³n cruzadagbmbin permite plantear gradient boosting
   para binarias
1228.
1229. cruzadaxgbmbin<-
1230.   function(data=data,vardep="vardep",
1231.            listconti="listconti",listclass="listclass",
1232.            grupos=4,sinicio=1234,repe=5,
1233.            min_child_weight=20,eta=0.1,nrounds=100,max_depth=2,
1234.            gamma=0,colsample_bytree=1,subsample=1,alpha=0,lambda
   =0,lambda_bias=0)
1235.   {
1236.
1237.     # PreparaciÃ³n del archivo
1238.
1239.     # b)pasar las categÃ³ricas a dummies
1240.
1241.     if  (listclass!=c(""))
1242.     {
1243.       databis<-data[,c(vardep,listconti,listclass)]
1244.       databis<- dummy.data.frame(databis, listclass, sep = ".")
1245.     }  else   {
1246.       databis<-data[,c(vardep,listconti)]
1247.     }
1248.
1249.     # c)estandarizar las variables continuas
1250.
1251.     # Calculo medias y dtipica de datos y estandarizo (solo las
   continuas)
1252.
1253.     means <-apply(databis[,listconti],2,mean)
1254.     sds<-sapply(databis[,listconti],sd)
1255.
1256.     # Estandarizo solo las continuas y uno con las categoricas
1257.
1258.     datacon<-scale(databis[,listconti],
   center = means, scale = sds)
1259.     numerocont<-which(colnames(databis)%in%listconti)
1260.     databis<-cbind(datacon,databis[,-numerocont,drop=FALSE ])
1261.
1262.     databis[,vardep]<-as.factor(databis[,vardep])
1263.
1264.     formu<-formula(paste("factor(",vardep,")~.",sep=""))
1265.
1266.     # Preparo caret
1267.
1268.     set.seed(sinicio)
1269.     control<-
   trainControl(method = "repeatedcv",number=grupos,repeats=repe,
1270.                            savePredictions = "all",classProbs=TRU
   E)
1271.
1272.     # Aplico caret y construyo modelo
1273.
1274.     xgbmgrid <-expand.grid( min_child_weight=min_child_weight,
```

```
                                eta=eta,nrounds=nrounds,max_depth=ma
   x_depth,
                                gamma=gamma,colsample_bytree=colsamp
   le_bytree,subsample=subsample)

     xgbm<- train(formu,data=databis,
                  method="xgbTree",trControl=control,
                  tuneGrid=xgbmgrid,objective = "binary:logistic"
   ,verbose=FALSE,
                  alpha=alpha,lambda=lambda,lambda_bias=lambda_bi
   as)

     print(xgbm$results)

     preditest<-xgbm$pred


     preditest$prueba<-strsplit(preditest$Resample,"[.]")
     preditest$Fold <- sapply(preditest$prueba, "[", 1)
     preditest$Rep <- sapply(preditest$prueba, "[", 2)
     preditest$prueba<-NULL

     tasafallos<-function(x,y) {
       confu<-confusionMatrix(x,y)
       tasa<-confu[[3]][1]
       return(tasa)
     }

     # Aplicamos funciÃ³n sobre cada RepeticiÃ³n

     medias<-preditest %>%
       group_by(Rep) %>%
       summarize(tasa=1-tasafallos(pred,obs))

     # CalculamoS AUC  por cada RepeticiÃ³n de cv
     # Definimnos funciÃ³n

     auc<-function(x,y) {
       curvaroc<-roc(response=x,predictor=y)
       auc<-curvaroc$auc
       return(auc)
     }

     # Aplicamos funciÃ³n sobre cada RepeticiÃ³n

     mediasbis<-preditest %>%
       group_by(Rep) %>%
       summarize(auc=auc(obs,Yes))

     # Unimos la info de auc y de tasafallos

     medias$auc<-mediasbis$auc

     return(medias)

   }


medias300<-cruzadaxgbmbin(data=tfmbis, vardep="sentido_fallo",
```

```
                            listconti=c("PC7","PC15","PC27","PC54"
   ,"PC25","PC52","PC22","PC42","PC43","PC4","PC111","PC13","PC175","P
   C126","PC154","PC193","PC144","PC200","PC141","PC96","PC59","PC125"
   ,"PC199","PC95","PC176","PC198","PC71","PC74","PC78"),
                            listclass=c("tribunal.1", "tribunal.2"
   , "tribunal.3"),
                            grupos=4,sinicio=1234,repe=5,
                            min_child_weight=10,eta=0.001,nrounds=
   5000,max_depth=6,
                            gamma=0,colsample_bytree=1,subsample=1
   ,
                            alpha=0,lambda=0,lambda_bias=0)

medias300$modelo="xgbm1"
medias300$auc
medias300$tasa

medias301<-cruzadaxgbmbin(data=tfmbis, vardep="sentido_fallo",
                            listconti=c("PC7","PC15","PC27","PC54"
   ,"PC25","PC44","PC35","PC3","PC2","PC52","PC61","PC22","PC9","PC55"
   ,"PC42","PC36","PC43","PC64","PC40","PC4","PC32","PC105","PC69","PC
   140","PC117","PC142","PC26","PC76","PC131","PC193","PC24","PC115","
   PC150","PC56","PC134","PC125","PC112","PC63","PC46","PC176","PC16",
   "PC104","PC20","PC14","PC166","PC30","PC71"),
                            listclass=c(""),
                            grupos=4,sinicio=1234,repe=5,
                            min_child_weight=10,eta=0.1,nrounds=50
   0,max_depth=6,
                            gamma=0,colsample_bytree=1,subsample=1
   ,
                            alpha=0,lambda=0,lambda_bias=0)

medias301$modelo="xgbm2"
medias301$auc
medias301$tasa

medias302<-cruzadaxgbmbin(data=tfmbis, vardep="sentido_fallo",
                            listconti=c("PC7","PC15","PC27","PC54"
   ,"PC44","PC35","PC52","PC43","PC42","PC17","PC62","PC12","PC4","PC1
   ","PC168","PC45","PC40","PC5","PC13","PC105","PC63","PC188","PC187"
   ),
                            listclass=c("tribunal.1", "tribunal.2"
   , "tribunal.3"),
                            grupos=4,sinicio=1234,repe=5,
                            min_child_weight=10,eta=0.03,nrounds=1
   000,max_depth=6,
                            gamma=0,colsample_bytree=1,subsample=1
   ,
                            alpha=0,lambda=0,lambda_bias=0)

medias302$modelo="xgbm3"
medias302$auc
medias302$tasa


medias303<-cruzadaxgbmbin(data=tfmbis, vardep="sentido_fallo",
                            listconti=c("PC1", "PC2", "PC3", "PC4"
   , "PC5", "PC6", "PC7", "PC8", "PC9",
                                        "PC10", "PC11", "PC12", "P
   C13", "PC14", "PC15", "PC16", "PC17",
```

```
1371.                                                "PC18", "PC19", "PC20", "P
   C21", "PC22", "PC23", "PC24", "PC25",
1372.                                                "PC26", "PC27", "PC28", "P
   C29", "PC30", "PC31", "PC32", "PC33",
1373.                                                "PC34", "PC35", "PC36", "P
   C37", "PC38", "PC39", "PC40", "PC41",
1374.                                                "PC42", "PC43", "PC44", "P
   C45", "PC46", "PC47", "PC48", "PC49",
1375.                                                "PC50", "PC51", "PC52", "P
   C53", "PC54", "PC55", "PC56", "PC57",
1376.                                                "PC58", "PC59", "PC60", "P
   C61", "PC62", "PC63", "PC64", "PC65",
1377.                                                "PC66", "PC67", "PC68", "P
   C69", "PC70", "PC71", "PC72", "PC73",
1378.                                                "PC74", "PC75", "PC76", "P
   C77", "PC78", "PC79", "PC80", "PC81",
1379.                                                "PC82", "PC83", "PC84", "P
   C85", "PC86", "PC87", "PC88", "PC89",
1380.                                                "PC90", "PC91", "PC92", "P
   C93", "PC94", "PC95", "PC96", "PC97",
1381.                                                "PC98", "PC99", "PC100", "
   PC101", "PC102", "PC103", "PC104",
1382.                                                "PC105", "PC106", "PC107",
    "PC108", "PC109", "PC110", "PC111",
1383.                                                "PC112", "PC113", "PC114",
    "PC115", "PC116", "PC117", "PC118",
1384.                                                "PC119", "PC120", "PC121",
    "PC122", "PC123", "PC124", "PC125",
1385.                                                "PC126", "PC127", "PC128",
    "PC129", "PC130", "PC131", "PC132",
1386.                                                "PC133", "PC134", "PC135",
    "PC136", "PC137", "PC138", "PC139",
1387.                                                "PC140", "PC141", "PC142",
    "PC143", "PC144", "PC145", "PC146",
1388.                                                "PC147", "PC148", "PC149",
    "PC150", "PC151", "PC152", "PC153",
1389.                                                "PC154", "PC155", "PC156",
    "PC157", "PC158", "PC159", "PC160",
1390.                                                "PC161", "PC162", "PC163",
    "PC164", "PC165", "PC166", "PC167",
1391.                                                "PC168", "PC169", "PC170",
    "PC171", "PC172", "PC173", "PC174",
1392.                                                "PC175", "PC176", "PC177",
    "PC178", "PC179", "PC180", "PC181",
1393.                                                "PC182", "PC183", "PC184",
    "PC185", "PC186", "PC187", "PC188",
1394.                                                "PC189", "PC190", "PC191",
    "PC192", "PC193", "PC194", "PC195",
1395.                                                "PC196", "PC197", "PC198",
    "PC199", "PC200"),
1396.                             listclass=c("genero.0", "genero.1", "t
   ribunal.1", "tribunal.2", "tribunal.3",
1397.                                                "fecha_reforma.1", "fecha_
   reforma.2", "fecha_reforma.3", "fecha_reforma.4"),
1398.                             grupos=4,sinicio=1234,repe=5,
1399.                             min_child_weight=5,eta=0.03,nrounds=10
   00,max_depth=6,
1400.                             gamma=0,colsample_bytree=1,subsample=1
   ,
1401.                             alpha=0,lambda=0,lambda_bias=0)
1402.
```

```
1403. medias303$modelo="xgbm4"
1404. medias303$auc
1405. medias303$tasa
1406.
1407. union2000<-rbind(medias300,medias301,medias302,medias303)
1408. par(cex.axis=1)
1409. boxplot(data=union2000,tasa~modelo,main="TASA FALLOS LOGÍSTICA
   XGBM + iter",col="pink")
1410. boxplot(data=union2000,auc~modelo,main="AUC LOGÍSTICA XGBM +
   iter",col="lightgreen")
1411.
1412.
1413. boxplot(data=union2000,tasa~modelo,main="TASA FALLOS XGBM +
   iter",col="pink")
1414. boxplot(data=union2000,auc~modelo,main="AUC
   XGBM",col="lightgreen")
1415.
1416.
1417.
1418. # *************************
1419. # TUNEADO SVM BINARIA
1420. # *************************
1421.
1422. #  SVM LINEAL: SOLO PARAMETRO C
1423.
1424. SVMgrid<-expand.grid(C=c(0.01,0.05,0.1,0.2,0.5,1,2,5,10))
1425.
1426. control<-
   trainControl(method = "cv",number=4,savePredictions = "all")
1427.
1428. SVM_lin1<- train(data=tfmbis,factor(sentido_fallo)~tribunal.1 +t
   ribunal.2 +tribunal.3 + PC7 + PC15 +    PC27 + PC54     + PC25  + P
   C52 + PC22 + PC42 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC1
   54 + PC193 + PC144+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+
   PC176+ PC198+ PC71+ PC74+ PC78,
1429.                  method="svmLinear",trControl=control,
1430.                  tuneGrid=SVMgrid,verbose=FALSE)
1431.
1432. SVM_lin1$results
1433. plot(SVM_lin1$results$C,SVM_lin1$results$Accuracy, main="SVM
   Lineal Set 1")
1434.
1435.
1436. SVM_lin2<- train(data=tfmbis,factor(sentido_fallo)~PC7+ PC15+
   PC27+   PC54+   PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC6
   1+   PC22+   PC9+    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+
     PC40+   PC4+    PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  P
   C26+   PC76+   PC131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC13
   4+  PC125+  PC112+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+
   PC14+   PC166+  PC30+   PC71,
1437.                  method="svmLinear",trControl=control,
1438.                  tuneGrid=SVMgrid,verbose=FALSE)
1439.
1440. SVM_lin2$results
1441. plot(SVM_lin2$results$C,SVM_lin2$results$Accuracy, main="SVM
   Lineal Set 2")
1442.
1443. SVM_lin3<- train(data=tfmbis,factor(sentido_fallo)~tribunal.1 +t
   ribunal.2 +tribunal.3 +PC7+     PC15+   PC27+   PC54+   PC35+   PC4
   4+   PC35+   PC52+   PC43+   PC42+   PC17+   PC62+   PC12+   PC4+
```

```
     PC1+    PC168+  PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   P
   C188+  PC187,
1444.                  method="svmLinear",trControl=control,
1445.                  tuneGrid=SVMgrid,verbose=FALSE)
1446.
1447. SVM_lin3$results
1448. plot(SVM_lin3$results$C,SVM_lin3$results$Accuracy)
1449.
1450. SVM_lin4<- train(data=tfmbis,factor(sentido_fallo)~.,method="svm
   Linear",trControl=control,
1451.                  tuneGrid=SVMgrid,verbose=FALSE)
1452.
1453. SVM_lin4$results
1454. plot(SVM_lin4$results$C,SVM_lin4$results$Accuracy)
1455.
1456.
1457. #validacion cruzada lineal
1458.
1459. cruzadaSVMbin<-
1460.   function(data=data,vardep="vardep",
1461.            listconti="listconti",listclass="listclass",
1462.            grupos=4,sinicio=1234,repe=5,
1463.            C=1,replace=TRUE)
1464.   {
1465.
1466.     # PreparaciÃ³n del archivo
1467.
1468.     # b)pasar las categÃ³ricas a dummies
1469.
1470.     if  (listclass!=c(""))
1471.     {
1472.       databis<-data[,c(vardep,listconti,listclass)]
1473.       databis<- dummy.data.frame(databis, listclass, sep = ".")
1474.     }  else   {
1475.       databis<-data[,c(vardep,listconti)]
1476.     }
1477.
1478.     # c)estandarizar las variables continuas
1479.
1480.     # Calculo medias y dtipica de datos y estandarizo (solo las
   continuas)
1481.
1482.     means <-apply(databis[,listconti],2,mean)
1483.     sds<-sapply(databis[,listconti],sd)
1484.
1485.     # Estandarizo solo las continuas y uno con las categoricas
1486.
1487.     datacon<-scale(databis[,listconti],
   center = means, scale = sds)
1488.     numerocont<-which(colnames(databis)%in%listconti)
1489.     databis<-cbind(datacon,databis[,-numerocont,drop=FALSE ])
1490.
1491.     databis[,vardep]<-as.factor(databis[,vardep])
1492.
1493.     formu<-formula(paste("factor(",vardep,")~.",sep=""))
1494.
1495.     # Preparo caret
1496.
1497.     set.seed(sinicio)
1498.     control<-
   trainControl(method = "repeatedcv",number=grupos,repeats=repe,
```

```
                           savePredictions = "all",classProbs=TRU
   E)

    # Aplico caret y construyo modelo

    SVMgrid <-expand.grid(C=C)

    SVM<- train(formu,data=databis,
                method="svmLinear",trControl=control,
                tuneGrid=SVMgrid,replace=replace)

    print(SVM$results)

    preditest<-SVM$pred

    preditest$prueba<-strsplit(preditest$Resample,"[.]")
    preditest$Fold <- sapply(preditest$prueba, "[", 1)
    preditest$Rep <- sapply(preditest$prueba, "[", 2)
    preditest$prueba<-NULL

    tasafallos<-function(x,y) {
      confu<-confusionMatrix(x,y)
      tasa<-confu[[3]][1]
      return(tasa)
    }

    # Aplicamos funciÃ³n sobre cada RepeticiÃ³n

    medias<-preditest %>%
      group_by(Rep) %>%
      summarize(tasa=1-tasafallos(pred,obs))

    # CalculamoS AUC  por cada RepeticiÃ³n de cv
    # Definimnos funciÃ³n

    auc<-function(x,y) {
      curvaroc<-roc(response=x,predictor=y)
      auc<-curvaroc$auc
      return(auc)
    }

    # Aplicamos funciÃ³n sobre cada RepeticiÃ³n

    mediasbis<-preditest %>%
      group_by(Rep) %>%
      summarize(auc=auc(obs,Yes))

    # Unimos la info de auc y de tasafallos

    medias$auc<-mediasbis$auc

    return(medias)

  }

medias91<-cruzadaSVMbin(data=tfmbis, vardep="sentido_fallo",
                        listconti=c("PC7","PC15","PC27","PC54","
  PC25","PC52","PC22","PC42","PC43","PC4","PC111","PC13","PC175","PC1
  26","PC154","PC193","PC144","PC200","PC141","PC96","PC59","PC125","
  PC199","PC95","PC176","PC198","PC71","PC74","PC78"),
```

```
                          listclass=c("tribunal.1", "tribunal.2",
   "tribunal.3"),
                          grupos=4,sinicio=1234,repe=5,
                          C=0.01)

medias91$modelo="SVM1"
medias91$auc
medias91$tasa


medias92<-cruzadaSVMbin(data=tfmbis, vardep="sentido_fallo",
                          listconti=c("PC7","PC15","PC27","PC54","
   PC25","PC44","PC35","PC3","PC2","PC52","PC61","PC22","PC9","PC55","
   PC42","PC36","PC43","PC64","PC40","PC4","PC32","PC105","PC69","PC14
   0","PC117","PC142","PC26","PC76","PC131","PC193","PC24","PC115","PC
   150","PC56","PC134","PC125","PC112","PC63","PC46","PC176","PC16","P
   C104","PC20","PC14","PC166","PC30","PC71"),
                          listclass=c(""),
                          grupos=4,sinicio=1234,repe=5,
                          C=0.05)

medias92$modelo="SVM2"
medias92$auc
medias92$tasa


medias93<-cruzadaSVMbin(data=tfmbis, vardep="sentido_fallo",
                          listconti=c("PC7","PC15","PC27","PC54","
   PC44","PC35","PC52","PC43","PC42","PC17","PC62","PC12","PC4","PC1",
   "PC168","PC45","PC40","PC5","PC13","PC105","PC63","PC188","PC187"),
                          listclass=c("tribunal.1", "tribunal.2",
   "tribunal.3"),
                          grupos=4,sinicio=1234,repe=5,
                          C=0.03)

medias93$modelo="SVM3"
medias93$auc
medias93$tasa


medias94<-cruzadaSVMbin(data=tfmbis, vardep="sentido_fallo",
                          listconti=c("PC1", "PC2", "PC3", "PC4",
   "PC5", "PC6", "PC7", "PC8", "PC9",
                                      "PC10", "PC11", "PC12", "PC1
   3", "PC14", "PC15", "PC16", "PC17",
                                      "PC18", "PC19", "PC20", "PC2
   1", "PC22", "PC23", "PC24", "PC25",
                                      "PC26", "PC27", "PC28", "PC2
   9", "PC30", "PC31", "PC32", "PC33",
                                      "PC34", "PC35", "PC36", "PC3
   7", "PC38", "PC39", "PC40", "PC41",
                                      "PC42", "PC43", "PC44", "PC4
   5", "PC46", "PC47", "PC48", "PC49",
                                      "PC50", "PC51", "PC52", "PC5
   3", "PC54", "PC55", "PC56", "PC57",
                                      "PC58", "PC59", "PC60", "PC6
   1", "PC62", "PC63", "PC64", "PC65",
                                      "PC66", "PC67", "PC68", "PC6
   9", "PC70", "PC71", "PC72", "PC73",
```

```
                                          "PC74", "PC75", "PC76", "PC7
   7", "PC78", "PC79", "PC80", "PC81",
                                          "PC82", "PC83", "PC84", "PC8
   5", "PC86", "PC87", "PC88", "PC89",
                                          "PC90", "PC91", "PC92", "PC9
   3", "PC94", "PC95", "PC96", "PC97",
                                          "PC98", "PC99", "PC100", "PC
   101", "PC102", "PC103", "PC104",
                                          "PC105", "PC106", "PC107", "
   PC108", "PC109", "PC110", "PC111",
                                          "PC112", "PC113", "PC114", "
   PC115", "PC116", "PC117", "PC118",
                                          "PC119", "PC120", "PC121", "
   PC122", "PC123", "PC124", "PC125",
                                          "PC126", "PC127", "PC128", "
   PC129", "PC130", "PC131", "PC132",
                                          "PC133", "PC134", "PC135", "
   PC136", "PC137", "PC138", "PC139",
                                          "PC140", "PC141", "PC142", "
   PC143", "PC144", "PC145", "PC146",
                                          "PC147", "PC148", "PC149", "
   PC150", "PC151", "PC152", "PC153",
                                          "PC154", "PC155", "PC156", "
   PC157", "PC158", "PC159", "PC160",
                                          "PC161", "PC162", "PC163", "
   PC164", "PC165", "PC166", "PC167",
                                          "PC168", "PC169", "PC170", "
   PC171", "PC172", "PC173", "PC174",
                                          "PC175", "PC176", "PC177", "
   PC178", "PC179", "PC180", "PC181",
                                          "PC182", "PC183", "PC184", "
   PC185", "PC186", "PC187", "PC188",
                                          "PC189", "PC190", "PC191", "
   PC192", "PC193", "PC194", "PC195",
                                          "PC196", "PC197", "PC198", "
   PC199", "PC200"),
                           listclass=c("genero.0", "genero.1", "tri
   bunal.1", "tribunal.2", "tribunal.3",
                                          "fecha_reforma.1", "fecha_re
   forma.2", "fecha_reforma.3", "fecha_reforma.4"),
                           grupos=4,sinicio=1234,repe=5,
                           C=0.03)

medias94$modelo="SVM4"
medias93$auc
medias93$tasa

union90<-rbind(medias91,medias92,medias93, medias94)

par(cex.axis=1.5)
boxplot(data=union90,tasa~modelo,main="TASA FALLOS SVM
   LIN",col="pink")
boxplot(data=union90,auc~modelo,main="AUC SVM
   LIN",col="lightgreen")

#  SVM Polinomial: PARÃMETROS C, degree, scale

SVMgrid<-expand.grid(C=c(0.01,0.05,0.1,0.2,0.5,1,2,5,10),
                     degree=c(2,3),scale=c(0.1,0.5,1,2,5))

control<-trainControl(method = "cv",
```

```
1636.                       number=4,savePredictions = "all")
1637.
1638. SVM_pol1<- train(data=tfmbis,factor(sentido_fallo)~tribunal.1 +t
   ribunal.2 +tribunal.3 + PC7 + PC15 +    PC27 + PC54     + PC25  + P
   C52 + PC22 + PC42 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC1
   54 + PC193 + PC144+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+
   PC176+ PC198+ PC71+ PC74+ PC78,
1639.                 method="svmPoly",trControl=control,
1640.                 tuneGrid=SVMgrid,verbose=FALSE)
1641.
1642. SVM_pol1
1643. plot(SVM_pol1)
1644. SVM_pol1$results
1645.
1646.
1647. SVM_pol2<- train(data=tfmbis,factor(sentido_fallo)~PC7+ PC15+
   PC27+   PC54+   PC25+   PC44+   PC35+   PC3+    PC2+    PC52+   PC6
   1+   PC22+   PC9+    PC55+   PC42+   PC36+   PC166+  PC43+   PC64+
     PC40+   PC4+    PC32+   PC105+  PC69+   PC140+  PC117+  PC142+  P
   C26+   PC76+   PC131+  PC193+  PC24+   PC115+  PC150+  PC56+   PC13
   4+  PC125+  PC112+  PC63+   PC46+   PC176+  PC16+   PC104+  PC20+
   PC14+   PC166+  PC30+   PC71,
1648.                  method="svmPoly",trControl=control,
1649.                  tuneGrid=SVMgrid,verbose=FALSE)
1650.
1651. SVM_pol2
1652.
1653. SVM_pol2$results
1654.
1655.
1656. SVM_pol3<- train(data=tfmbis,factor(sentido_fallo)~tribunal.1 +t
   ribunal.2 +tribunal.3 +PC7+     PC15+   PC27+   PC54+   PC35+   PC4
   4+   PC35+   PC52+   PC43+   PC42+   PC17+   PC62+   PC12+   PC4+
     PC1+    PC168+  PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   P
   C188+  PC187,
1657.                  method="svmPoly",trControl=control,
1658.                  tuneGrid=SVMgrid,verbose=FALSE)
1659.
1660. SVM_pol3
1661. plot(SVM_pol3, main="tuneado SVM Poly set 3")
1662.
1663. SVM_pol3$results
1664.
1665.
1666. SVM_pol4<- train(data=tfmbis,factor(sentido_fallo)~.,method="svm
   Poly",trControl=control,
1667.                  tuneGrid=SVMgrid,verbose=FALSE)
1668.
1669. SVM_pol4
1670.
1671. SVM_pol4$results
1672.
1673. #validacion cruzada para SVM polinomial
1674.
1675. cruzadaSVMbinPoly<-
1676.   function(data=data,vardep="vardep",
1677.            listconti="listconti",listclass="listclass",
1678.            grupos=4,sinicio=1234,repe=5,
1679.            C=1,degree=2,scale=1)
1680.   {
1681.
```

```
1682.     # PreparaciÃ³n del archivo
1683.
1684.     # b)pasar las categÃ³ricas a dummies
1685.
1686.     if  (listclass!=c(""))
1687.     {
1688.       databis<-data[,c(vardep,listconti,listclass)]
1689.       databis<- dummy.data.frame(databis, listclass, sep = ".")
1690.     }  else   {
1691.       databis<-data[,c(vardep,listconti)]
1692.     }
1693.
1694.     # c)estandarizar las variables continuas
1695.
1696.     # Calculo medias y dtipica de datos y estandarizo (solo las
   continuas)
1697.
1698.     means <-apply(databis[,listconti],2,mean)
1699.     sds<-sapply(databis[,listconti],sd)
1700.
1701.     # Estandarizo solo las continuas y uno con las categoricas
1702.
1703.     datacon<-scale(databis[,listconti],
   center = means, scale = sds)
1704.     numerocont<-which(colnames(databis)%in%listconti)
1705.     databis<-cbind(datacon,databis[,-numerocont,drop=FALSE ])
1706.
1707.     databis[,vardep]<-as.factor(databis[,vardep])
1708.
1709.     formu<-formula(paste("factor(",vardep,")~.",sep=""))
1710.
1711.     # Preparo caret
1712.
1713.     set.seed(sinicio)
1714.     control<-
   trainControl(method = "repeatedcv",number=grupos,repeats=repe,
1715.                            savePredictions = "all",classProbs=TRU
   E)
1716.
1717.     # Aplico caret y construyo modelo
1718.
1719.     SVMgrid <-expand.grid(C=C,degree=degree,scale=scale)
1720.
1721.     SVM<- train(formu,data=databis,
1722.                 method="svmPoly",trControl=control,
1723.                 tuneGrid=SVMgrid,replace=replace)
1724.
1725.     print(SVM$results)
1726.
1727.     preditest<-SVM$pred
1728.
1729.     preditest$prueba<-strsplit(preditest$Resample,"[.]")
1730.     preditest$Fold <- sapply(preditest$prueba, "[", 1)
1731.     preditest$Rep <- sapply(preditest$prueba, "[", 2)
1732.     preditest$prueba<-NULL
1733.
1734.     tasafallos<-function(x,y) {
1735.       confu<-confusionMatrix(x,y)
1736.       tasa<-confu[[3]][1]
1737.       return(tasa)
1738.     }
```

```
1739.
1740.     # Aplicamos funciÃ³n sobre cada RepeticiÃ³n
1741.
1742.     medias<-preditest %>%
1743.       group_by(Rep) %>%
1744.       summarize(tasa=1-tasafallos(pred,obs))
1745.
1746.     # CalculamoS AUC  por cada RepeticiÃ³n de cv
1747.     # Definimnos funciÃ³n
1748.
1749.     auc<-function(x,y) {
1750.       curvaroc<-roc(response=x,predictor=y)
1751.       auc<-curvaroc$auc
1752.       return(auc)
1753.     }
1754.
1755.     # Aplicamos funciÃ³n sobre cada RepeticiÃ³n
1756.
1757.     mediasbis<-preditest %>%
1758.       group_by(Rep) %>%
1759.       summarize(auc=auc(obs,Yes))
1760.
1761.     # Unimos la info de auc y de tasafallos
1762.
1763.     medias$auc<-mediasbis$auc
1764.
1765.     return(medias)
1766.
1767.   }
1768.
1769.
1770.
1771. medias191<-cruzadaSVMbinPoly(data=tfmbis,
   vardep="sentido_fallo",
1772.                              listconti=c("PC7","PC15","PC27","PC
   54","PC25","PC52","PC22","PC42","PC43","PC4","PC111","PC13","PC175"
   ,"PC126","PC154","PC193","PC144","PC200","PC141","PC96","PC59","PC1
   25","PC199","PC95","PC176","PC198","PC71","PC74","PC78"),
1773.                              listclass=c("tribunal.1", "tribunal
   .2", "tribunal.3"),
1774.                              grupos=4,sinicio=1234,repe=5,
1775.                              C=0.01)
1776.
1777. medias191$modelo="SVM_poly1"
1778. medias191$tasa
1779. medias191$auc
1780.
1781.
1782. medias192<-cruzadaSVMbinPoly(data=tfmbis,
   vardep="sentido_fallo",
1783.                              listconti=c("PC7","PC15","PC27","PC
   54","PC25","PC44","PC35","PC3","PC2","PC52","PC61","PC22","PC9","PC
   55","PC42","PC36","PC43","PC64","PC40","PC4","PC32","PC105","PC69",
   "PC140","PC117","PC142","PC26","PC76","PC131","PC193","PC24","PC115
   ","PC150","PC56","PC134","PC125","PC112","PC63","PC46","PC176","PC1
   6","PC104","PC20","PC14","PC166","PC30","PC71"),
1784.                              listclass=c(""),
1785.                              grupos=4,sinicio=1234,repe=5,
1786.                              C=0.05)
1787.
1788. medias192$modelo="SVM_poly2"
```

```
medias192$tasa
medias192$auc

medias193<-cruzadaSVMbinPoly(data=tfmbis,
  vardep="sentido_fallo",
                             listconti=c("PC7","PC15","PC27","PC
  54","PC44","PC35","PC52","PC43","PC42","PC17","PC62","PC12","PC4","
  PC1","PC168","PC45","PC40","PC5","PC13","PC105","PC63","PC188","PC1
  87"),
                             listclass=c("tribunal.1", "tribunal
  .2", "tribunal.3"),
                             grupos=4,sinicio=1234,repe=5,
                             C=0.05)

medias193$modelo="SVM_poly3"
medias193$tasa
medias193$auc

medias194<-cruzadaSVMbinPoly(data=tfmbis,
  vardep="sentido_fallo",
                             listconti=c("PC1", "PC2", "PC3", "P
  C4", "PC5", "PC6", "PC7", "PC8", "PC9",
                                         "PC10", "PC11", "PC12",
   "PC13", "PC14", "PC15", "PC16", "PC17",
                                         "PC18", "PC19", "PC20",
   "PC21", "PC22", "PC23", "PC24", "PC25",
                                         "PC26", "PC27", "PC28",
   "PC29", "PC30", "PC31", "PC32", "PC33",
                                         "PC34", "PC35", "PC36",
   "PC37", "PC38", "PC39", "PC40", "PC41",
                                         "PC42", "PC43", "PC44",
   "PC45", "PC46", "PC47", "PC48", "PC49",
                                         "PC50", "PC51", "PC52",
   "PC53", "PC54", "PC55", "PC56", "PC57",
                                         "PC58", "PC59", "PC60",
   "PC61", "PC62", "PC63", "PC64", "PC65",
                                         "PC66", "PC67", "PC68",
   "PC69", "PC70", "PC71", "PC72", "PC73",
                                         "PC74", "PC75", "PC76",
   "PC77", "PC78", "PC79", "PC80", "PC81",
                                         "PC82", "PC83", "PC84",
   "PC85", "PC86", "PC87", "PC88", "PC89",
                                         "PC90", "PC91", "PC92",
   "PC93", "PC94", "PC95", "PC96", "PC97",
                                         "PC98", "PC99", "PC100"
  , "PC101", "PC102", "PC103", "PC104",
                                         "PC105", "PC106", "PC10
  7", "PC108", "PC109", "PC110", "PC111",
                                         "PC112", "PC113", "PC11
  4", "PC115", "PC116", "PC117", "PC118",
                                         "PC119", "PC120", "PC12
  1", "PC122", "PC123", "PC124", "PC125",
                                         "PC126", "PC127", "PC12
  8", "PC129", "PC130", "PC131", "PC132",
                                         "PC133", "PC134", "PC13
  5", "PC136", "PC137", "PC138", "PC139",
                                         "PC140", "PC141", "PC14
  2", "PC143", "PC144", "PC145", "PC146",
                                         "PC147", "PC148", "PC14
  9", "PC150", "PC151", "PC152", "PC153",
```

```
                                                "PC154", "PC155", "PC15
   6", "PC157", "PC158", "PC159", "PC160",
                                                "PC161", "PC162", "PC16
   3", "PC164", "PC165", "PC166", "PC167",
                                                "PC168", "PC169", "PC17
   0", "PC171", "PC172", "PC173", "PC174",
                                                "PC175", "PC176", "PC17
   7", "PC178", "PC179", "PC180", "PC181",
                                                "PC182", "PC183", "PC18
   4", "PC185", "PC186", "PC187", "PC188",
                                                "PC189", "PC190", "PC19
   1", "PC192", "PC193", "PC194", "PC195",
                                                "PC196", "PC197", "PC19
   8", "PC199", "PC200"),
                                 listclass=c("genero.0", "genero.1",
    "tribunal.1", "tribunal.2", "tribunal.3",
                                                "fecha_reforma.1", "fec
   ha_reforma.2", "fecha_reforma.3", "fecha_reforma.4"),
                                 grupos=4,sinicio=1234,repe=5,
                                 C=0.05)

medias194$modelo="SVM_poly4"
medias194$tasa
medias194$auc

union190<-rbind(medias191,medias192,medias193,medias194)

par(cex.axis=1)
boxplot(data=union190,tasa~modelo,main="TASA FALLOS SVM
   POLY",col="pink")
boxplot(data=union190,auc~modelo,main="AUC SVM
   POLY",col="lightgreen")


#RBF SVM
SVMgrid<-expand.grid(C=c(0.01,0.05,0.1,0.2,0.5,1,2,5,10),
                     sigma=c(0.1,0.5,1,2,5))

control<-trainControl(method = "cv",
                      number=4,savePredictions = "all")

SVM_RBF00<- train(data=tfmbis,factor(sentido_fallo)~tribunal.1 +
   tribunal.2 +tribunal.3 + PC7 + PC15 +   PC27 + PC54     + PC25  + P
   C52 + PC22 + PC42 + PC43 + PC4 + PC111 + PC13 + PC175 + PC126 + PC1
   54 + PC193 + PC144+ PC200+  PC141+ PC96+ PC59+ PC125+ PC199+ PC95+
   PC176+ PC198+ PC71+ PC74+ PC78,
                  method="svmRadial",trControl=control,
                  tuneGrid=SVMgrid,verbose=FALSE)
SVM_RBF00
SVM_RBF00$results
plot(SVM_RBF00)


SVM_RBF002<- train(data=tfmbis,factor(sentido_fallo)~PC7+
   PC15+   PC27+   PC54+   PC25+   PC44+   PC35+   PC3+    PC2+    PC5
   2+   PC61+   PC22+   PC9+    PC55+   PC42+   PC36+   PC166+  PC43+
     PC64+   PC40+   PC4+    PC32+   PC105+  PC69+   PC140+  PC117+  P
   C142+  PC26+   PC76+   PC131+  PC193+  PC24+   PC115+  PC150+  PC56
   +   PC134+  PC125+  PC112+  PC63+   PC46+   PC176+  PC16+   PC104+
   PC20+   PC14+   PC166+  PC30+   PC71,
                  method="svmRadial",trControl=control,
```

```
1863.                   tuneGrid=SVMgrid,verbose=FALSE)
1864.
1865. SVM_RBF002
1866.
1867. SVM_RBF002$results
1868. plot(SVM_RBF002)
1869.
1870.
1871. SVM_RBF003<- train(data=tfmbis,factor(sentido_fallo)~tribunal.1
   +tribunal.2 +tribunal.3 +PC7+   PC15+   PC27+   PC54+   PC35+   PC4
   4+   PC35+   PC52+   PC43+   PC42+   PC17+   PC62+   PC12+   PC4+
     PC1+    PC168+  PC45+   PC40+   PC5+    PC13+   PC105+  PC63+   P
   C188+  PC187,
1872.                   method="svmRadial",trControl=control,
1873.                   tuneGrid=SVMgrid,verbose=FALSE)
1874.
1875. SVM_RBF003
1876. plot(SVM_RBF003)
1877.
1878. SVM_RBF003$results
1879.
1880.
1881. SVM_RBF004<- train(data=tfmbis,factor(sentido_fallo)~.,method="s
   vmRadial",trControl=control,
1882.                   tuneGrid=SVMgrid,verbose=FALSE)
1883.
1884. SVM_RBF004
1885. plot(SVM_RBF004)
1886.
1887. SVM_RBF004$results
1888.
1889. #validación cruzada RBF
1890.
1891. cruzadaSVMbinRBF<-
1892.   function(data=data,vardep="vardep",
1893.            listconti="listconti",listclass="listclass",
1894.            grupos=4,sinicio=1234,repe=5,
1895.            C=1,sigma=1)
1896.   {
1897.
1898.     # PreparaciÃ³n del archivo
1899.
1900.     # b)pasar las categÃ³ricas a dummies
1901.
1902.     if  (listclass!=c(""))
1903.     {
1904.       databis<-data[,c(vardep,listconti,listclass)]
1905.       databis<- dummy.data.frame(databis, listclass, sep = ".")
1906.     }  else   {
1907.       databis<-data[,c(vardep,listconti)]
1908.     }
1909.
1910.     # c)estandarizar las variables continuas
1911.
1912.     # Calculo medias y dtipica de datos y estandarizo (solo las
   continuas)
1913.
1914.     means <-apply(databis[,listconti],2,mean)
1915.     sds<-sapply(databis[,listconti],sd)
1916.
1917.     # Estandarizo solo las continuas y uno con las categoricas
```

```
1918.
1919.      datacon<-scale(databis[,listconti],
   center = means, scale = sds)
1920.      numerocont<-which(colnames(databis)%in%listconti)
1921.      databis<-cbind(datacon,databis[,-numerocont,drop=FALSE ])
1922.
1923.      databis[,vardep]<-as.factor(databis[,vardep])
1924.
1925.      formu<-formula(paste("factor(",vardep,")~.",sep=""))
1926.
1927.      # Preparo caret
1928.
1929.      set.seed(sinicio)
1930.      control<-
   trainControl(method = "repeatedcv",number=grupos,repeats=repe,
1931.                              savePredictions = "all",classProbs=TRU
   E)
1932.
1933.      # Aplico caret y construyo modelo
1934.
1935.      SVMgrid <-expand.grid(C=C,sigma=sigma)
1936.
1937.      SVM<- train(formu,data=databis,
1938.                  method="svmRadial",trControl=control,
1939.                  tuneGrid=SVMgrid,replace=replace)
1940.
1941.      print(SVM$results)
1942.
1943.      preditest<-SVM$pred
1944.
1945.      preditest$prueba<-strsplit(preditest$Resample,"[.]")
1946.      preditest$Fold <- sapply(preditest$prueba, "[", 1)
1947.      preditest$Rep <- sapply(preditest$prueba, "[", 2)
1948.      preditest$prueba<-NULL
1949.
1950.      tasafallos<-function(x,y) {
1951.        confu<-confusionMatrix(x,y)
1952.        tasa<-confu[[3]][1]
1953.        return(tasa)
1954.      }
1955.
1956.      # Aplicamos funciÃ³n sobre cada RepeticiÃ³n
1957.
1958.      medias<-preditest %>%
1959.        group_by(Rep) %>%
1960.        summarize(tasa=1-tasafallos(pred,obs))
1961.
1962.      # CalculamoS AUC  por cada RepeticiÃ³n de cv
1963.      # Definimnos funciÃ³n
1964.
1965.      auc<-function(x,y) {
1966.        curvaroc<-roc(response=x,predictor=y)
1967.        auc<-curvaroc$auc
1968.        return(auc)
1969.      }
1970.
1971.      # Aplicamos funciÃ³n sobre cada RepeticiÃ³n
1972.
1973.      mediasbis<-preditest %>%
1974.        group_by(Rep) %>%
1975.        summarize(auc=auc(obs,Yes))
```

```
1976.
1977.     # Unimos la info de auc y de tasafallos
1978.
1979.     medias$auc<-mediasbis$auc
1980.
1981.     return(medias)
1982.
1983.   }
1984.
1985.
1986.
1987.
1988. medias1191<-cruzadaSVMbinRBF(data=tfmbis,
   vardep="sentido_fallo",
1989.                              listconti=c("PC7","PC15","PC27","PC
   54","PC25","PC52","PC22","PC42","PC43","PC4","PC111","PC13","PC175"
   ,"PC126","PC154","PC193","PC144","PC200","PC141","PC96","PC59","PC1
   25","PC199","PC95","PC176","PC198","PC71","PC74","PC78"),
1990.                              listclass=c("tribunal.1", "tribunal
   .2", "tribunal.3"),
1991.                              grupos=4,sinicio=1234,repe=5,
1992.                              C=0.01)
1993.
1994. medias1191$modelo="SVMRBF1"
1995.
1996. medias01191<-cruzadaSVMbinRBF(data=tfmbis,
   vardep="sentido_fallo",
1997.                              listconti=c("PC7","PC15","PC27","PC
   54","PC25","PC52","PC22","PC42","PC43","PC4","PC111","PC13","PC175"
   ,"PC126","PC154","PC193","PC144","PC200","PC141","PC96","PC59","PC1
   25","PC199","PC95","PC176","PC198","PC71","PC74","PC78"),
1998.                              listclass=c("tribunal.1", "tribunal
   .2", "tribunal.3"),
1999.                              grupos=4,sinicio=1234,repe=5,
2000.                              C=5, sigma=0.1)
2001.
2002. medias01191$modelo="SVMRBF01"
2003. medias01191$tasa
2004. medias01191$auc
2005.
2006.
2007.
2008. medias1192<-cruzadaSVMbinRBF(data=tfmbis,
   vardep="sentido_fallo",
2009.                              listconti=c("PC7","PC15","PC27","PC
   54","PC25","PC44","PC35","PC3","PC2","PC52","PC61","PC22","PC9","PC
   55","PC42","PC36","PC43","PC64","PC40","PC4","PC32","PC105","PC69",
   "PC140","PC117","PC142","PC26","PC76","PC131","PC193","PC24","PC115
   ","PC150","PC56","PC134","PC125","PC112","PC63","PC46","PC176","PC1
   6","PC104","PC20","PC14","PC166","PC30","PC71"),
2010.                              listclass=c(""),
2011.                              grupos=4,sinicio=1234,repe=5,
2012.                              C=0.05)
2013.
2014. medias1192$modelo="SVMRBF2"
2015.
2016. medias01192<-cruzadaSVMbinRBF(data=tfmbis,
   vardep="sentido_fallo",
2017.                              listconti=c("PC7","PC15","PC27","PC
   54","PC25","PC44","PC35","PC3","PC2","PC52","PC61","PC22","PC9","PC
   55","PC42","PC36","PC43","PC64","PC40","PC4","PC32","PC105","PC69",
```

```
   "PC140","PC117","PC142","PC26","PC76","PC131","PC193","PC24","PC115
   ","PC150","PC56","PC134","PC125","PC112","PC63","PC46","PC176","PC1
   6","PC104","PC20","PC14","PC166","PC30","PC71"),
                                listclass=c(""),
                                grupos=4,sinicio=1234,repe=5,
                                C=2, sigma = 0.1)

medias01192$modelo="SVMRBF02"
medias01192$tasa
medias01192$auc

medias1193<-cruzadaSVMbinRBF(data=tfmbis,
   vardep="sentido_fallo",
                                listconti=c("PC7","PC15","PC27","PC
   54","PC44","PC35","PC52","PC43","PC42","PC17","PC62","PC12","PC4","
   PC1","PC168","PC45","PC40","PC5","PC13","PC105","PC63","PC188","PC1
   87"),
                                listclass=c("tribunal.1", "tribunal
   .2", "tribunal.3"),
                                grupos=4,sinicio=1234,repe=5,
                                C=0.05)

medias1193$modelo="SVMRBF3"

medias01193<-cruzadaSVMbinRBF(data=tfmbis,
   vardep="sentido_fallo",
                                listconti=c("PC7","PC15","PC27","PC
   54","PC44","PC35","PC52","PC43","PC42","PC17","PC62","PC12","PC4","
   PC1","PC168","PC45","PC40","PC5","PC13","PC105","PC63","PC188","PC1
   87"),
                                listclass=c("tribunal.1", "tribunal
   .2", "tribunal.3"),
                                grupos=4,sinicio=1234,repe=5,
                                C=10, sigma = 0.1)

medias01193$modelo="SVMRBF03"
medias01193$tasa
medias01193$auc


medias1194<-cruzadaSVMbinRBF(data=tfmbis,
   vardep="sentido_fallo",
                                listconti=c("PC1", "PC2", "PC3", "P
   C4", "PC5", "PC6", "PC7", "PC8", "PC9",
                                            "PC10", "PC11", "PC12",
    "PC13", "PC14", "PC15", "PC16", "PC17",
                                            "PC18", "PC19", "PC20",
    "PC21", "PC22", "PC23", "PC24", "PC25",
                                            "PC26", "PC27", "PC28",
    "PC29", "PC30", "PC31", "PC32", "PC33",
                                            "PC34", "PC35", "PC36",
    "PC37", "PC38", "PC39", "PC40", "PC41",
                                            "PC42", "PC43", "PC44",
    "PC45", "PC46", "PC47", "PC48", "PC49",
                                            "PC50", "PC51", "PC52",
    "PC53", "PC54", "PC55", "PC56", "PC57",
                                            "PC58", "PC59", "PC60",
    "PC61", "PC62", "PC63", "PC64", "PC65",
                                            "PC66", "PC67", "PC68",
    "PC69", "PC70", "PC71", "PC72", "PC73",
```

```
                                                "PC74", "PC75", "PC76",
    "PC77", "PC78", "PC79", "PC80", "PC81",
                                                "PC82", "PC83", "PC84",
    "PC85", "PC86", "PC87", "PC88", "PC89",
                                                "PC90", "PC91", "PC92",
    "PC93", "PC94", "PC95", "PC96", "PC97",
                                                "PC98", "PC99", "PC100"
   , "PC101", "PC102", "PC103", "PC104",
                                                "PC105", "PC106", "PC10
   7", "PC108", "PC109", "PC110", "PC111",
                                                "PC112", "PC113", "PC11
   4", "PC115", "PC116", "PC117", "PC118",
                                                "PC119", "PC120", "PC12
   1", "PC122", "PC123", "PC124", "PC125",
                                                "PC126", "PC127", "PC12
   8", "PC129", "PC130", "PC131", "PC132",
                                                "PC133", "PC134", "PC13
   5", "PC136", "PC137", "PC138", "PC139",
                                                "PC140", "PC141", "PC14
   2", "PC143", "PC144", "PC145", "PC146",
                                                "PC147", "PC148", "PC14
   9", "PC150", "PC151", "PC152", "PC153",
                                                "PC154", "PC155", "PC15
   6", "PC157", "PC158", "PC159", "PC160",
                                                "PC161", "PC162", "PC16
   3", "PC164", "PC165", "PC166", "PC167",
                                                "PC168", "PC169", "PC17
   0", "PC171", "PC172", "PC173", "PC174",
                                                "PC175", "PC176", "PC17
   7", "PC178", "PC179", "PC180", "PC181",
                                                "PC182", "PC183", "PC18
   4", "PC185", "PC186", "PC187", "PC188",
                                                "PC189", "PC190", "PC19
   1", "PC192", "PC193", "PC194", "PC195",
                                                "PC196", "PC197", "PC19
   8", "PC199", "PC200"),
                                listclass=c("genero.0", "genero.1",
    "tribunal.1", "tribunal.2", "tribunal.3",
                                                "fecha_reforma.1", "fec
   ha_reforma.2", "fecha_reforma.3", "fecha_reforma.4"),
                                grupos=4,sinicio=1234,repe=5,
                                C=0.05)

medias1194$modelo="SVMRBF4"


medias01194<-cruzadaSVMbinRBF(data=tfmbis,
   vardep="sentido_fallo",
                                listconti=c("PC1", "PC2", "PC3", "P
   C4", "PC5", "PC6", "PC7", "PC8", "PC9",
                                                "PC10", "PC11", "PC12",
    "PC13", "PC14", "PC15", "PC16", "PC17",
                                                "PC18", "PC19", "PC20",
    "PC21", "PC22", "PC23", "PC24", "PC25",
                                                "PC26", "PC27", "PC28",
    "PC29", "PC30", "PC31", "PC32", "PC33",
                                                "PC34", "PC35", "PC36",
    "PC37", "PC38", "PC39", "PC40", "PC41",
                                                "PC42", "PC43", "PC44",
    "PC45", "PC46", "PC47", "PC48", "PC49",
```

```
                                                "PC50", "PC51", "PC52",
    "PC53", "PC54", "PC55", "PC56", "PC57",
                                                "PC58", "PC59", "PC60",
    "PC61", "PC62", "PC63", "PC64", "PC65",
                                                "PC66", "PC67", "PC68",
    "PC69", "PC70", "PC71", "PC72", "PC73",
                                                "PC74", "PC75", "PC76",
    "PC77", "PC78", "PC79", "PC80", "PC81",
                                                "PC82", "PC83", "PC84",
    "PC85", "PC86", "PC87", "PC88", "PC89",
                                                "PC90", "PC91", "PC92",
    "PC93", "PC94", "PC95", "PC96", "PC97",
                                                "PC98", "PC99", "PC100"
   , "PC101", "PC102", "PC103", "PC104",
                                                "PC105", "PC106", "PC10
   7", "PC108", "PC109", "PC110", "PC111",
                                                "PC112", "PC113", "PC11
   4", "PC115", "PC116", "PC117", "PC118",
                                                "PC119", "PC120", "PC12
   1", "PC122", "PC123", "PC124", "PC125",
                                                "PC126", "PC127", "PC12
   8", "PC129", "PC130", "PC131", "PC132",
                                                "PC133", "PC134", "PC13
   5", "PC136", "PC137", "PC138", "PC139",
                                                "PC140", "PC141", "PC14
   2", "PC143", "PC144", "PC145", "PC146",
                                                "PC147", "PC148", "PC14
   9", "PC150", "PC151", "PC152", "PC153",
                                                "PC154", "PC155", "PC15
   6", "PC157", "PC158", "PC159", "PC160",
                                                "PC161", "PC162", "PC16
   3", "PC164", "PC165", "PC166", "PC167",
                                                "PC168", "PC169", "PC17
   0", "PC171", "PC172", "PC173", "PC174",
                                                "PC175", "PC176", "PC17
   7", "PC178", "PC179", "PC180", "PC181",
                                                "PC182", "PC183", "PC18
   4", "PC185", "PC186", "PC187", "PC188",
                                                "PC189", "PC190", "PC19
   1", "PC192", "PC193", "PC194", "PC195",
                                                "PC196", "PC197", "PC19
   8", "PC199", "PC200"),
                                  listclass=c("genero.0", "genero.1",
    "tribunal.1", "tribunal.2", "tribunal.3",
                                                "fecha_reforma.1", "fec
   ha_reforma.2", "fecha_reforma.3", "fecha_reforma.4"),
                                  grupos=4,sinicio=1234,repe=5,
                                  C=10, sigma = 0.1)

medias01194$modelo="SVMRBF04"
medias01194$tasa
medias01194$auc

union01190<-
   rbind(medias01191,medias01192,medias01193,medias01194)
par(cex.axis=1)
boxplot(data=union01190,tasa~modelo,main="TASA FALLOS SVM
   RBF",col="pink")
boxplot(data=union01190,auc~modelo,main="AUC SVM
   RBF",col="lightgreen")

```

```
union1190<-rbind(medias1191,medias1192,medias1193,medias1194)

par(cex.axis=1)
boxplot(data=union1190,tasa~modelo,main="TASA FALLOS SVM
   RBF",col="pink")
boxplot(data=union1190,auc~modelo,main="AUC SVM
   RBF",col="lightgreen")

union11190<-
   rbind(medias91,medias92,medias93,medias94,medias191,medias192,media
   s193,medias194,medias1191,medias1192,medias1193,medias1194)

par(cex.axis=0.7)
boxplot(data=union11190,tasa~modelo,main="TASA FALLOS todos
   SVM",col="pink")
boxplot(data=union11190,auc~modelo,main="AUC todos
   SVM",col="lightgreen")


union011190<-
   rbind(medias91,medias92,medias93,medias94,medias191,medias192,media
   s193,medias194,medias01191,medias01192,medias01193,medias01194)

par(cex.axis=1)
boxplot(data=union011190,tasa~modelo,main="TASA FALLOS todos
   SVM",col="pink")
boxplot(data=union011190,auc~modelo,main="AUC todos
   SVM",col="lightgreen")

#comparamos con los otros mejores modelos
union00010<-
   rbind(medias1,medias2,medias3,medias4,medias04,medias5,
   medias6,medias7,medias100,medias101,medias102,medias103,medias300,m
   edias301,medias302,medias303)
par(cex.axis=1.75)
boxplot(data=union00010,tasa~modelo,main="TASA FALLOS
   comparativa",col="lightblue")
boxplot(data=union00010,auc~modelo,main="AUC
   comparativa",col="lightgrey")

unionfinal<-
   rbind(medias2,medias6,medias102,medias301,medias01193)
boxplot(data=unionfinal,tasa~modelo,main="T.Fallos mejores
   modelos",col="orange")
boxplot(data=unionfinal,auc~modelo,main="AUC mejores
   modelos",col="lightgreen")
```

8.5 Código R Índice de Youden

```
library(sas7bdat)
library(nnet)
library(dummies)
library(MASS)
library(reshape)
library(caret)
library(readr)
library(dplyr)

   # Lectura y esquema de variables
```

```
11.   tfm<-read.csv("C:/Users/Mélani EV/Desktop/TFM
   Jurimetrics/BD_juridica_csv.csv")
12.   dput(names(tfm))
13.
14.   continuas<-
   c("PC1", "PC2", "PC3", "PC4", "PC5", "PC6", "PC7", "PC8", "PC9",
15.                "PC10", "PC11", "PC12", "PC13", "PC14", "PC15", "PC
   16", "PC17",
16.                "PC18", "PC19", "PC20", "PC21", "PC22", "PC23", "PC
   24", "PC25",
17.                "PC26", "PC27", "PC28", "PC29", "PC30", "PC31", "PC
   32", "PC33",
18.                "PC34", "PC35", "PC36", "PC37", "PC38", "PC39", "PC
   40", "PC41",
19.                "PC42", "PC43", "PC44", "PC45", "PC46", "PC47", "PC
   48", "PC49",
20.                "PC50", "PC51", "PC52", "PC53", "PC54", "PC55", "PC
   56", "PC57",
21.                "PC58", "PC59", "PC60", "PC61", "PC62", "PC63", "PC
   64", "PC65",
22.                "PC66", "PC67", "PC68", "PC69", "PC70", "PC71", "PC
   72", "PC73",
23.                "PC74", "PC75", "PC76", "PC77", "PC78", "PC79", "PC
   80", "PC81",
24.                "PC82", "PC83", "PC84", "PC85", "PC86", "PC87", "PC
   88", "PC89",
25.                "PC90", "PC91", "PC92", "PC93", "PC94", "PC95", "PC
   96", "PC97",
26.                "PC98", "PC99", "PC100", "PC101", "PC102", "PC103",
    "PC104",
27.                "PC105", "PC106", "PC107", "PC108", "PC109", "PC110
   ", "PC111",
28.                "PC112", "PC113", "PC114", "PC115", "PC116", "PC117
   ", "PC118",
29.                "PC119", "PC120", "PC121", "PC122", "PC123", "PC124
   ", "PC125",
30.                "PC126", "PC127", "PC128", "PC129", "PC130", "PC131
   ", "PC132",
31.                "PC133", "PC134", "PC135", "PC136", "PC137", "PC138
   ", "PC139",
32.                "PC140", "PC141", "PC142", "PC143", "PC144", "PC145
   ", "PC146",
33.                "PC147", "PC148", "PC149", "PC150", "PC151", "PC152
   ", "PC153",
34.                "PC154", "PC155", "PC156", "PC157", "PC158", "PC159
   ", "PC160",
35.                "PC161", "PC162", "PC163", "PC164", "PC165", "PC166
   ", "PC167",
36.                "PC168", "PC169", "PC170", "PC171", "PC172", "PC173
   ", "PC174",
37.                "PC175", "PC176", "PC177", "PC178", "PC179", "PC180
   ", "PC181",
38.                "PC182", "PC183", "PC184", "PC185", "PC186", "PC187
   ", "PC188",
39.                "PC189", "PC190", "PC191", "PC192", "PC193", "PC194
   ", "PC195",
40.                "PC196", "PC197", "PC198", "PC199", "PC200")
41.
42.   options(max.print=999999)
43.
44.   tfm$tribunal<-factor(tfm$tribunal)
```

```
45.   tfm$fecha_reforma<-factor(tfm$fecha_reforma)
46.   tfm$sentido_fallo<-factor(tfm$sentido_fallo)
47.   tfm$genero<-factor(tfm$genero)
48.
49.   summary(tfm)
50.
51.   categoricas<-c("tribunal","fecha_reforma","genero")
52.
53.   # a)Eliminar las observaciones con missing en alguna variable
54.
55.   tfm2<-na.omit(tfm,(!is.na(tfm)))
56.
57.   # b)pasar las categÃ³ricas a dummies
58.
59.   tfm3<- dummy.data.frame(tfm2, categoricas, sep = "_")
60.
61.   # c)estandarizar las variables continuas
62.
63.   # Calculo medias y dtipica de datos y estandarizo (solo las
   continuas)
64.
65.   means <-apply(tfm3[,continuas],2,mean)
66.   sds<-sapply(tfm3[,continuas],sd)
67.
68.   tfmbis<-scale(tfm3[,continuas], center = means, scale = sds)
69.   numerocont<-which(colnames(tfm3)%in%continuas)
70.   tfmbis<-cbind(tfmbis,tfm3[,-numerocont])
71.
72.   tfmbis$sentido_fallo<-ifelse(tfmbis$sentido_fallo==1,"Yes","No")
73.
74.   dput(names(tfmbis))
75.
76.   #RANDOM FOREST CON VALIDACION CRUZADA
77.   cruzadarfbin<-
78.     function(data=data,vardep="vardep",
79.              listconti="listconti",listclass="listclass",
80.              grupos=4,sinicio=1234,repe=5,nodesize=20,
81.              mtry=2,ntree=50,replace=TRUE,sampsize=1)
82.     {
83.
84.       databis=data
85.       #
86.       # estandarizar las variables continuas
87.
88.       # Calculo medias y dtipica de datos y estandarizo (solo las
   continuas)
89.
90.       means <-apply(databis[,listconti],2,mean)
91.       sds<-sapply(databis[,listconti],sd)
92.
93.       # Estandarizo solo las continuas y uno con las categoricas
94.
95.       datacon<-scale(databis[,listconti],
   center = means, scale = sds)
96.       numerocont<-which(colnames(databis)%in%listconti)
97.       databis<-cbind(datacon,databis[,-numerocont,drop=FALSE ])
98.
99.       databis[,vardep]<-as.factor(databis[,vardep])
100.
101.      formu<-formula(paste("factor(",vardep,")~.",sep=""))
102.
```

```
103.      # Preparo caret
104.
105.      set.seed(sinicio)
106.      control<-
   trainControl(method = "repeatedcv",number=grupos,repeats=repe,
107.                           savePredictions = "all",classProbs=TRU
   E)
108.
109.      # Aplico caret y construyo modelo
110.
111.      rfgrid <-expand.grid(mtry=mtry)
112.
113.      if  (sampsize==1)
114.      {
115.        rf<- train(formu,data=databis,
116.                   method="rf",trControl=control,
117.                   tuneGrid=rfgrid,nodesize=nodesize,replace=repla
   ce,ntree=ntree)
118.      }
119.
120.      else  if  (sampsize!=1)
121.      {
122.        rf<- train(formu,data=databis,
123.                   method="rf",trControl=control,
124.                   tuneGrid=rfgrid,nodesize=nodesize,replace=repla
   ce,sampsize=sampsize,
125.                   ntree=ntree)
126.      }
127.
128.
129.      print(rf$results)
130.
131.      preditest<-rf$pred
132.
133.      preditest$prueba<-strsplit(preditest$Resample,"[.]")
134.      preditest$Fold <- sapply(preditest$prueba, "[", 1)
135.      preditest$Rep <- sapply(preditest$prueba, "[", 2)
136.      preditest$prueba<-NULL
137.
138.      tasafallos<-function(x,y) {
139.        confu<-confusionMatrix(x,y)
140.        tasa<-confu[[3]][1]
141.        return(tasa)
142.      }
143.
144.      # Aplicamos funciÃ³n sobre cada RepeticiÃ³n
145.
146.      medias<-preditest %>%
147.        group_by(Rep) %>%
148.        summarize(tasa=1-tasafallos(pred,obs))
149.
150.      return(rf)
151.
152.    }
153.
154.
155.  medias6<-cruzadarfbin(data=tfmbis,
156.                        vardep="sentido_fallo",listconti=c("PC7","
   PC15","PC27","PC54","PC44","PC35","PC52","PC43","PC42","PC17","PC62
   ","PC12","PC4","PC1","PC168","PC45","PC40","PC5","PC13","PC105","PC
   63","PC188","PC187"),
```

```
                          listclass=c("tribunal.1","tribunal.2","tribunal.3"),
                          grupos=4,sinicio=1234,repe=5,nodesize=10,
                          mtry=18,ntree=200,replace=TRUE, sampsize = 500)


  tfmbis$sentido_fallo<-ifelse(tfmbis$sentido_fallo=="Yes",1,0)

  #Busqueda del mejor punto de corte para el ganador
  sensEspCorte<-function(modelo,dd,nombreVar,ptoCorte,evento){
    probs <-predict(modelo,newdata=dd,type="prob")
    cm<-confusionMatrix(data=factor(ifelse(probs[,2]>ptoCorte,1,0)),reference=factor(dd[,nombreVar]),positive=evento)
    c(cm$overall[1],cm$byClass[1:2])
  }

  posiblesCortes<-seq(0,1,0.01)
  rejilla<-data.frame(t(rbind(posiblesCortes,sapply(posiblesCortes,function(x)
   sensEspCorte(medias6,tfmbis,"sentido_fallo",x,"1")))))
  rejilla$Youden<-rejilla$Sensitivity+rejilla$Specificity-1
  plot(rejilla$posiblesCortes,rejilla$Youden, main= "Corte que maximiza Indice de Youden", ylab="Indice de Youden", xlab="posibles Cortes")
  plot(rejilla$posiblesCortes,rejilla$Accuracy,main= "Corte que maximiza la exactitud del modelo", ylab="Precisión", xlab="posibles Cortes")
  rejilla$posiblesCortes[which.max(rejilla$Youden)]
  rejilla$posiblesCortes[which.max(rejilla$Accuracy)]
  sensEspCorte(medias6,tfmbis,"sentido_fallo",0.58,"1") #el punto optimo segun el indice de Youden
  sensEspCorte(medias6,tfmbis,"sentido_fallo",0.54,"1") #el punto optimo segun accuracy
```

8.6 Script R nubes de palabras

```
library("tm")
library("SnowballC")
library("wordcloud")
library("RColorBrewer")

#para elegirlo sobre la marcha en la ruta donde tengamos el txt:

text <- readLines(file.choose(),encoding="UTF-8")

  docs <- Corpus(VectorSource(text), readerControl = list(language = "sp")) #VectorSource() function creates a corpus of character vectors
  inspect(docs)

  toSpace <- content_transformer(function (x , pattern ) gsub(pattern, " ", x))
  docs <- tm_map(docs, toSpace, "/")
  docs <- tm_map(docs, toSpace, "@")
  docs <- tm_map(docs, toSpace, "\\|")
```

```
18.   # Convertir texto en minuscula
19.   docs <- tm_map(docs, content_transformer(tolower))
20.   # quitar numberos
21.   docs <- tm_map(docs, removeNumbers)
22.   # quitar  stopwords
23.   docs <- tm_map(docs, removeWords, stopwords("spanish"))
24.   # Eliminar algunas palabras a modo prueba
25.   docs <- tm_map(docs,
   removeWords, c"condeno", "absolver", "absuelvo", "autor", "responsa
   ble"))
26.   # eliminar signos de puntuación
27.   docs <- tm_map(docs, removePunctuation)
28.   # Eliminar espacios en blanco extra
29.   docs <- tm_map(docs, stripWhitespace)
30.
31.   #funcion TermDocumentMatrix():
32.
33.   dtm <- TermDocumentMatrix(docs)
34.   m <- as.matrix(dtm)
35.   v <- sort(rowSums(m),decreasing=TRUE)
36.   d <- data.frame(word = names(v),freq=v)
37.   head(d, 10)
38.
39.   #generar nube de palabras
40.   set.seed(1234)
41.   wordcloud(words = d$word, freq = d$freq, min.freq = 1,
42.             max.words=200, random.order=FALSE, rot.per=0.35,
43.             colors=brewer.pal(8, "Dark2"))
44.
45.
46.   #Explorar los términos frecuentes y asociaciones según palabras
47.   findFreqTerms(dtm, lowfreq = 4)
48.
49.   findAssocs(dtm, terms = "imprudencia", corlimit = 0.3)
50.
51.
52.   #Graficarlo :
53.   barplot(d[1:10,]$freq, las = 2, names.arg = d[1:10,]$word,
54.           col ="lightblue", main ="Palabras mas frecuentes",
55.           ylab = "Frecuencia de palabras")
```