FACULTAD DE ESTUDIOS ESTADÍSTICOS

MÁSTER EN MINERÍA DE DATOS E INTELIGENCIA DE NEGOCIOS

Curso 2019/2020

Trabajo de Fin de Máster

TITULO: Estrategias de mercado en Ensayos Clínicos

***Alumna*: Elianni María Agüero Selva**

***Tutor*: David Lora Pablos**

Septiembre de 2020

UNIVERSIDAD COMPLUTENSE
MADRID

Estrategias de mercado en Ensayos Clínicos

Trabajo de Fin de Máster en Minería de Datos e Inteligencia de Negocios

Autora
Elianni María Agüero Selva

Director
David Lora Pablos

Máster en Minería de Datos e Inteligencia de Negocios
Facultad de Estudios Estadísticos
Universidad Complutense de Madrid

Agradecimientos

No puedo empezar a escribir este trabajo sin antes mostrar mi agradecimiento a mi tutor David por toda la ayuda que me ha ofrecido durante el proyecto. Gracias por darme la oportunidad de participar en este proyecto.

A mis padres y a mis abuelos por todo el apoyo y la confianza que me han dado en todo momento, sin ellos no hubiese sido posible superar esta etapa. A mis amigos de la carrera y a las *compis* del máster por ayudarme siempre que lo he necesitado y por todos los buenos momentos.

Por último, y especialmente, a Dani. Sin tu ayuda no hubiese sido capaz de avanzar tanto. Gracias por toda esa paciencia infinita que tienes conmigo, por tus ánimos, por las risas, por todos los buenos momentos que hemos pasado juntos y los que nos quedan.

Abstract

A clinical trial is an experimental study conducted to evaluate the efficacy or safety of various treatments. Usually these trials are carried out by sectors of the pharmaceutical industry and involve a long, risky and very expensive process.

Currently, there is a great deal of information about the characteristics of the trials that are carried out worldwide. Some of them, sometimes, are interrupted without the possibility of continuing with it, which supposes a great economic loss.

The aim of this work is to carry out a data mining project in which we use techniques of *machine learning* to detect and analyze what are the characteristics that should be given in a clinical trial so that it can be completed without interruption.

Keywords

clinical trials, data mining, machine learning, prediction models, cross validation, failure rate.

Resumen

Un ensayo clínico es un estudio experimental que se realiza para evaluar la eficacia o seguridad de diversos tratamientos. Por lo general estos ensayos son llevados a cabo por sectores de la industria farmacéutica y suponen un proceso largo, arriesgado y muy costoso.

Actualmente, existe una gran cantidad de información acerca de las características de los ensayos que se realizan a nivel mundial. Algunos de ellos, en ocasiones, son interrumpidos sin posibilidad de continuar con el mismo, lo que supone una gran pérdida económica.

El objetivo del presente trabajo es realizar un proyecto de minería de datos en el que utilicemos técnicas de *machine learning* para detectar y analizar cuáles son las características que se deben dar en un ensayo clínico para que este pueda finalizar sin interrupción.

Palabras clave

ensayos clínicos, minería de datos, *machine learning*, modelos de predicción, validación cruzada, tasa de fallos.

Índice

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

"Investigar es ver lo que todo el mundo ha visto, y pensar lo que nadie más ha pensado"
— Albert Szent-Györgyi

1.1. Motivación

Hoy en día, la cantidad de datos que se está generando constantemente en cualquier sector es enorme. No se queda atrás el sector farmacéutico que posee un gran avance tecnológico abarcando campos como la biología, bioquímica, farmacología, ingeniería, estadística, entre otros. Gracias a este avance tecnológico, las empresas farmacéuticas están teniendo grandes progresos en el desarrollo e investigación de sus productos y servicios. Además, se desarrollan actividades de *marketing*, relaciones públicas, administración, producción y control de calidad.

La minería de datos o *data mining* es una disciplina que consiste en descubrir patrones ocultos y/o relaciones complejas o desconocidas que expliquen el comportamiento de conjuntos de datos. Esta disciplina ayuda a las empresas farmacéuticas a descubrir pautas para mejorar la calidad de los métodos de investigación y administración de medicamentos, entre otras. Un ejemplo de estas investigaciones que llevan a cabo son los ensayos clínicos. Un ensayo clínico es un experimento controlado en voluntarios que se utiliza para evaluar la eficacia y seguridad de nuevos tratamientos. Las compañías farmacéuticas promueven ensayos clínicos para evaluar la calidad de estos tratamientos. Esto implica un proceso largo, arriesgado y muy costoso. Durante la realización de un ensayo se pueden producir desviaciones del protocolo como la interrupción del tratamiento, el abandono por falta de eficacia o efectos adversos, entre otras causas. Esto genera una gran pérdida económica para estas compañías y es por tanto un riesgo que deben tener en cuenta.

Con el presente trabajo nos proponemos aplicar la minería de datos a los ensayos clínicos, con el objetivo de minimizar el riesgo y por tanto las posibles pérdidas de los ensayos fallidos. Buscaremos las características de los

mismos que más influyen en que un estudio no se vea interrumpido y llegue a terminarse de forma satisfactoria.

Con este trabajo esperamos ayudar en los costes de investigación y desarrollo de la industria farmacéutica. El objetivo sería lograr minimizar en la medida de lo posible los ensayos que acaban siendo fallidos, y que por tanto suponen un enorme coste sin ningún beneficio. De esta forma, las empresas no se verían en la necesidad de compensar dichos costes con el precio de otros tratamientos fructíferos. Esto devendría en un abaratamiento de los demás tratamientos sin perjuicio de los ingresos de la empresa, lo cual beneficiaría en último lugar al paciente.

1.2. Objetivos

Como hemos comentado anteriormente, nos hemos propuesto como objetivo determinar y analizar cuáles son las características de partida que debe tener un estudio para asegurar que pueda finalizarse correctamente. Para ello seguiremos los siguientes pasos:

- Extracción y selección de variables de *Database for Aggregated Analysis of* `ClinicalTrials.gov` (AACT).
- Comprensión de los datos obtenidos mediante la exploración y modificación de los mismos.
- Creación de diversos modelos de predicción utilizando técnicas estadísticas y basadas en *machine learning*.
- Comparación de resultados y selección del mejor modelo.
- Análisis de resultados y obtención de conclusiones.

Capítulo 2

Estado del Arte

"Siempre que te pregunten si puedes hacer un trabajo,
contesta que sí y ponte enseguida
a aprender cómo se hace"
— Franklin D. Roosevelt

En este capítulo daremos una visión sobre algunos de los temas que trataremos y desarrollaremos en el presente trabajo.

2.1. Ensayos clínicos

Según Pallás y Villa (2019), cuando hablamos de *investigación clínica*, nos referimos a la base de todos los avances médicos. A través de este tipo de investigación se intenta buscar la manera de descubrir nuevas formas de diagnosticar, prevenir, tratar y entender las enfermedades que afectan a todos los seres humanos.

Un *ensayo clínico* es un estudio experimental llevado a cabo por equipos de investigación que se realiza en personas sanas y/o enfermas. Este estudio permite evaluar la eficacia o seguridad que tienen algunas intervenciones, o tratamientos. Todo estudio se realiza siguiendo un protocolo de investigación estrictamente controlado. Es decir, no se puede empezar el estudio sin antes haber sido aprobado por un comité ético, independientemente de los investigadores y los promotores del estudio. Además, los nuevos tratamientos sólo podrán lanzarse al mercado siempre y cuando se haya demostrado, previamente, que no suponen un riesgo para la población. Para ello, se realizan evaluaciones en animales donde comprueban la eficacia y seguridad de los tratamientos y sólo si los resultados de estas evaluaciones son favorables se comienzan los estudios en humanos. Gracias a los ensayos clínicos se avanza en la investigación de nuevos tratamientos y se conocen nuevos aspectos de las enfermedades.

Los ensayos clínicos se llevan a cabo siguiendo una serie de fases consecutivas. Si un tratamiento nuevo tiene éxito en una fase, se pasa a la siguiente.

- **Fase I** - Evaluación de la seguridad: se selecciona un grupo reducido de personas sanas y se les aplica el tratamiento. En esta fase se evalúa la seguridad del mismo y se determina el rango seguro de aplicación del mismo. Por ejemplo, en el caso de un fármaco, el rango de dosificación. El objetivo será entender cómo actúa el tratamiento en el cuerpo y cómo reacciona el cuerpo frente al mismo.
- **Fase II** - Evaluación de la eficacia: en esta fase ya se incluyen pacientes. El tratamiento se administra a un mayor número de personas que padecen alguna enfermedad. En esta segunda fase se evalúan los efectos adversos a corto plazo y los riesgos de seguridad asociados al tratamiento en investigación.
- **Fase III** - Confirmación de hallazgos en una población mayor de pacientes: en esta fase ya se cuenta con un grupo de personas que padecen alguna enfermedad bastante amplio (cientos o miles). El objetivo, como en las fases anteriores, es conocer la seguridad, los beneficios y la eficacia del tratamiento que se está investigando. Además, se comparará con otros tratamientos utilizados habitualmente. Si los resultados en esta fase son favorables, se solicita la autorización del tratamiento para uso clínico.
- **Fase IV** – Revisión del tratamiento en práctica clínica. La postcomercialización son estudios que se llevan a cabo cuando el tratamiento ya está comercializado y esto es lo que se realiza en la fase cuatro. El objetivo es recopilar información sobre los riesgos o beneficios obtenidos tras su utilización clínica diaria. Evalúa la seguridad y eficacia del fármaco a largo plazo.

El rol de investigador dentro del ensayo se encarga de analizar los datos y tomar las decisiones en función de los resultados obtenidos. Además, son ellos los que determinan si se debe pasar de una fase a otra.

Otras de las principales características (Sánchez-Caro y Abellán, 2006) que están presentes en un ensayo es la *aleatorización*. Esta característica consiste en que cada participante del ensayo será asignado al azar a los distintos grupos del mismo. En algunos casos dichos grupos reciben distintos fármacos o tratamientos o incluso placebo. Se considera la mejor forma de garantizar que los pacientes formen grupos similares. De esta forma se podrán hacer comparaciones no sesgadas entre los efectos de los tratamientos.

En términos de ensayos, cuando se habla de *enmascaramiento* o estudio ciego se refiere a ocultar el tratamiento que reciben los pacientes para evitar condicionar la evaluación de los resultados. Según el grado de enmascaramiento podemos diferenciar varios tipos:

- Abiertos o no enmascarados: tanto el paciente como el médico conocen el tratamiento.
- Ciego o simple ciego: el médico conoce el tratamiento, pero no el paciente.
- Doble ciego: tanto el paciente como el médico desconocen el tratamiento.

- Triple ciego: el paciente, el investigador o médico y la persona responsable de analizar los datos desconocen el tratamiento.

Una vez que los ensayos clínicos muestren que un nuevo tratamiento es seguro y eficaz, puede convertirse en una práctica estándar.

2.2. Inteligencia Artificial

La inteligencia artificial o IA es una rama de la informática que pretende responder a la pregunta *¿pueden las máquinas pensar?* del matemático Alan Turing (Warwick y Shah, 2016). Este concepto se refiere a la simulación de la inteligencia humana en máquinas programadas para pensar como los humanos e imitar sus acciones.

La principal característica de la inteligencia artificial es su capacidad de racionalizar y tomar decisiones para lograr un objetivo específico. En el libro "*Artificial Intelligence: A Modern Approach*" publicado por Russell y Norvig, se analizan cuatro enfoques distintos que han definido a la inteligencia artificial a lo largo de los años: pensar como un humano, pensamiento racional, comportamiento humano y actuar de forma racional.

Gracias al desarrollo tecnológico de la computación, a día de hoy es posible llevar a cabo sistemas cada vez más complejos de IA. Concretamente, los modelos más desarrollados son los conocidos como Aprendizaje Automático o *Machine Learning*.

2.2.1. Aprendizaje automático

Según Sancho Caparrini (2017) aprendizaje es el "*proceso a través del cual se adquieren o modifican habilidades, destrezas, conocimientos, conductas o valores como resultado del estudio, la experiencia, la instrucción, el razonamiento y la observación*".

Cuando hablamos de aprendizaje automático o *machine learning* (Alpaydin, 2020; Jake, 2020) nos referimos a una disciplina correspondiente a una de las ramas de la Inteligencia Artificial. Esta consiste en desarrollar técnicas que permitan a las máquinas aprender de manera automática. Es decir, se crean algoritmos capaces de generalizar comportamientos y reconocer patrones a partir de cierta información. Usando técnicas basadas en aprendizaje automático, se persigue intentar predecir comportamientos futuros a partir de la información obtenida del pasado.

Dependiendo del tipo de problema que se aborde, esta disciplina puede clasificarse en:

- Aprendizaje supervisado: al algoritmo se le proporcionan los datos de entrada y los resultados esperados correspondientes a cada dato. De esta forma es el propio algoritmo el que aprende de sus errores y se va corrigiendo para intentar aproximarse lo máximo posible al resultado.

- Aprendizaje no supervisado: sólo se le proporcionan al algoritmo los datos de entrada. En este caso, el algoritmo realizará un modelo de agrupaciones o *clusters* en el que busque conjuntos de datos similares entre sí. Un posible ejemplo sería dado datos clínicos de análisis de tumores, obtener una clasificación en benignos o malignos.

Dependiendo del tipo de objetivo que intentamos predecir, podemos clasificar los problemas de *machine learning* de la siguiente manera:

- Regresión: se intenta predecir un valor real, es decir una variable con valores infinitos. Por ejemplo, predecir cualquier fenómeno meteorológico como el porcentaje de precipitación o la temperatura.
- Clasificación: se intenta predecir o clasificar valores finitos. Por ejemplo, la clasificación de un tipo de vehículo (turismo, motocicleta, camión). Cuando la clasificación se realiza entre dos valores se denomina clasificación binaria.
- *Ranking*: se intenta establecer y predecir un orden óptimo dentro de un conjunto de valores. Por ejemplo, el orden de relevancia con el que buscador *Google* nos devuelve los resultados cuando realizamos una búsqueda.

Es importante tener en cuenta a qué tipo de problema de *machine learning* nos estamos enfrentando. Ya que dependiendo de cómo sea el problema tomaremos unas medidas u otras para medir el error cometido entre la predicción y el valor real.

Dadas las capacidades de procesamiento actuales, se pueden desarrollar modelos de predicción cada vez más elaborados. Para ello uno de los principales requisitos es poseer una gran cantidad de información. Esta información que no podría ser procesada mediante métodos manuales, es la que da lugar a la disciplina llamada *Minería de datos*.

2.3. Minería de datos

La minería de datos (Aluja, 2001) se conoce como el proceso automático que persigue descubrir patrones ocultos y/o relaciones complejas o desconocidas que expliquen el comportamiento de grandes conjuntos de datos. Es una disciplina que se basa en la combinación estadística, técnicas de inteligencia artificial, aprendizaje automático e incluso la gestión de bases de datos.

En la actualidad, los proyectos de minería de datos están siendo muy utilizados en los ámbitos empresariales como la comprensión del *marketing* de los consumidores, el análisis de productos, comercio electrónico, la tendencia de inversión, las investigaciones científicas o la seguridad, entre otras. Es decir, en todas aquellas áreas donde se generen una gran cantidad de datos. Una de las principales características de esta disciplina es que se hace más hincapié en el poder predictivo y en el resultado de los modelo que en la interpretación de los mismos.

En minería de datos se trabaja con datos abundantes y heterogéneos. Es decir, conjuntos de datos en los que tenemos un gran número de variables y observaciones. En cuanto a las variables (Calviño Martínez, 2019) podemos clasificarlas en dos tipos:

- Según su función:
  - Identificativas: identifican las observaciones. No son objetos de estudio
  - *Input*, de entrada o independientes: variables predictoras
  - Objetivo o dependiente: variable que se pretende predecir
  - Rechazadas, variables eliminadas antes de la fase de modelización
- Según su tipología:
  - Continuas, cuantitativas o de intervalo: toman valores infinitos.
  - Nominales, cualitativas o categóricas: toman un número finito de valores. A las variables que toman solo dos valores se les denomina binarias o dicotómicas.

2.3.1. Metodología SEMMA

Dentro de la minería de datos (Azevedo y Santos, 2008) existen distintas metodologías en las que nos basamos para desarrollar un proyecto. Metodologías como *KDD* (*Knowledge discovery in databases*) y *CRISP-DM* (*CRoss-Industry Standard Process for Data Mining*) tienen un gran uso en la actualidad (Hlaing y Thaw, 2019; Wirth y Hipp, 2000). No obstante, en el desarrollo del presente trabajo nos centraremos en la metodología *SEMMA* puesto que usaremos *software* proporcionado por el instituto SAS como *SAS Base*[1] y *SAS Enterprise Miner*[2].

El acrónimo SEMMA significa *Sample* (muestrear), *Explore* (explorar), *Modify* (modificar), *Model* (modelizar) y *Assess* (evaluar). El Instituto SAS[3] considera cinco fases para el proceso de minería de datos:

1. *Sample*: muestrear los datos. Es decir, si la base de datos de la que disponemos es demasiado grande, debemos tomar una muestra lo suficientemente grande para que contenga toda la información y lo suficientemente pequeña para poder ser procesada.

2. *Explore*: exploración de los datos para conocer la naturaleza de los mismos y así detectar relaciones, anomalías y tendencias.

3. *Modify*: modificación de los datos creando, seleccionando y transformando las variables para facilitar el proceso de modelización.

4. *Model*: creación de modelos que se ajusten a los datos para poder predecir la variable objetivo.

[1] Acceder a `https://www.sas.com/en_us/software/base-sas.html` para más información.
[2] Acceder a `https://www.sas.com/en_us/software/enterprise-miner.html` para más información.
[3] Para más información acceder a `https://www.sas.com/es_es/home.html`

5. *Assess*: comprobación de la calidad de las predicciones y comparación de los modelos obtenidos.

Es importante destacar que en los proyectos de minería de datos no siempre intervienen todas las fases de esta metodología y, además no siguen necesariamente un orden secuencial. Es decir, todas ellas pueden repetirse y modificar el orden en el que están descritas.

Capítulo 3

Técnicas empleada

"No inventé nada nuevo. Simplemente junté los descubrimientos de otros hombres que trabajaron durante siglos"
— Henry Ford

En este capítulo hablaremos sobre las distintas técnicas que se van a utilizar en el desarrollo del presente trabajo. Expondremos los motivos por los cuales hemos decidimos usarlas y hablaremos sobre su funcionamiento.

3.1. Regresión

Según Kuhn et al. (2013), los modelos de regresión son modelos matemáticos que nos permiten predecir una variable y o variable objetivo, a partir de un conjunto de variables de entrada ($x_1, x_2, \ldots, x_m$). Estos modelos de regresión intentan modelizar la relación entre las variables *inputs* y la variable objetivo a través de una determinada función $y = f(x_1, x_2, ..., x_m) + \epsilon$, donde ϵ representa el término de error.

En el caso de la regresión lineal suponemos que la relación entre las variables independientes y la variable dependiente es lineal. Por tanto, se sustituye la función genérica f por una recta: $y = \hat{y} + \epsilon$, donde $\hat{y}$ representa el valor predicho:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m,$$

esto se puede interpretar como, por cada incremento unitario que tienen las variables x_i, la variable $\hat{y}$ deberá aumentar β_i unidades. La estimación de los parámetros β_i se realizará buscando el valor que minimice el error cometido por el modelo. Este error viene dado por $y - \hat{y}$.

Por otro lado, si hablamos de regresión logística, nos ocupa el caso en el que la variable objetivo es dicotómica. Esto es, una variable de respuesta bina-

ria que indica la *ocurrencia* o *evento* de un determinado suceso. Por ejemplo, puede tomar los valores 0 (*no*) ó 1 (*si*) donde esta última represente la *categoría prioritaria* o *categoría de interés*. Sin embargo, no podemos asegurar obtener *0s* y *1s* utilizando la ecuación anterior, por lo que interpretaremos los resultados como la probabilidad de que la $\hat{y}$ tome el valor 1. Por otra parte, tampoco podemos asegurar que los resultados estén dentro del intervalo (0, 1) por lo que entran en juego funciones de enlace o *link*[1] (Calviño Martínez, 2019) como las funciones de distribución.

Por tanto, el modelo de regresión logística persigue intentar modelizar la probabilidad del suceso (1) o la categoría de interés. Para ello, se estima la probabilidad de que ocurra dicho suceso como:

$$p_1 = P(Y = 1|x_1, x_2, \ldots, x_m) = \frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_m x_m)}}$$

de forma análoga, tenemos p_0 como la probabilidad de que no ocurra:

$$p_0 = 1 - p_1 = P(Y = 0|x_1, x_2, \ldots, x_m) = \frac{e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_m x_m)}}{1 + e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_m x_m)}}$$

En la Figura 3.1 podemos ver un ejemplo gráfico de la diferencia de aplicar un modelo de regresión lineal y un modelo de regresión logística.

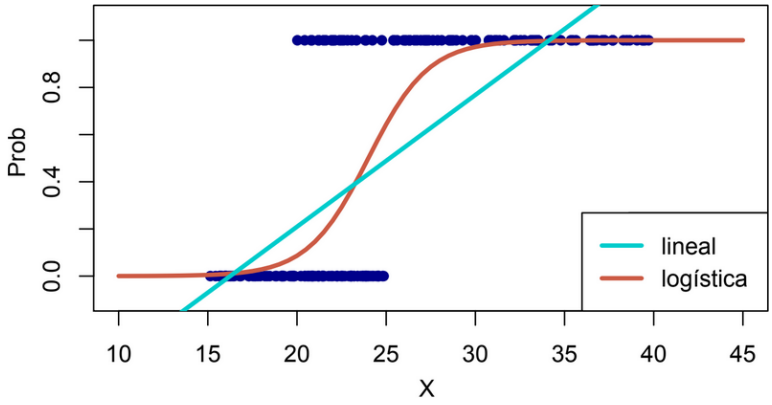


Figura 3.1: Esquema regresión. Fuente: Calviño Martínez (2019)

Destacamos que los modelos de regresión no solo funcionan como modelos predictivos, sino que a menudo se utilizan como métodos de selección de variables automáticos. Entre ellos podemos nombrar:

- *Forward* o hacia adelante: parte de un modelo en el que solo hay constantes, no se utiliza información de ninguna variable: $y = \beta_0$. Se van seleccionando las variables que produzcan una mejora en comparación al

[1] Función que permite relacionar las variables regresoras con las probabilidades. Asegura que dichas probabilidades se encuentran en el intervalo (0, 1).

modelo anterior. Las variables que han sido seleccionadas no se pueden eliminar.

- *Backward* o hacia atrás: Parte de un modelo que contiene todas las variables y se irá eliminando, de una en una, aquella que aporte menos hasta que todas las que queden sean significativas. Una vez que se elimine una variable del modelo, no se podrá seleccionar de nuevo.
- *Stepwise* o paso a paso: mezcla las técnicas de selección anteriores. Comienza el proceso de forma similar al método *forward*, con la diferencia de que las variables que se han seleccionado si pueden ser eliminadas. Esta eliminación se hace siguiendo el método *backward*.

3.2. Redes neuronales

Las redes neuronales artificiales (Jain et al., 1996) se inspiran en las redes neuronales biológicas. Este modelo, a diferencia de modelos clásicos como los de regresión, tienen la capacidad de adaptarse para encontrar cualquier relación entre las variables *input* y la objetivo, aunque sea desconocida.

En dichas redes se crean, internamente, capas ocultas o artificiales compuestas por nodos. El objetivo es construir una función generalista de la forma $y = f(x_1, x_2, \ldots, x_m)$ que relacione los nodos *input* con el *output*. Una red neuronal generalmente está formada por:

- Capa *input*: capa que recibe los datos de entrada. En el caso que nos ocupa, diremos que son las variables *inputs* del conjunto.
- Capa *output*: capa que devuelve los resultados obtenidos de la red neuronal. Esto es, variable objetivo o dependiente.
- Capas ocultas: capas intermedias e interconectadas que se encargan del procesamiento de la información. Puede contener uno o más nodos ocultos.
- *Weights*: pesos de cada entrada. Sirven para dar más importancia a una entrada que a otra.
- Función de activación: función limitadora o umbral. El objetivo es modificar el resultado o imponer un límite que no se debe sobrepasar antes de propagarse a otra neurona.

Podemos ver en la Figura 3.2 un esquema de una red neural artificial que contiene cuatro nodos *input* (X_1, X_2, X_3 y X_4), un nodo *output* (Y) y una capa oculta con cuatro nodos ocultos (H_1, H_2, H_3 y H_4).

La construcción matemática consiste en crear una función de combinación lineal $\sum$ donde intervienen las variables independientes (Portela García-Miguel, 2020). Estas variables se combinan multiplicando cada una de ellas por los parámetros y sumándolos. Realizado este proceso se activan, es decir, se les aplica una función f no lineal que hace la labor de complicar o hacer más flexible la función.

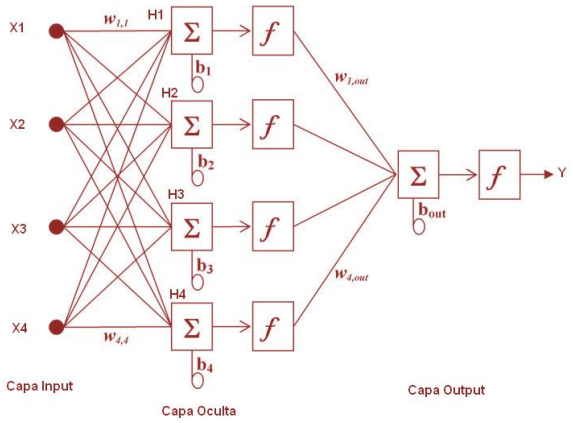


Figura 3.2: Esquema red neuronal. Fuente: (Portela García-Miguel, 2020)

Por tanto, basándonos en el esquema anterior, y suponiendo que apliquemos la función de activación tangente hiperbólica $tanh(x) = 1 - \frac{2}{1+e^{2x}}$, podemos definir la función:

$$\begin{aligned} y &= tanh(W_{i,out}(tanh(w_{11}X1 + w_{21}X2 + w_{31}X3 + w_{41}X4 + b_1)) \\ &+ (tanh(w_{12}X1 + w_{22}X2 + w_{32}X3 + w_{42}X4 + b_2)) \\ &+ (tanh(w_{13}X1 + w_{23}X2 + w_{33}X3 + w_{43}X4 + b_3)) \\ &+ (tanh(w_{14}X1 + w_{24}X2 + w_{34}X3 + w_{44}X4 + b_4)) \\ &+ b_{out}) \end{aligned}$$

El objetivo computacional concreto es estimar los valores de los parámetros w_{ij} (pesos) y b_j (*bias* o sesgo) del modelo. Para ello, se utilizan técnicas o algoritmos de optimización que van variando los valores de los parámetros de manera iterativa, hasta cumplir el objetivo.

3.3. Árboles de decisión

Según Aguirre (2019) los árboles de decisión son métodos de estimación no paramétricos. Se caracterizan por su capacidad de modelar datos heterogéneos (variables continuas y discretas) y por la buena tolerancia al ruido introducido por variables no relevantes. Además, no requieren asunciones teóricas de los datos. Desde un punto de vista matemático (Artalejo et al., 2011), se puede decir que los árboles son estructuras jerárquicas que se construyen de manera inductiva.

El objetivo es seguir una serie de reglas que se basan en el valor de las variables de entrada, con el fin de segmentar la población en subconjuntos más pequeños que tengan ciertas características en común. Es decir, separan lo máximo posible los datos con el objetivo de que todas aquellas observaciones que tengan el mismo valor de la variable dependiente estén en un subgrupo. Algunos términos y conceptos básicos de los árboles son:

- Nodo raíz: nodo superior del árbol.
- Nodo hijo: nodo conectado con otro cuando se aleja de la raíz.
- Nodo padre: lo contrario al nodo hijo.
- Nodo interno: nodo con al menos un hijo.
- Nodo hoja: un nodo sin hijos.
- Camino: una sucesión de nodos en la que cada nodo es padre del siguiente.
- Rama: cualquier camino que empieza en la raíz y acaba en una hoja.
- Nivel o profundidad: longitud del camino que va desde la raíz hasta al nodo. El nivel de la raíz es 1.
- Talla o altura del árbol: profundidad de la rama más larga del árbol.
- Grado o *aridad* de un nodo: número de hijos. La *aridad* de un árbol es el máximo de las *aridades* de todos sus nodos.

En la Figura 3.3 podemos observar un ejemplo de la representación gráfica de un árbol utilizando dos variables explicativas. Podemos ver cómo se separan los datos formando subgrupos en los que cada uno tiene un valor de predicción.

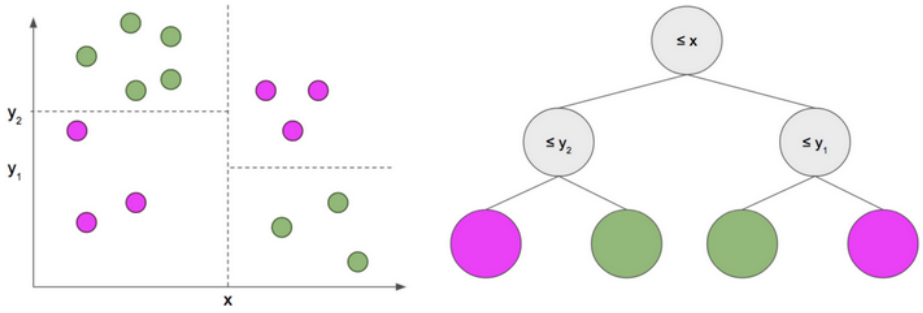


Figura 3.3: Esquema árbol de decisión. Fuente: Orellana-Alvear (2018)

Son denominados árboles de regresión y clasificación o CART, por sus siglas en inglés, aquellos en los que la variable a predecir es continua y discreta, respectivamente. Para el desarrollo de este proyecto nos atañe los árboles de clasificación.

Otra característica de los árboles es que aportan medidas de importancia a las variables. Esto es, permiten crear un *ranking* de variables. Además, permiten trabajar con valores *missing* o atípicos. No obstante, esta técnica tiene algunos inconvenientes, como por ejemplo la realización de predicciones muy *toscas*, ya que a todas las observaciones nuevas les da el mismo valor. Esto puede generar que haya una gran varianza.

3.3.1. Técnicas basadas en árboles

Estas desventajas de los modelos de árboles no han podido solucionarse mejorando los algoritmos o minimizando los errores de las funciones. Sin embargo, se han creado otras técnicas que se basan en la combinación del resultado de varios árboles. Esta técnica de combinación se conoce con el nombre de ensamblado o *ensemble learning*. Los métodos *ensemble* (Koren, 2009; Dietterich et al., 2002) consisten en la construcción de predicciones a partir de la combinación de las predicciones de varios modelos. De esta forma, se comprueba si realizando dicha combinación puede producir una mejora respecto a los resultados discretos.

Estos métodos no solo se basan en combinar distintas técnicas o algoritmos de predicción, como los que ya hemos comentado y otros que comentaremos más adelante, sino que pueden realizar el ensamblado utilizando un mismo tipo de predictor. Dentro de ellas encontramos las técnicas basadas en la generación de estructuras arborescentes como las que veremos a continuación.

3.3.1.1. *Bagging*

Según Breiman (1996) la técnica *bagging*, diminutivo de *bootstrap aggregation*, consiste en crear diferentes modelos de árboles usando submuestras aleatorias con reemplazo de los datos, para luego combinar los resultados. Sigue los pasos que veremos a continuación:

1. Se selecciona el conjunto de datos de entrenamiento y se divide en distintos subconjuntos de datos con o sin reemplazamiento.

2. Con cada una de estas submuestras obtenidas se crea un modelo predictivo, obteniendo por tanto modelos diferentes.

3. Se construye un único modelo predictivo, que es el resultado de promediar todos los modelos creados.

La ventaja de utilizar la técnica *bagging* es que no hay dependencia de los datos y se evita la inestabilidad generada por un árbol simple. Como consecuencia, se reduce la varianza. Además, se suavizan las predicciones obteniendo valores más sutiles y adaptados, ya que no se aplicará el mismo valor predictivo como ocurre en los árboles de decisión.

No obstante tenemos una desventaja, y es que en esta técnica se usan siempre las mismas variables para construir todos los árboles. Por lo que, si estamos en presencia de una variable dominante, la mayoría de los modelos generados serán iguales o muy parecidos.

Es importante destacar que en el proceso de *bagging*, el número de árboles generados no es un parámetro por el que debemos preocuparnos. Esto es así ya que por mucho que se incremente el número de unidades, no se aumenta el riesgo de sobreajuste del modelo. Cuando se alcanzada un determinado número de árboles, el valor del error se estabiliza.

3.3.1.2. *Random Forest*

Esta técnica (Breiman, 2001) es una extensión de *bagging*, o dicho de otra manera: *bagging* es un caso particular de *random forest*. Consiste en incorporar aleatoriedad en las variables utilizadas para segmentar cada nodo del árbol. De forma análoga al modelo anterior, describiremos los pasos que sigue esta técnica:

1. Se selecciona el conjunto de datos de entrenamiento y se divide en distintos subconjuntos de datos con o sin reemplazamiento.
2. En cada nodo, seleccionamos subgrupos de variables entre todas las originales y de ese subgrupo de variables, se escoge la mejor para la partición del nodo.
3. Con cada una de estas submuestras obtenidas se crea un modelo predictivo, obteniendo modelos diferentes.
4. Se construye un único modelo predictivo, que es el resultado de promediar todos los modelos creados.

Por tanto, la ventaja de esta técnica es que se evita que haya variables dominantes y se aprovecha mejor la información del resto de variables del conjunto. De esta forma nos aseguramos de que los árboles puedan ser muy diferentes y capten sutilezas importantes, evitando así problemas de sobreajuste. Además, tal y como ocurre en *bagging*, tampoco hay que tener en cuenta el número de árboles generado, ya que llegados a un punto el error se estabiliza.

3.3.1.3. *Gradient boosting*

Según Amat Rodrigo (2019) *gradient boosting* es otra estrategia de *ensemble*. A diferencia de los dos métodos anteriores, consiste en un algoritmo iterativo. La idea es ajustar, de forma secuencial, varios modelos de árboles. Con cada modelo que se genera se utiliza la información del modelo anterior para aprender de sus errores. De esta forma se intenta mejorar el resultado en cada iteración.

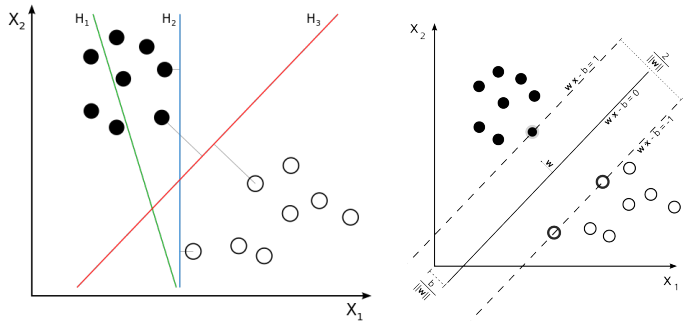
Tanto en *bagging* como en *random forest* la cantidad de árboles generado no es un parámetro que pueda afectar a los modelos. Sin embargo, en *boosting*, si creamos demasiados árboles, puede conllevar a sobreajuste. Para evitar este problema (Friedman, 2001) podemos ajustar un parámetro llamado *learning rate* o *shrinkage* (tasa de aprendizaje) que gradúa cuánto se va corrigiendo el error en cada iteración. Hará falta un número mayor de iteraciones (árboles) cuando se introduce un valor de *shrinkage* muy pequeño, mientras que para tasas moderadas el número de iteraciones necesarias es inferior.

3.4. *Support Vector Machine*

La técnica *support vector machine* o SVM (Amat Rodrigo, 2015; Kowalczyk, 2017; Portela García-Miguel, 2020) constituye un método para la resolución

de problemas de clasificación y regresión. El objetivo es crear una separación lineal de clases con métodos algebraicos, mediante la búsqueda de un hiperplano de separación. Para ello, se basa en tres ideas importantes:

- Hiperplano y *Maximal Margin Classifier* (Vapnik y Chervonenkis, 1964): en un espacio de p dimensiones, se define un hiperplano como un subespacio plano de p-1 dimensiones. Dada definición de hiperplano se buscará aquel que separe las clases sin fallos. Sin embargo, esto puede resultar en un número infinito de posibles hiperplanos en los que existe separación lineal perfecta, como se muestra en la Figura 3.4a (p. ej., recta H_2 y H_3). Sin embargo, se considera una restricción muy estricta puesto que exige una separación perfecta y no siempre se da dicha situación.



(a) (b)

Figura 3.4: SVM *maximal margin*. Fuente (Portela García-Miguel, 2020)

- *Support vector classifier* o *soft margin* SVM (Cortes y Vapnik, 1995): para no cometer sobreajuste del modelo, se debe permitir observaciones mal clasificadas por los separadores. Basándose en la idea anterior, esta técnica permitir que existan observaciones mal separadas, pero penalizando por la distancia al margen de dichas observaciones. Esto implica añadir una constante de error y otra de penalización, C, que está relacionada inversamente con la anchura del margen. Si C toma valores muy grandes estará penalizando mucho la mala clasificación ysi toma valores muy pequeños está siendo más permisivo.

- *Kernel* (Boser et al., 1992): este método se puede considerar como una extensión del anterior, con la diferencia de que puede aumentar la dimensión de los datos. Como podemos observar en la Figura 3.5, las separaciones lineales generadas cuando se aumenta el número de dimensiones, se convierten en límites de separación no lineales si los proyectamos en el espacio original. El objetivo de esta técnica es tener un algoritmo más universal que permita establecer relaciones no lineales.

  El inconveniente es que al aumentar el número de dimensiones aumentamos el número de variables artificiales y se complica el proceso de optimización. Por esta razón interviene lo que se conoce como *truco del kernel*. Según Portela García-Miguel (2020), *cualquier algoritmo que de-*

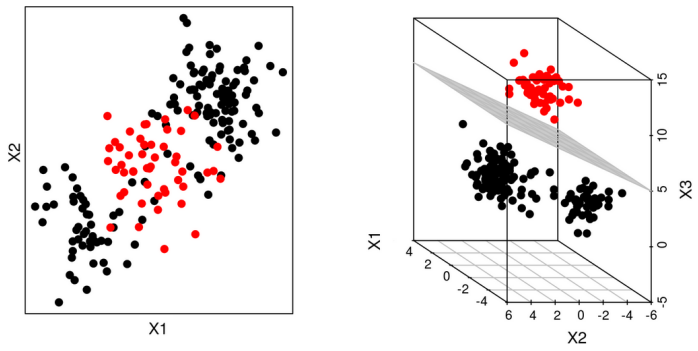


Figura 3.5: Máquinas de vector soporte. Fuente: (Amat Rodrigo, 2015)

pende de productos escalares permite trabajar en una dimensión controlada a través de una función llamada kernel. El presente trabajo estudiaremos las funciones *kernels* más habituales:

- ***Kernel* lineal**: modelo básico de SVM. Si se emplea un *kernel* lineal, el clasificador SVM obtenido es equivalente al *support vector classifier* visto anteriormente. Sigue la función $K(x_i, x_j) = x_i \cdot x_j$.
- ***Kernel* polinomial**: modelo SVM que permite extenderse a separaciones no lineales: $K(x_i, x_j) = (x_i \cdot x_j + C)^d$, donde la constante C representa la penalización y la d (*degree*) el grado del polinomio, en este caso la dimensión.
- ***Kernel radial basis function* (RBF)**: también conocido como *kernel* Gaussiano, es más general que el modelo anterior. A diferencia de este solo consta de un parámetro σ que controla el comportamiento del *kernel*: $K(x_i, x_j) = exp\left(-\gamma \parallel x_i - x_j \parallel^2\right)$.

Esta idea del *kernel* es la que hace a *support vector machine* ser uno de los algoritmos más populares.

3.5. Validación de modelos

Si creamos modelos de predicción como los estudiados anteriormente, podemos utilizar algunas técnicas que generan buenos resultados para la validación de dichos modelos:

- ***Train-test***: Una aproximación es la de dividir el conjunto de datos en dos sub-conjuntos disjuntos: entrenamiento y *test*.

  En primer lugar, el conjunto de entrenamiento lo emplearemos como hasta ahora para entrenar cada uno de los diferentes modelos. Para cada uno de estos, mediremos su desempeño con el conjunto de datos de *test*. De esta forma, al ser observaciones sobre las que no se había entrenado el modelo, podemos obtener una aproximación del desempeño real que tendría cada uno de los diferentes modelos.

- **Validación cruzada**: La validación cruzada es una herramienta muy útil para desarrollar, ajustar y evaluar la calidad de los modelos de minería de datos. Mediante este método buscamos aumentar la validación del grado de desempeño real de los modelos.

  Mediante este enfoque entrenaremos n modelos y escogeremos aquel que mejores resultados dé. Para ello en primer lugar dividimos los datos en n conjuntos iguales de forma aleatoria. A continuación, seleccionamos un modelo para la fase de *test* y los $n-1$ restantes para la fase de entrenamiento y medimos el desempeño. Luego se van escogiendo sucesivamente todos los n conjuntos para *test* y el resto para entrenamiento. De esta forma se habrán entrenado y probado n modelos. Por último, escogeremos como modelo ganador aquel que haya minimizado el error medio en su respectivo conjunto de *test*.

Debemos tener en cuenta que, al evaluar los modelos de predicción sobre distintos conjuntos de datos, obtendremos resultados distintos. Por tanto, debemos evaluar la media del error que genera el modelo (sesgo) y la variabilidad del mismo (varianza).

3.6. Medidas de clasificación

Cuando se construyen modelos, como los que hemos visto anteriormente, donde la variable a predecir es binaria, podemos obtener la matriz de confusión. Esta es una herramienta que permite visualizar el desempeño de un algoritmo. Contiene el número de observaciones que han sido bien y/o mal clasificadas. En la Tabla 3.1 podemos ver como se hace dicha clasificación:

	Predicción = 0	**Predicción = 1**
Realidad = 0	Verdadero negativo (VN)	Falso positivo (FP)
Realidad = 1	Falso negativo (FN)	Verdadero positivo (VP)

Tabla 3.1: Matriz de confusión

donde podemos generar las siguientes métricas:

- Tasa de aciertos o exactitud (*accuracy*): $\frac{VN+VP}{VN+FP+FN+VP}$
- Tasa de fallos o precisión (*precision*): $\frac{FP+FN}{VN+FP+FN+VP}$
- *Sensibilidad* o tasa de verdaderos positivos. Esta medida mide la capacidad que tiene el modelo que capturar los *eventos* (1): $\frac{VP}{FN+VP}$

- *Especificidad* o tasa de verdaderos negativos. Capacidad que tiene el modelo para capturar los *no-eventos* (0): $\frac{VN}{VN+FP}$

Otra de las métricas de evaluación y verificación del rendimiento de modelos de clasificación es el área bajo la curva *ROC*. Esta se obtiene a partir del valor la *sensibilidad* y 1 - *especificidad* (tasa de falsos negativos). Se considerará un modelo con buen rendimiento aquel cuyo área bajo la curva *ROC* tenga un valor cercano a 1, y se considerará un modelo con bajo rendimiento aquel que este más cercano al valor 0,5.

Capítulo 4

Preparación de los Datos

"Ya no estamos en la era de la información.
Estamos en la era de la gestión
de la información."
— Chris Hardwick

En este capítulo analizaremos las tres primeras fases de la metodología SEMMA. Esto es, las fases *sample* (muestra), *explore* (exploración) y *modify* (modificación).

En primer lugar, analizaremos la base de datos de la que vamos a extraer la información. Seleccionaremos las variables y datos más relevantes para nuestro estudio. Una vez completado este proceso, realizaremos una depuración de los datos obtenidos.

4.1. Muestra

En esta fase *sample* explicaremos el proceso de extracción de nuestros datos. Como ya hemos comentado el propósito del presente trabajo es conocer y analizar cuáles son las características que permitan que un ensayo clínico pueda completarse sin interrupciones.

Los datos que utilizaremos para la realización del presente trabajo los hemos extraído de la base de datos de la AACT[1]. Se trata de una base de datos relacional de acceso público que contiene toda la información sobre cada ensayo clínico registrado en `ClinicalTrials.gov`.

En la Figura 4.1 se muestra un fragmento de cómo está estructurada dicha base de datos, donde cada caja representa una tabla. Dentro de cada tabla existe una gran cantidad de variables que tienen información sobre cada ensayo. No obstante, hemos realizado una selección de variables en función del interés clínico y bibliográfico (Califf et al., 2012).

[1] *Database for Aggregated Analysis of* `ClinicalTrials.gov`. Para más información acceder a `https://www.ctti-clinicaltrials.org/aact-database`.

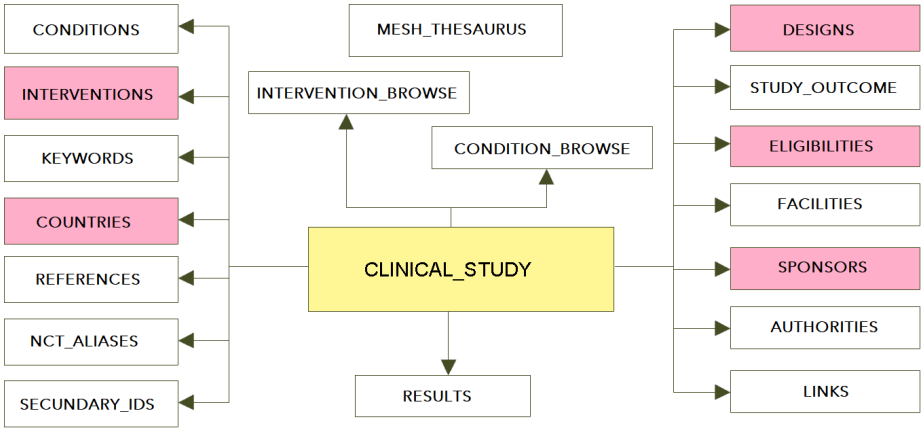


Figura 4.1: Diagrama relacional base de datos AACT (fragmento)

4.1.1. Extracción, tratamiento y descripción de las variables

La tabla principal es `clinical_study`, que contiene el identificador único de cada ensayo clínico. De esta tabla hemos seleccionado una serie de variables que describiremos más adelante. En este proceso, hemos aplicado algunos filtros. En primer lugar, hemos seleccionado aquellos ensayos que están en fase 3 y fase 4 ya que se considera que son estudios que están en fases avanzadas, e interrumpirlos puede suponer un gran coste. Además, seleccionamos aquellos ensayos comenzados entre enero de 2005 y septiembre de 2019. Es importante destacar que desde el comienzo de un ensayo hasta la siguiente fase del mismo hay un largo período de tiempo. Por tanto, al aplicar dichos filtros el ensayo más reciente que tenemos es de marzo de 2019. En la Tabla 4.1 podemos observar qué representan las variables seleccionadas[2].

Variables	Descripción
`enrollment`	El número total estimado o el número real de individuos que están inscritos en el estudio. El hecho de que sea estimado o real depende de la variable `enrollment_type`.
`enrollment_type`	Representa el estado actual del número total estimado o el número real de individuos que están inscritos en el estudio. Presenta los valores: • *actual*: número real de participantes. • *anticipated*: número estimado de participantes.
`has_dmc`	indica si se ha nombrado un comité de vigilancia de datos para el estudio[3]. Se clasifica en: • *t* (Si) • *f* (No)

Continúa en la siguiente página...

[2]Acceder a `https://prsinfo.clinicaltrials.gov/definitions.html` para obtener más información.

[3]El comité de supervisión de datos (*Data Monitoring Committee* o DMC) es un grupo de científicos independientes que son nombrados para supervisar la seguridad e integridad científica de un ensayo clínico en seres humanos. Además, se encarga de hacer recomendaciones al patrocinador con respecto a la detención del ensayo por eficacia, daños o por inutilidad.

Variables	Descripción
nct_id	Identificador único del ensayo.
number_of_arms	Número de grupos o subgrupos que componen el ensayo.
phase	La fase actual del ensayo clínico. Presenta las categorías: • *phase 3* • *phase 4*
overall_status	Indica el estado actual del ensayo clínico: • *terminated*: estudio interrumpido y no se reanudará • *completed*: estudio concluido con normalidad

Tabla 4.1: Descripción de las variables (tabla clinical_study)

La variable objetivo binaria seleccionada es overall_status donde la categoría *completed* representa la categoría de interés.

La siguiente tabla analizada es designs que contiene las principales características del ensayo. En la Tabla 4.2 podemos observar cuáles han sido las variables seleccionadas.

Variables	Descripción
allocation	Indica el método por el cual los pacientes son asignados a los distintos grupos del ensayo. Contiene los valores: • *randomized*: los pacientes se asignan a un grupo de forma aleatoria. • *non-randomized*: los pacientes se asignan a los grupos de manera intencionada.
intervention_model	Indica el tipo de estrategia usada para asignar los tratamientos a los pacientes. Contiene los valores: • *single group*: estudios de un solo grupo. • *parallel*: se asignan a uno o más grupos de forma paralela. • *crossover*: los pacientes reciben un tipo de tratamiento durante las primeras fases del estudio y otro tipo durante las últimas fases. • *factorial*: reciben dos o más tratamientos por separado y/o combinados. Se evalúan con otros grupos de control.
masking	Enmascaramiento. *Estudio ciego*. Toma los valores: • *none (open label)*: no se utiliza ningún tipo de *máscara*. Los involucrados en el estudio conocen el tratamiento. • *single*: una parte, investigador o participante, desconoce el tratamiento; también es conocido como *single-masked study* • *double*, *triple* o *quadruple*: dos, tres o cuatro de las partes involucradas desconocen el tratamiento.
primary_purpose	Principal objetivo del tratamiento evaluado en el ensayo. Contiene los siguientes valores: • *treatment*: tratar una enfermedad, síndrome o afección. • *prevention*: prevenir el desarrollo de una enfermedad o condición de salud específica. • *diagnostic*: identificar una enfermedad o condición de salud.

Continúa en la siguiente página...

Variables	Descripción
	• *supportive care*: se intenta maximizar la comodidad, minimizar los efectos secundarios o mitigar el deterioro de la salud del paciente. • *screening*: identificar una enfermedad o los factores de riesgo de la misma. Se evalúa en individuos en los que se desconoce si padece o no dicha enfermedad. • *health services research*: tratamientos para evaluar los procesos, gestión, organización o financiación de la asistencia sanitaria. • *basic science*: tratamientos para examinar el mecanismo básico de acción (*p. ej.*, la fisiología o la biomecánica de una intervención). • *device feasibility*: se estudia un producto o un dispositivo médico para evaluar la viabilidad del mismo. Los estudios se realizan para confirmar el diseño y las especificaciones de funcionamiento de los mismos antes de iniciar un ensayo clínico completo. • *other*

Tabla 4.2: Descripción de las variables (tabla `designs`)

De `eligibilities` hemos seleccionado las variables que se muestran en la Tabla 4.3. Las variables que representan edades máximas y mínimas están en formato cadena de texto y representadas en minutos, hora, días, semanas, meses y años. Por ese motivo, hemos tratado los datos para convertirlos a enteros que miden las edades en días.

Variables	Descripción
`gender`	Indica el sexo de los individuos que participan en el estudio. Contiene los valores: • *female* • *male* • *all*
`healthy_volunteers`	Indica si se permite o no la participación de personas sanas. Contiene los valores: • *no* • *accepts healthy volunteers*
`maximum_age`	Edad máxima que deben tener los participantes para participar en el estudio clínico. El valor *NA* (valor ausente) indica que no hay límite. Está representada en días.
`minimum_age`	Edad mínima que deben tener los participantes para participar en el estudio clínico. El valor *NA* (valor ausente) indica que no hay límites. Está representada en días.

Tabla 4.3: Descripción de las variables (tabla `eligibilities`)

La tabla `sponsors` contiene información sobre las entidades que dirigen y las que colaboran en el ensayo. Dado que puede haber más de una entidad colaboradora, hemos creado una variable que contiene el total de entidades que intervienen en el ensayo. Podemos ver dichas variables en la Tabla 4.4.

Variables	Descripción
`agency_class`	Indica la categoría de los patrocinadores. Presenta los valores: • *industry* • *nih* • *u.s fed* • *other*
`n_collaborators`	Número de entidades colaboradoras que intervienen en el estudio.

Tabla 4.4: Descripción de las variables (tabla `sponsors`)

De la tabla `countries` hemos seleccionado los países en los que se han realizado cada ensayo. Dado que esta variable contenía más de 80 categorías decidimos generar otra con el nombre de `region` que clasifica dichos países en regiones geográficas (ver Tabla 4.5).

Variables	Descripción
`n_countries`	Cantidad de países en los que se ha realizado el ensayo clínico.
`region`	Región donde se ha realizado el ensayo clínico: • *Africa* • *Arab States* • *Asia Pacific* • *Middle East* • *Europe* • *North America* • *South Latin America*

Tabla 4.5: Descripción de las variables (tabla `countries`)

Por último, de la tabla `interventions` seleccionamos la variable que se muestra en la Tabla 4.6.

Variables	Descripción
`intervention_type`	Indica el tipo de tratamiento especificado a cada grupo que compone el ensayo clínico. Toma los valores: • *drug*: incluyendo placebo • *device*: incluyendo placebo • *biological/vaccine* • *procedure/surgery* • *radiation* • *behavioral*: *p. ej.,* psicoterapia, asesoramiento sobre el estilo de vida. • *genetic*: incluyendo transferencia de genes, células madre y ADN recombinante. • *dietary supplement*: vitaminas, minerales. • *diagnostic Test*: imágenes, *in-vitro* • *other*

Tabla 4.6: Descripción de las variables (tabla `interventions`)

Con esto obtuvimos un total de 40821 observaciones. Dada la gran cantidad de información obtenida y las limitaciones técnicas del equipo donde se va a realizar el estudio, decidimos seleccionar una muestra aleatoria del 11 % de los datos[4].

[4]Todo este proceso de muestreo y pre-procesado de los datos se ha desarrollado en el entorno de desarrollo *RStudio*,podemos verlo accediendo a `https://github.com/eliiags/TFM-MineriaDatos`

4.2. Exploración

Una vez realizado el proceso de extracción y tratamiento de las variables, vamos a comenzar el proceso de depuración de nuestro *dataset*. Dos de las fases más importantes de la metodología SEMMA son la fase de exploración (*explore*) y la fase de modificación (*modify*), las cuales nos van a permitir conocer la naturaleza de nuestros datos. Esto nos va a ayudar a realizar los cambios o modificaciones necesarias para que los modelos de predicción aprendan bien de los datos y no saquen conclusiones erróneas de los mismos.

Depuración es el proceso mediante el cual arreglamos, modificamos o depuramos los datos para que cumplan con las especificaciones esperables de un *dataset*. Es decir, que no haya valores *anómalos* y que cumplan las características que se le tienen que exigir dependiendo del modelo que se vaya a utilizar, entre otras especificaciones que debemos de tener en cuenta para que los modelos generen los mejores resultados posibles. Dentro de la metodología SEMMA esta fase de depuración incluye tanto la fase de exploración como la fase de modificación de los datos.

Para desarrollar la fase de exploración utilizaremos el *software SAS Enterprise Miner*, ya que nos parece una forma más sencilla e intuitiva de visualizar y explorar nuestros datos.

4.2.1. Asignación de roles y clasificación de las variables

Hemos generado un *dataset* que contiene una serie de variables *input* con diferentes características. Hemos importamos nuestro conjunto de datos al *SAS Enterprise Miner*, como se muestra en la Figura 4.2 y hemos clasificado las variables según su función (rol) y tipología (nivel).

Nombre	Rol	Nivel
agency_class	**Input**	**Nominal**
allocation	**Input**	**Binario**
enrollment	**Input**	**Intervalo**
enrollment_type	**Input**	**Binario**
gender	**Input**	**Nominal**
has_dmc	**Input**	**Binario**
healthy_volunteers	**Input**	**Binario**
intervention_model	**Input**	**Nominal**
intervention_type	**Input**	**Nominal**
masking	**Input**	**Nominal**

Nombre	Rol	Nivel
maximum_age	**Input**	**Intervalo**
minimum_age	**Input**	**Intervalo**
nct_id	**ID**	**Nominal**
number_of_arms	**Input**	**Intervalo**
n_collaborators	**Input**	**Intervalo**
n_countries	**Input**	**Intervalo**
overall_status	**Objetivo**	**Binario**
phase	**Input**	**Binario**
primary_purpose	**Input**	**Nominal**
region	**Input**	**Nominal**

Figura 4.2: Asignación de roles y clasificación de las variables

4.2.2. Análisis descriptivo del conjunto de datos y detección de errores

El *nodo Data Mining Database* o *nodo DMDB* nos proporciona información y estadísticas resumidas tanto de las variables de clase como de intervalo del

conjunto de datos. Tenemos un *dataset* que consta de 4495 observaciones y un total de 19 variables. De ellas, 6 son de intervalo, 7 nominales y 6 binarias, en la que se incluye además la variable objetivo.

Las variables de intervalo (ver Figura 4.3) `maximun_age`, `minimun_age` y `number_of_arms` presentan valores ausentes (recuadro rojo). En cuanto a la descripción de las variables relacionadas con la edad, se especifica que si presentan valores ausentes significa que no hay límites de edad establecidos, por lo que más adelante estableceremos valores mínimos y máximos. El resto de variables se encuentran dentro de los límites. No obstante, podemos señalar que la variable `enrollment` (recuadro azul) presenta valores de 1 y 227000 como valores mínimos y máximos respectivamente, contando con una media de 592,17. Esto podría indicar que dicha variable puede contener valores atípicos.

Variable	Ausente	N	Mínimo	Máximo	Media
enrollment	0	4495	1.0000	227000	592.17
maximum_age	2429	2066	0.0014	47450	22273.71
minimum_age	169	4326	30.4170	27375	7335.63
n_collaborators	0	4495	0.0000	29	0.52
n_countries	0	4495	1.0000	30	2.06
number_of_arms	69	4426	1.0000	24	2.35

Figura 4.3: Análisis descriptivo - variables de intervalo

En cuanto a las variables de clase (ver Figura 4.4), podemos apreciar que no presentan errores ni valores ausentes.

Variable	Tipo	niveles	Ausente
agency_class	C	4	0
allocation	C	2	0
enrollment_type	C	2	0
gender	C	3	0
has_dmc	C	2	0
healthy_volunteers	C	2	0
intervention_model	C	4	0
intervention_type	C	8	0
masking	C	5	0
overall_status	C	2	0
phase	C	2	0
primary_purpose	C	7	0
region	C	7	0

Figura 4.4: Análisis descriptivo - variables de clase

El *nodo Explorador de estadísticos* es otra de las herramientas multipropósito de *SAS Enterprise Miner* que nos permite examinar las distribuciones y estadísticos de las variables del conjunto de datos. Vamos a utilizar dicho nodo para analizar el porcentaje de observaciones que representan a las categorías

de las variables de clase. En las Figuras 4.5 y 4.6 se muestran los histogramas de dichas variables.

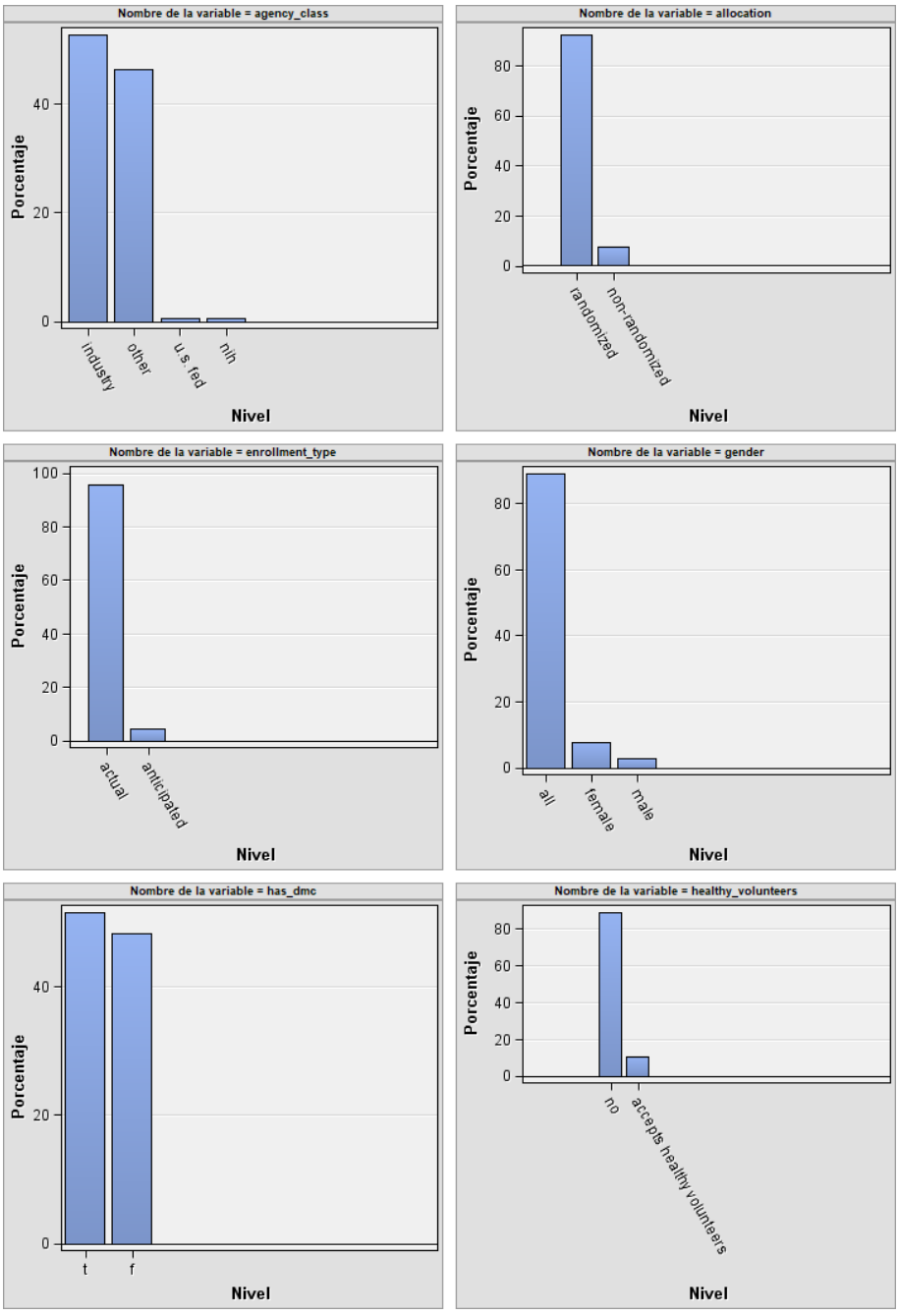


Figura 4.5: Histograma. Niveles de variables (Parte 1)

Con los resultados obtenidos hemos comprobado que las variables `region`, `agency_class`, `intervention_model`, `intervention_type` y `primary_purpose`

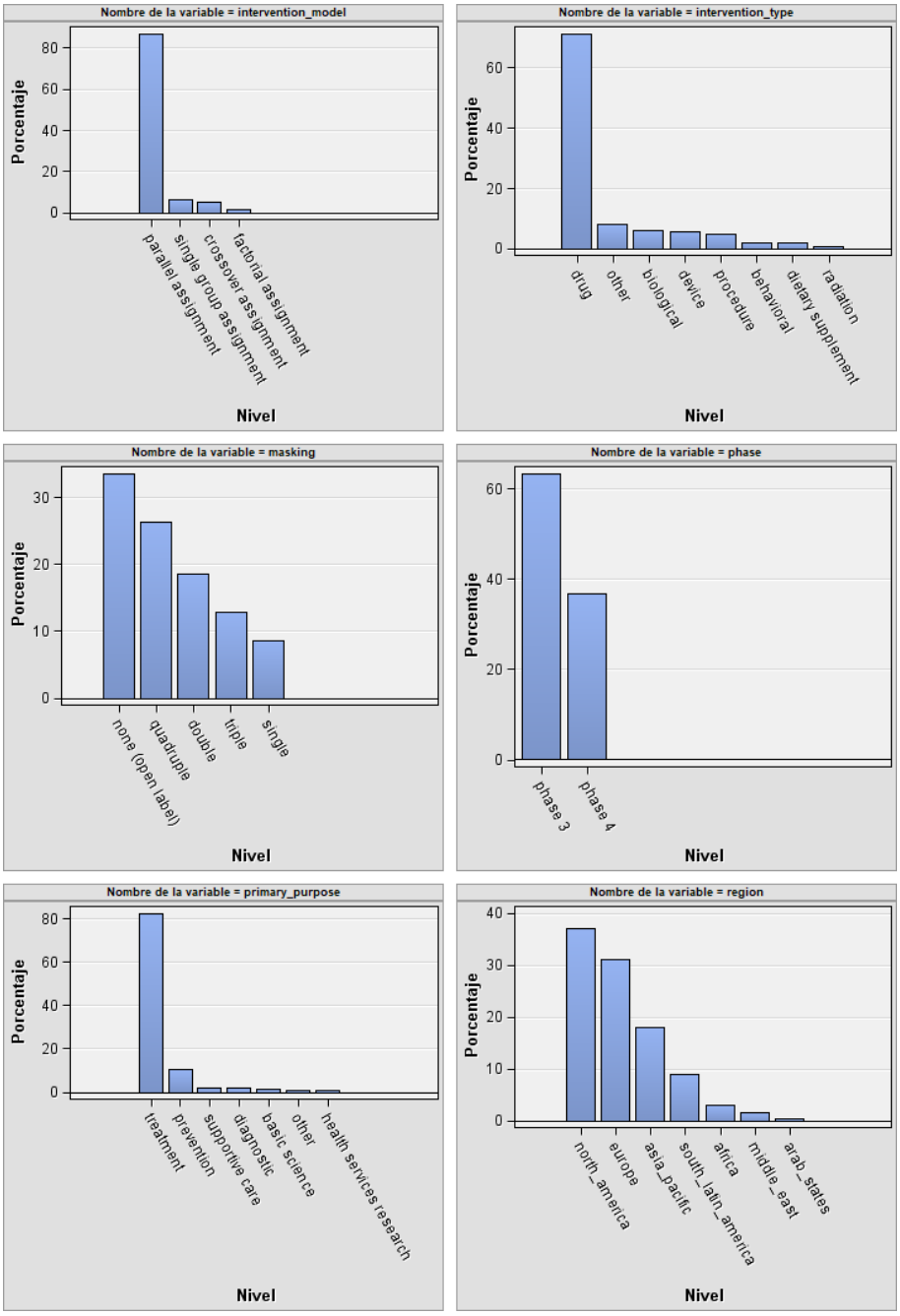


Figura 4.6: Histograma. Niveles de variables (Parte 2)

contienen categorías con muy pocas observaciones. Por tanto, en apartados posteriores se considerará su reagrupación. Este proceso es importante ya que debemos asegurarnos que todas las categorías de las variables contienen un número suficiente de observaciones. En nuestro caso, dado que tenemos

un total de 4495 observaciones vamos a considerar que cada categoría tenga al menos el 2 % de representación, correspondiente a aproximadamente 90 observaciones. De esta forma evitaremos que los modelos que generemos saquen conclusiones erróneas de nuestros datos.

4.3. Modificación

En la fase de modificación de la metodología SEMMA daremos paso a la corrección de todos los errores detectados en la fase anterior.

4.3.1. Corrección de los errores detectados

Para corregir estos errores con valores válidos vamos a utilizar el *nodo Reemplazo*. En la Figura 4.7 podemos observar la recategorización de aquellas variables que estaban representadas por menos del 2 % del total de observaciones. En cuanto a la variable `intervention_model` hemos considerado no reagrupar sus niveles de forma manual ya que representan estrategias muy diferentes de asignación de tratamientos.

Variable	Valor formateado	Valor de reemplazo	ocurrencias
agency_class	industry		2369
agency_class	other		2075
agency_class	u.s. fed	**other**	27
agency_class	nih	**other**	24
region	north_america		1669
region	europe		1398
region	asia_pacific		808
region	south_latin_america		404
region	africa	**a_me_as**	131
region	middle_east	**a_me_as**	69
region	arab_states	**a_me_as**	16

Variable	Valor formateado	Valor de reemplazo	ocurrencias
intervention_type	drug		3213
intervention_type	other		351
intervention_type	biological		262
intervention_type	device		254
intervention_type	procedure		218
intervention_type	behavioral	**other**	89
intervention_type	dietary supplement	**other**	82
intervention_type	radiation	**other**	26
primary_purpose	treatment		3723
primary_purpose	prevention		464
primary_purpose	supportive care		100
primary_purpose	diagnostic	**other**	79
primary_purpose	basic science	**other**	67
primary_purpose	other		34
primary_purpose	health services research	**other**	28

Figura 4.7: Recategorización de variables de clase

4.3.2. Tratamiento de datos atípicos y faltantes

Un dato atípico o *outliers* (Calviño Martínez, 2019) es un dato que está numéricamente distante. Es decir, es un valor excesivamente grande o pequeño en comparación con el resto de valores. La detección de los datos atípicos es un proceso importante ya que si no se detectan podrían tener una gran influencia en los resultados. A continuación, explicaremos algunas de las técnicas que se utilizan para detectar estos valores en variables continuas:

- Desviación típica: se considera que los datos son atípicos cuando están entre 3 y 6 desviaciones típicas de distancia de la media. Este método

se basa en la distribución normal y por tanto sólo es válido si las distribuciones son aproximadamente simétricas.

- Mediana de las distancias absolutas a la mediana o MAD *(Median Absolute Deviation)*: se consideran datos atípicos aquellos que disten más de un número *k* (habitualmente entre 8 y 15) de *MADs* de la mediana. A diferencia del método anterior, este es más adecuado para distribuciones asimétricas, pero no es válido cuando la mediana es igual a 0.
- Percentiles extremos: detecta como atípico el 1 % de los valores más grandes y el 1 % de los valores más pequeños. Este método tiene como ventaja la rapidez, pero el inconveniente es que podría detectar como atípicos algunos valores que no lo son. Por tanto, este método se aplicará cuando no sea posible aplicar ninguno de los dos anteriores.

En la Figura 4.8 podemos observar qué técnicas hemos aplicado a cada variable: método *MAD* a las variables enmarcadas en color rojo; desviación típica a las azules y percentil extremo a las verdes.

Variable	Media	Mínimo	Mediana	Máximo	Asimetría	Curtosis
enrollment	592.1669	1	164	227000	47.20819	2723.639
maximum age	35878.43	0.001389	47450	47450	-0.84516	-0.44149
minimum age	7059.833	0	6570	27375	1.707023	4.708154
n collaborators	0.520356	0	0	29	7.74625	93.84842
n countries	2.063849	1	1	30	3.975578	19.08459
number of arms	2.346362	1	2	24	6.124748	86.66173

Figura 4.8: Métodos para detección de valores atípicos

Para realizar este proceso hemos utilizado nuevamente el *nodo Reemplazo*. Es importante destacar que los datos atípicos dan lugar a valores *raros* o poco frecuente, por lo que debemos tener en cuenta el número de atípicos detectados con respecto al total de observaciones. En este caso solo consideraremos atípicos aquellos valores que representen menos del 5 % del total de observaciones. Observamos en la Figura 4.9 que solo se han detectado atípicos en la variable `enrollment` y representan el 5,89 % del total de los datos. Por tanto, deshacemos este proceso y no aplicaremos dichas técnicas.

Variable	Rol	Etiqueta	Entrenamiento
enrollment	INPUT	enrollment	265
maximum_age	INPUT	maximum_age	0
minimum_age	INPUT	minimum_age	0
n_collaborator	INPUT	n_collaborators	0
n_countries	INPUT	n_countries	0
number_of_arms	INPUT	number_of_arms	0

Figura 4.9: Valores atípicos detectados

Recordemos que los valores ausentes que presentan las variables relacionadas con la edad mínima y máxima significan que no hay límites de edad, por

lo que haremos una imputación simple de dichos valores. Imputar no es más que sustituir los datos faltantes por valores reales. Para ello hemos decidido utilizar el *software Microsoft Excel* ya que es una manera rápida y sencilla de establecer el valor 0 (0 años) a los datos ausentes de la variable `minimun_age` y el valor 47450 (130 años) a los de la variable `maximun_age`. En cuanto a la variable `number_of_arms` utilizaremos el *nodo Imputar* aplicando el método de *Distribución*, de esta forma se asignarán valores aleatorios a las datos ausentes y no se perderá variabilidad.

Una vez concluido este proceso ya no tenemos datos ausentes, los valores mínimos y máximos son razonables y no hay datos atípicos.

4.3.3. Análisis de la relación de las variables input con la variable objetivo

Desde el punto de vista estadístico (Calviño Martínez, 2019) se dice que *dos variables están relacionadas si, al conocer el valor de una de ellas para cierta observación, podemos sacar conclusiones sobre el valor de la otra variable sobre la misma observación*. Analizar esta relación nos servirá para determinar cuáles son las variables que aportan más información en los modelos predictivos. Para medir dicha relación debemos tener en cuenta la tipología de las variables, ya que dependiendo de cómo sea la variable tendremos que usar una medida u otra. En nuestro caso, dado que nuestra variable objetivo es binaria tendremos en cuenta los valores del estadístico χ^2.

Este estadístico es un método útil para detectar relaciones, sobre todo las no lineales. Requiere que las dos variables a analizar sean de clase, o de lo contrario discretizar las variables de intervalo. Esto es así ya que necesita que los datos estén en forma de tabla de contingencia, como la que se muestra a continuación. Donde C_i y D_j representan las clases en las que está dividida la variable de interés, n_{ij} el número de observaciones que toman las categorías C_i y D_j, y $f_{ij} = \frac{n_{ij}}{n}$ la frecuencia relativa de ese cruce de categorías.

	D_1	D_2	$\dots$	D_l	Total
C_1	n_{11}	n_{12}	$\dots$	n_{1l}	$n_{1\cdot}$
C_2	n_{21}	n_{22}	$\dots$	n_{2l}	$n_{2\cdot}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
C_k	n_{k1}	n_{k2}	$\dots$	n_{kl}	$n_{k\cdot}$
Total	$n_{\cdot 1}$	$n_{\cdot 2}$	$\dots$	$n_{\cdot l}$	n

$$\chi^2 = n \sum_{i=1}^{k} \sum_{j=1}^{l} \frac{(f_{ij} - f_{i\cdot} f_{\cdot j})^2}{f_{i\cdot} f_{\cdot j}}$$

Si al calcular la relación observamos que el estadístico χ^2 toma valores muy pequeños se puede decir que las variables no están relacionadas. Sin embargo, el principal inconveniente de esta medida es que no es adimensional. Por este motivo, para valores diferentes del 0 no podemos definir exactamente la relación que tienen dos variables, y menos aún comparar las relaciones entre diferentes variables respecto a la objetivo.

Por este motivo, emplearemos el estadístico *V de Cramer*. Este está basado en el estadístico χ^2 con la ventaja de que su valor está acotado entre 0 y 1.

El valor 0 indica que las variables no tienen ningún tipo de relación, mientras que el valor 1 significa que la relación es directa. Esto nos permitirá establecer un *ranking* donde podamos ver cuál es el valor *V de Cramer* de todas las variables *input* frente a la objetivo. Esta medida viene dada por:

$$V = \sqrt{\frac{\chi^2}{n \cdot min(l-1, k-1)}}$$

Para este proceso usaremos el *nodo Explorador de estadísticos*. Antes de analizar estas relaciones vamos a crear una nueva variable de intervalo con valores aleatorios. Esto nos ayudará a saber cuánto de bien o mal predice esta variable aleatoria y así compararla con el resto de las variables *input* del conjunto. Si la variable aleatoria obtiene valores de *V de Cramer* similares a los de otra variable *input* podemos deducir que dicha variable no está aportando información al conjunto. Utilizaremos el *nodo Transformar variable* para crear la `aleatoria`.

En la Figura 4.10 podemos observar que la variable `aleatoria` ocupa la posición nueve comenzando por la derecha, con un valor *V de Cramer* de 0,035. Esto significa que todas las variables *input* que se encuentran a la derecha no son útiles para predecir, ya que tienen menos relación que una variable que acabamos de crear. En contrapartida, las variables que sí presentan una relación con la variable dependiente son: `region`, `healthy_volunteers`, `has_dmc` e `intervention_type`. No obstante, apreciamos que el valor más alto es 0,138 por lo que no podemos afirmar que tienen una relación alta.

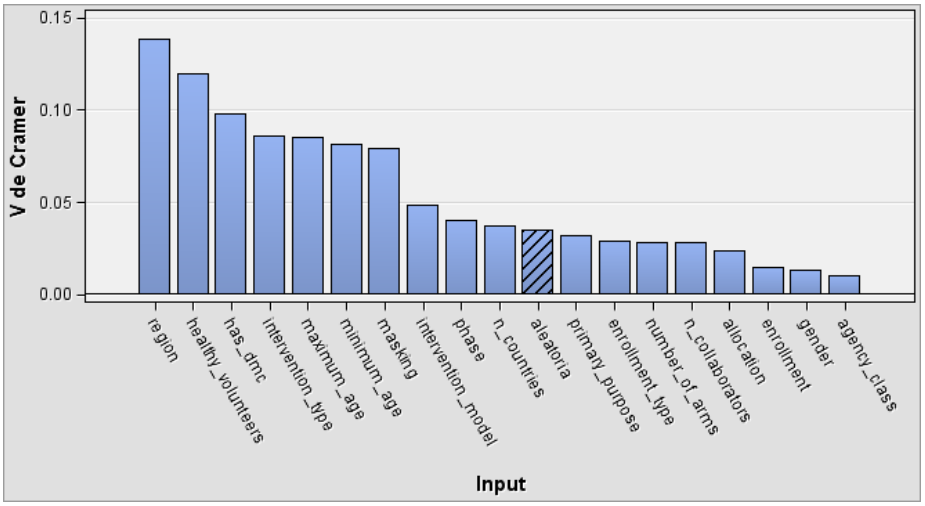


Figura 4.10: Relación entre variables originales: medida *V de Cramer*

4.3.4. Transformación de variables

La transformación de datos persigue intentar maximizar la relación existente entre variables. El objetivo de esta fase es aplicar algún tipo de transformación (logarítmica, exponencial, entre otras) a las variables. Es decir, modi-

ficarlas de manera que a los modelos de predicción posteriores les sea más fácil detectar la relación entre ellas. Utilizaremos el *nodo Transformar variables* para llevar a cabo este proceso de transformación.

Como ya sabemos nuestro objetivo es predecir una variable binaria, por tanto seleccionaremos el método de transformación *Mejor*. Este método calculará todas las distintas trasformaciones posibles y escogerá el que genere un mayor coeficiente X^2 entre la variable de entrada y la objetivo.

En la Figura 4.11 podemos observar cómo han variado los valores de *V de Cramer*. Existe una notable diferencia en la variable `enrollment` donde, tras hacer una discretización, ha pasado de tener un coeficiente *V de Cramer* de 0,014 a 0,367. Al resto variables de intervalo del conjunto también se les ha aplicado una transformación, pero no han obtenido notables diferencias en sus resultados. Cabe mencionar que al hacer este proceso hemos dejado en el conjunto las variables originales y las transformadas.

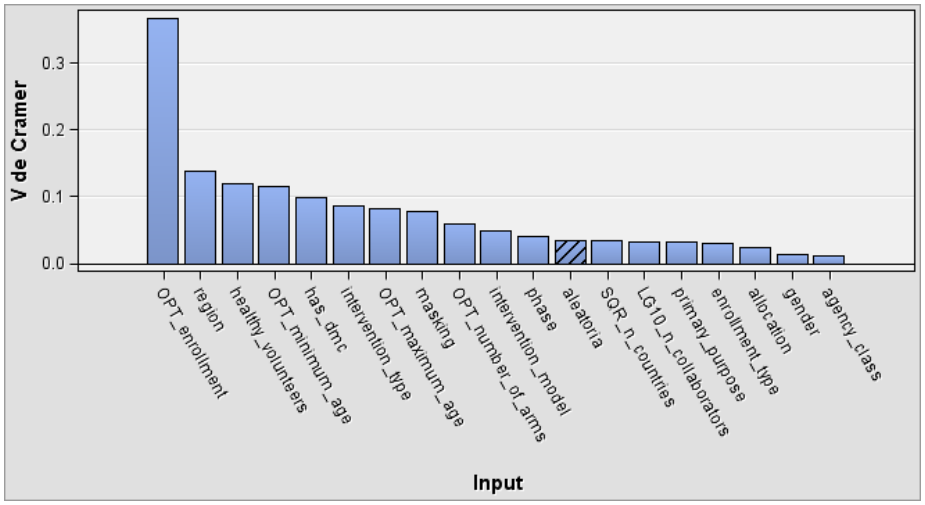


Figura 4.11: Relación entre variables transformadas: medida *V de Cramer*

4.3.5. Selección inicial de variables

Muchos de los modelos predictivos que se van a crear en el presente trabajo tienen sus propios mecanismos de selección de variables. El proceso realizado en el apartado anterior ha generado un gran número de variables en nuestro conjunto. A la hora de crear modelos, esto podría implicar un aumento en el tiempo de cómputo, ya que los propios modelos deben seleccionar variables entre todas las del *dataset*. El objetivo de esta última fase de depuración persigue reducir ese tiempo computacional. Haciendo esta preselección eliminaremos las variables que consideremos que no nos van a servir para predecir nuestro objetivo.

Abrimos un inciso para adelantar que en el presente trabajo crearemos una serie de modelos usando técnicas de *machine learning*, por lo que hemos tomado la decisión de convertir las variables *input* categóricas a variables

dummies. Esto es, se crea una variable de intervalo por cada categoría que tengan las variables discretas de nuestro conjunto estableciendo valores 0 y 1. Creemos este proceso necesario ya que a la hora de aplicar técnicas de selección de variables podemos obtener como resultado *sets* más interesantes para analizar, y además no perdemos información de las variables.

Selección de variables en *SAS Enterprise Miner*

Vamos a utilizar diversos nodos que nos automatizarán los procesos de preselección de variables. Se han generado los siguientes *sets* (ver Tabla 4.7):

Sets	Descripción
Set 1	Variables seleccionadas de forma manual siguiendo el criterio: variables de entradas con un coeficiente χ^2 mayor que la `aleatoria`.
Set 2	Utilizando el *nodo Selección de variables*, seleccionaremos aquellas cuyo valor de R^2 es mayor que 0,005.
Set 3	Se seleccionan variables utilizando un modelo de árbol de clasificación.

Tabla 4.7: Descripción de los *sets* de variables (1-3)

Una vez seleccionados los *sets*, rechazamos la variable `aleatoria` puesto que no es objeto de estudio. Además, vamos a seleccionar otros conjuntos con las siguientes características (ver Tabla 4.8):

Sets	Descripción
Set 4	Contendrá todas las variables originales sin transformación.
Set 5	Contendrá todas las variables del conjunto, es decir las originales y las que se les ha aplicado una transformación.

Tabla 4.8: Descripción de los *sets* de variables (4-5)

Selección de variables en *SAS Base*

Para realizar esta selección crearemos modelos de regresión utilizando los *sets* de variables 4 y 5. Es decir, haremos el proceso utilizando un conjunto con las variables originales y el otro con las variables transformadas. Esto se realizará con el procedimiento `logistic` del entorno *SAS Base*.

Antes de continuar, observamos en la Figura 4.12 la proporción de valores *terminated* (0) y *completed* (1) de nuestra variable objetivo.

Como comentamos en el apartado 3.1, los modelos de regresión tienen sus propios mecanismos de selección de variables. Por ello, de cada conjunto de datos (*sets* 4 y 5) realizaremos 3 pruebas utilizando los métodos *forward*, *backward* y *stepwise*. Para que esta selección de variables sea más fiable, utilizaremos la técnica *training-test* repetido con 66 repeticiones utilizando semillas diferentes. Esto es, se hará una partición de los datos *train-test* 80-20, donde el 80 % de los datos irán al conjunto de entrenamiento y el 20 % restante al conjunto de prueba. Dado que los valores de la muestra están desbalanceados, esta partición de los datos se realizará usando la opción de estratificación.

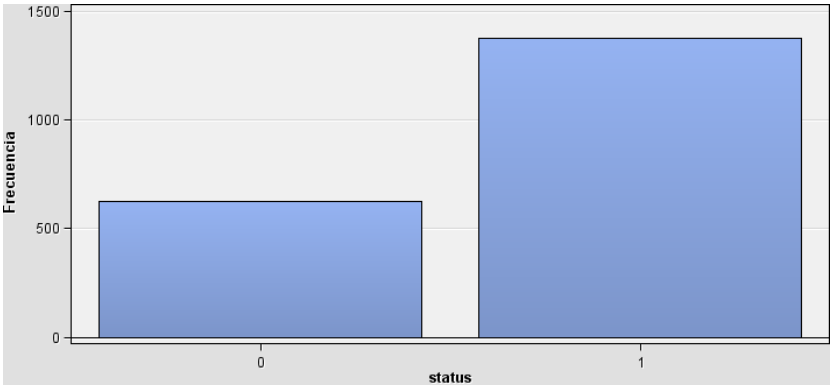


Figura 4.12: Frecuencia de valores de la variable objetivo

Debemos asegurarnos que en cada submuestra *train* y *test* haya suficiente representación de ambas categorías.

Como resultado de estas pruebas obtendremos unas tablas de frecuencia con los conjuntos de variables que más veces han sido seleccionadas. De las 6 pruebas realizadas en total (tres para el *set* 4 y tres para el *set* 5), seleccionaremos los dos conjuntos de variables que más veces han aparecido, por lo que tenemos un total de 12 conjuntos. Estos 12 *sets* se ven reducidos a 6 por las coincidencias de resultados generados por todos los modelos. Con estos 6 *sets* más los cinco generados anteriormente tenemos un total de once. En la Tabla 4.9 podemos observar las características de los nuevos *sets* creados.

Sets	**Descripción**
Set 6	Primer conjunto con mayor frecuencia de aparición utilizando el *set* 4 y aplicando el método *stepwise*.
Set 7	Segundo conjunto con mayor frecuencia de aparición utilizando el *set* 4 y aplicando el método *stepwise*.
Set 8	Primer conjunto con mayor frecuencia de aparición utilizando el *set* 5 y aplicando el método *backward*.
Set 9	Segundo conjunto con mayor frecuencia de aparición utilizando el *set* 5 y aplicando el método *backward*.
Set 10	Primer conjunto con mayor frecuencia de aparición utilizando el *set* 5 y aplicando el método *stepwise*.
Set 11	Segundo conjunto con mayor frecuencia de aparición utilizando el *set* 5 y aplicando el método *stepwise*.

Tabla 4.9: Descripción de los *sets* de variables (6-11)

Con este proceso de selección de variables damos por concluida la preparación y depuración de nuestros datos.

Capítulo 5

Modelización

"No podemos resolver problemas pensando
de la misma manera que
cuando los creamos"
— Albert Einstein

En el presente capítulo desarrollaremos la fase modelizar (*model*) de la metodología SEMMA. Como se ha comentado anteriormente, el objetivo de esta fase es hallar el modelo que nos permita predecir la variable objetivo. Por tanto, crearemos diversos modelos predictivos utilizando técnicas clásicas y otras más avanzadas como las técnicas de *machine learning*. De esta forma podremos analizar los requisitos que debe tener un ensayo clínico para asegurar que puede completarse con normalidad.

5.1. Regresión logística

Con todos los *sets* de variables que hemos generado en el capítulo anterior, haremos un estudio para analizar el sesgo y la varianza de los modelos con respecto a la tasa de fallos de todas las combinaciones de variables. Para ello, volveremos a crear modelos de regresión logística utilizando el método *stepwise*. En este caso, en vez de usar técnicas de remuestreo aplicaremos la validación cruzada repetida ya que es la opción más fiable para el análisis de datos.

Se ha realizado una partición de la muestra en 10 grupos con 30 repeticiones variando la semilla. Como resultados se han calculado los errores promedios cometidos en cada ejecución de validación cruzada para cada semilla. Podemos ver dichos resultados en la Figura 5.1 donde se muestran diagramas de cajas que resumen los resultados obtenidos. Observamos que los modelos con peores resultados son los que contienen las variables originales del conjunto. Esto se debe a la poca relación que ya observamos que existía entre las variables de entrada y la variable dependiente.

Rehacemos el gráfico para ver con mejor detalle los resultados obtenidos

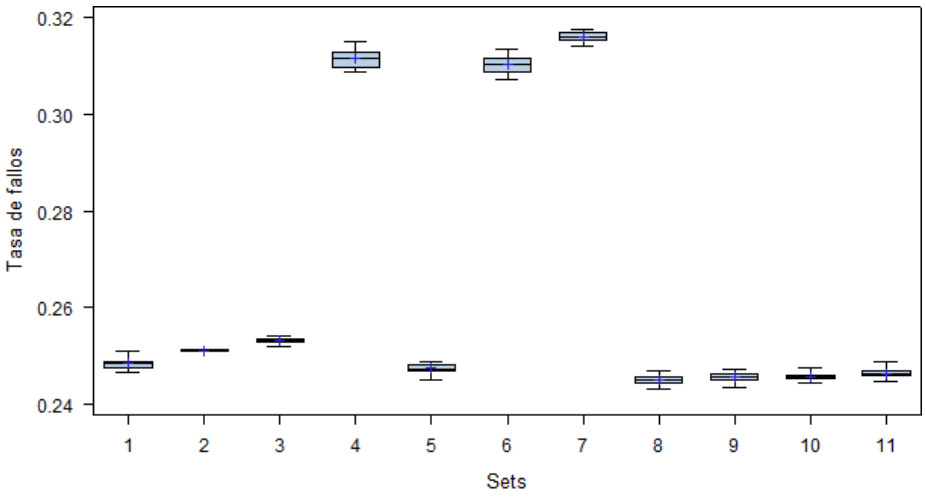


Figura 5.1: Resultados regresión logística (*stepwise*). Remuestro repetido

en el resto de modelos (ver Figura 5.2) y observamos que todos los conjuntos salvo el *Set* 2 presentan una varianza muy parecida. Nos llaman especialmente la atención los *sets* 8, 9 y 10 ya que presentan un sesgo más bajo que el resto de modelos. No obstante, volveremos a realizar una pequeña prueba más con estos tres modelos variando la semilla.

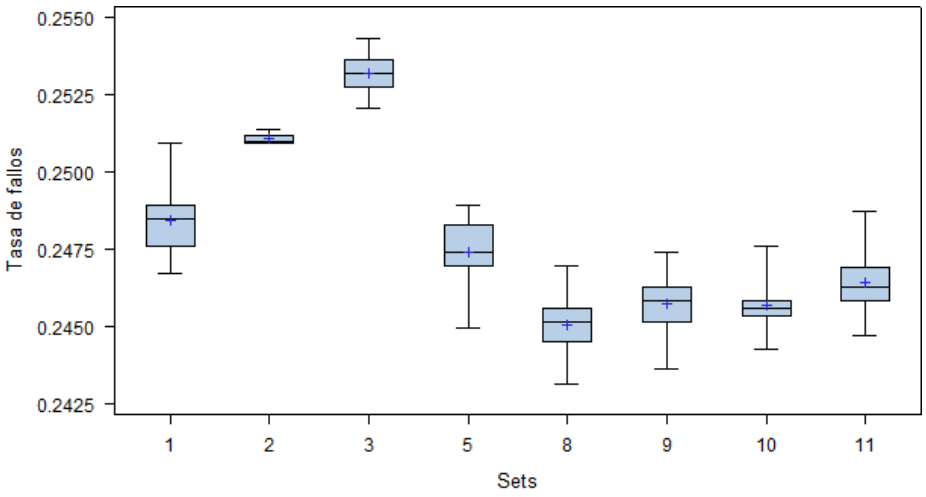


Figura 5.2: Resultados regresión logística (*stepwise*). Remuestro repetido

Podemos observar los resultados en la Figura 5.3. En cuanto al sesgo y varianza los tres modelos generan resultados muy parecidos. No obstante, nos vamos a decantar por el modelo 10 ya que parece ser el modelo más estable y además contiene un total de 18 parámetros. Esto supone cuatro parámetros menos que los dos anteriores.

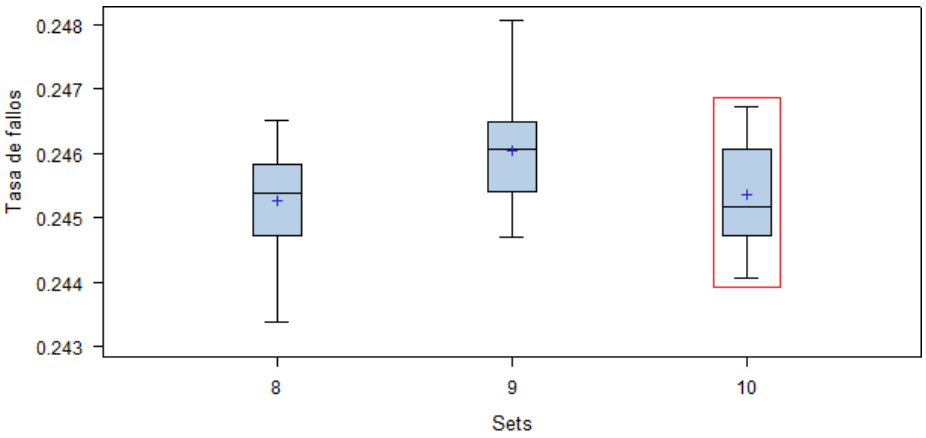


Figura 5.3: Resultados regresión logística (*stepwise*). Validación cruzada

5.2. Redes neuronales

Con el conjunto de datos que mejor resultado obtuvo en el apartado anterior, vamos a crear una serie de modelos de redes neuronales, Para ello variaremos el número de nodos ocultos, el algoritmo de optimización y la función de activación, con el objetivo de encontrar los pesos que minimicen el error. De forma análoga al modelo de regresión, al final de todo este proceso seleccionaremos el mejor modelo predictivo.

Número de nodos

En primer lugar, debemos calcular el número *ideal* de nodos que debería tener nuestro modelo de red en la capa oculta. Lo podemos calcular (Portela García-Miguel, 2020) siguiendo la ecuación:

$$h(k+1)+h+1 = observaciones \ / \ p,$$

donde h es el número de nodos ocultos de la red, k el número de nodos *input* (variables de entrada) y p el número de parámetros.

Calcularemos el numero óptimo de nodos teniendo en cuenta que tenemos entre 20, 25 y 30 observaciones por parámetros. Además, contamos con un total de 4495 observaciones, no obstante, si tomamos el 80 % de los datos para el conjunto *train* nos quedan aproximadamente 3596. Recordemos además que tenemos 18 variables. Así que podemos despejar h:

$$h(18+1)+h+1 \quad = \quad 3596 \ / \ 20 \quad \longrightarrow \quad h = 8{,}94$$

$$h(18+1)+h+1 \quad = \quad 3596 \ / \ 25 \quad \longrightarrow \quad h = 7{,}142$$

$$h(18+1)+h+1 \quad = \quad 3596 \ / \ 30 \quad \longrightarrow \quad h = 5{,}943$$

por tanto, crearemos modelos de redes que tengan entre 5 y 9 nodos ocultos.

Utilizaremos la sentencia `proc neural` para crear modelos de redes neuronales en *SAS Base*. Siguiendo el paso anterior, vamos a generar un bucle de 15 iteraciones en el que podemos comprobar cómo se comporta la red a medida que le vamos añadiendo nodos en la capa oculta. Para este primer estudio, se han creado modelos utilizando el algoritmo de optimización `levmar` (Levenberg-Marquardt) y la función de activación tangente hiperbólica (`tanh`). Además, se ha realizado un entrenamiento preliminar de 2 rondas con 3 iteraciones cada una, con esto se persigue intentar evitar el problema de los mínimos locales.

Contrastamos los resultados con la tasa de clasificación errónea (*MISC*, por sus siglas en inglés). Como podemos observar en la Figura 5.4, a medida que va aumentando el número de nodos, va disminuyendo el error, salvo para casos específicos como en modelos con 5, 8 y 12 nodos que aumenta ligeramente. Esto significa que modelos con un mayor número de nodos en la capa oculta, o incluso con más de una capa, se adaptan mejor al conjunto de datos. Sin embargo, esto puede suponer sobreajuste del modelo.

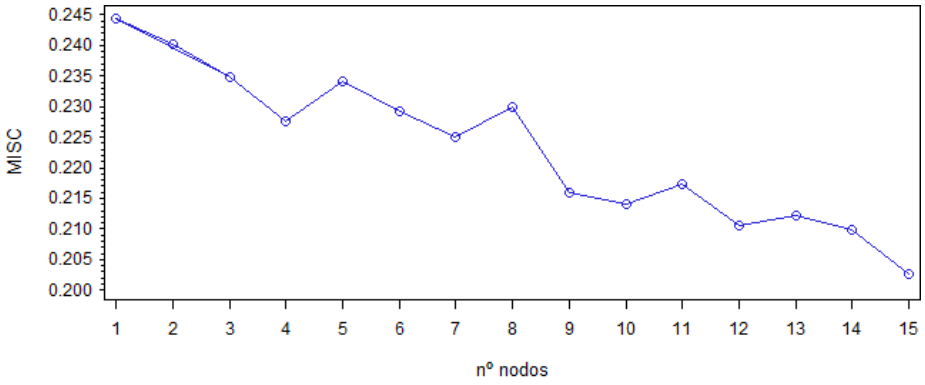


Figura 5.4: Tasa de clasificación errónea en función del número de nodos

La prueba anterior se ha realizado utilizando la totalidad de los datos. Por tanto, realizaremos una prueba más exhaustiva para determinar el número de nodos óptimos. En este caso utilizaremos la técnica de validación cruzada repetida con 5 grupos y 22 repeticiones. Estos modelos de red se crearán con las mismas características que los anteriores.

En la Figura 5.5 podemos volver a comprobar que el error disminuye a medida que se añaden más nodos ocultos a la red. Sin embargo, como ya sabemos esto podría ocasionar sobreajuste del modelo.

Teniendo en cuenta el número de nodos óptimos obtenidos siguiendo la ecuación anterior y comparando los resultados generados en estas dos últimas pruebas (ver Figuras 5.4 y 5.5), hemos tomado la decisión de ajustar modelos de redes con siete nodos en la capa oculta.

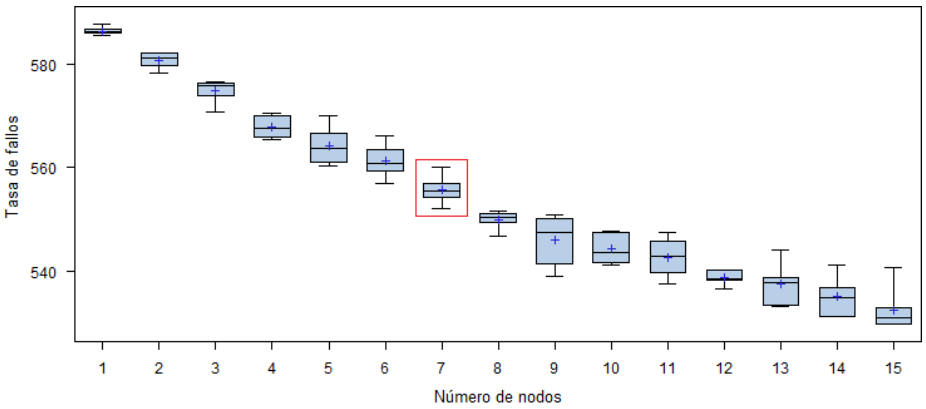


Figura 5.5: Tasa de fallos en función del número de nodos. Validación cruzada

Algoritmos de optimización

Una vez seleccionado el número de nodos en la capa oculta de nuestro modelo de redes neuronales vamos a estudiar cuál es el mejor algoritmo de optimización (Matignon, 2005) que más se ajusta a los datos. El objetivo de dicho algoritmo de optimización es ajustar los pesos w para intentar minimizar la función de error o la función objetivo. Para ello probaremos los siguientes: `levmar` (*Levenberg-Marquardt*), `quanew` (*Quasi-Newton*), `congra` (*Conjugate Gradient Descent*), `dbldog` (*Double dogleg*), `trureg` (*Trust region*) y `bprop` (*backpropagation*). Para este último algoritmo, haremos dos pruebas en las que se han establecido dos valores distintos de los coeficientes *learning rate*[1] y *momentum*[2].

Esta prueba se ha realizado empleando validación cruzada repetida de 5 grupos y 22 repeticiones. Crearemos modelos de redes con cada uno de estos algoritmos de optimización, redes que contendrán siete nodos. Además, los probaremos utilizando la función de activación tangente hiperbólica (`tanh`). No obstante, en el siguiente apartado realizaremos pruebas para comprobar cuál es la mejor función de activación que más se ajuste a nuestros datos.

En la Figura 5.6 podemos observar los resultados obtenidos. Los valores de *learning rate* y *momentum* establecidos para el modelo `bprop1` han sido 0,1 y 0,9, respectivamente. Para el modelo `bprop2` hemos fijado los valores *learning rate* 0,5 y *momentum* 0,1. Estos modelos han generado una diferencia significativa en su tasa de fallos en comparación al resto de modelos.

Hemos decidido seleccionar el modelo `bprop2` ya que obtiene resultados ligeramente mejores que el `bprop1`. No obstante, volveremos a realizar futuras pruebas modificando sus parámetros para estudiar si varían o no sus resultados.

[1] *learning rate* refleja en qué medida se van a cambiar los pesos w en cada iteración, haciendo que el proceso de optimización sea lento para valores muy pequeños con el inconveniente de poder quedarse estancado en mínimos locales.

[2] *momentum* hace que en cada iteración del algoritmo los pesos vayan cambiando acercándose al mínimo. Los valores altos en estos parámetros suponen cambios en los valores de los pesos más bruscos.

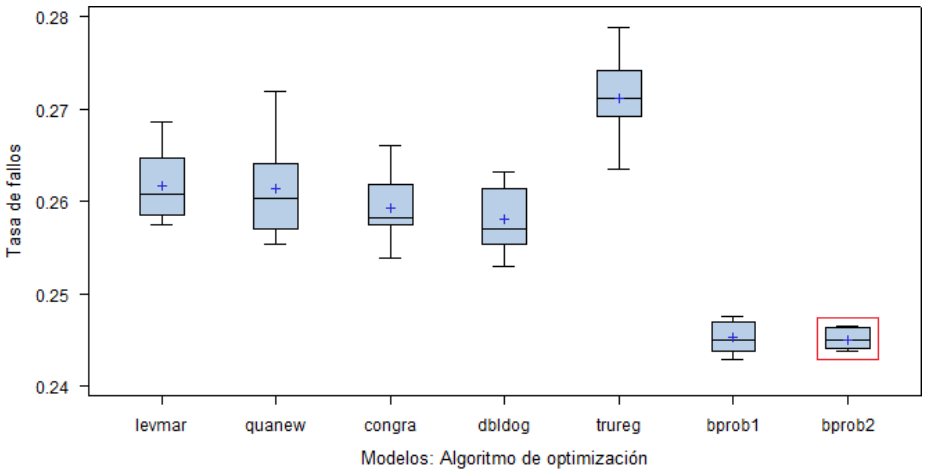


Figura 5.6: Mejores algoritmos de optimización

Función de activación

Seleccionando el algoritmo de optimización `bprop` y siete nodos ocultos, volveremos a utilizar la técnica de validación cruzada repetida para estudiar cuál es la función de activación del modelo de red que mejor se ajusta a nuestros datos. La función de activación (Matignon, 2005) define una salida a partir de un valor de entrada. En la Tabla 5.1 podemos ver las distintas funciones de activación que aplicaremos a nuestros modelos de redes. En la segunda columna de dicha tabla podemos ver el conjunto de valores de salida que puede tomar una función y en la tercera, podemos ver la función matemática correspondiente.

Nombre	**Rango**	**Función** $f(x)$
Tangente hiperbólica (`tanh`)	(-1, 1)	$tanh(x) = 1 - \frac{2}{1+e^{2x}}$
Softmax (`sof`)	(0, 1)	$\frac{e^x}{\sum exponentials}$
Logística (`log`)	(0, 1)	$\frac{1}{1+e^{-x}}$
Arcotangente (`arc`)	(-1, 1)	$arctan(x) \cdot \frac{2}{\pi}$
Seno (`sin`)	(0, 1]	$sin(x)$
Gaussiana (`gau`)	(0, 1]	e^{-x^2}
Exponencial (`exp`)	(0, ∞)	exp^x

Tabla 5.1: Funciones de activación

Tras aplicar dichas funciones mostramos los resultados en la Figura 5.7 donde podemos observar que las funciones *softmax*, *logística* y *gaussiana* obtienen los peores resultados en cuanto a la tasa de fallo. Por tanto, rehacemos

el gráfico para poder visualizar en detalle el resto de modelos.

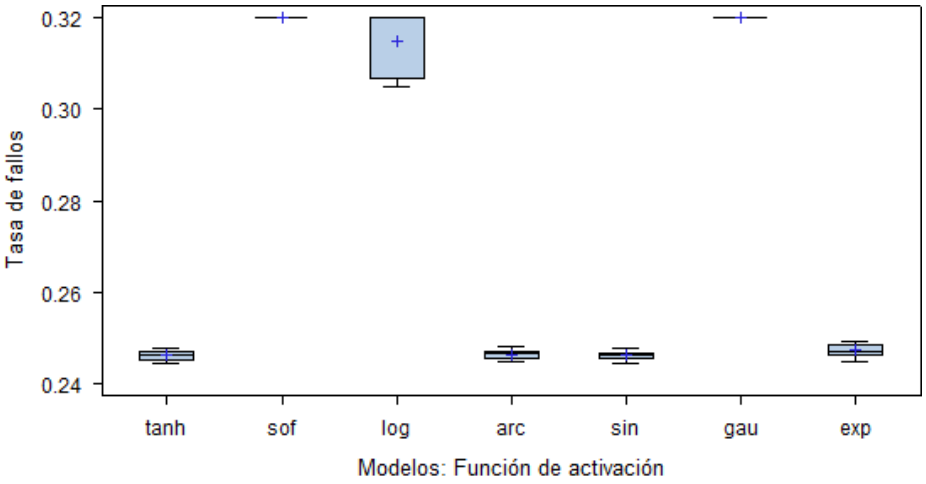


Figura 5.7: Mejores funciones de activación

En la Figura 5.8 podemos observar que para los tres primeros modelos, no existe una notable diferencia en cuanto a los valores de sesgo y varianza. Los valores respecto a la media y la mediana son los mismos para las funciones `tanh` (tangente hiperbólica) y `sin` (seno). Además, los mínimos son iguales y los máximos muy similares, aunque el intervalo entre el primer cuartil (Q1) y el tercer cuartil (Q3) es más uniforme en la función seno aun siendo una función más simple. Es por ello que hemos decidido seleccionar esta función para nuestro modelo de redes.

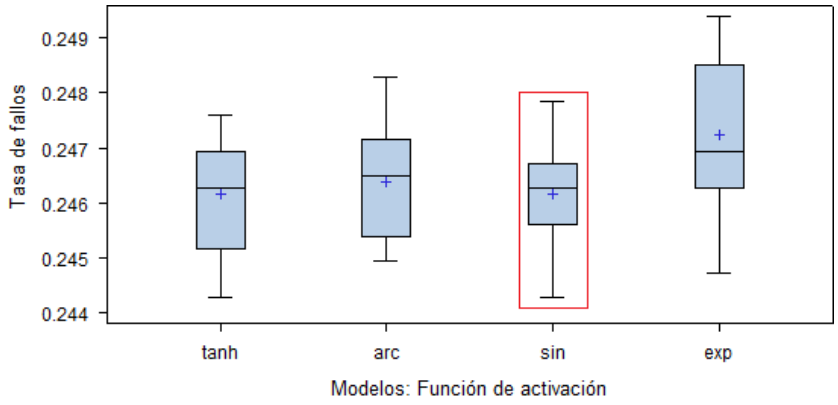


Figura 5.8: Mejores funciones de activación (ampliado)

En resumen, el mejor ajuste que podemos aplicar a nuestro modelo de red neuronal es: siete nodos en la capa oculta usando *backpropagation* como algoritmo de optimización y aplicando la función de activación seno. No obstante, como hemos comentado anteriormente, vamos a crear otros modelos donde hagamos variaciones de los parámetros *learning rate* y *momentum* para estudiar su comportamiento.

Para ello definimos tres nuevas configuraciones de los parámetros de *backpropagation*, así como dos semillas aleatorias. De esta forma realizamos dos pruebas usando validación cruzada repetida de 5 grupos y 22 repeticiones, y aplicando dichas semillas. Podemos observar en la Figura 5.9 que un ajuste (enmarcado en rojo) de los parámetros del algoritmo *backpropagation* ha mejorado al modelo que teníamos definido (enmarcado en negro). Por tanto, tomamos la decisión de seleccionar este modelo.

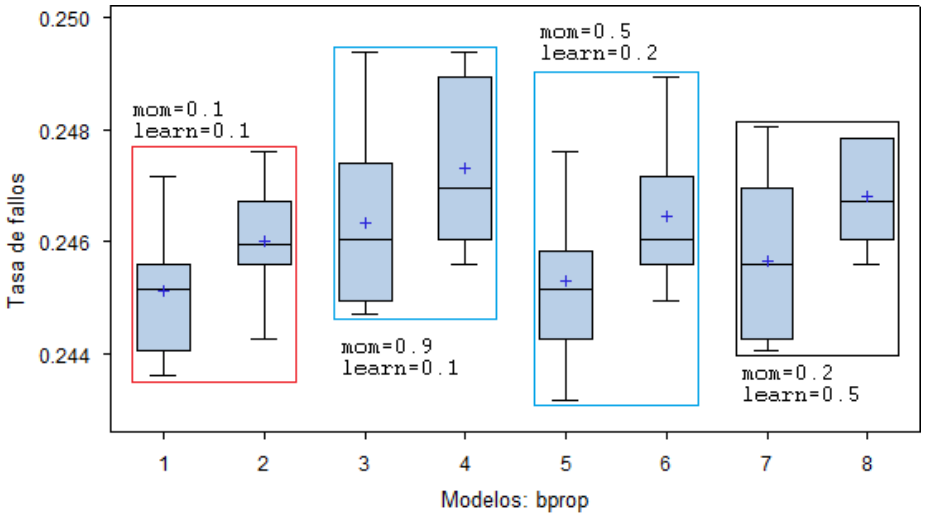


Figura 5.9: Resultados de variaciones de parámetros de modelos `bprop`

Análisis del modelo ganador

Llegados a este punto del desarrollo del trabajo tomaremos el modelo ganador y estudiaremos la posibilidad de aplicar *early stopping*. Esta técnica consiste en dividir la muestra total en submuestras de entrenamiento y validación, y detener el proceso de estimación cuando el error en los datos de validación comience a aumentar. Hemos realizado un total de 15 pruebas aplicando la técnica *early stopping* variando la semilla aleatoria. Las Figuras 5.10a y 5.10b muestran los resultados de dos de las ejecuciones realizadas. El valor del *early stopping* en todas las pruebas realizadas fue superior a 160. No obstante, cabe destacar que no hay mucha diferencia entre el error generado por el conjunto de entrenamiento y el de validación, por lo que no parece que se llegue a generar sobreajuste.

Realizado este proceso, damos por concluido el estudio de modelos de redes neuronales.

5.3. Árbol de clasificación

Como hemos comentado anteriormente, los árboles de decisión son algoritmos iterativos que consisten en segmentar o dividir los datos en regiones

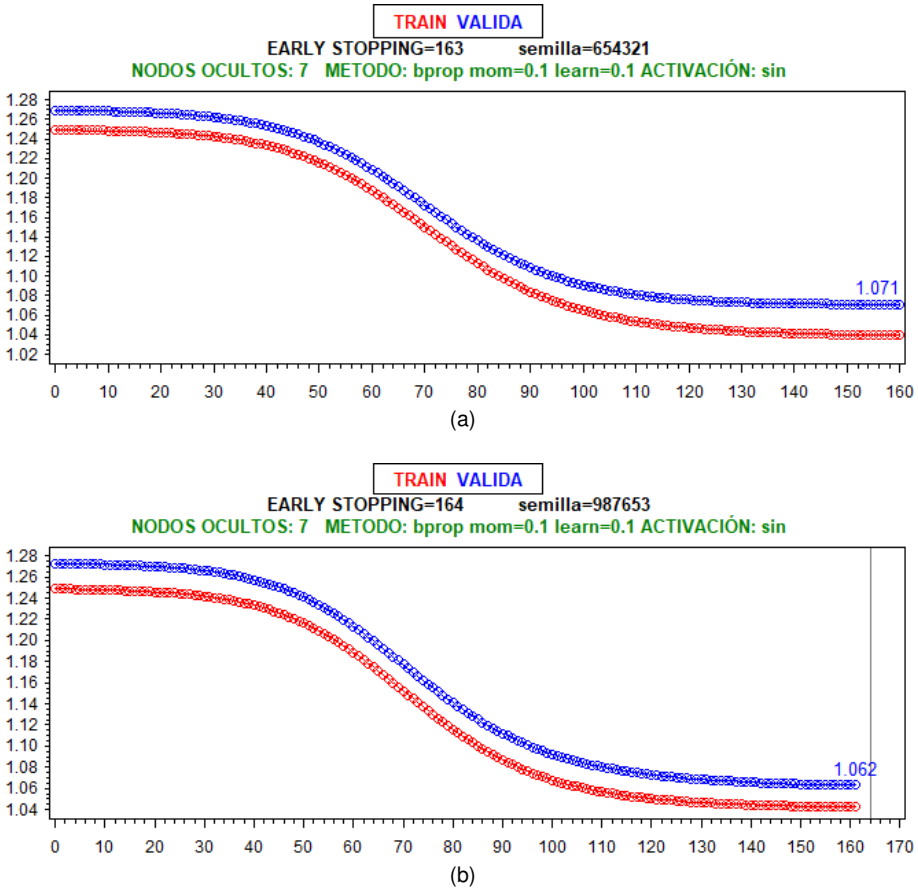


Figura 5.10: Estudio de *early stopping*

basados en intervalos de las variables independientes. Son algoritmos que poseen una gran ventaja ya que tienen una gran potencia descriptiva. Es decir, los resultados se pueden explicar fácilmente.

Dado que para el desarrollo del presente proyecto haremos uso de modelos basados en la generación de estructuras arborescentes, vamos a generar un modelo de árbol de clasificación utilizando el *software SAS Enterprise Miner*. De esta forma, podemos hacer un análisis previo y conocer cómo podrían comportarse los modelos que crearemos en apartados posteriores. Podemos ver un fragmento del árbol generado en la Figura 5.11.

Este árbol binario lo hemos creado con una profundidad máxima de 10 y con 50 como número mínimo de observaciones que debe haber en las hojas. Además se ha usado el índice de *Gini*[3] como criterio para la división de nodos. En la Figura 5.12 podemos observar las variables que el modelo ha seleccionado, y además la importancia que ha establecido a cada una de ellas. Observamos que la variable *dummy* `enrollment` ha sido la más importante

[3]Acceder a `https://www.investopedia.com/terms/g/gini-index.asp` para obtener más información.

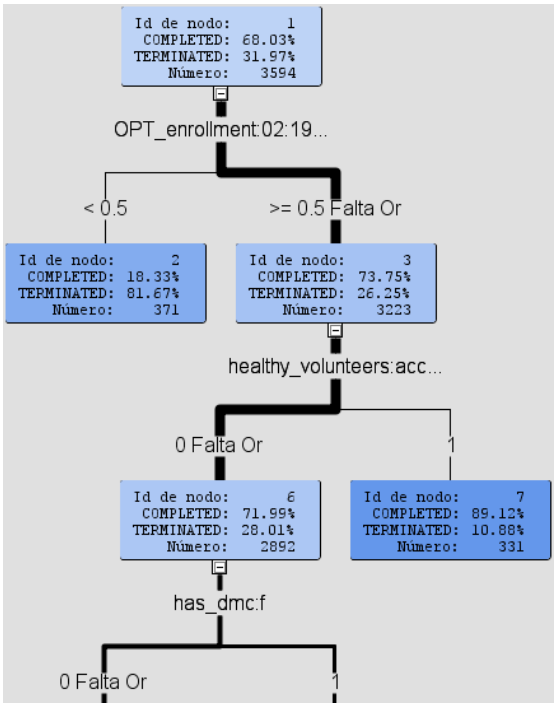


Figura 5.11: Resultados árbol de clasificación (fragmento)

con respecto al resto de variables. Por tanto, podríamos decir que es una variable predominante dentro del conjunto. No obstante, tendríamos que tener en cuenta todos los parámetros y ajustes que hemos aplicado para construir el árbol, dado que si cambiamos uno de ellos podríamos obtener resultados totalmente distintos. Es por ello que realizaremos otro tipo de técnicas.

Importancia de la variable

Nombre de la variable	Etiqueta	Número de reglas de división	Importancia
TI_OPT_enrollment2	OPT_enrollment:02:19.5-high, MISSING	1	1.0000
TI_healthy_volunteers1	healthy_volunteers:accepts healthy volun	1	0.2921
TI_has_dmc1	has_dmc:f	1	0.2816
TI_REP_phase1	REP_phase:phase3	1	0.2113
TI_OPT_minimum_age4	OPT_minimum_age:04:18067.5-high	1	0.1850
TI_REP_region4	REP_region:north_america	1	0.1817
TI_REP_masking2	REP_masking:none	1	0.0991

Figura 5.12: Árbol de clasificación. Importancia de las variables

5.4. *Bagging*

Como hemos visto en el apartado 3.3.1.1, la técnica *bagging* se basa en árboles de decisión. En *SAS Base* podemos utilizar el procedimiento `hpforest`

para crear modelos de *bagging*, y además como veremos en apartados posteriores, modelos de *random forest*. De forma análoga a los modelos de redes, en este apartado vamos a estudiar cómo se ajustan los modelos de *bagging* a nuestros datos. Para ello, iremos modificando y ajustando parámetros tales como el tamaño mínimo de observaciones en las hojas finales, la profundidad máxima del árbol, el *p-valor* para creación de nuevos nodos o el porcentaje del conjunto de datos *train* para la construcción de los árboles, entre otros.

En primer lugar, estudiaremos el número óptimo de observaciones que ha de haber en las hojas. Para ello, realizaremos un bucle de 17 iteraciones comenzando con 5 observaciones e incrementando de 4 en 4. Para este proceso aplicaremos la técnica de validación cruzada repetida haciendo 5 grupos y 22 repeticiones. Todos los árboles creados son binarios. Además, hemos establecido el valor 0,01 a la constante *p-valor*. En este proceso se construirán 200 árboles, es decir, se harán 200 iteraciones del modelo y cada árbol se construirá utilizando el 80 % del total de observaciones.

Hemos realizado tres pruebas con estas mismas características de modelos, pero variando el valor de profundidad máxima de los árboles. En las Figuras 5.13, 5.14 y 5.15, podemos ver los resultados obtenidos para árboles a los que se les ha establecido un nivel máximo de profundidad de 5, 10 y 15, respectivamente.

A pesar de que hay valores muy distintos en los resultados, podemos apreciar que en los tres casos cuando el árbol tiene un máximo de 41 y 45 observaciones, la tasa de fallos aumenta. En los tres diagramas hemos enmarcado en color rojo los tres modelos que consideramos que tienen un mejor comportamiento en cuanto a sesgo y varianza. Debemos destacar que en el diagrama de la Figura 5.13 hemos seleccionado el modelo 25 en lugar del 5, a pesar de que este último presenta mejores resultados. Esto es debido a que en general árboles con pocas observaciones en sus hojas dan lugar a sobreajuste. Esto ocurre porque tienen pocos datos para estimar los valores. También hemos tenido en cuenta que tener un número mayor de observaciones puede producir un aumento en el sesgo. Por tanto, haremos otras pruebas para evaluar el comportamiento de estos modelos seleccionados.

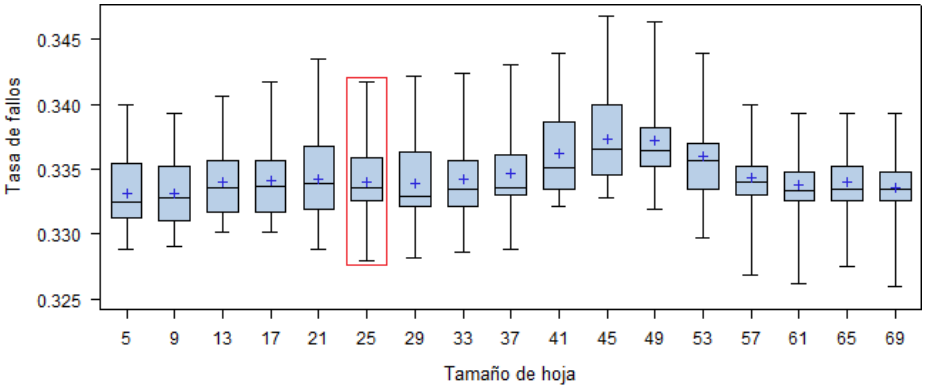


Figura 5.13: Estudio del número de observaciones por hojas. Profundidad 5

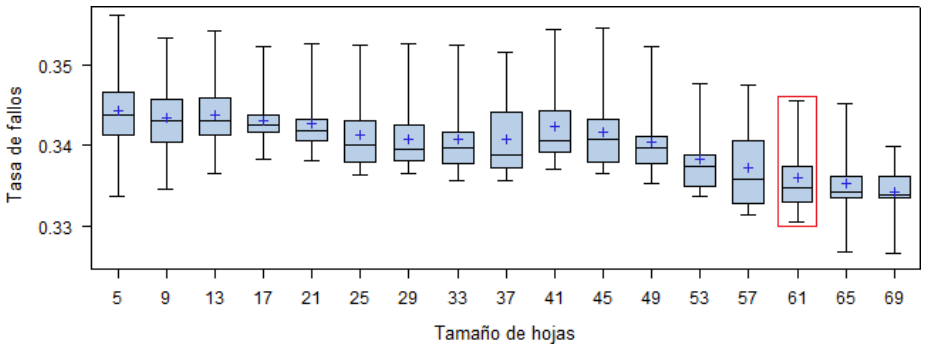


Figura 5.14: Estudio del número de observaciones por hojas. Profundidad 10

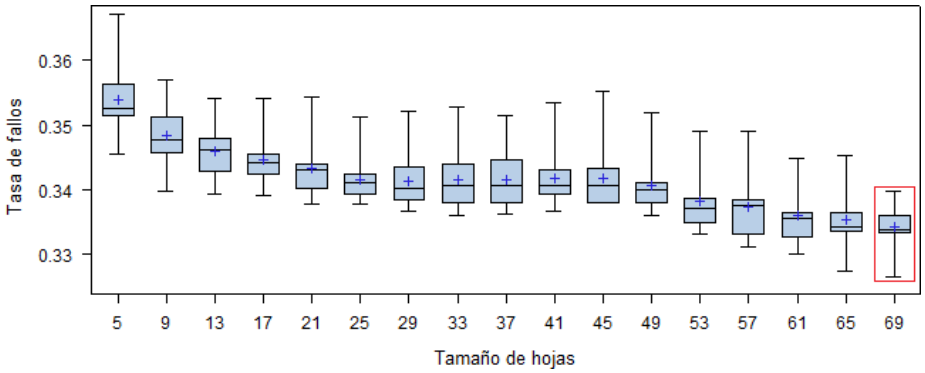


Figura 5.15: Estudio del número de observaciones por hojas. Profundidad 15

Dado que la principal característica de los modelos *bagging* es la cantidad de observaciones que se utilizan para crear los árboles, la siguiente prueba que realizaremos es analizar cómo se comporta el modelo *bagging* a medida que cambiamos el porcentaje de la muestra a utilizar.

Utilizando los tres modelos seleccionados anteriormente y aplicando validación cruzada repetida, hemos creado modelos usando el 50, 60, 70, 80 y 90 % de la totalidad de las observaciones.

Observamos en la Figura 5.16 que el modelo 1, con 25 observaciones en las hojas y una profundidad máxima de 15, tiene un comportamiento ligeramente distinto en comparación con el resto. Los modelos 2 y 3 presentan un aumento en el sesgo a medida que aumenta el porcentaje de observaciones. No obstante, dado que los resultados de la varianza son muy parecidos, nos quedaremos con los modelos que han generado un sesgo más bajo (enmarcados en color rojo).

Nuestro próximo análisis será variar el parámetro *p-valor*, ya que hasta ahora hemos generado los modelos usando el valor 0,01. Este valor influye en la creación del árbol. Esto es, si establecemos un valor muy alto se crearán árboles más complejos y si es moderado, árboles más simples. Por tanto, queremos saber cómo se comportan los modelos seleccionados al variar dicho parámetro. Para ello, utilizaremos validación cruzada repetida de 5 grupos y

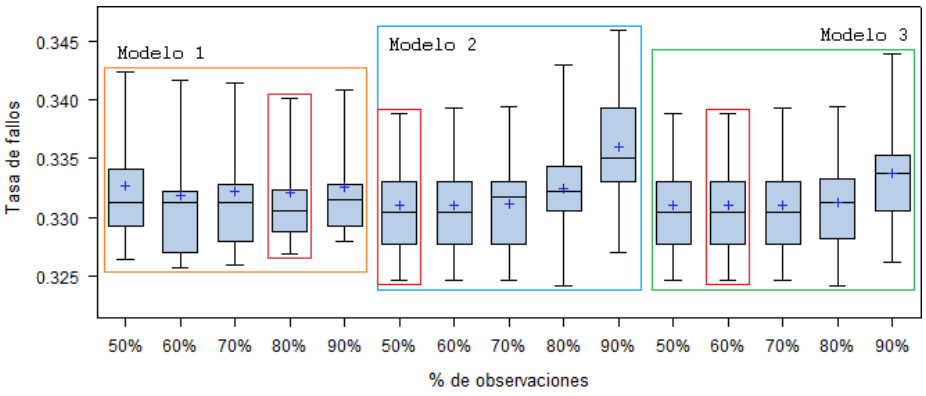


Figura 5.16: Modelos *bagging*: porcentaje de observaciones

22 repeticiones. Mostramos el resultado en la Figura 5.17 y observamos que para los tres modelos, los mejores resultados se obtienen con un *p-valor* de 0,01.

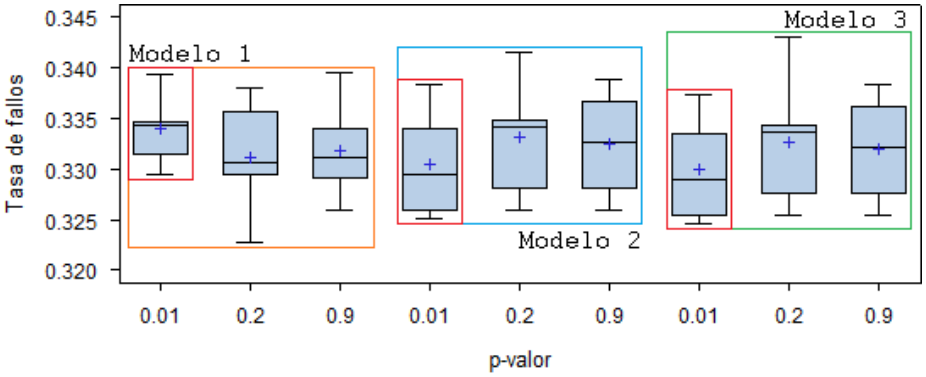


Figura 5.17: Modelos *bagging*: parámetro *p-valor*

En este apartado no vamos a seleccionar un único modelo ganador puesto que esta técnica de *machine learning* tiene una estrecha relación con otras que veremos a continuación. Por tanto, y por ahora, nos quedaremos con los tres modelos generados que contienen las características que mostramos en la Tabla 5.2.

	Modelo 1	**Modelo 2**	**Modelo 3**
Nº de observaciones en hojas	25	61	69
Profundidad máxima	5	10	15
Porcentaje de observaciones	80 %	50 %	60 %
p-valor	0,01	0,01	0,01

Tabla 5.2: Características modelos *bagging*

5.5. *Random forest*

Como hemos visto en el apartado 3.3.1.2, esta técnica es una extensión de *bagging*, o dicho de otra manera: *bagging* es un caso particular de *random forest*.

Recordemos que esta técnica consiste en incorporar aleatoriedad en las variables *input* utilizadas para segmentar cada nodo del árbol. Por tanto, la ventaja de esta técnica es que se evita que haya variables dominantes y se aprovecha mejor la información del resto de variables del conjunto. De esta forma nos aseguramos de que los árboles sean muy diferentes y capten sutilezas importantes, evitando así problemas de sobreajuste.

El presente apartado lo dedicaremos a crear diversos modelos de *random forest*. Para ello, tomaremos los modelos con las mismas combinaciones de parámetros generados en el apartado anterior (ver Tabla 5.2).

Aplicando nuevamente validación cruzada repetida de 5 grupos y 22 repeticiones, crearemos modelos *random forest* en los que variaremos el número de variables de entrada a utilizar. Teniendo en cuenta que nuestro conjunto de datos tiene 18 variables, crearemos un bucle de 16 iteraciones en el que se seleccionarán entre 3 y 18 variables. El modelo que contiene 18 variables supone por tanto un modelo *bagging*. No obstante, lo hemos añadido para comparar sus resultados con los modelos de *random forest*.

En las Figuras 5.18, 5.19 y 5.20 se muestran los resultados para los modelos 1, 2 y 3, respectivamente. Todas las pruebas coinciden en que modelos con pocas variables generan mejores resultados en cuanto a la tasa de fallos. Sin embargo, observamos que la varianza aumenta, lo que puede generar que el modelo sea más inestable. Para los modelos 2 y 3, observamos que a partir de 10 y hasta el total de variables genera errores muy similares.

De igual forma que en las pruebas anteriores, hemos enmarcado en rojo los modelos que consideramos que han presentado mejores resultados en cuanto a la tasa de fallos y a la variabilidad.

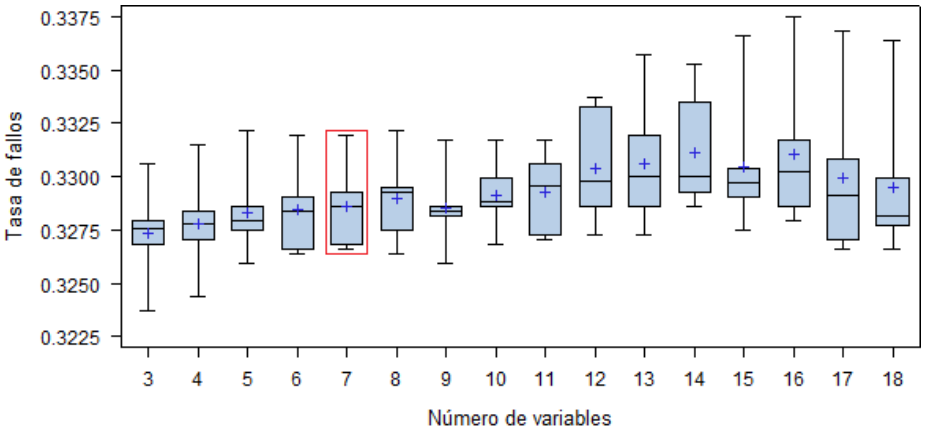


Figura 5.18: Modelo 1: Estudio de la variación del número de variables

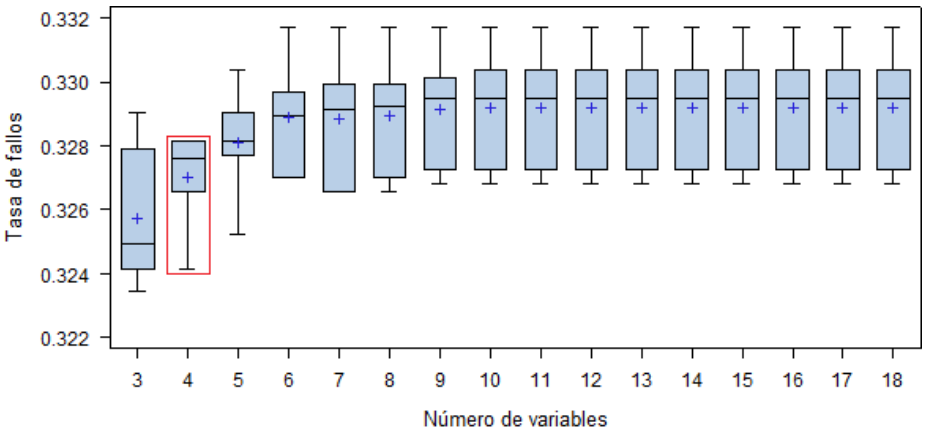


Figura 5.19: Modelo 2: Estudio de la variación del número de variables

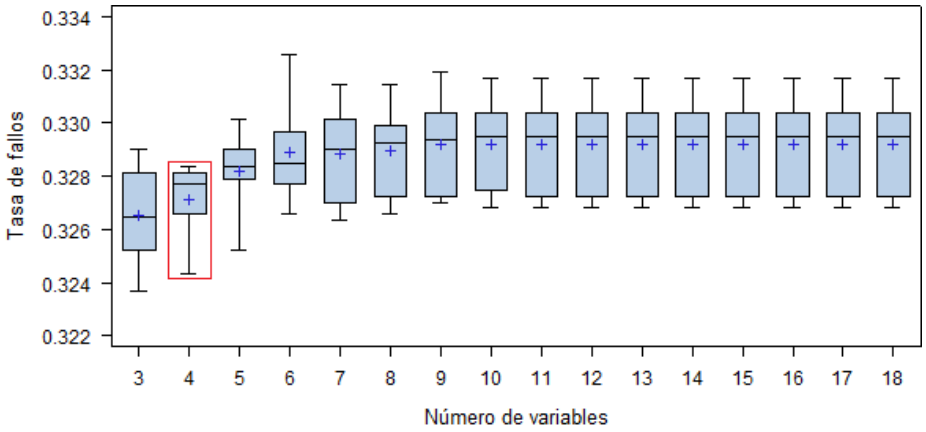


Figura 5.20: Modelo 3: Estudio de la variación del número de variables

Llegados a este punto, debemos seleccionar el mejor modelo de *random forest*. Para ello realizaremos un último estudio con los tres modelos distintos. Este estudio se realizará utilizando validación cruzada repetida, y para hacerlo más exhaustivo utilizaremos 10 grupos con 22 repeticiones cada uno.

En la Figura 5.21 se muestran los resultados y el modelo seleccionado como ganador.

5.6. *Gradient boosting*

Como ya sabemos, a diferencia de las dos técnicas anteriores, *gradient boosting* se trata de un algoritmo iterativo. En cada iteración, es decir, en cada nuevo árbol que se cree, éste utilizará los errores del anterior para mejorar su error de predicción. En este caso interviene un parámetro llamado *shrinkage*, que se refiere a la tasa de aprendizaje que gradúa cuánto se corrige el error en cada iteración. Cuando se establece un valor reducido de *shrinkage* es

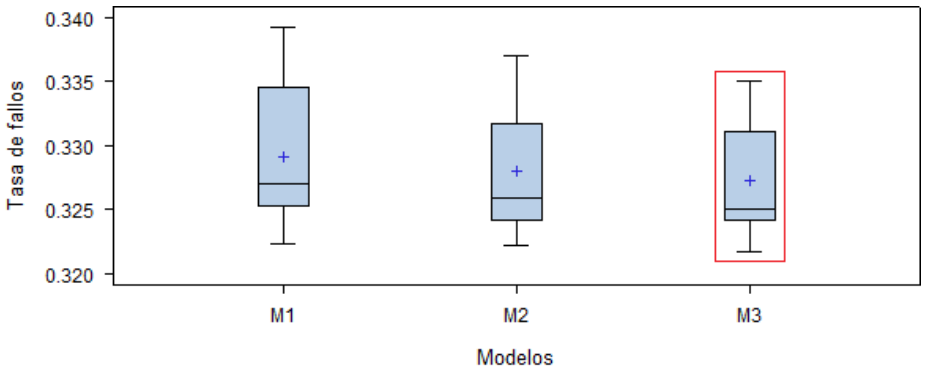


Figura 5.21: Modelos *random forest*

posible que se necesiten más iteraciones, mientras que para valores grandes el número de iteraciones necesarias será menor.

Para crear modelos de *grandient boosting* en *SAS Base* utilizaremos el procedimiento `treeboost`. En primer lugar, vamos a hacer variaciones del parámetro *shrinkage* con los valores 0,001, 0,01, 0,03, 0,05, 0,1 y 0,3 para modelos con 50, 100 y 300 iteraciones. Esto se ha realizado mediante validación cruzada repetida donde se ha hecho una división en 5 grupos con 22 repeticiones. El resto de parámetros lo hemos establecido con el ajuste generado en el mejor modelo *random forest*. Observamos los resultados en la Figura 5.22.

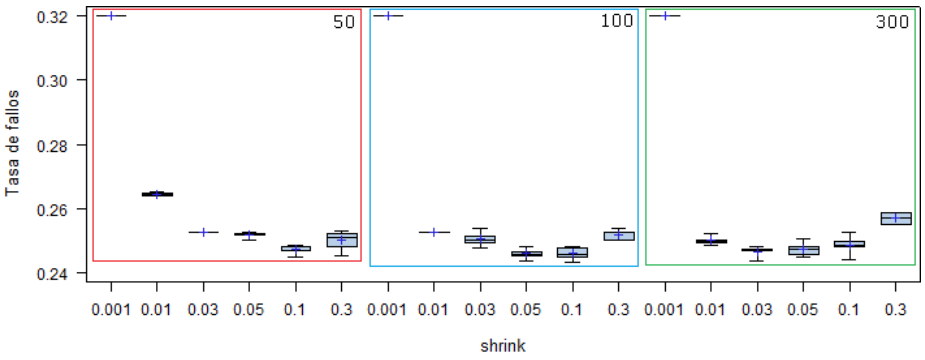


Figura 5.22: Resultados variaciones parámetro *shrinkage*

Rehacemos el gráfico eliminando aquellos modelos que han generado una tasa de fallos más alta para, de esta forma, poder visualizar mejor los resultados. Podemos ver en la Figura 5.23 que hemos seleccionado como mejor modelo aquel con un valor de *shrinkage* de 0,05 y 100 iteraciones. En comparación al resto de resultados, este modelo seleccionado presenta uno de los errores más bajo. Además, en cuanto a su varianza parece ser más estable que el resto.

Las pruebas anteriores las hemos realizado estableciendo el mismo valor de los parámetros del modelo *random forest*. Ahora bien, vamos a realizar un estudio más. Teniendo en cuenta el valor de *shrinkage* y el número de iteraciones del modelo, variamos el número de observaciones por hoja. Para

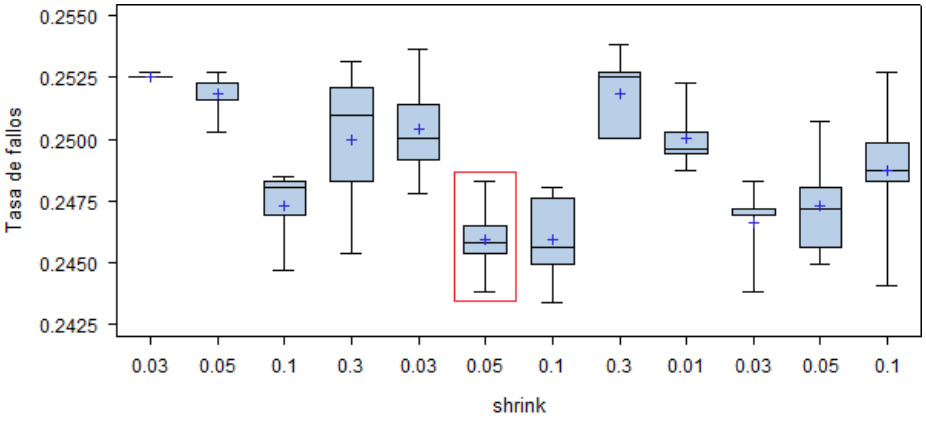


Figura 5.23: Resultados variaciones parámetro *shrinkage* (ampliado)

ello, de forma análoga a como se hizo en el estudio de *bagging*, haremos un bucle de 17 iteraciones comenzando con 5 observaciones e incrementando de 4 en 4. Para este proceso, aplicaremos la técnica de validación cruzada repetida haciendo 5 grupos y 22 repeticiones.

Observamos los resultados en la Figura 5.24. A medida que aumenta el número de observaciones en las hojas finales de los árboles, disminuye el error del modelo. Además, podemos apreciar que la varianza también disminuye ligeramente.

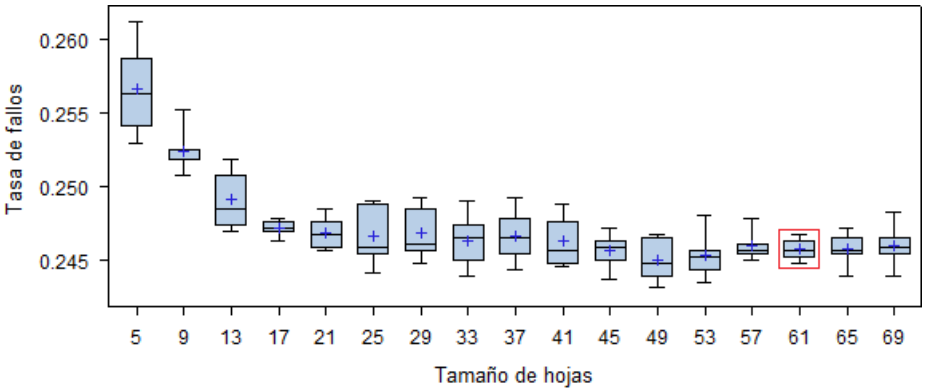


Figura 5.24: Número de observaciones por hojas

Hemos de destacar que el modelo 61 (con 61 observaciones) ha presentado mejores resultados que el modelo 69 que usamos en la prueba anterior. Por tanto, vamos a seleccionar dicho modelo como ganador.

5.7. *Support vector machine*

De forma análoga a todas las pruebas que hemos ido haciendo durante el desarrollo del presente trabajo, crearemos modelos de *support vector machi-*

ne. Para crear modelos SVM en *SAS Base* haremos uso del `proc hpsvm`.

De las tres ideas expuestas en el apartado 3.4 nos basaremos en la idea del *kernel*. Crearemos modelos en los que iremos variando el tipo de *kernel* utilizado, el parámetro de penalización C, el parámetro γ y los grados del polinomio d, entre otros. Además, para el proceso de validación cruzada repetida estableceremos 5 grupos y 22 repeticiones.

En la Tabla 5.3 podemos ver las configuraciones que hemos establecido para cada uno de los modelos.

	***kernel* lineal**	
Mod.	C	
1	0,05	
2	0,1	
3	0,5	
4	1	
5	5	
6	10	

	***kernel* polinomial**	
Mod.	C	d
7	0,05	2
8	0,1	3
9	0,5	2
10	1	3
11	5	2
12	10	3

	***kernel* RBF**	
Mod.	C	$1/\gamma$
13	0,05	1
14	0,1	0,5
15	0,5	1
16	1	5
17	5	1

Tabla 5.3: Configuración de modelos SVM

En la Figura 5.25 observamos los resultados generados por cada uno de los modelos. En los modelos con *kernel* lineal, podemos observar que hay variaciones en la varianza, aunque con respecto a los fallos casi todos se mantienen en la media. Con respecto a los modelos polinomial y gaussiano, observamos que a medida que se aumenta el coeficiente de penalización aumenta el sesgo del modelo. Esto podría deberse a que se sobreajustan los mismos. Destacamos el modelo número 14 que ha generado mejores resultados en cuanto a sesgo y varianza que el resto. No obstante, de cada uno de los *kernels* hemos seleccionado el mejor modelo (enmarcados en color rojo).

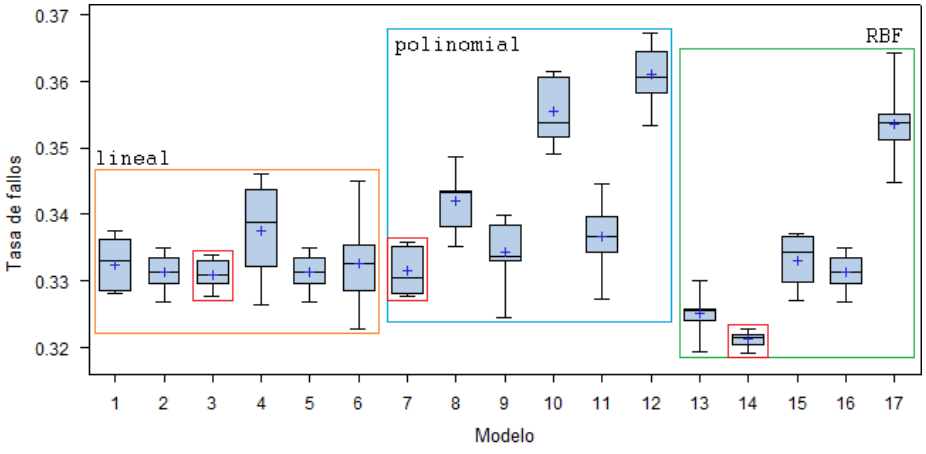


Figura 5.25: Resultados modelos SVM

Una vez generados y seleccionados los modelos de *support vector machine* que mejor se ajustan a nuestros datos, podemos dar por finalizada la fase *model*.

Capítulo 6

Resultados

"El mundo exige resultados. No le cuentes
a otros tus dolores del parto.
Muéstrales al niño."
— Indira Gandhi

En este capítulo desarrollaremos la última fase de la metodología SEMMA, es decir, la fase de comprobación de la calidad de los modelos (*access*). Haremos una comparación de todos los modelos generados, analizaremos sus resultados y seleccionaremos el mejor de ellos.

6.1. Comparación de modelos

En la Tabla 6.1 mostramos un resumen de las características de los modelos seleccionados.

Nótese que hemos incluido un modelo de *bagging*. En el apartado 5.4 comentamos que no seleccionaríamos el modelo ganador *bagging* puesto que se mejoraría la técnica usando *random forest*. No obstante, para realizar esta comparación vamos a seleccionar uno a modo ilustrativo.

Técnica	**Modelo**	**Características**
Regresión	1	- Método: *stepwise*
Red neuronal	2	- Número de nodos de la capa: 7 - Algoritmo optimización: bprop *learn*=0,1 *mom*=0,1 - Función de activación: seno (sin)
Bagging	3	- Número de variables: 18 - Porcentaje de observaciones: 50 % - Máximo de observaciones por hoja: 61 - *p-valor*: 0,01 - Aridad máxima del árbol: 2 - Profundidad máxima: 10 - Número de iteraciones: 200

Continúa en la siguiente página...

Técnica	Modelo	Características
Random forest	4	- Número de variables: 4 - Porcentaje de observaciones: 60 % - Máximo de observaciones por hoja: 69 - *p-valor*: 0,01 - Aridad máxima del árbol: 2
Gradient boosting	5	- Máximo de observaciones por hoja: 61 - Valor *shrink*: 0,05 - Aridad máxima del árbol: 2 - Profundidad máxima: 15 - Número de iteraciones: 100
Support vector machine	6	- *kernel*: lineal - penalización C: 0,5
	7	- *kernel*: polinomial - penalización C: 0,05 - grado d: 2
	8	- *kernel*: RBF - penalización C: 0,1 - $1/\gamma$: 0,5

Tabla 6.1: Características modelos ganadores

Para realizar este estudio comparativo de modelos volveremos a ejecutarlos ajustando los parámetros correspondientes y empleando la técnica de validación cruzada repetida con 10 grupos y 22 repeticiones.

En la Figura 6.1 podemos visualizar los resultados. Destacan los modelos regresión logística, redes neuronales y *gradient boosting*, que han obtenido resultados sensiblemente mejores que el resto.

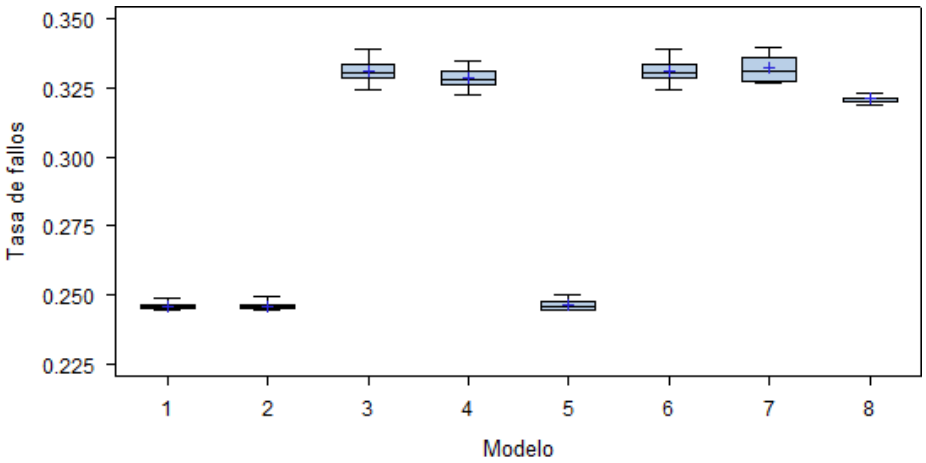


Figura 6.1: Comparación de modelos

Rehacemos el gráfico para visualizar con mejor detalle los resultados (ver Figura 6.2). Observamos que el modelo de regresión logística presenta menor tasa de fallos y una varianza ligeramente menor que el resto. Por tanto, podemos escoger el modelo de regresión logística como nuestro modelo ganador.

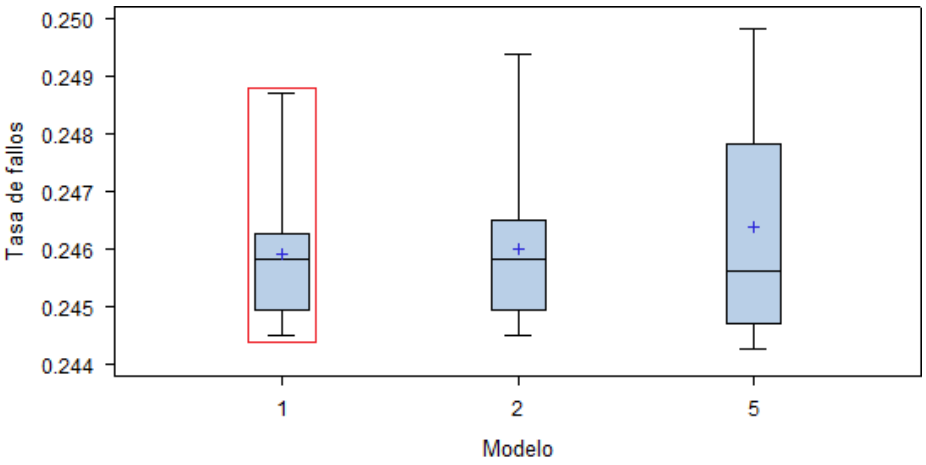


Figura 6.2: Comparación de modelos (ampliado)

La ventaja de tener un modelo de regresión es que podemos explicar en qué se basa el algoritmo para sacar dichas conclusiones. Esto es, podemos interpretar el valor de los parámetros del modelo. Recordemos que el modelo de regresión logística persigue intentar modelizar la probabilidad la categoría de interés. Para ello, se estima la probabilidad de que ocurra dicho suceso como:

$$p_1 = P(Y = 1|x_1, x_2, \ldots, x_m) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m)}}$$

Por lo que podemos definir que:

$$\log\left(\frac{p_1}{1 - p_1}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m$$

Este término $\log\left(\frac{p_1}{1-p_1}\right)$ se conoce como *logit* y es el logaritmo del *odds ratio*. Esto significa que representa el cociente entre la probabilidad de que ocurra el suceso y la probabilidad de que no ocurra. Por tanto,

$$\begin{aligned}
\log(odds) &= -1{,}206 + 0{,}288 \cdot TI_agency_class2 + 0{,}589 \cdot TI_allocation1 \\
&+ 3{,}018 \cdot TI_OPT_enrollment2 + 0{,}624 \cdot TI_enrollment_type1 \\
&+ 0{,}534 \cdot TI_has_dmc1 - 0{,}922 \cdot TI_healthy_volunteers1 \\
&+ 1{,}1065 \cdot TI_intervention_model1 + 0{,}867 \cdot TI_intervention_type1 \\
&+ 0{,}251 \cdot TI_intervention_type3 + 0{,}502 \cdot TI_intervention_type4 \\
&- 0{,}400 \cdot TI_masking2 - 0{,}458 \cdot TI_OPT_minimun_age2 \\
&- 1{,}009 \cdot TI_OPT_minimun_age4 + 0{,}0013 \cdot SQR_n_countries \\
&- 0{,}363 \cdot TI_phase1 - 0{,}546 \cdot TI_region3 - 0{,}742 \cdot TI_region4 \\
&- 0{,}349 \cdot TI_region5
\end{aligned}$$

Para interpretar los parámetros debemos hablar de su exponencial. Si tomamos el ejemplo de la variable `n_countries` a la que se le ha aplicado una

transformación de la raíz cuadrada, observamos que el parámetro β es positivo $(0{,}0013 \cdot SQR\_n\_countries)$. Esto significa que, al aumentar el número de países en los que se realiza el ensayo, la probabilidad será de 0,23 de que nuestra categoría de interés sea positiva. Es decir, que el ensayo se termine sus cuatro fases.

En contrapartida, si tomamos un ejemplo con parámetro negativo como $-0{,}922 \cdot TI\_healthy\_volunteers1$, nos indica que la probabilidad de que un ensayo se culmine correctamente disminuye un 0,10 % si se permite que participen personas sanas.

A continuación, mostramos el valor de la matriz de confusión obtenida para el modelo de regresión logística. Esta matriz se ha obtenido utilizando el conjunto *test* que contiene el 20 % del total de observaciones. En la tabla 6.2 podemos observar que el porcentaje de verdaderos positivos ha sido de 64,63 % y el de verdaderos negativos de 10,68 %. Además, observamos que ha habido un total de 222 observaciones mal clasificadas.

		Predicción		
		terminated	***completed***	**Total**
Real	***terminated***	96 10,68 %	43 4,78 %	139 15,46 %
	completed	179 19.91 %	581 64,63 %	760 84,54 %
	Total	275 30,59 %	624 69,41 %	899 100 %

Tabla 6.2: Matriz de confusión para el modelo ganador

Con los resultados anteriores, podemos determinar el valor de la curva ROC. Podemos observar en la Figura 6.3 que tiene un valor de 0,77. Este valor se considera regular, no obstante recordemos que el modelo de regresión logística ha sido el modelo con menor sesgo y variabilidad entre todos los generados.

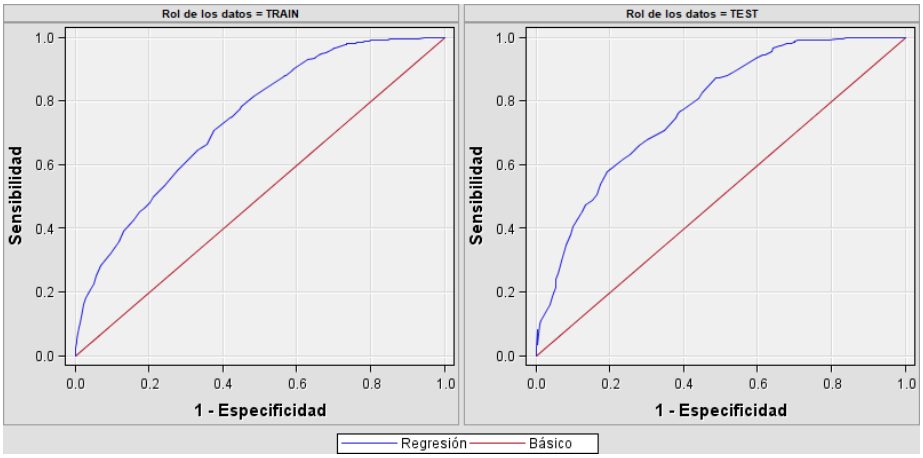


Figura 6.3: Curva ROC

El punto de corte óptimo también podemos obtenerlo a partir de los valores de la especificidad y la sensibilidad. En la Figura 6.4 podemos observar dicho punto de corte, aproximadamente 0,7.

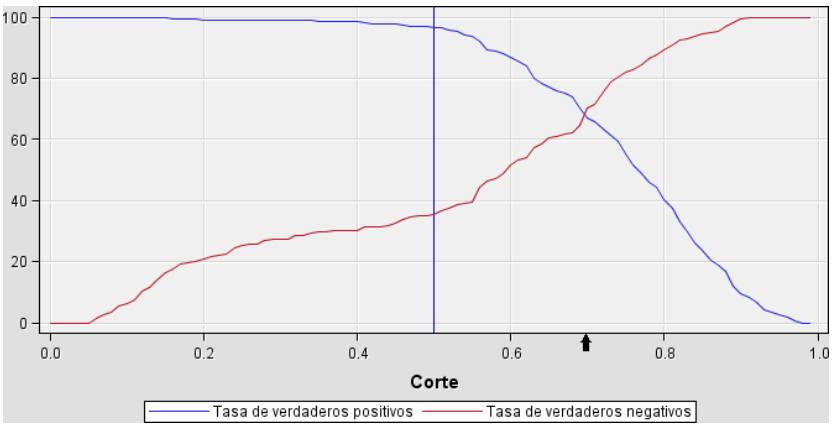


Figura 6.4: Punto de corte óptimo

6.2. Modelos de ensamblado

En el desarrollo del presente trabajo, hemos podido crear y analizar modelos basados en diversas técnicas de *machine learning* como las redes neuronales, que tienen una filosofía de función continua o función de aproximación; algoritmos basados en árboles, que se basan en buscar o definir regiones; y *support vector machine*, que suponen un técnica geométrica. Todos estos algoritmos aprovechan y utilizan al máximo sus técnicas para llegar a realizar predicciones. Por tanto, la cuestión a resolver en este apartado es si se podrían aprovechar las diversas maneras de resolver el problema que tienen estos algoritmos para llegar a una posición común. Esto nos lleva a los métodos de ensamblado o *ensemble*.

Como hemos comentado en apartados anteriores, los métodos *ensemble* consisten en la construcción de predicciones a partir de la combinación de varios modelos. Es decir, utilizan las predicciones de diferentes algoritmos para predecir la variable dependiente. Esta técnica la utilizaremos para examinar si es posible reducir aún más el sesgo de los modelos obtenidos hasta ahora.

En la Tabla 6.3[1] podemos ver las descripciones de los modelos *ensamble*. En el modelo 29 (ponderaciones) usamos todos los modelos, pero hemos intentado establecer un *peso* y dar más valor a los modelos que obtuvieron mejores resultados (regresión, redes y *gradient boosting*). Para este estudio usaremos validación cruzada repetida de 10 grupos y 40 iteraciones. En cuanto a los que incluyen modelos de *support vector machine*, hemos seleccionado *kernel* RBF puesto que fue el que mejor resultado obtuvo.

[1] Aclaraciones de la Tabla: LOG (regresión logística), RED (redes neuronales), RF (*random forest*), GB (*gradient boosting*), SVM (*support vector machine*)

Modelo	Descripción
1	LOG
2	RED
3	RF
4	GB
5	SVM
6	LOG + RED
7	LOG + RF
8	LOG + GB
9	LOG + SVM
10	RED + RF
11	RED + GB
12	RED + SVM
13	RF + GB
14	RF + SVM
15	GB + SVM

Modelo	Descripción
16	LOG + RED + RF
17	LOG + RED + GB
18	LOG + RED + SVM
19	LOG + RF + GB
20	LOG + RF + SVM
21	LOG + GB + SVM
22	RED + RF + GB
23	RED + RF + SVM
24	RF + GB + SVM
25	LOG + RED + RF + GB
26	LOG + RED + RF + SVM
27	RED + RF + GB + SVM
28	todos los modelos
29	ponderaciones

Tabla 6.3: Descripción modelos *ensamble*

Los resultados se muestran en las Figuras 6.5 y 6.6. Nos llama especialmente la atención que casi todos los modelos *ensamble* que incluyen *support vector machine* (modelos 5, 9, 12, 14, 15, 20, 23 y 24) obtienen un sesgo más alto. Lo mismo ocurre en algunos que incluyen *random forest* (3, 7, 10, 13 y 14) aunque el error no es tan alto.

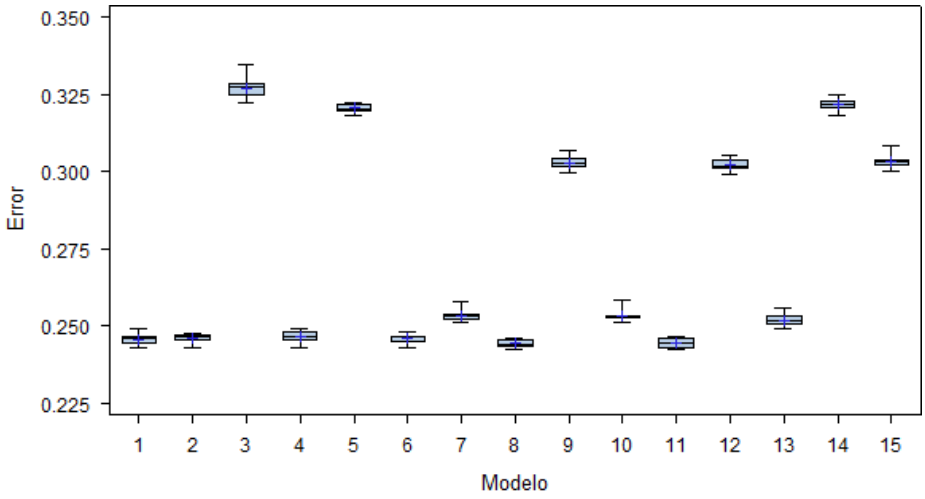


Figura 6.5: Resultados modelos *ensamble* (Parte 1)

Volvemos a hacer el gráfico eliminando aquellos modelos con peores resultados (ver Figura 6.7). Los modelos 1, 2 y 4 representan los modelos de regresión logística, redes neuronales y *grandient boosting*, respectivamente. Como hemos variado la semilla y el número de repeticiones observamos que el modelo de regresión ha presentado una varianza un poco más alta en comparación al estudio del apartado anterior. Sin embargo, la media de error de los tres modelos se mantiene.

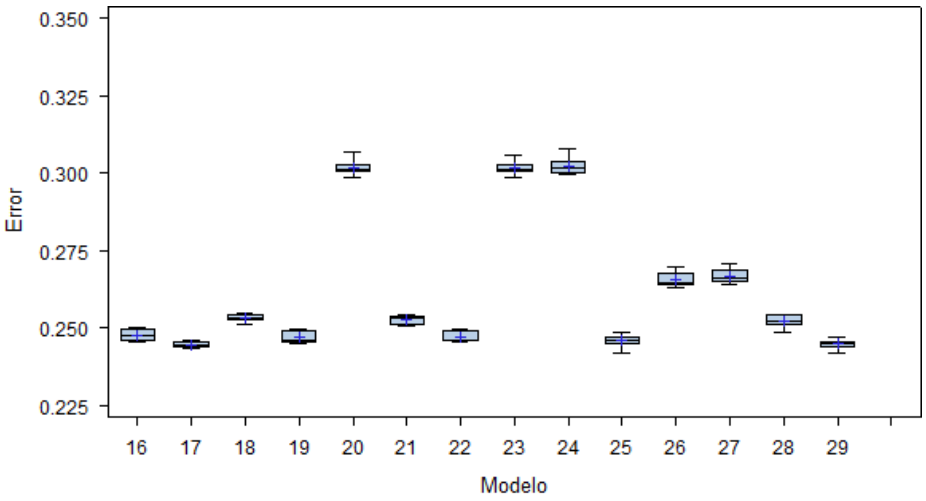


Figura 6.6: Resultados modelos *ensamble* (Parte 2)

Observamos que hay dos modelos que han presentado los errores más bajos, 8 y 11, que son el resultado de combinar regresión y redes con *gradient boosting*. Esta combinación ha mejorado los resultados de predicción. No obstante, destacamos que la combinación de estos tres modelos (LOG + RED + GB) modelo 17 en la Tabla 6.3, presenta un sesgo ligeramente mayor pero reduce la varianza. Es por ello que hemos decidido seleccionar este como modelo ganador.

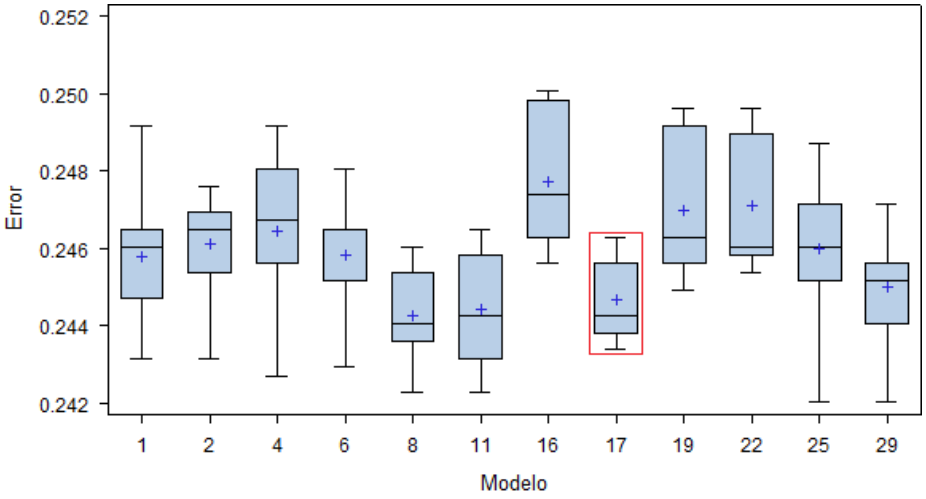


Figura 6.7: Resultado modelos *ensamble* (ampliado)

Capítulo 7

Conclusiones y Trabajo Futuro

"Nunca se es demasiado viejo para establecer
un nuevo objetivo, o para soñar
un nuevo sueño"
— CS Lewis

Poniendo punto final al desarrollo del proyecto llegamos a una serie de conclusiones e ideas que se podrían tener en cuenta para futuros desarrollos.

En primer lugar, podemos decir que, dado que las variables del conjunto no tenían una buena relación con la variable objetivo, ha sido necesario aplicar una transformación. Cuando se realizó la misma hubo una notable diferencia con respecto a la variable `enrollment` (número total estimado o número real de individuos inscritos en el estudio) que pasó de tener un valor *V de Cramer* de 0,014 a 0,367.

Al realizarse este cambio, pudimos apreciar que dicha variable ha sido la más influyente en los modelos generados. Esto se puede apreciar en el modelo de árbol de clasificación donde aparece como primera en el *ranking* de importancia de variables; y además, en el modelo de regresión seleccionado como ganador. Esta variable ha tenido un coeficiente β de 3,018. Esto significa que dicha variable influye positivamente en la probabilidad de que ocurra el evento, es decir, que el ensayo culmine correctamente.

Por otra parte, de todos los modelos generados utilizando diversas técnicas de *machine learning*, hemos podido comprobar que el de regresión logística ha obtenido los mejores resultados. Con esto podemos decir que a pesar de existir modelos extremadamente potentes que por lo general generan buenos resultados, no podemos subestimar modelos más sencillos como los de regresión, ya que los resultados obtenidos siempre dependen de nuestros datos.

También hemos podido observar que combinando diversos modelos que han generado buenos resultados podemos reducir aún más la tasa de fallos.

Trabajo Futuro

Como trabajo futuro proponemos realizar este mismo estudio en otros entornos de desarrollo utilizando otros lenguajes de programación y librerías. De esta forma, podremos probar otros modelos potentes como *xgboost* (Chen y Guestrin, 2016).

Además, se podrían incluir otras variables de la base de datos de la *AACT* para analizar su comportamiento. Por ejemplo, teniendo en cuenta que la variable más importante ha sido el número de participantes involucrados en el estudio, nos planteamos la siguiente pregunta: ¿qué ocurriría con estudios multicéntricos con pocos pacientes en cada centro y que juntan un gran número de sujetos? Para ello, deberíamos considerar incluir en el conjunto de datos la variable que indique el número de centros en los que se ha realizado el ensayo clínico. De esta forma podremos conocer cuántos participantes hay de media en cada centro y ver si este parámetro también influye en los resultados.

Apéndice A

Anexo

Todo el código desarrollado en el presente trabajo se puede observar en `https://github.com/eliiags/TFM-MineriaDatos`.

Bibliografía

Y así, del mucho leer y del poco dormir, se le secó el celebro de manera que vino a perder el juicio.

Miguel de Cervantes Saavedra

AGUIRRE, C. ML Part 1: Introducción a los arboles de decisión. 2019. Disponible en `https://www.cristobal-aguirre.com/arboles-de-decision` (último acceso, Agosto, 2020).

ALPAYDIN, E. *Introduction to machine learning*. MIT press, 2020.

ALUJA, T. La minería de datos, entre la estadística y la inteligencia artificial. *Qüestiió: quaderns déstadística i investigació operativa*, páginas 479–498, 2001.

AMAT RODRIGO, J. Máquinas de Vector Soporte (Support Vector Machines, SVMs). 2015. Disponible en `https://www.cienciadedatos.net/documentos/34_maquinas_de_vector_soporte_support_vector_machines` (último acceso, Agosto, 2020).

AMAT RODRIGO, J. Árboles de predicción: random forest, gradient boosting y C5.0. 2019. Disponible en `https://www.cienciadedatos.net/documentos/33_arboles_de_prediccion_bagging_random_forest_boosting#Gradient_Boosting` (último acceso, Agosto, 2020).

ARTALEJO, M. R., CALERO, P. A. G. y MARTÍN, M. A. G. *Estructuras de datos. Un enfoque moderno*. Editorial Complutense, 2011.

AZEVEDO, A. y SANTOS, M. KDD, SEMMA and CRISP-DM: A parallel overview. En *IADIS European Conference Data Mining*, páginas 182–185. 2008.

BOSER, B. E., GUYON, I. M. y VAPNIK, V. N. A training algorithm for optimal margin classifiers. En *Proceedings of the fifth annual workshop on Computational learning theory*, páginas 144–152. 1992.

BREIMAN, L. Bagging predictors. *Machine learning*, vol. 24(2), páginas 123–140, 1996.

BREIMAN, L. Random forests. *Machine learning*, vol. 45(1), páginas 5–32, 2001.

CALIFF, R. M., ZARIN, D. A., KRAMER, J. M., SHERMAN, R. E., ABERLE, L. H. y TASNEEM, A. Characteristics of clinical trials registered in clinicaltrials. gov, 2007-2010. *Jama*, vol. 307(17), páginas 1838–1847, 2012.

CALVIÑO MARTÍNEZ, A. "Técnicas y Metodología de la Minería de Datos". Facultad de Estudios Estadísticos, Universidad Complutense de Madrid, 2019.

CHEN, T. y GUESTRIN, C. Xgboost: A scalable tree boosting system. En *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, páginas 785–794. 2016.

CORTES, C. y VAPNIK, V. Support-vector networks. *Machine learning*, vol. 20(3), páginas 273–297, 1995.

DIETTERICH, T. G. ET AL. Ensemble learning. *The handbook of brain theory and neural networks*, vol. 2, páginas 110–125, 2002.

FRIEDMAN, J. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, vol. 29, páginas 1189–1232, 2001.

HLAING, K. S. y THAW, Y. M. K. K. Applications, Techniques and Trends of Data Mining and Knowledge Discovery Database. *International Journal of Trend in Scientific Research and Development (IJTSRD)*, 2019.

JAIN, A. K., MAO, J. y MOHIUDDIN, K. M. Artificial Neural Networks: A Tutorial. *Computer*, vol. 29(3), página 31–44, 1996. ISSN 0018-9162.

JAKE, F. Machine Learning. 2020. Disponible en `https://www.investopedia.com/terms/m/machine-learning.asp` (último acceso, Agosto, 2020).

KOREN, Y. The Bellkor solution to the Netflix Grand Prize. *Netflix prize documentation*, vol. 81(2009), páginas 1–10, 2009.

KOWALCZYK, A. Support vector machines succinctly. *Syncfusion Inc*, 2017.

KUHN, M., JOHNSON, K. ET AL. *Applied Predictive Modeling*, vol. 26. Springer, 2013.

MATIGNON, R. *Neural Network modeling using SAS Enterprise Miner*. AuthorHouse, 2005.

ORELLANA-ALVEAR, J. Árboles de decisión y Random Forest. 2018. Disponible en `https://bookdown.org/content/2031/` (último acceso, Agosto, 2020).

PALLÁS, J. M. A. y VILLA, J. J. *Métodos de investigación clínica y epidemiológica*. Elsevier, 2019.

PORTELA GARCÍA-MIGUEL, J. "Técnicas de Machine Learning". Facultad de Estudios Estadísticos, Universidad Complutense de Madrid, 2020.

RUSSELL, S. y NORVIG, P. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edición, 2009. ISBN 0136042597.

SÁNCHEZ-CARO, J. y ABELLÁN, F. Ensayos Clínicos en España. *Aspectos científicos, Bioéticos y Jurídicos. 1ª ed. Granada: Comares*, 2006.

SANCHO CAPARRINI, F. Introducción al Aprendizaje Automático. 2017. Disponible en `http://www.cs.us.es/~fsancho/?e=75` (último acceso, Agosto, 2020).

VAPNIK, V. N. y CHERVONENKIS, A. On a perceptron class. *Automation and Remote Control*, vol. 25, páginas 112–120, 1964.

WARWICK, K. y SHAH, H. Can machines think? a report on turing test experiments at the royal society. *Journal of experimental & Theoretical artificial Intelligence*, vol. 28(6), páginas 989–1007, 2016.

WIRTH, R. y HIPP, J. CRISP-DM: Towards a standard process model for data mining. En *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, páginas 29–39. Springer-Verlag London, UK, 2000.

–¿Qué te parece desto, Sancho? – Dijo Don Quijote –
Bien podrán los encantadores quitarme la ventura,
pero el esfuerzo y el ánimo, será imposible.

Segunda parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes

–Buena está – dijo Sancho –; fírmela vuestra merced.
–No es menester firmarla – dijo Don Quijote–,
sino solamente poner mi rúbrica.

Primera parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes