FACULTAD DE ESTUDIOS ESTADÍSTICOS
MÁSTER EN MINERÍA DE DATOS E INTELIGENCIA DE NEGOCIOS
Curso 2017/2018

Trabajo de Fin de Máster

TÍTULO: Minería de datos como herramienta para optimizar la detección del fraude empresarial

Alumno: Pablo Contreras Núñez

Tutores: Javier Castro Cantalejo / Rosa Espínola Vílchez

Junio de 2018

UNIVERSIDAD COMPLUTENSE
MADRID

Dedicatoria:

A mi tío Fer y a mis abuelos

Muchas gracias a todos los profesores, amigos y familiares que me han acompañado en estos últimos cinco años.

Este trabajo no hubiera sido posible sin todos vosotros.

Índice

1. Introducción

Actualmente vivimos en una sociedad en la que, por muy incomprensible que parezca, cuanto más se desarrolla económicamente, más codiciosa se vuelve. No importa lo que tengamos o lo que hayamos conseguido, siempre queremos más y casi siempre prima la recompensa personal sobre la colectiva. Es precisamente este afán por desear más de lo que se tiene lo que lleva a la gente a cometer **fraude** en muchas ocasiones, siendo esta avaricia la causa más común del fraude. En otras ocasiones, otro motivo que provoca el fraude es el hecho de poder conseguir aquello que los individuos y las empresas no son capaces de alcanzar de forma legítima.

Un ejemplo de ello son los casos de corrupción que se han producido a nivel mundial en los últimos años, destacando los de grandes compañías como Volkswagen, grandes bancos como Lehman Brothers o incluso la FIFA y jugadores profesionales. Sin olvidarnos de los políticos. Mientras que los motivos de cometer fraude para los políticos o jugadores profesionales suele ser la avaricia, en las grandes compañías y empresas normalmente se debe a la presión de los mercados por mejorar los resultados trimestre a trimestre lo que las empuja a presentar resultados no reales, no respetar las reglas del juego vigentes, sobornos… Aunque estos sean los casos más sonados, la realidad es que el fraude también se da a menor escala en el día a día (facturas sin IVA, pequeños autónomos cobrando y pagando en negro…). Sin embargo, sólo los primeros están mal vistos por la sociedad lo que indica una **falta de concienciación** que provoca que sea muy difícil erradicarlo.

El Banco Mundial estima que el coste global del fraude supone un 5% del valor de la economía mundial [1] lo que sin duda lo convierte en unos de los crímenes más importantes de nuestra época. En el **ámbito fiscal**, esto repercute directamente sobre la sociedad puesto que **se reduce el ingreso de dinero en las arcas públicas** lo que conlleva un menor gasto público (sobre todo en educación y sanidad) sobre el que se sustenta el Estado de Bienestar, se generan situaciones de **competencia desleal** y es una gran **injusticia** ya que los defraudadores se benefician igual de toda la infraestructura material y social que mantiene el resto de la sociedad con sus contribuciones. En el **ámbito empresarial**, el coste del fraude va más allá del impacto económico (**disminución de ingresos**). Así, hay otras perdidas como el **daño a la reputación** y sus consecuentes perdidas de negocio y sin ir más lejos, los **costes asociados con la investigación** y la reparación de los problemas que permitieron el fraude. Además, existen una serie de impactos del fraude que son aplicables a cualquier ámbito como son el **aumento de tarifas/impuestos**, **deterioro de la confianza** de los accionistas/socios/población y el hecho de **animar a cometer fraude** a otras personas si no es detectado.

A pesar del importante riesgo que supone el fraude en el mundo empresarial y aunque empiece a existir una creciente preocupación, **muchas empresas no cuentan aún con sistemas o procedimientos de actuación para prevenir y responder ante el fraude** lo que implica una ventaja para el defraudador al facilitarlo o hacerlo posible. El hecho de cometer fraude se vería sensiblemente reducido si el defraudador creyese que va a ser detectado o que el precio a pagar, de detectarse, fuese tan elevado que no compensase el beneficio obtenido.

2. Justificación del proyecto

El problema del fraude no tiene una solución única y lo ideal en estos casos sería poder **evitar el fraude antes de que tuviese lugar**. Dado que esto resulta muy complicado en muchas ocasiones, otro aspecto importante es la **detección del fraude lo antes posible** y su **seguimiento a posteriori** ya que a medida que se van identificando casos y aplicando medidas correctivas para evitarlo, nuevas formas de cometerlo surgen debido a la picaresca de los defraudadores. El presente trabajo pretende ser un **ejemplo metodológico** para ayudar en la lucha contra el fraude que está presente en nuestro día a día.

Sin embargo, existen diversidad de estudios previos que abordan el mismo problema. Por similitud a la metodología que se va a llevar a cabo en este trabajo, destacan, por ejemplo, Altman (1968) pionero en usar un análisis discriminante para clasificar a un conjunto de empresas. Martin (1977) utilizó la regresión logística (logit y probit) para el estudio de la cuestión, evitando algunos inconvenientes metodológicos del análisis discriminante (ausencia de normalidad de las variables, la no interpretación, etc.). Bell *et al.* (1990) usó por primera vez las redes neuronales que son más robustas que las técnicas anteriores, aunque requiere mayor tiempo computacional. Frydman *et al.* (1985), por otro lado, usaron algoritmos de particiones recursivas como los árboles de decisión.

En España, los primeros trabajos sobre el tema aparecen a mediados de los años 80. Los trabajos de Laffarga *et al.* (1985; 1987) pueden considerarse pioneros en nuestro país. El primero aplica ANOVA y análisis discriminante mientras que el segundo incorpora la regresión logística. Gabás (1990) utilizó árboles de decisión y Mar Molinero (1991) o Serrano Cinca (1996) publican modelos de inteligencia artificial (redes neuronales). Más recientemente, destacan Correa *et al.* (2003) y Acosta y Fernández (2007).

Estos son simplemente algunos de los ejemplos más conocidos de toda literatura existente relativa al problema que se va a tratar en el trabajo y constituyen una guía metodológica básica para tratar el problema con garantías.

3. Objetivos principales y secundarios

El **objetivo principal** de este trabajo se centra en intentar maximizar las opciones de recuperar lo defraudado y de, por tanto, **aumentar la efectividad y la rentabilidad de las campañas de inspección** realizadas actualmente por la institución objeto de estudio para detectar el fraude puesto que los resultados actuales no son los deseados. Esto es así ya que utilizan técnicas tradicionales para obtener la información sobre qué usuarios inspeccionar a partir de su base de datos y no consiguen una efectividad acorde a los recursos económicos empleados. Para conseguir este objetivo principal se plantean una serie de **objetivos secundarios**:

- **Procesamiento de los archivos recibidos** con toda la información para su correcta lectura.
- **Análisis exploratorio** de los datos para conocer la información de partida y la realidad de la situación a tratar.

- **Depuración racional** de los datos en base al conocimiento del problema de forma que se conserve la integridad de las variables.
- **Creación de nuevas variables explicativas y variables respuesta** con el objetivo de ayudar a cumplir el siguiente punto.
- **Predecir lo mejor posible la probabilidad de que una observación sea fraudulenta** a través de técnicas estadísticas más modernas como son la *regresión logística*, *redes neuronales*, *random forest, gradient boosting* y *ensamble de modelos* para que pueda ser inspeccionada.
- **Determinar los factores que permitan distinguir** a las observaciones solventes que cumplen con sus obligaciones de los morosos, que las incumplen o se retrasan en su cumplimiento.
- Definir un **plan de acción** sobre las inspecciones a realizar por parte de la institución a partir de las probabilidades de cometer fraude calculadas previamente.
- Realizar un **post - análisis** con los resultados en campo para mejorar los resultados previos.

4. Naturaleza de los datos

Debido a razones de confidencialidad con la empresa cuyos datos serán utilizados en el presente trabajo, los mismos han sido camuflados en una realidad diferente, aunque parecida a la original debido a lo cual en ocasiones se presentarán problemas de interpretación.

En este nuevo contexto ficticio, la unidad de análisis de este trabajo son pequeños comercios/empresas, la población objeto de estudio son los pequeños comercios y empresas de dos determinados municipios y la institución afectada por el fraude es Hacienda. Los datos recogen **información diaria** sobre determinadas características de estos comercios a lo largo del tiempo, obteniéndose 1.248.134 observaciones y un total de 569 comercios. Se recibió toda esta información en formato Excel y en cuatro tipos de archivos:

1. Archivos con los datos históricos de cada empresa: se recibió un archivo Excel independiente para cada comercio, es decir, 569 archivos diferentes. En ellos se recogían características de los comercios como los ingresos diarios, los gastos diarios, el consumo de energía diario, etc.
2. Archivo con el resultado de las inspecciones: en él se recogen todas las inspecciones llevadas a cabo por la institución con su correspondiente fecha y resultado, es decir, si habían encontrado fraude o no.
3. Archivo con información sociodemográfica: en él se detalla la localización de cada comercio, así como el sexo y el nivel de estudios del propietario de cada comercio.
4. Archivo con el nivel de riesgo: dicho archivo recoge una variable elaborada por la institución que indica el grado de peligrosidad de que un comercio cometa fraude en base a su histórico.

Con el objetivo de facilitar la lectura del **primer tipo de archivos** éstos se renombraron desde 1 hasta 569 y se reestructuraron: se eliminó la cabecera y se añadió la información de dicha cabecera

(nombre del comercio y el código del comercio) como columnas. A la hora de realizar la lectura también se realizaron una serie de acciones para depurar los datos:

- Se eliminaron variables no útiles.
- Se reconvirtió el tipo de algunas variables (normalmente variables que originalmente eran de tipo carácter a numéricas).
- Se asignaron como datos perdidos cuando las variables presentaban valor 0 excepto para alguna variable en concreto que veremos más adelante ya que se consideraba que dicho valor podía indicar fraude.
- Se recodificaron las fechas y se corrigieron las fechas repetidas.
- Se cortaron los archivos de forma que se eliminaron los años incompletos.

Respecto al **archivo Excel con el resultado de las inspecciones**, éste presentaba dos hojas, una en la que se recogían las inspecciones que habían resultado fraudulentas y otra en la que se recogían las no fraudulentas. Ambas hojas se unieron en una sola para su lectura, se eliminaron los años incompletos y las variables no útiles y se llevó a cabo una comprobación que consistió en verificar que no existía una inspección fraudulenta y no fraudulenta el mismo día para el mismo comercio.

Para el **tercer tipo de archivo** se recodificaron las variables de forma que tuviesen finalmente tres categorías (dos con información y una tercera con valores ausentes) y se eliminaron las variables no útiles. Para el **cuarto archivo** simplemente se eliminaron las variables no útiles.

Todo **este procedimiento se tuvo que repetir varias veces** ya que los datos fueron enviados en varias tandas y en ocasiones incorrectos por lo que fue un trabajo largo y pesado. Una vez se depuraron todos los archivos, éstos se unieron de forma que al final cada uno de los 569 archivos recogía toda la información (incluida en los cuatro archivos) de una determinada estación.

Tabla 4.1. Ejemplo del formato de un archivo Excel depurado y unido

COMERCIO 1									
Comercio	**Fecha**	**Variables**			**Variable**	**Variables**			**Variable**
1	01JAN11	9056	…	30.5	Sin fraude	A	1	0	1
1	02JAN11	6513	…	28.3	.	A	1	0	1
1	…	…	…	…	…	A	1	0	1
1	31DEC16	4640	…	27.4	Con fraude	A	1	0	1

En azul las variables del primer tipo de archivo, en roja las del segundo tipo de archivo, en verde las del tercer tipo de archivo y en amarillo las del cuarto tipo de archivo.

Estos 569 archivos se dividieron a su vez en 2192 archivos. Cada uno de estos 2192 archivos **recoge la información histórica y de inspecciones de todos los comercios para un mismo día y serán los que se usen en este trabajo**. La temporalidad total de los datos tras su depuración es de **seis años**, de ahí que se creen 2192 archivos, uno para cada día. Así, por ejemplo, el archivo 365 contiene

toda la información de las 569 empresas para el día 31 de diciembre del primer año. Puede ser que de alguna empresa por el motivo que fuera no se dispusiese de datos de algunas variables para ese día en concreto. En ese caso se decidió añadir la empresa al archivo, pero dando valores ausentes a dichas variables de forma que al final cada archivo diario recoge a todas las empresas (una única fila por empresa). Por tanto, disponemos finalmente de **569 series temporales** (una por comercio) recogidas en 2192 archivos.

Tabla 4.2. Ejemplo del formato de un archivo diario final usado en el trabajo

DÍA 1									
Comercio	**Fecha**	**Variables**			**Variable**	**Variables**			**Variable**
1	01JAN11	9056	…	30.5	Sin fraude	A	1	0	1
2	01JAN11	3546	…	29.5	.	B	.	.	2
…	01JAN11	…	…	…	…	…	…	…	…
569	01JAN11	.	…	.	Con fraude	A	2	1	4

5. Metodología

El software mayormente utilizado en este trabajo ha sido el SAS base en su versión 9.4. Las **técnicas estadísticas programadas**, utilizadas para predecir la probabilidad de que un usuario fuese fraudulento y que se van a describir en los siguientes apartados son la **regresión logística, las redes neuronales, random forest, gradient boosting y ensamble de modelos**. Puede encontrarse el código en el anexo, a partir del apartado 12.5.

Los modelos construidos con las cinco técnicas se compararán a través de una serie de **medidas de comparación** que también se presentan en este apartado.

5.1. Descripción de las técnicas estadísticas

Señalar que la descripción que se presenta en los siguientes apartados es una presentación general de las técnicas que se usan en este trabajo. Más adelante, a la hora de construir los modelos con las correspondientes técnicas se detallará más a fondo la metodología y parámetros de cada técnica.

5.1.1. Regresión logística

El objetivo de la **regresión logística**, como el de cualquier regresión estadística, es el de analizar las relaciones entre la variable respuesta y las variables explicativas. Es decir, determinar si existe relación entre la variable dependiente y las variables independientes, medir el signo de dicha relación en caso de que exista y estimar la probabilidad de que se produzca el suceso definido en función de las variables explicativas.

En regresión logística, la variable dependiente es categórica (normalmente dicotómica, como es nuestro caso) y la naturaleza de las variables explicativas es flexible: pueden ser tanto continuas como categóricas. En caso de variables explicativas con más de dos categorías, deben ser introducidas en el modelo definiendo variables ***dummy*** (variable dicotómica). Si la variable explicativa consta de *k* categorías (*k>2)* entonces deben crearse *k-1* variables dummy que serán las que se incluyan en el modelo sustituyendo a la variable explicativa con más de dos categorías original. La manera más normal de definir estas variable *dummy* es la siguiente: si el sujeto pertenece a la categoría de referencia entonces las variables *dummy* toman el valor 0; si el sujeto pertenece a la segunda categoría entonces la primera variable *dummy* toma el valor 1 y el resto 0 y así sucesivamente hasta completar todas las categorías. A cada una de estas variables dummy le corresponderá su respectivo coeficiente en el modelo. Las variables continuas, sin embargo, no necesitan de un tratamiento previo, aunque a veces es recomendable reconvertirlas en variables de intervalos para facilitar y mejorar la interpretación.

En definitiva, el objetivo del modelo de regresión logística consiste en predecir la probabilidad de que suceda un determinado suceso en función de los valores de las variables explicativas, es decir, determinar:

Ecuación 5.1.1.1.

$$P[Y = 1/X_1, X_2 \dots X_k]$$

donde *y* hace referencia a la variable dependiente binaria que toma valor '1' cuando se elige una determinada opción objeto de estudio (en nuestro caso, cometer fraude) y '0' en otro caso. Por otro lado, las variables *X* son las variables explicativas elegidas.

Los modelos de elección binaria especifican una relación entre las características de cada individuo y la probabilidad de la siguiente forma:

Ecuación 5.1.1.2

$$P[Y = 1/X_1, X_2 \dots X_k] = G(\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k)$$

cuyas hipótesis iniciales son:

- Las observaciones $Y_1 \dots Y_n$ son independientes
- Cada Y_i sigue una distribución de Bernoulli

y donde *G* es una función que asume valores que se hallan estrictamente entre cero y uno. Los tres modelos de elección binaria más comunes que se diferencian en la función *G* utilizada en su especificación son: ***cloglog, probit*** y ***logit***, siendo ésta última la más famosa y que tiene la siguiente estructura:

Ecuación 5.1.1.3.

$$P[Y = 1/X_1, X_2 \dots X_k] = \frac{1}{1 + e^{-\beta_0 - \beta_1 x_1 - \cdots - \beta_k x_k}} = \frac{e^{\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k}}{1 + e^{\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k}}$$

Para estimar el vector de parámetros $\beta_0, \beta_1, \dots \beta_k$ se utiliza el método de **máxima verosimilitud**. Una vez estimados $(\hat{\beta}_0, \hat{\beta}_1, \dots \hat{\beta}_k)$ ya se puede estimar la probabilidad del suceso y la interpretación del modelo se realiza mediante los ***odds ratio*** que se calculan como el cociente entre el ***odds*** de una categoría de una variable explicativa (i) y el *odds* de la categoría de referencia de dicha variable explicativa. Los *odds* a su vez se definen como el cociente entre la probabilidad de que el suceso ocurra y la probabilidad de que no ocurra.

Ecuación 5.1.1.4.

$$Odds\ Ratio = \frac{odds(i)}{odds(0)} = \frac{\frac{P[Y=1/X=i]}{1-P[Y=1/X=i]}}{\frac{P[Y=1/X=0]}{1-P[Y=1/X=0]}} = \frac{e^{\beta_0+\beta_i x_i}}{e^{\beta_0}} = e^{\beta_i x_i}$$

El *odds ratio* es por tanto una medida de asociación que aproxima cuanto más probable, o improbable, es que un resultado esté presente entre los individuos con $X=i$ que entre los individuos en la categoría de referencia ($X=0$). En el caso de las variables explicativas continuas, el *odds ratio* determina cuanto más probable, o improbable, es que un resultado esté presente en los individuos con $X=x$ que en los individuos con $X=x-1$, lo cual en ocasiones aporta muy poca información, de ahí que a veces sea preferible 'categorizar' dichas variables continuas.

Para contrastar si una variable incluida en el modelo es significativa y debe quedarse en él se pueden utilizar de manera conjunta el contraste de **razón de verosimilitudes**, que compara el máximo de la función de verosimilitud del modelo con y sin la variable, y el **contraste de Wald**. También se pueden utilizar los intervalos de confianza para determinar si una variable es significativa. Así, si el intervalo de confianza para el coeficiente/odds ratio incluye al cero/uno entonces la variable asociada a dicho coeficiente no será significativa mientras que si el intervalo de confianza no incluye al cero/uno entonces dicha variable será significativa.

Esta es sin duda una de las técnicas más importantes del trabajo y su gran ventaja es que se pueden interpretar los resultados, es decir, los coeficientes de las variables independientes o explicativas que hayan resultado seleccionadas. Además, esta técnica es robusta frente a la heterocedasticidad, no requiere que las variables se distribuyan con normalidad multivariante y permite trabajar con muestras no proporcionales.

5.1.2. Redes neuronales

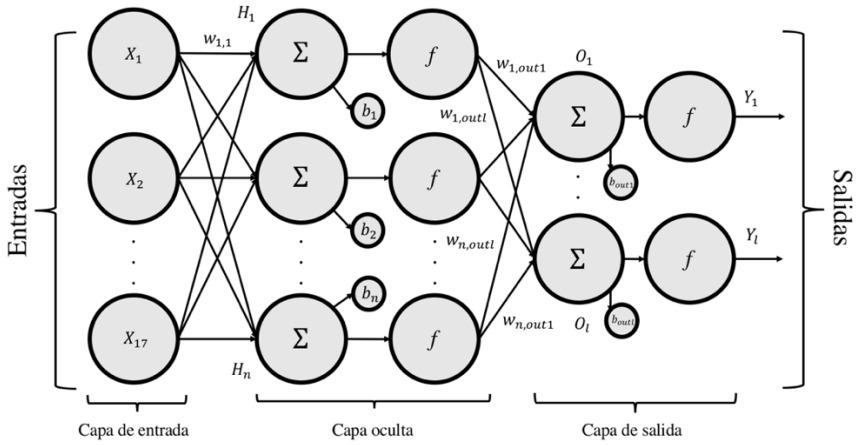
Una **red neuronal artificial** es un modelo matemático inspirado en el comportamiento biológico de las neuronas y en la estructura del cerebro, y que es utilizada para resolver un amplio rango de problemas. Se utilizan sobre todo para situaciones donde no hay fórmulas matemáticas explicitas (no conocemos las relaciones entre las variables input y la objetivo, las relaciones no son lineales, etc.), la **predicción es más importante que la explicación** y cuando **se dispone de muchos datos**, características que se cumplen en nuestro caso.

Una red neuronal consta de ***neuronas* (*nodos*) y *conexiones***, a través de las cuales los nodos se comunican entre sí y se traspasan la información. Las neuronas o nodos de características similares se agrupan en ***capas***, de manera que cuando se conectan dos capas, cada nodo de la primera capa está conectado a cada nodo de la segunda capa. Se distinguen tres tipos de capas:

- **Capa de entrada**: reciben la información del exterior.
- **Capas ocultas**: donde se ejecutan las partes computacionales de la red. Puede haber varias.
- **Capa de salida**: proporciona el resultado del trabajo de la red al exterior. Puede haber varios nodos de salida. En el trabajo se utiliza un único nodo de salida puesto que solo hay una variable objetivo.

Las conexiones son unidireccionales y no pueden formar bucles. Así, los nodos de la capa de entrada sólo pueden estar conectados a los nodos de la/s capa/s oculta/s o a los nodos de la capa de salida; los nodos de la/s capa/s oculta/s pueden estar conectados tanto a los nodos de la capa de entrada como a los nodos de la capa de salida y los nodos de la capa de salida no pueden estar conectados a ningún otro tipo de nodo.

Figura 5.1.2.1. Estructura de una red neuronal con una única capa oculta y varios nodos de salida.



A cada conexión se le asigna un ***peso numérico*** $(\boldsymbol{w_{ij}})$. Este constituye el principal recurso de memoria de largo plazo, y el aprendizaje usualmente se realiza con la actualización de tales pesos.

Después de estandarizar las variables input para acotar sus valores, la capa de entrada se conecta con la capa oculta mediante una ***función de combinación*** (Σ, la más común es la lineal), donde los pesos hacen el papel de parámetros a estimar. Al parámetro b_j se le conoce como ***bias*** o sesgo.

Ecuación 5.1.2.1.

$$H_1 = w_{1,1}X_1 + w_{2,1}X_2 + \cdots + w_{m,1}X_m + b_1$$
$$H_2 = w_{1,2}X_1 + w_{2,2}X_2 + \cdots + w_{m,2}X_m + b_2$$
$$\dots$$
$$H_n = w_{1,n}X_1 + w_{2,n}X_2 + \cdots + w_{m,n}X_m + b_n$$

Tras aplicar la función de combinación, aplicamos a cada nodo de la capa oculta (*H*) una ***función de activación*** (*F*, la más común es *tanh*) que opera con los valores recibidos de las neuronas de la capa anterior, y que tiene en cuenta el peso numérico de la conexión por la que se recibieron dichos valores. Así, una neurona dará más importancia a la información que le llegue por una conexión de peso mayor que a aquella que le llegue por una conexión de menor peso.

Ecuación 5.1.2.2.

$$F(H_1) = F(w_{1,1}X_1 + w_{2,1}X_2 + \cdots + w_{m,1}X_m + b_1)$$
$$F(H_2) = F(w_{1,2}X_1 + w_{2,2}X_2 + \cdots + w_{m,2}X_m + b_2)$$
$$\cdots$$
$$F(H_n) = F(w_{1,n}X_1 + w_{2,n}X_2 + \cdots + w_{m,n}X_m + b_n)$$

Finalmente, aplicamos las funciones de combinación y activación de la capa/s oculta/s a la capa de salida que nos proporcionará los valores predichos para la variable objetivo por la red neuronal:

Ecuación 5.1.2.3.

$$Combinacion \begin{cases} Y_1 = w_{1,out1}H_1 + w_{2,out1}H_2 + \cdots + w_{m,out1}H_m + b_{out1} \\ \cdots \\ Y_l = w_{1,outl}H_1 + w_{2,outl}H_2 + \cdots + w_{m,outl}H_m + b_{outl} \end{cases}$$

$$Activación \begin{cases} Y_1 = F(w_{1,out1}H_1 + w_{2,out1}H_2 + \cdots + w_{m,out1}H_m + b_{out1}) \\ \cdots \\ Y_l = F(w_{1,outl}H_1 + w_{2,outl}H_2 + \cdots + w_{m,outl}H_m + b_{outl}) \end{cases}$$

En el entrenamiento (estimación de los parámetros *w* y *b*) de una red neuronal tanto el peso numérico de las conexiones como el sesgo para cada nodo se modifican (según un **algoritmo de aprendizaje**), con el fin de que los resultados generados por la red coincidan con (o se aproximen a) los resultados esperados (se reduzca el error). El funcionamiento de los algoritmos de optimización por aproximación es muy similar para todos:

1. Tomar unos valores iniciales para los pesos aleatoriamente.
2. Calcular la **función de error** con esos valores.
3. Calcular una función de decrecimiento de la función de error.
4. Retocar los valores de los pesos en esa dirección de decrecimiento.
5. Volver al paso 2.

Como ya se ha comentado, el algoritmo de aprendizaje o de optimización estiman los parámetros de cara a minimizar la función de error. Como regla general, para regresión se toma como función objetivo (de error) a minimizar la suma de cuadrados de los errores, pero en nuestro caso (clasificación binaria) se utiliza la función de verosimilitud basada en la distribución de Bernoulli. También es importante señalar que en regresión no se plantea una función de activación de la capa oculta a la capa de salida, pero en clasificación se suele utilizar la función *softmax*. Tras todo este proceso se obtienen

las predicciones en forma de una **probabilidad** de pertenecer a cada clase de la variable respuesta como en regresión logística. En los métodos basados en árboles la predicción es directamente la clase de la variable respuesta a la que se asigna la observación.

Una de las grandes ventajas de las redes neuronales es que, pese a que los datos estén distorsionados o incompletos, éstas son capaces de extraer la información esencial de los datos; son muy flexibles. Además, no requiere realizar supuestos ni sobre las funciones que va a aproximar ni sobre la distribución de las variables independientes, permite aplicar todo tipo de transformaciones a los datos a través de la función de activación y posee una gran capacidad predictiva.

Sin embargo, los resultados carecen de interpretación ya que no se conoce como se procesa la información internamente y es muy difícil determinar el número de capas y el número de nodos que debe tener cada capa. Por otro lado, para garantizar que una red neuronal funcione con garantías se debe disponer de muchos datos (evitar sobreajuste) y requiere mayor tiempo computacional que otras técnicas por la gran cantidad de parámetros a estimar.

5.1.3. Árboles de decisión

Un **árbol de decisión** es un modelo de predicción utilizado para modelar construcciones lógicas sobre el contenido de bases de datos. Es una forma gráfica y analítica de representar todos los eventos que pueden surgir a partir de una decisión asumida en cierto momento y nos ayudan a tomar la mejor decisión en términos probabilísticos ante un abanico de decisiones. Los valores que pueden tomar las entradas y las salidas pueden ser discretos o continuos. Cuando se utilizan valores discretos se habla de árboles de clasificación y cuando son continuos de árboles de regresión.

Para construir un árbol de decisión se parte de un nodo inicial (***nodo raíz***) y nos preguntamos cómo dividir en dos partes homogéneas el conjunto de datos utilizando una de las variables explicativas consideradas. La variable escogida ha de hacer los dos conjuntos lo más homogéneos posibles con respecto a la variable objetivo pero heterogéneos entre ellos. Se elige una variable y un punto de corte (que minimicen el error) de forma que aquellas observaciones del conjunto de datos que tengan un valor en dicha variable menor que el punto de corte establecido pasan a un nuevo nodo mientras que las observaciones que superan dicho punto de corte para la variable pasan a otro nuevo nodo. De forma que del nodo raíz ahora han surgido dos nuevos nodos. Estos nodos que a su vez se dividen en otros nodos se denominan ***nodos padre*** o ***nodos no terminales***. Los árboles que se usaran en este trabajo son árboles de clasificación **binarios** donde cada nodo padre se divide en dos nodos descendientes.

El proceso de seleccionar una variable y un punto de corte anterior se repite para cada nodo de forma que al final cada observación del conjunto de datos quede clasificada dentro de lo que se llama un ***nodo terminal*** u ***hoja***, que son aquellos que no se dividen y se les asigna una etiqueta que caracterizará a todas las observaciones que hayan ido a parar a dicho nodo terminal.

Por lo tanto, para construir un árbol de decisión hay que tomar las siguientes decisiones:

- Seleccionar las variables y sus puntos de corte óptimos para hacer las divisiones.
- Cuando se considera que un nodo es terminal y cuándo se continúa dividiendo (criterios de parada).
- La asignación de etiquetas o clases a los nodos terminales.

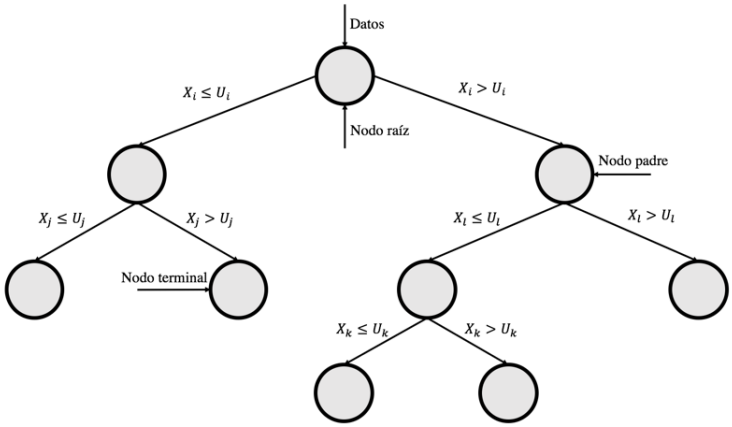
Para decidir que variable va a usarse para hacer la división en cada nodo se calcula primero la proporción de observaciones que pasan por el nodo para cada uno de los grupos. Si se denomina a los nodos como *t = 1,2, ..., T* y *P(g/t)* a las probabilidades de que las observaciones que lleguen al nodo t pertenezcan a cada una de las categorías de la variable respuesta, se define la **impureza de Gini** del nodo *t* como:

Ecuación 5.1.3.1.

$$I(t) = 1 - \sum_{g=1}^{G} p(g|t)^2$$

La variable que se introduce en un nodo es la que minimiza esta impureza que resulta de la división de un nodo. La clasificación de las observaciones en los nodos terminales se hace asignando todas las observaciones del nodo al grupo más probable en ese nodo, es decir, el grupo con máxima *p(g|t)*. Se deja crecer al árbol hasta que cada nodo tenga como mínimo un número de observaciones que escoja el investigador. Si el número de variables al final es muy grande y por tanto de nodos, será necesario ***podar*** o simplificar el árbol hasta que tenga un tamaño adecuado, es decir, podar el árbol de forma que se produzca el menor cambio posible de puridad respecto al original.

Figura 5.1.3.1. Estructura de un árbol de decisión



Las principales ventajas de los árboles de decisión son que fácilmente interpretables, permite encontrar interacciones y reglas difíciles de encontrar con otros métodos, no hay asunciones teóricas, aportan medidas de importancia de las variables, permiten tratar los valores ausentes de una manera eficiente y son bastante robustos frente a los atípicos. En contra, suelen tener poco poder predictivo y gran varianza. Dichas desventajas no han podido ser solucionadas mejorando las funciones de error o algoritmos de construcción, pero sí combinando el resultado de muchos árboles.

5.1.4. Random forest

Random forest es un algoritmo que utiliza un conjunto de modelos formados por árboles de decisión para predecir un valor de salida. Como se ha descrito antes, un árbol de decisión es una colección de nodos y aristas organizados de forma jerárquica en los que un dato es introducido y es sometido a un conjunto de test binarios en cada nodo hasta que llega a una hoja o nodo terminal.

El esquema del algoritmo del random forest es:

1. Se selecciona una muestra aleatoria (con/sin reemplazamiento, estratificación, etc.) de la base de datos original y de igual o no tamaño. Al seleccionarse con reemplazamiento es posible que en la muestra no estén presentes todos los datos del conjunto de entrenamiento original.
2. Los datos que no forman parte de este 'nuevo' conjunto de entrenamiento para aplicar el random forest pueden pasar a formar un conjunto de validación sobre el que evaluar el modelo (OOB). En nuestro caso no usaremos este método ya que disponemos de unos datos de validación independientes.
3. Aplicar un árbol a los datos del paso 1. En cada nodo de los árboles, la búsqueda de la variable que mejor divide a los datos no se realiza sobre todas las variables (**bagging**) si no que se realiza sobre un subconjunto aleatorio *m* de las mismas. Para buscar la variable que mejor clasifique a los datos se utiliza el índice de impureza de Gini definido en la descripción de los árboles de clasificación.
4. Los pasos anteriores son repetidos varias veces, de forma que se obtiene al final un conjunto de árboles de decisión entrenados sobre diferentes conjuntos de datos.
5. Cada nueva entrada es evaluada por el conjunto de árboles de decisión. Para clasificarla se pueden utilizar dos estrategias:

    1. Majority voting: cada árbol de decisión clasificará esta entrada en una categoría y la categoría final de la entrada será la mayoritaria, es decir, la categoría que más veces haya sido asignada por los distintos árboles de clasificación. En caso de regresión se hará por el valor promedio de los resultados.
    2. Promediar las probabilidades estimadas y obtener una clasificación a partir de un punto de corte, que será la que se use en este trabajo.

Al igual que en redes neuronales, una ventaja que presenta esta técnica es que es robusta frente a datos atípicos, muy importante debido a la naturaleza de nuestros datos. Además, aumenta la estabilidad, la capacidad predictiva y reduce la varianza respecto a los árboles de decisión. Aunque presenta una gran desventaja frente a los arboles de decisión que es la pérdida de interpretación. Sólo se puede elaborar un gráfico de importancia de las variables según el número de veces utilizadas en el algoritmo o siguiendo el criterio de Gini.

5.1.5. Gradient Boosting

Antes de explicar el algoritmo de gradient boosting se van a repasar algunos conceptos ya comentados para comprender mejor la técnica de este apartado.

Para mejorar la sensibilidad de un algoritmo de clasificación (como puede ser un árbol de decisión), en vez de procesar solamente una muestra inicial, se generan T submuestras de la muestra inicial de igual tamaño, tomadas aleatoriamente y con reemplazamiento. A estas nuevas muestras se las conoce como muestras **bootstrap**.

Los algoritmos que utilizan **bagging** (como Random Forest) buscan minimizar la varianza de los errores. El bagging consiste en que el clasificador final para una observación se forma agregando todos los clasificadores $C_t(t = 1 \dots T)$, de forma que para clasificar una observación se toman todas las clasificaciones obtenidas de las T submuestras y se clasifica en la clase que tenga más votos de todos los clasificadores.

En cambio, los algoritmos que utilizan **boosting** (como el Gradient Boosting) lo que persiguen es minimizar el sesgo de los errores. Al contrario que el bagging que guarda una sucesión de clasificadores, el boosting mantiene un peso para cada observación en el modelo. Cuanto mayor es el peso, mayor es la influencia de la observación en el clasificador. En cada nueva iteración los pesos de las observaciones se van actualizando de forma que los pesos de las observaciones mal clasificadas se incrementan para que en la siguiente iteración se haga más énfasis en ellas.

De forma muy inocente, el algoritmo de gradient boosting consiste en:

1. Ajustar un modelo a los datos, $F_1(x) = y$
2. Ajustar un modelo a los residuos del modelo anterior, $h_1(x) = y - F_1(x)$
3. Crear un nuevo modelo corrigiendo los errores del modelo del punto 1, $F_2(x) = F_1(x) + h_1(x)$

De forma simple se puede generalizar esta idea insertando más modelos que corrijan los errores del modelo previo:

Ecuación 5.1.5.1.

$$F(x) = F_1(x) \rightarrow F_2(x) = F_1(x) + h_1(x) \dots \rightarrow F_M(x) = F_{M-1}(x) + h_{M-1}(x)$$

donde $F_1(x)$ es el modelo inicial ajustado. Es importante señalar que h_m puede ser un modelo cualquiera que sirva para mejorar algún modelo anterior más débil. Esta es una de las ventajas del gradient boosting. En la práctica, sin embargo, h_m suele ser un **árbol de decisión** (como en nuestro caso y presente trabajo).

Sin embargo, en gradient boosting, en el paso 2. se ajusta un modelo no sobre los residuos del modelo anterior, si no sobre el gradiente de la función de pérdida $L\,(y, F_1(x))$ con respecto a los valores predichos por $F_1(x)$. Es decir, se busca optimizar dicha **función de pérdida**. Dicha función

debe ser diferenciable y para clasificaciones binarias como las del presente trabajo se suele utilizar la **función logística o Deviance** ($L(y_i, F(x_i)) = \log(1 + e^{-2y_i F_i(x_i)}\ donde\ y_i = 1, 0$).

El algoritmo queda de la siguiente forma:

1. Iniciamos el algoritmo con un valor constante: el porcentaje del evento de interés en los datos ($F_1(x)$).
2. Calculamos los pseudo residuos (gradiente de L con respecto a los valores predichos por $F_1(x)$): $r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]$ para $i=1...n$
3. Ajustar un árbol de decisión (de clasificación en nuestro caso) $h_m(x)$ a los pseudo residuos.
4. Actualizar $F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$ donde γ_m es un parámetro de regularización encargado de definir el peso de cada predicción.

En resumen, el algoritmo gradient boosting consiste en repetir la construcción de árboles de decisión, modificando ligeramente las predicciones iniciales cada vez, intentando ir minimizando los residuos en la función de decrecimiento. Al plantear diferentes árboles cada vez, el proceso va ajustando las predicciones cada vez más a los datos, y de alguna manera unos árboles corrigen a otros con lo cual la flexibilidad y adaptación del método mejora respecto a la construcción de un único árbol.

Las principales ventajas, como toda técnica basada en árboles de decisión, es que es robusto frente a datos atípicos, eficiente a la hora de tratar los valores ausentes, permite detectar interacciones y no tiene problema si existe multicolinealidad. Además, como ya se ha comentado, reduce el sesgo de los árboles de decisión. Posee una gran capacidad predictiva. Presenta, al igual que en random forest, la desventaja de que se pierda la interpretabilidad de los resultados.

5.1.6. Ensamble de modelos

El **ensamble de modelos** consiste en combinar las predicciones (sobre los datos test) de unos modelos primarios con el objetivo de mejorar las predicciones de dichos modelos. De hecho, ya hemos visto dos técnicas que lo realizan: random forest y gradient boosting, ya que principalmente lo que hacen es ajustar varios árboles y combinar las predicciones de dichos árboles. Las técnicas básicas para combinar los modelos son bagging (bootstrap averaging, como en random forest), boosting (como en gradient boosting) y ***stacking*** que es en la que nos vamos a centrar en el presente apartado. Existen tres opciones básicas de stacking:

- Combinar las predicciones (probabilidades) de las anteriores técnicas usando algún tipo de medida estadística (media, mediana, etc.)
- Majority Voting (ver apartado 5.1.4.)
- Introducir las predicciones de otras técnicas como variables independientes de otra técnica.

En este trabajo se utilizarán la primera y tercera opción ya que son las que mejor suelen funcionar. En general, se recomienda hacer ensambles de modelos poco correlacionados entre sí, de forma que se complementen unos a otros. No obstante, también se pueden hacer ensambles de modelos correlacionados para reducir la varianza del error (por ejemplo, ensamble de un modelo de redes y otro modelo de redes, pero con más nodos u otra función de activación, etc.).

La principal ventaja del ensamble de modelos es que reducen la varianza del error en general, es decir, casi nunca empeoran los modelos primarios. Además, son robustos, unos modelos se corrigen a otros. Las desventajas es que se aumenta la complejidad, a menudo se produce sobre ajuste y que los resultados no son interpretables como sucede en redes neuronales, random forest y gradient boosting.

5.2. Medidas de comparación de modelos

En los próximos apartados se tratará de encontrar el mejor modelo dentro de cada técnica estadística para luego compararlos entre ellos y obtener un modelo final con el que lograr el objetivo del estudio. Se generarán multitud de modelos y la metodología utilizada para compararlos se presenta a continuación.

En función de los parámetros que varían dentro de cada técnica y de la longitud de los intervalos definidos a partir de los puntos de cortes y a partir de los cuales una observación se considerará o no fraudulenta es obvio que el grado de acierto de los modelos irá también variando. Por ejemplo, si en un momento determinado el punto de corte es de 0.5 y la probabilidad estimada de un modelo determinado es de 0.51 para una observación concreta, entonces dicha observación será clasificada como fraudulenta. Como el punto de corte irá variando, es fácil ver como en algunas ocasiones la observación será clasificada como fraudulenta y en otras como no fraudulenta (con 0.51 por ejemplo), lo que modificará el grado de acierto de un modelo a otro. En este tipo de datos donde las clases de la variable objetivo están muy desequilibradas como veremos más adelante, utilizar medidas como la tasa de fallos puede ser muy optimista. Dado que en el presente trabajo nos importa sobre todo una clase (1, fraude) de la variable objetivo, se utilizarán medidas que hagan más énfasis en ella como la **sensibilidad** o la **precisión**. Esta precisión o **bondad de ajuste** se construye para todos los modelos como:

Ecuación 5.2.1.

$$Bondad\ de\ Ajuste/Precisión = \frac{VP}{VP + FP}$$

donde VP hace referencia a los verdaderos positivos (en nuestro caso, positivo es igual a ser fraudulento), es decir, aquellas observaciones que eran fraudulentas y que el modelo ha determinado como fraudulentas; y FP a los falsos positivos, es decir, observaciones que el modelo ha clasificado como fraudulentas y que en realidad no lo eran. A mayor bondad de ajuste, mejor será el modelo. Se puede decir que **esta es la medida de comparación más importante**. Su interpretación es la siguiente: de las veces que se predice que una observación es fraudulenta, muestra el porcentaje de acierto.

Tabla 5.2.1. Matriz de confusión

		Fraude	
		1	**-1**
Fraude estimado	**1**	VP (Verdadero positivo)	FP (Falso positivo)
	-1	FN (Falso negativo)	VN (Verdadero negativo)

A partir de la tabla anterior también se pueden calcular la **sensibilidad** y la **especificidad**. La sensibilidad es la probabilidad de estimar como fraudulento a un comercio fraudulento mientras que la especificidad es la probabilidad de estimar como no fraudulento a un comercio no fraudulento. Lo deseado es que ambos valores sean elevados ya que eso es un indicio de que el modelo discrimina adecuadamente a los usuarios.

Ecuación 5.2.2.

$$Sensibilidad\ [P(+/Fr)] = \frac{VP}{VP + FN} \qquad Especificidad\ [P(-/NoFr)] = \frac{VN}{VN + FP}$$

Otra medida para comparar los mejores modelos óptimos de cada técnica estadística y los submodelos dentro de cada técnica es el **número de parámetros** a estimar en el modelo. Así, siguiendo el *criterio de parsimonia*, entre dos modelos con un grado de acierto parecido será mejor aquel que haya obtenido dicho grado de acierto con menos parámetros a estimar.

La última medida de comparación que se considerará es el **número de investigados** que propone cada modelo, es decir, el número total de comercios que el modelo clasifica como fraudulentos. La institución está capacitada para realizar 158 visitas mensuales (unas 5 por día). No obstante, la temporalidad de nuestros datos (y, por ende, de los modelos) es diaria por lo que lo óptimo sería que la institución ejecutara diariamente el modelo que salga ganador y fuese a inspeccionar a los 5 comercios que el modelo les proponga cada día. En este caso, dado que se disponen de dos años de validación, para que un modelo fuera considerado válido debería clasificar en ellos como fraudulentos a un mínimo de 3792 comercios (158*24 meses).

Sin embargo, **la institución no posee la capacidad operativa para ejecutar diariamente el modelo sino solo una vez al mes**. Dado que se desconoce qué día se va a ejecutar, es necesario que el modelo proponga 158 inspecciones cada vez que se ejecute. Por tanto, **para que un modelo sea considerado válido, debe clasificar como fraudulentos como mínimo a 113760** (24 meses en los dos años de validación * 158 número de inspecciones mensuales capaz de realizar la institución * 30 días en un mes) **comercios en los dos años de validación**. En las tablas para comparar los modelos se presentarán las visitas por ejecución del modelo (es decir, comercios que el modelo clasifica como fraudulentos en un día, que han de ser 158 como mínimo), no las visitas totales en los dos años de validación.

6. Variables

6.1. Análisis descriptivo y depuración de las variables originales

Es muy importante realizar en primer lugar un **análisis descriptivo** de los datos antes de estimar los modelos estadísticos para conseguir el objetivo del estudio ya que es necesario conocer la calidad de la información de la que se dispone y de la que partimos puesto que la calidad de los modelos dependerá de la de los datos de partida.

Inicialmente se parte de las siguientes variables contenidas en los 2192 archivos, estructuradas según el archivo del que provienen (las dos primeras variables son comunes en todos ellos). Como se puede comprobar, hay variables estáticas que no cambian de valor a lo largo del tiempo y variables no estáticas que sí lo hacen. Recordar que han sido camufladas.

- ***Código comercio***: código que sirve para identificar a un comercio en los datos. No puede haber códigos de comercio repetidos por día (y, por ende, por archivo). Se dispone de información de 569 comercios con un código de comercio distinto asociado a cada uno.
- ***Día***: indica el marco temporal de los datos. Toma los valores de 1 a 2192 lo que supone unos 6 años. Así, por ejemplo, el día 1 puede hacer referencia al 24 de septiembre de 1995 y el día 731 al 24 de septiembre de 1997 ya que 1996 es bisiesto. Por lo tanto, el día 2192 haría referencia al 24 de septiembre de 2001. No se indica el marco temporal real por confidencialidad.

- ***Ventas***: variable que hace referencia al valor de las ventas realizadas por un pequeño comercio en un día, es decir, sus ingresos diarios.
- ***Costes***: esta variable expresa el valor de los gastos diarios del pequeño comercio en las materias primas, equipos y suministros que utilizan para producir sus bienes, productos y servicios.
- ***Salario de los empleados***: variable que informa sobre los salarios diarios totales de los empleados contratados por parte del propietario del pequeño comercio.
- ***Impuestos***: como su nombre indica, esta variable revela el valor total de los impuestos diarios sobre las ventas a pagar a la institución.
- ***Consumo de energía***: variable que recoge el valor del consumo diario de la luz y que sirve para determinar si un inmueble está habitado, el grado de uso que se le da y que corresponda con lo declarado.

- ***Resultado***: variable que recoge el resultado de las inspecciones (si se encontró fraude o no) llevadas a cabo por la institución. Las categorías que presenta la variable se presentan en la siguiente tabla. En general, las cuatro primeras categorías indican que se encontró alguna anomalía en la inspección, la quinta indica que no se encontró ninguna anomalía y los valores perdidos en este caso indican el número de días que no se realizó inspección.

Tabla 6.1.1. Distribución de la variable 'resultado' en los datos

		Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Resultado	**Abierto**	268	2.66	268	2.66
	Dentro de plazo	2281	22.66	2549	25.32
	En curso	35	0.35	2584	25.67
	Fuera de plazo	48	0.48	2632	26.14
	Sin irregularidad	7435	73.86	10067	100
Número de missings = 1030065					

Como se puede apreciar, únicamente en el 0.96% de los días de los datos se realizó alguna inspección. Además, de todas las inspecciones realizadas, sólo el 26.14% de ellas presentaron anomalías.

- ***Localización:*** variable con dos categorías que indica el municipio al que pertenece el comercio.

Tabla 6.1.2. Distribución de la variable 'localización' en los datos

		Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Localización	**A**	438	80.66	438	80.66
	B	105	19.34	543	100
Número de missings = 26					

- ***Sexo:*** variable con dos categorías que indica el sexo del propietario del comercio.

Tabla 6.1.3. Distribución de la variable 'sexo' en los datos

		Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Sexo	**H**	212	39.04	212	39.04
	M	331	60.96	543	100
Número de missings = 26					

- ***Nivel de estudios:*** variable con dos categorías que indica si el propietario del comercio posee un nivel de estudios universitario (1) o no (0).

Tabla 6.1.4. Distribución de la variable 'nivel de estudios' en los datos

		Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Estudios	**1**	262	48.88	262	48.88
	0	274	51.12	536	100
Número de missings = 33					

- ***Código alerta***: variable con cuatro categorías que indica cómo de fraudulento es un comercio a partir de su histórico. A mayor código de alerta, mayor es la probabilidad de que el comercio sea fraudulento según la institución.

Tabla 6.1.5. Distribución de la variable 'código de alerta' en los datos

		Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Código alerta	**1**	169	30.39	165	30.39
	2	11	2.03	176	32.41
	3	91	16.76	267	49.17
	4	276	50.83	543	100
Número de missings = 26					

En el anexo 12.1. se presentan los siguientes estadísticos sobre las variables **continuas** sin depurar:

- Histograma
- Máximo
- Mínimo
- Media
- Mediana
- Kurtosis
- Desviación típica

Para las variables **categóricas**, a parte de las tablas de frecuencias incluidas en este apartado, se representa un gráfico de sectores en el anexo para observar la distribución de la variable. En caso de valores perdidos se presenta como una categoría más.

Debido a la confidencialidad de los datos, el análisis descriptivo no se puede interpretar con claridad y debido al enmascaramiento de las variables algunas de ellas pueden presentar resultados incongruentes.

En cuanto a la **depuración**, en general, a la vista de los histogramas, se observaron comportamientos extraños en los valores extremos de las variables por lo que se sustituyeron los **valores atípicos,** de dos variables en concreto, por valores perdidos dado que, independientemente

del tamaño o el tipo del comercio, presentaban valores muy anómalos para cualquier situación posible. Más allá de esto, no se detectaron errores en los valores de las variables. Los datos ausentes se tratarán más adelante, tras la creación de las variables explicativas.

6.2. Creación de las variables explicativas

A partir de las variables originales depuradas se van a construir otras nuevas posibles variables explicativas que intenten ayudar a predecir o a explicar mejor la variable respuesta. No hay una metodología escrita sobre cómo crear estas nuevas variables. Es el investigador el que, a través de su conocimiento sobre la realidad sobre la que está trabajando y de los datos disponibles, intuye o piensa variables que pueden ayudar a explicar la variable respuesta. Seguramente muchas de ellas no resulten significativas, pero ha de ser el modelo el que lo decida.

Dado que la naturaleza de los datos es de tipo temporal, las variables que se crean son de tipo "agrupadoras" para que la información de cada observación no se limite a la de únicamente ese día (para una empresa determinada), si no que también recoja información agregada del pasado (de esa empresa determinada) para que los modelos intenten encontrar patrones de comportamiento a lo largo del tiempo que indiquen fraude. Si no se hiciese esto, los modelos predecirían cada observación como si fuese independiente, lo cual no es correcto en esta situación. Por ejemplo, supongamos que para una empresa determinada se tiene que estaba cometiendo fraude un día determinado. Predecir el fraude únicamente teniendo en cuenta la información del día en el que se le detectó fraude no es correcto, ya que seguramente su comportamiento pasado también influya.

El máximo marco temporal "agregador" que se usa para crear las variables es de 365 días por lo que en los modelos se incluirán los datos a partir del día/archivo 365 de los 2192, ya que de esta forma se dispondrá de **todas las variables para todas las observaciones**.

6.2.1. Variables históricas

Utilizando las variables originales que recogían la información histórica de cada comercio (*ventas*, *costes*, *salario de los empleados*, *impuestos* y *consumo de energía*) del primer tipo de archivo "*azul*" se han construido nuevas variables de la siguiente forma:

- Calculando la **media** y la **desviación típica** para cada empresa de cada una de dichas variables agrupando los datos de los últimos 7, 14, 28, 91 y 365 días, es decir, calcular la media y la desviación típica de las variables para cada comercio una semana, dos semanas, tres semanas, tres meses y un año atrás al día correspondiente en el que nos encontremos.
- Contando el **número de datos perdidos**, el **número de datos válidos** y el **porcentaje de valores perdidos** en cada variable para cada comercio agrupando los datos igual que en el punto anterior.
- Creando una variable auxiliar en la que para cada uno de los últimos 28 días toma valor 1 si el valor de la variable es mayor que la media de la variable, -1 si es menor y 0 en otro caso (para

cada empresa). Se realiza una media ponderada de dicha variable auxiliar de forma que se tienen más en cuenta los cambios sufridos más cerca al día correspondiente que los lejanos.

- Utilizando la variable auxiliar del punto anterior, se busca el valor mínimo que alcanza la suma de dicha variable en los últimos 28 días, es decir, el máximo de cambios negativos. La variable representa por tanto el máximo número de días en los que un determinado comercio presentó valores en la mencionada variable por debajo de la media de dicha variable en los últimos 28 días.
- Contando los días que he de retroceder para encontrar el máximo calculado en el punto anterior. Si vale 1 será que lo he encontrado en el día actual, 2 en el anterior y así sucesivamente hasta un posible valor máximo de 28.

A continuación, y a modo de ejemplo para el resto de variables (*costes*, *salario de los empleados*, *impuestos* y *consumo de energía*), se presenta la codificación de las variables creadas para la variable *ventas*. En el anexo 12.2. se pueden encontrar las tablas con la codificación de las variables creadas para el resto de variables, aunque son muy similares.

Tabla 6.2.1.1. Codificación de las variables creadas para la variable 'ventas'

Nombre de la variable creada	Descripción	Nombre de la variable creada	Descripción
VentasMedias365	Media de las ventas de los últimos 365 días	***PorcenMissVentas7***	Porcentaje de valores ausentes en los últimos 7 días
VentasMedias91	Media de las ventas de los últimos 91 días	***NumMissVentas365***	Número de valores ausentes en los últimos 365 días
VentasMedias28	Media de las ventas de los últimos 28 días	***NumMissVentas91***	Número de valores ausentes en los últimos 91 días
VentasMedias14	Media de las ventas de los últimos 14 días	***NumMissVentas28***	Número de valores ausentes en los últimos 28 días
VentasMedias7	Media de las ventas de los últimos 7 días	***NumMissVentas14***	Número de valores ausentes en los últimos 14 días
VentasDesv365	Desviación típica de las ventas de los últimos 365 días	***NumMissVentas7***	Número de valores ausentes en los últimos 7 días
VentasDesv91	Desviación típica de las ventas de los últimos 91 días	***NumDatosVentas365***	Número de datos válidos en los últimos 365 días

VentasDesv28	Desviación típica de las ventas de los últimos 28 días	***NumDatosVentas91***	Número de datos válidos en los últimos 91 días
VentasDesv14	Desviación típica de las ventas de los últimos 14 días	***NumDatosVentas28***	Número de datos válidos en los últimos 28 días
VentasDesv7	Desviación típica de las ventas de los últimos 7 días	***NumDatosVentas14***	Número de datos válidos en los últimos 14 días
PorcenMissVentas365	Porcentaje de valores ausentes en los últimos 365 días	***NumDatosVentas7***	Número de datos válidos en los últimos 7 días
PorcenMissVentas91	Porcentaje de valores ausentes en los últimos 91 días	***CambiosPonderadosVentas***	Media ponderada de los cambios respecto al valor medio de los últimos 28 días
PorcenMissVentas28	Porcentaje de valores ausentes en los últimos 28 días	***MaxCambioVentas***	Número máximo de días con cambios negativos en los últimos 28 días
PorcenMissVentas14	Porcentaje de valores ausentes en los últimos 14 días	***LugarMaxCambioVentas***	Indica el lugar donde se ha producido el máximo de la variable anterior

Hasta aquí se crean un total de 140 variables, 28 por cada variable. La variable *ventas* dado que es una variable muy importante se ha abordado desde diferentes perspectivas a parte de la original:

- Se crea una nueva variable (***Ventas_estand***) que recoge las ventas estandarizadas. Concretamente esta variable se genera calculando la mediana de las ventas de todos los comercios en un día y restando dicha mediana al valor de las ventas de cada comercio para ese día. Con esta variable se construyen 13 nuevas de forma similar a las anteriores:

    - Calculando la media y la desviación típica para cada empresa de la variable *ventas_estand* agrupando los datos de los últimos 7, 14, 28, 91 y 365 días (***VentasMediasEstand365***, ***VentasMediasEstand91***, ***VentasMediasEstand28***, ***VentasMediasEstand14***, ***VentasMediasEstand7***, ***VentasDesvEstand365***, ***VentasDesvEstand91***, ***VentasDesvEstand28***, ***VentasDesvEstand14***, ***VentasDesvEstand7***).
    - Creando una variable auxiliar en la que para cada uno de los últimos 28 días toma valor 1 si el valor de *ventas_estand* es mayor que la media de la variable en los últimos 28

días, -1 si es menor y 0 en otro caso. Se realiza una media ponderada de dicha variable auxiliar de forma que se tienen más en cuenta los cambios sufridos más cerca al día correspondiente que los lejanos (***CambiosPonderadosVentasEstand***).

- Utilizando la variable auxiliar del punto anterior, se busca el valor mínimo que alcanza la suma de dicha variable en los últimos 28 días, es decir, el máximo de cambios negativos (***MaxCambioVentasEstand***).
- Contando los días que he de retroceder para encontrar el máximo calculado en el punto anterior (***LugarMaxCambioVentasEstand***).

- Dado que se sospecha que unas ventas nulas pueden ser indicio de alguna actividad fraudulenta (de ahí que en la lectura de los datos los valores nulos para ventas no se reconvirtieran en valores perdidos como sí se hizo en el resto de variables históricas) se crea una nueva variable (***Ventas0***) que tomará los mismos valores que *ventas* con la diferencia de que cuando la variable *ventas* presenta un valor perdido, a la variable *ventas0* se le asigna un 0. Con esta variable se construyen 3 nuevas:

  - Nuevamente se crea una variable auxiliar en la que para cada uno de los últimos 28 días toma valor 1 si el valor de *ventas0* es mayor que la media de la variable en los últimos 28 días, -1 si es menor, 0 en otro caso y se realiza una media ponderada de dicha variable auxiliar (***CambiosPonderadosVentas0***).
  - Utilizando la variable auxiliar del punto anterior, se busca el valor mínimo que alcanza la suma de dicha variable en los últimos 28 días, es decir, el máximo de cambios negativos (***MaxCambioVentas0***).
  - Contando los días que he de retroceder para encontrar el máximo calculado en el punto anterior (***LugarMaxCambioVentas0***).

- Por último, se ha calculado una variable (***PorcenVariacionVentas***) que indica el porcentaje de variación de las ventas en un día de la semana respecto al mismo día una, dos y tres semanas antes. Por ejemplo, para un miércoles determinado, se calcula la media de las ventas de los miércoles de las tres semanas anteriores y se resta dicha media al valor de las ventas del miércoles actual y se divide nuevamente entre la media. Si el valor de las ventas del miércoles es missing entonces la variable tomará valor -1 y si el valor de la media de los tres miércoles anteriores es missing entonces la variable tomará el valor 1.

Se crean por tanto un total de 162 variables en este apartado. Todas ellas se usarán como variables explicativas en los modelos de predicción a excepción de *ventas_estand*, *ventas0* y las variables originales (ya que se presupone que las variables creadas a partir de ellas contienen y aportan más información). Este es el grupo de variables más numeroso y se considera esencial a la hora de poder detectar el fraude.

6.2.2. Variables relativas a las inspecciones

Utilizando la variable original *resultado* que recogía la información con el resultado de las inspecciones llevadas a cabo por la institución del segundo tipo de archivo "*rojo*" se ha construido una nueva variable (***Fraude***) de la siguiente forma:

- Como ya se comentó en la descripción de la variable original *resultado*, para construir esta nueva variable se ha considerado que una inspección resultó fraudulenta si presentó alguna de las primeras cuatro categorías y como no fraudulenta si presentó la quinta categoría. Por tanto, la variable *fraude* consta de tres categorías. La primera de ellas (-1) indica que al comercio se le realizó una investigación un determinado día y que no se le encontraron anomalías (quinta categoría de *resultado*). La segunda (0) indica que no se le realizó ninguna inspección ese día al comercio por lo que no se dispone de información sobre si estaba siendo fraudulento o no (missings en *resultado*). La última de ellas y la tercera (1) indica que un determinado día se inspeccionó al comercio y se le descubrió algún tipo de comportamiento fraudulento (cuatro primeras categorías de *resultado*).

A partir de la variable *fraude* se construyen otras nuevas variables de la siguiente manera:

- Contando el número de inspecciones que resultaron fraudulentas para cada comercio en los últimos 365, 91 y 28 días, es decir, en el último año, últimos tres meses y últimas tres semanas (***NumFraudes365***, ***NumFraudes91*** y ***NumFraudes28*** respectivamente).
- Contando el número de inspecciones que resultaron no fraudulentas en los mismos periodos que el punto anterior (***NumNoFraudes365***, ***NumNoFraudes91*** y ***NumNoFraudes28*** respectivamente).
- Contando el número total de inspecciones llevadas a cabo por la institución (independientemente de su resultado) en los mismos periodos que el primer punto (***NumVisitas365***, ***NumVisitas91*** y ***NumVisitas28*** respectivamente).

6.2.3. Variables sociodemográficas

Utilizando las variables originales *localización*, *sexo* y *nivel de estudios* del tercer tipo de archivo "*verde*" se crean una serie de nuevas variables:

- Para cada categoría de cada una de las variables se calcula el porcentaje de fraudes (usando la variable *fraude*) que tiene cada categoría. En el caso de valores perdidos se calcula la media general de todas las categorías. Dado que las tres variables originales presentan dos categorías, estas nuevas tres nuevas variables (***MediaFraudeLoc***, ***MediaFraudeSexo*** y ***MediaFraudeEstudios***) tendrán tres posibles valores, uno para cada una de las dos categorías de la variable y un tercer valor que será la media general asociada a los datos perdidos.

A partir de las variables *MediaFraudeSexo* y *MediaFraudeEstudios* se pretende recoger la idea de la "*moralidad fiscal*" de cada individuo. Más allá de la posible existencia de otros factores explicativos, de los costes de oportunidad de la renta evadida, de las probabilidades de ser detectados y sancionados y de la mayor o menor aversión al riesgo, la honestidad o "*moral fiscal"* individual es una variable relevante a la hora de decidir el cumplimiento de las obligaciones tributarias. De hecho, existen trabajos como por ejemplo los de Alm y Torgler (2006) y Vogel (1974) en los que se encontraron relaciones positivas entre la moral fiscal y el hecho de ser mujer y relaciones positivas entre la moral fiscal y el poseer estudios elevados.

En definitiva, se crean un total de 175 variables de las cuales 172 se usarán como variables explicativas en los modelos de predicción. A estas 172 variables explicativas se le suma la variable original *código de alerta* por lo que **en total se tienen 173 variables explicativas**.

Hay que mencionar que se realizó un análisis descriptivo de todas las variables creadas para comprobar que tomaban valores razonables y esperados, pero no se muestran en este trabajo debido al elevado número de variables creadas. Dado que las variables originales con las que se han construido las variables presentaban **valores ausentes**, alguna de las variables construidas también (concretamente, las variables que calculaban la media o la desviación típica). Es necesario tratar estos valores ausentes puesto que a la hora de construir los modelos (a excepción de random forest y gradient boosting) solo se utilizan aquellas observaciones completas por lo que la presencia de valores ausentes puede suponer una gran pérdida de información y un sesgo. Además, a la hora de validar el modelo, sólo se podrá calcular la probabilidad de fraude para una observación si no hay datos ausentes en las variables seleccionadas por el modelo. Así pues, estos valores 'missings' se sustituyeron por el valor cero ya que se comprobó que era un valor válido en todos los casos para que los modelos lo entendiesen como un valor que indicaba ausencia de información.

6.3. Creación de la variable respuesta

Si recordamos, uno de los objetivos de este trabajo es el de predecir la probabilidad de que un comercio cometa fraude en el futuro por lo que, como es lógico, se necesita una variable respuesta que contenga la información de si un comercio cometió fraude o no para poder predecirla con los modelos de predicción. Hasta ahora, la variable que podría usarse como variable respuesta sería *resultado* y por ende *fraude*. Sin embargo, *fraude* sólo considera como fraudulenta a una empresa el día que se le realizó la inspección y se le encontró alguna anomalía, pero seguramente **el hecho de cometer fraude un día determinado implica que un periodo de tiempo anterior también se estaba cometiendo fraude**. Es por ello por lo que se definen nueve posibles variables respuestas (*Fr1, Fr5, Fr10, Fr15, Fr30, Fr61, Fr91, Fr182* y *Fr365*) en las que se considera que un comercio estaba siendo fraudulento el día anterior, los cinco días anteriores y así hasta un año anterior al día en el que se le detectó fraude. Por ejemplo, *Fr30* tomará el valor 1 (que indica fraude) los treinta días anteriores a un día en el que un comercio presentó una inspección fraudulenta. Si por ejemplo a un comercio determinado se le detectó fraude en una inspección el 1 de julio, *Fr30* tomará el valor 1 en todo el mes de junio (los 30 días anteriores).

El motivo de la creación de estas variables respuesta no es otro que el de mejorar la capacidad predictiva de los modelos. Para escoger aquella variable respuesta con mayor capacidad de predicción se realizan varios análisis de **regresión logística** con las variables *Fr1 … Fr365* como variables respuesta y como variables explicativas las 173 descritas en el apartado anterior. La metodología de la regresión logística se explica en el apartado 7.2.1. aunque en este caso, por motivos de tiempos de ejecución, sólo se considera como método de selección de variables el método *stepwise* cuyos p-valores de entrada y de salida irán variando en el mismo rango que se especifica en dicho apartado, pero siempre tomando el mismo valor ambos. Para construir los modelos se elabora un archivo de entrenamiento para cada posible variable respuesta y un archivo de validación único siguiendo los pasos del apartado 7.1. Se elaboran por tanto 504 modelos para cada variable respuesta, es decir, un total de 4536 modelos.

Los 504 modelos para cada variable respuesta se ordenan de forma descendiente por la bondad de ajuste y se selecciona el primero que cumpla la condición de realizar al menos 113760 investigaciones en los dos años de validación para comparar su bondad de ajuste con la actual de la institución. El número de inspecciones también se va modificando para comparar la evolución de la efectividad de las posibles variables respuestas a medida que varía el número de investigaciones a realizar. A continuación, se presenta la tabla correspondiente:

Tabla 6.3.1. Elección de la variable respuesta.

	Inspecciones en los dos años de validación									
	100000 (139 por ejecución)		**120000 (167 por ejecución)**		**140000 (194 por ejecución)**		**160000 (222 por ejecución)**		**180000 (250 por ejecución)**	
Variable	**% de mejora absoluto**	**% de mejora relativo**	**% de mejora absoluto**	**% de mejora relativo**	**% de mejora absoluto**	**% de mejora relativo**	**% de mejora absoluto**	**% de mejora relativo**	**% de mejora absoluto**	**% de mejora relativo**
Fr1	0	0	0	0	0	0	0	0	0	0
Fr5	13.48	55.58	10.48	42.58	10.17	41.31	8.87	36.59	8.87	36.59
Fr10	9.31	38.32	9.31	38.32	9.23	37.99	5.78	23.78	5.78	23.78
Fr15	7.98	32.85	7.98	32.85	7.77	31.97	4.65	19.14	4.65	19.14
Fr30	6.23	25.68	6.23	25.68	6.23	25.68	3.12	12.87	3.12	12.87
Fr61	9.67	39.51	5.38	21.99	5.38	21.99	5.38	21.99	2.43	9.93
Fr91	8.16	31.97	8.16	31.97	6.91	27.07	4.23	16.56	4.23	16.56
Fr182	7.54	22.08	7.54	22.08	7.54	22.08	4.24	12.41	4.24	12.41
Fr365	12.24	23.32	9.43	17.97	9.43	17.97	9.43	17.97	9.43	17.97

Lo primero que llama la atención y que se puede observar es que el porcentaje de mejora apenas varía a medida que cambia el número de inspecciones. Esto se debe a que, al estimar los modelos, los 'saltos' entre el número de inspecciones que propone un modelo y otro son muy altos por lo que a pesar de realizar el corte por el número de inspecciones siempre se selecciona el mismo modelo o uno muy parecido. Si se dispusiese de más tiempo de ejecución, se podrían variar más los parámetros de la regresión logística para intentar mitigar esta situación. También se puede observar como a medida que aumentamos el número de inspecciones, el grado de mejora de los modelos baja como es lógico.

En cualquier caso, todos los modelos construidos con las posibles variables respuesta mejoran el grado de acierto actual de la institución excepto *Fr1*. Esto, sin duda, es un buen síntoma ya que la regresión logística es la técnica más básica y presenta mejora respecto a la institución. Con el resto de técnicas se espera mejorar en mayor o menor medida los resultados de logística, pero si la regresión logística no hubiese presentado mejora entonces habría que haber creado nuevas variables y pensado en más soluciones.

La mejor candidata a convertirse en variable respuesta es ***Fr5*** ya que sus modelos son los que mayor grado de acierto presentan y el que mayor porcentaje de mejora ofrece con diferencia para las 158 visitas mensuales (capacidad de la institución). Por tanto, la variable respuesta pasa a ser *Fr5* que tomará valor 1 en un día si en los cinco días siguientes se le inspeccionó y se le detectó fraude, 0 si no se le realizó inspección en dichos días y -1 si se le realizó inspección, pero no se detectó ninguna anomalía. Es decir, **consideraremos que un comercio estaba siendo fraudulento cinco días atrás si hoy tuvo una inspección con anomalías**. Por último, es importante entender que el hecho de que la variable respuesta sea *Fr5* no quiere decir que dicha variable respuesta prediga bien el fraude solo cinco días más allá. Recordemos que ha sido la que mejor ha funcionado en los dos años de validación.

Tabla 6.3.2. Distribución de la variable respuesta 'Fr5' en los datos.

		Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Fraude	**-1**	37003	3.57	37003	3.57
	0	987147	95.17	1024150	98.73
	1	13137	1.27	1037287	100

7. Modelos de predicción

7.1. Introducción

Tras describir la metodología y las variables a utilizar, es hora de pasar a la acción y construir los modelos con los cuales se intentará lograr un objetivo muy importante del presente trabajo que no es otro que el de calcular la probabilidad de que un comercio sea fraudulento para que la institución pueda investigar o inspeccionar a aquellos que mayor probabilidad tengan y aumentar la efectividad de sus campañas.

Para ello se usarán las técnicas estadísticas descritas: ***regresión logística***, ***redes neuronales***, ***random forest, gradient boosting*** y ***ensamble de modelos*** con las cuales se construirán diferentes modelos con los datos de **entrenamiento**. Estos modelos se evaluarán con los datos de **validación** (test) generando las tablas resúmenes y con ellas y a través de las medidas de comparación se elegirá el mejor modelo predictivo de cada técnica y posteriormente el mejor modelo general.

El conjunto de datos de entrenamiento abarca los 1096 días (tres años) siguientes a los 365 suprimidos tras crear las variables explicativas y los 731 días restantes formarán el conjunto de datos

de validación. Cabe mencionar que para el conjunto de entrenamiento sólo se han seleccionado a los clientes a los que se les haya inspeccionado en ese tiempo, es decir**, no se ha considerado la categoría 0** de la variable respuesta ya que al no conocer el resultado de la inspección no pueden usarse para generar patrones de comportamiento. En el archivo de validación sí que se incluye dicha categoría ya que debe simular el comportamiento real cuando el modelo se ponga en funcionamiento y en ese caso no conoceremos nada de la variable respuesta. En cualquier caso, la precisión es idéntica quitándolos o no, lo único que varía es la información sobre el número de usuarios visitados.

El motivo de realizar una partición de la base de datos es evitar el **sobreajuste** y poder valorar cómo funcionan realmente los modelos al no utilizar para dicha validación datos con los que han sido construidos los modelos (capacidad de generalización de los modelos). **La partición temporal de los archivos de entrenamiento y validación se realiza así ya que es como se hará en la vida real**, es decir, con los datos presentes entrenaremos los modelos y con los futuros veremos si hemos acertado o no, manteniendo el orden temporal de los datos.

Tabla 7.1.1. Frecuencia de Fr5 en los datos de entrenamiento.

		Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Fraude	**-1**	20540	72.31	20540	72.31
	1	7865	27.69	28405	100

Tabla 7.1.2. Frecuencia de Fr5 en los datos de validación.

		Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Fraude	**-1**	16463	3.99	16463	3.99
	0	391359	94.74	407822	98.72
	1	5272	1.28	413094	100

Como podemos observar en las tablas, **la clase objetivo (1) de la variable respuesta presenta una frecuencia relativa mucho menor que la otra clase**. Esto es un problema muy común en situaciones de fraude de este tipo, donde se dispone de muchas observaciones, pero pocas de ellas resultaron fraudulentas. Un problema evidente de este tipo de datos es que se dispone de poca información para representar la clase objetivo y además la heterogeneidad suele ser mayor en los sucesos *raros* que en los *no raros* lo que unido a la poca información dificulta aún más su representación. Por otra parte, los métodos de selección de variables y de construcción de modelos pueden no funcionar bien para este tipo de datos puesto que hay un sesgo implícito hacia la clase mayor.

Para solucionar este problema existen técnicas de balanceo como ***oversampling*** (remuestrar la clase minoritaria) y ***undersampling*** (eliminar observaciones de la clase mayoritaria). No obstante, eliminar información nunca suele ser bueno, y crear observaciones repetidas de la clase minoritaria tampoco puesto que no aporta información nueva y puede llevar a sesgos indeseados de la clase minoritaria. En realidad, no importa tanto la frecuencia relativa de la clase objetivo si no el número absoluto de observaciones de dicha clase. En este caso, disponemos de 7865 observaciones fraudulentas en los

datos de entrenamiento que se consideran suficientes para poder encontrar patrones de comportamiento por lo que no aplicaremos ninguna de las dos técnicas anteriores y evitaremos así distorsionar la realidad.

7.2. Regresión logística

7.2.1. Metodología y construcción de modelos

La base de datos cuenta con muchas variables explicativas por lo que es necesario determinar aquellas variables que están más correlacionadas con la variable respuesta y que por tanto más aportarán a la explicación de la misma. Esto se hará con los **métodos de selección de variables** y que pueden ser:

- **Forward**: se parte de un modelo que no contiene ninguna variable explicativa y se comienza incluyendo la variable más correlacionada con la variable respuesta. A continuación, se añade la segunda más correlacionada y así sucesivamente hasta que entre las variables no seleccionadas ninguna discrimine de forma significativa.
- **Backward**: actúa de forma inversa. Se parte de un modelo que incluye todas las variables explicativas consideradas y se van eliminado del modelo una a una según su capacidad explicativa hasta que todas las variables restantes en el modelo discriminan significativamente.
- **Stepwise**: es una combinación de los dos algoritmos anteriores. En primer lugar, se procede como en el método forward, pero a diferencia de éste en el que cuando una variable entra en el modelo ya no puede volver a salir, en el método stepwise es posible que la inclusión de una nueva variable haga que otra que estaba en el modelo resulte redundante y sea eliminada del modelo.

Dentro de estos métodos de selección de variables a su vez irá variando otro parámetro: **el p-valor**, valor a partir del cual se determinará si una variable es significativa o no en el modelo. Podrá tomar los siguientes 8 valores: 0.2, 0.1, 0.05, 0.01, 0.001, 0.0001, 0.00001 y 0.000001. Este p-valor se conoce como probabilidad de entrada en el método forward y probabilidad de salida en el método de backward. El método stepwise combina ambas probabilidades.

Otro parámetro que cambiará será la **transformación** que podrá tomar los valores: *logit*, *probit* y *cloglog*. El **número de parámetros** a estimar es igual al número de variables explicativas que resulten significativas más la constante.

Por último, la regresión logística (ver apartado 5.1.1.) proporcionará una probabilidad (entre 0 y 1) con la cual se podrán clasificar las observaciones como fraudulentas o como no fraudulentas. Para hacer esta catalogación se elige un **punto de corte** (*pmin*) hasta el cual se considerará que la observación no es fraudulenta y otro punto de corte (*pmax*) hasta (desde *pmin*) el cual se considerará que la observación es fraudulenta. El *pmin* variara de 0.1 a 0.7 en intervalos de 0.1 y el *pmax* de 0.8 a

1 con los mismos intervalos. Este parámetro no es propio de la regresión logística, si no que se añade a posteriori por parte del investigador.

Debido a que la naturaleza de la mayoría de las variables explicativas es de tipo continua, se va a ajustar un modelo inicialmente **sólo con los efectos principales**, es decir, no se van a considerar las interacciones de orden dos ni superiores para no añadirle demasiada complejidad al modelo.

En total se estiman 5037 modelos resultantes de todas las combinaciones posibles de los diferentes parámetros que pueden ir variando en la regresión logística.

7.2.2. Selección del modelo óptimo

A continuación, se muestra una tabla para comparar los modelos construidos de regresión logística de entre los cuales sólo nos hemos quedado con aquellos que tuvieran más de 158 investigaciones diarias. En ella aparecen los 15 modelos con mayor bondad de ajuste ordenados de mayor a menor. La tabla está construida sobre el archivo de validación.

Tabla 7.2.2.1. Resumen mejores modelos de regresión logística

Método	Función	VP	FP	FN	VN	Visitas por ejecución	Parámetro	Bondad Ajuste	p entrada	p salida	pmin	pmax
STEPWISE	Probit	3517	5797	1755	10666	193	26	0.3776	0.001	0.2	0.3	1
STEPWISE	Probit	3517	5797	1755	10666	193	26	0.3776	0.001	0.1	0.3	1
STEPWISE	Probit	3517	5797	1755	10666	193	26	0.3776	0.001	0.05	0.3	1
STEPWISE	Probit	3517	5797	1755	10666	193	26	0.3776	0.001	0.01	0.3	1
FORWARD	Probit	3517	5797	1755	10666	193	26	0.3776	0.001	0	0.3	1
STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.2	0.001	0.3	1
STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.2	0.0001	0.3	1
STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.1	0.001	0.3	1
STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.1	0.0001	0.3	1
STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.05	0.001	0.3	1
STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.05	0.0001	0.3	1
STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.01	0.001	0.3	1
STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.01	0.0001	0.3	1

STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.001	0.001	0.3	1
STEPWISE	Logit	3519	5806	1753	10657	189	18	0.3773	0.001	0.0001	0.3	1

Se puede ver que los cinco primeros modelos son idénticos y los diez restantes también (solo varían los p - valores pero el modelo final es el mismo). Los cinco primeros presentan una bondad de ajuste del 37.8% pero, sin embargo, vamos a considerar, siguiendo el criterio de parsimonia, como el **mejor modelo de logística al sexto modelo en la tabla** ya que presenta una bondad de ajuste muy similar, pero 8 parámetros menos a estimar. Sus características son:

- Método de selección de variables: Stepwise
- Función o transformación: Logit
- Número de parámetros a estimar: 18
- P-valor de entrada: 0.2
- P-valor de salida: 0.001
- Puntos de corte:
  - Pmin: 0.3
  - Pmax: 1

Por tanto, el modelo clasificará como fraudulentos a aquellos comercios que obtengan una probabilidad mayor a 0.3. Como es lógico, este punto de corte está cercano a la frecuencia de unos en los datos de entrenamiento (27.69%). A partir de los valores de la tabla se pueden calcular la sensibilidad y la especificidad del modelo óptimo:

$$Sensibilidad = \frac{3519}{3519 + 1753} = 0.667 \qquad Especificidad = \frac{10657}{10657 + 5806} = 0.647$$

Es decir, la probabilidad de que el modelo clasifique correctamente a los comercios fraudulentos es de un 66.70% y de que clasifique correctamente a los comercios no fraudulentos es de un 64.70% mientras que la tasa de verdaderos positivos es de 0.3773 (bondad de ajuste).

A continuación, se presenta una tabla con las estimaciones de los coeficientes asociados a las variables que han resultado significativas en el modelo seleccionado, así como sus errores estándar, el estimador de Wald (que permite contrastar si una variable es significativa en el modelo), el p-valor asociado a dicho contraste y la estimación del OR. Con todo ello y aplicando la ecuación 5.1.1.3. lograremos la estimación de la probabilidad deseada.

Tabla 7.2.2.2. Estimaciones regresión logística óptima

Variable	$\widehat{\beta}_i$	Error estand.	Wald	p - valor	OR	Intervalo confianza OR (95%)	
Constante	-1.1794	0.1964	36.0681	<.0001	-	-	-
NumDatosVentas14	-0.0399	0.00843	22.336	<.0001	0.961	0.945	0.977

CambiosPonderadosVentas	0.2592	0.0433	35.8552	<.0001	1.296	1.191	1.411
CostesMedios7	-0.0423	0.00429	97.3728	<.0001	0.959	0.951	0.967
CostesDesv365	-0.3052	0.0293	108.2338	<.0001	0.737	0.696	0.781
CostesDesv7	0.1974	0.0264	55.9680	<.0001	1.218	1.157	1.283
EnergíaMedia365	0.0193	0.00435	19.8011	<.0001	1.020	1.011	1.208
EnergíaDesv365	0.1139	0.0177	41.3519	<.0001	1.121	1.082	1.160
PorcentajeMissEnergía7	-1.1498	0.1624	50.0963	<.0001	0.317	0.230	0.435
NumMissEnergía365	-0.00246	0.000297	68.7840	<.0001	0.998	0.997	0.998
CambiosPonderadosCostes	-0.1520	0.0284	28.6250	<.0001	0.859	0.812	0.908
MaxCambioImpuestos	0.0431	0.00827	27.1056	<.0001	1.044	1.027	1.061
NumVisitas365	0.2420	0.0102	560.7958	<.0001	1.274	1.249	1.300
NumFraude91	0.2631	0.0378	48.3988	<.0001	1.301	1.208	1.401
NumNoFraude91	-0.9069	0.0350	672.7107	<.0001	0.404	0.377	0.432
NumFraude28	2.5697	0.0838	940.0221	<.0001	13.062	11.083	15.394
NumNoFraude28	-1.6435	0.0802	420.2393	<.0001	0.193	0.165	0.226
CostesMedios91	7.4363	0.3122	567.4155	<.0001	1.244	1.147	1.754

Como es lógico, todas las variables son significativas, con un nivel de confianza del 99%. Esto se corrobora mirando no solo los p-valores si no también los intervalos de confianza para los OR. Si el intervalo de confianza de éste contiene al 1 entonces la variable no es significativa, pero en este caso ninguno incluye al 1.

Debido al origen cuantitativo de las variables su interpretación no interesa en este caso ya que un OR en este caso nos indica cuanto más probable, o improbable, es que un resultado esté presente en los individuos con $X=x$ que en los individuos con $X=x-1$. Sin embargo, se puede matizar el hecho de que las variables con una estimación positiva indica que un aumento en dichas variables aumenta la probabilidad de cometer fraude (si el resto de variables se mantienen constantes) mientras que aquellas que presentan una estimación negativa indica que un aumento en dichas variables disminuye la probabilidad. Así, por ejemplo, la variable *NumDatosVentas14,* que indica el número de datos válidos en los últimos catorce días, presenta un coeficiente negativo lo que significa que, a mayor número de datos válidos en los últimos catorce días, menor es la probabilidad de cometer fraude. En cambio, la variable *NumVisitas365*, que indica el número de inspecciones realizadas en el último año, presenta un coeficiente positivo lo que significa que, a mayor número de inspecciones, mayor es la probabilidad de cometer fraude. Como vemos, parecen lógicos los resultados.

También es interesante la interpretación de variables relacionadas como pueden ser *NumFraude91* y *NumFraude28*. Al estar relacionadas (solo cambia la temporalidad de agrupación), la interpretación se hace de forma conjunta. En este caso *NumFraude28* presenta un coeficiente mayor que *NumFraude91* lo que indica que a la hora de predecir el fraude, son más importantes los fraudes cercanos que los cometidos hace 3 meses (aunque estos también sean importantes).

7.3. Random forest

7.3.1. Metodología y construcción de modelos

Al igual que en el resto de técnicas del trabajo, random forest posee unos criterios de parada o parámetros a controlar con los que se van a construir los modelos y que son los siguientes:

- **Número mínimo de observaciones por nodo final u hoja**: indica el número mínimo de observaciones que ha de contener un nodo para ser considerado un nodo final u hoja. En nuestro caso puede variar entre 1000, 700, 500, 100 y 50.
- **Número máximo de árboles a crear**: indica el número de iteraciones que se va a repetir el algoritmo. En nuestro caso se fija en 500 aunque con 200 suele ser suficiente.
- **Número de variables a tener en cuenta para dividir los datos en cada nodo**: indica el número de variables que se seleccionan aleatoriamente para crear el subconjunto del cual se decidirá qué variable se usa para dividir los datos en el nodo. En nuestro caso varía entre 20, 50 y 80.
- **Porcentaje de la base de datos usada para cada árbol**: indica el porcentaje de la base de datos original (datos de entrenamiento) que se usará para constituir la muestra con la que se construirá cada árbol. Esta muestra cambia para cada árbol. En nuestro caso está fijada en un 80%.
- **p - valor**: necesario para generar una regla de decisión. En nuestro caso puede tomar los valores 0.2, 0.1, 0.05 o 0.01. Cuanto más alto, árboles menos complejos (más sesgo, menos varianza).
- **Punto de corte**: el modelo clasificará como fraudulentos a todos aquellos clientes que tengan una probabilidad mayor que dicho punto de corte el cual oscila entre 0.2 y 0.7 en intervalos de 0.01.
- **Profundidad máxima**: distancia, en número de nodos, entre el nodo raíz y la hoja más alejada. En nuestro caso se fija en 50 que es el máximo que admite el software.

Por último, mencionar, como ya se comentó en la metodología (ver apartado 5.1.4.), que los árboles que se estimaran en este trabajo son **árboles de clasificación binarios** donde cada nodo padre se divide en dos nodos descendientes y que el criterio para la selección del punto de corte y variable es el **índice de Gini**.

En total se estiman 3059 modelos resultantes de todas las combinaciones posibles de los diferentes parámetros que pueden ir variando en random forest. Cabe destacar que todos los árboles creados partían de las 173 variables explicativas ya que las técnicas basadas en árboles, al igual que la logística con stepwise, backward y forward, poseen criterios para seleccionar variables. No obstante, se realizó un modelo de random forest con las 17 variables significativas del modelo óptimo de regresión logística pero los resultados fueron peores que los del apartado que se muestra a continuación por lo que no se muestran en el presente trabajo.

7.3.2. Selección del modelo óptimo

Los distintos modelos resultantes han sido ordenados de manera descendiente en función de su bondad de ajuste y en la siguiente tabla se presentan los 15 mejores modelos en términos de bondad de ajuste con sus valores correspondientes para los diferentes parámetros que realicen las visitas mínimas por ejecución. De nuevo, la tabla está construida sobre el archivo de validación.

Tabla 7.3.2.1. Resumen mejores modelos de Random Forest

VP	FP	FN	VN	Decisiones	Visitas por ejecución	Bondad Ajuste	P corte	Vars to try	Tamaño hoja	p - valor
3282	4576	1990	11887	7721	159	0.41766	0.32	80	1000	0.2
3282	4576	1990	11887	7721	159	0.41766	0.32	80	1000	0.1
3282	4576	1990	11887	7721	159	0.41766	0.32	80	1000	0.05
3282	4576	1990	11887	7721	159	0.41766	0.32	80	1000	0.01
3157	4460	2115	12003	7997	158	0.41447	0.30	20	1000	0.2
3157	4460	2115	12003	7997	158	0.41447	0.30	20	1000	0.1
3157	4460	2115	12003	7997	158	0.41447	0.30	20	1000	0.05
3157	4460	2115	12003	7997	158	0.41447	0.30	20	1000	0.01
3200	4546	2072	11917	16623	159	0.41312	0.30	20	500	0.2
3200	4546	2072	11917	16623	159	0.41312	0.30	20	500	0.1
3200	4546	2072	11917	16623	159	0.41312	0.30	20	500	0.05
3200	4546	2072	11917	16623	159	0.41312	0.30	20	500	0.01
3180	4543	2092	11920	11782	158	0.41176	0.30	20	700	0.2
3180	4543	2092	11920	11782	158	0.41176	0.30	20	700	0.1
3180	4543	2092	11920	11782	158	0.41176	0.30	20	700	0.05

La variable **decisiones** que aparece en la tabla resumen indica el número de divisiones que se realizan en cada modelo. En realidad, en la tabla solo hay cuatro modelos diferentes ya que en muchos de ellos solo varía el p – valor respecto al modelo anterior. En este caso parece claro que el mejor modelo es el que presenta una bondad de ajuste de 0.41766 y la siguiente estructura:

- Número mínimo de observaciones por nodo final u hoja: 1000
- Número de variables a tener en cuenta para dividir los datos en cada nodo: 80
- P-valor: 0.2 (aunque es indiferente puesto que con 0.1, 0.01 y 0.05 se obtienen los mismos resultados)
- Punto de corte: 0.32 (de nuevo, cercano a la frecuencia relativa original)
- Visitas por ejecución: 159.

Como en la regresión logística, a partir de los valores de la tabla se pueden calcular la sensibilidad y la especificidad del modelo óptimo:

$$Sensibilidad = \frac{3282}{3282 + 1990} = 0.6225 \qquad Especificidad = \frac{11887}{11887 + 4576} = 0.7220$$

Es decir, la probabilidad de que el modelo clasifique correctamente a los comercios fraudulentos es de un 62.25% y de que clasifique correctamente a los comercios no fraudulentos es de un 72.2% mientras que la tasa de verdaderos positivos es de 0.41766 (bondad de ajuste o precisión). Es decir, de todas las observaciones que el modelo clasifica como fraudulentas, un 41.766% lo son de verdad.

Como ya se comentó en la descripción de las técnicas, al contrario que la regresión logística, las técnicas como random forest, gradient boosting y redes neuronales presentan la desventaja de que no se pueden interpretar. No ofrecen información sobre la aportación de las diferentes variables input al modelo a través del signo y magnitud de los parámetros correspondientes, como sí ocurre en métodos clásicos como la regresión logística. Sin embargo, se elaboró un gráfico con la importancia de las variables usadas para crear el modelo de random forest óptimo. La importancia[1] de las variables se mide a través del número de veces que una variable se usó para dividir un nodo. A mayor número de veces usada, mayor importancia de la variable en el modelo creado. En el gráfico las variables están ordenadas de menor a mayor importancia y se ha incluido en el anexo 12.3. donde también se puede consultar la tabla con la que ha sido elaborado el grafico. Como se puede apreciar, cinco de las variables más importantes son *CambiosPonderadosCostes, NumVisitas365, NumFraude91, NumNoFraude91* y *NumFraude28* que también aparecían como significativas en la regresión logística óptima.

7.4. Gradient Boosting

7.4.1. Metodología y construcción de modelos

Nuevamente, para empezar, se van a detallar los parámetros que pueden ir variando dentro de la presente técnica, así como los posibles valores que podrán tomar a la hora de la construcción de los modelos. Como vemos, son muy parecidos a los de random forest, lo cual es lógico porque ambas técnicas están basadas en árboles de decisión.

- **Número mínimo de observaciones por nodo final u hoja**: al igual que en random forest, indica el número mínimo de observaciones que ha de contener un nodo para ser considerado un nodo final u hoja. En nuestro caso puede variar entre 1000, 700, 500, 100 y 50.
- **Parámetro de regularización**: puede tomar los valores 0.001, 0.01, 0.1, 0.2 y 0.3. Indica la rapidez con la que converge. Cuanto más alto, más rápido converge, pero va dando grandes bandazos por lo que es poco preciso. Si se pone muy bajo, entonces será más preciso, pero más lento por lo que se necesitarán mas iteraciones.

[1] Otra forma de medir la importancia de una variable en RF consiste en elaborar un modelo con la variable y otro sin ella y comparar resultados. Si ofrecen resultados muy diferentes es que la variable es importante.

- **Iteraciones**: para variables objetivo nominales, como en nuestro caso, se crea un árbol independiente para cada categoría de la variable objetivo en cada iteración. El número de iteraciones se fija en 300.
- **Punto de corte**: el modelo clasificará como fraudulentos a todos aquellos clientes (en los datos de validación, lógicamente) que tengan una probabilidad mayor que dicho punto de corte el cual oscila entre 0.2 y 0.7 en intervalos de 0.01.
- **Profundidad máxima**: distancia, en número de nodos, entre el nodo raíz y la hoja más alejada. En nuestro caso se fija en 50 que es el máximo que admite el software.

Como ya se comentó en la metodología (ver apartado 5.1.5.), los modelos que se utilizarán para mejorar las predicciones del modelo en la iteración anterior son los árboles de decisión. En concreto, para nuestro caso se usan arboles de clasificación binarios.

En total se estiman 1274 modelos resultantes de todas las combinaciones posibles de los diferentes parámetros que pueden ir variando en gradient boosting, partiendo de las 173 variables explicativas, como en random forest.

7.4.2. Selección del modelo óptimo

Se filtran los 1274 modelos de forma que nos quedamos solo con aquellos que cumplen con el número mínimo necesario de ejecuciones diarias en los dos años de validación y se ordenan de mayor a menor por la bondad de ajuste (calculada sobre el archivo de validación lógicamente) de forma que obtenemos la siguiente tabla (solo se muestran los 15 mejores modelos) para su comparación:

Tabla 7.4.2.1. Resumen mejores modelos de Gradient Boosting

VP	FP	FN	VN	Decisiones	Visitas por ejecución	Bondad Ajuste	P corte	Tamaño hoja	Param. regu
3286	4716	2004	11747	7474	159	0.4093	0.29	500	0.001
3181	4688	2091	11775	39269	162	0.4042	0.29	100	0.001
3022	4568	2250	11895	78610	160	0.3981	0.31	50	0.01
3454	5245	1818	11218	5078	169	0.3971	0.28	700	0.001
2817	4296	2455	12167	5313	158	0.3960	0.37	700	0.1
3403	5201	1869	11262	3651	169	0.3955	0.28	1000	0.001
3068	4704	2204	11759	38942	158	0.3948	0.32	100	0.1
3214	4931	2058	11532	7473	160	0.3946	0.33	500	0.1
3046	4687	2226	11776	79274	160	0.3939	0.29	50	0.001
3089	4761	2183	11702	78610	169	0.3935	0.3	50	0.01
2863	4438	2409	12025	5313	163	0.3921	0.36	700	0.1
2901	4505	2371	11958	3652	163	0.3917	0.36	1000	0.1

3152	4900	2120	11563	38942	164	0.3915	0.31	100	0.01
3390	5279	1882	11184	39269	179	0.3910	0.28	100	0.001
2767	4309	2505	12154	3625	158	0.3910	0.37	1000	0.2

El gradient boosting es un algoritmo que en la actualidad está muy de moda porque suele presentar buenos resultados y mejorar al random forest. Sin embargo, en este caso, si recordamos los resultados obtenidos por random forest, gradient boosting presenta unos peores. El mejor modelo de gradient boosting alcanza una precisión de 0.4093 mientras que el de random forest conseguía una de 0.4177. Vemos como el mejor modelo presenta un parámetro de regularización bajo, por lo que es más preciso y al fijar un numero de iteraciones alto consigue un sesgo menor que para parámetros más altos. Por tanto, las características del mejor modelo de gradient boosting son:

- Punto de corte a partir del cual se considerada una observación como fraudulenta: 0.29
- Tamaño de los nodos finales u hojas: 500
- Parámetro de regularización: 0.001
- Visitas por ejecución: 159

Como en todos los modelos anteriores, a partir de los valores de la tabla se pueden calcular la sensibilidad y la especificidad del modelo óptimo:

$$Sensibilidad = \frac{3286}{3286 + 2004} = 0.6212 \qquad Especificidad = \frac{11747}{11747 + 4716} = 0.7135$$

Es decir, la probabilidad de que el modelo clasifique correctamente a los comercios fraudulentos es de un 62.12% y de que clasifique correctamente a los comercios no fraudulentos es de un 71.35% mientras que la tasa de verdaderos positivos es de 0.4093 (bondad de ajuste).

Al igual que en random forest, se elabora un gráfico de la importancia de las variables en el modelo óptimo que puede consultarse en el anexo 12.4. La importancia, de nuevo, viene medida por el número de veces que una variable se usó en las decisiones. Como se puede ver en dicho gráfico, las variables más importantes son *CostesMedios91, CostesDesv365* y *NumVisitas365*. La mayoría de las variables más importantes de gradient boosting aparecen también en los modelos óptimos de regresión logística y random forest como significativas o importantes. Y hay otras que, aunque no sean la misma variable, ofrecen información muy parecida. Por ejemplo, *CostesDesv14* aparece como una variable importante en gradient boosting mientras que en regresión logística y random forest es *CostesDesv7*.

7.5. Redes neuronales

7.5.1. Metodología y construcción de modelos

Empezamos hablando sobre la selección de variables que siempre es un quebradero de cabeza y que en redes neuronales (ver apartado 5.1.2.) es un problema no resuelto y un punto flaco de las mismas.

Tanto en regresión logística como en random forest y gradient boosting se introdujeron inicialmente todas las variables explicativas disponibles, ya que éstas poseen una característica que las redes neuronales no y es el hecho de poder seleccionar aquellas variables explicativas que sirven para explicar mejor la variable respuesta a través de los métodos de selección de variables u otra serie de criterios.

No obstante, los métodos stepwise, forward y backward presentan un sesgo por el planteamiento lineal que llevan implícito por lo que también es adecuado probar conjuntos de variables no afectados por dichos planteamientos como pueden ser los obtenidos por técnicas basadas en árboles como random forest o gradient boosting. Si observamos las variables más importantes de dichos algoritmos, vemos como todas ellas son muy parecidas a las obtenidas mediante el método stepwise de logística, cambiando en algunos casos solo la temporalidad de la *agrupación*. Por ello, se decide construir las redes utilizando solo el conjunto de variables obtenido por stepwise. Por lo tanto, el número de nodos en la capa de entrada es de 17, uno por cada variable explicativa.

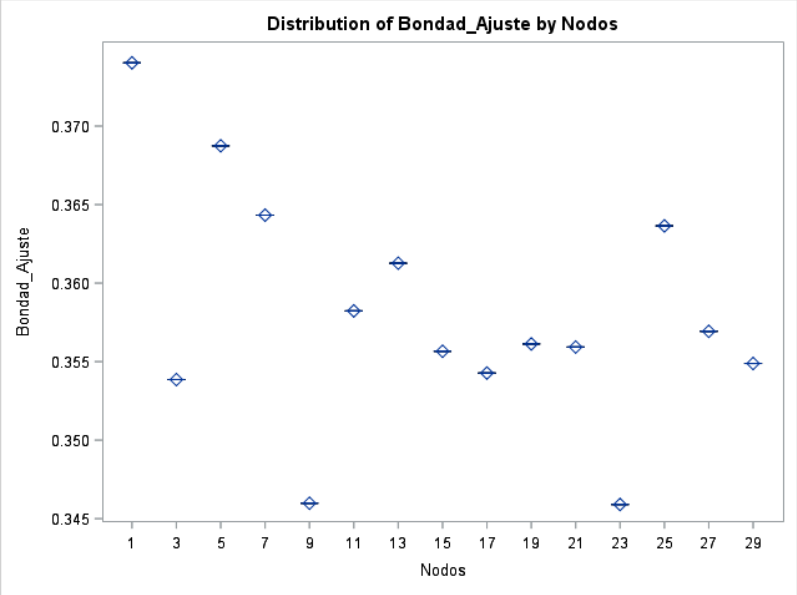
Al igual que con el resto de técnicas estadísticas, las redes neuronales poseen ciertos parámetros que pueden ir variando y que nos proporcionarán diferentes modelos cada vez con distintas bondades de ajuste. Algunos de ellos son:

- **Nodos en la/s capa/s oculta/s**: sólo se utilizará **una capa oculta** ya que según los teoremas de aproximación suele ser suficiente. Para conocer cuál es el rango de nodos óptimo para nuestros datos, se elabora el siguiente gráfico en el que calculamos la bondad del ajuste (sobre el archivo de validación) por diferentes redes con los nodos variando de 1 hasta 29 de 2 en 2. Se usa el algoritmo de optimización de Levenberg – Marquardt y la función de activación tangente hiperbólica, de momento, ya que es la combinación que mejor suele funcionar.

  Un elemento muy importante de las redes neuronales son los **pesos iniciales** con los cuales el algoritmo de aprendizaje comienza a optimizar la función objetivo. En ocasiones, debido a una mala elección de dichos pesos, el algoritmo puede quedarse estancado en mínimos locales. Para intentar mitigar esta situación, haremos training – test repetido (3 veces; dado el carácter temporal de los datos los conjuntos de entrenamiento y validación son los mismos en cada iteración, solo varia la semilla de los pesos iniciales) de forma que cada red tenga una semilla diferente para inicializar dichos pesos.

  Al igual que en la regresión logística y en random forest, las redes neuronales clasificaran a las observaciones dependiendo de un **punto de corte** elegido a posteriori por el investigador. Así, una observación a la que la red neuronal le ha otorgado una probabilidad de cometer fraude mayor que dicho punto de corte será considerada como fraudulenta. Este punto de corte se fijará de momento en 0.28 puesto que es la proporción de la categoría de interés en los datos de entrenamiento.

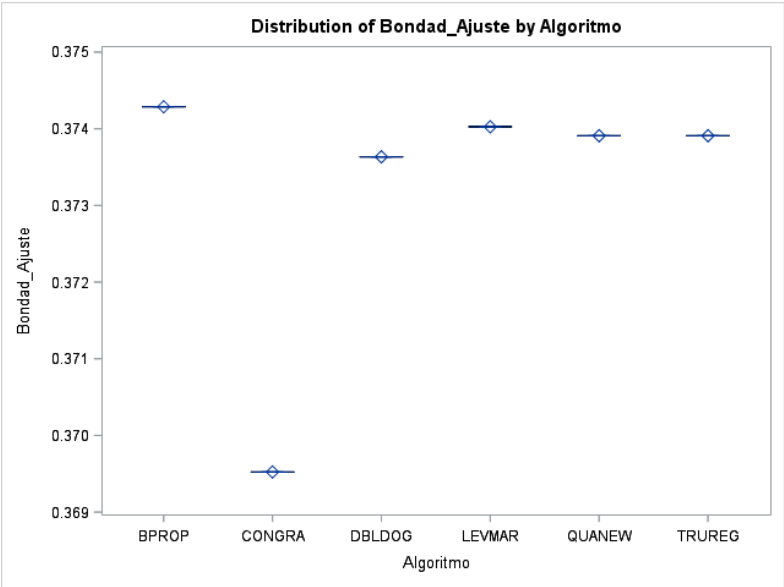
Ilustración 7.5.1.1. Distribución de la bondad de ajuste en función del número de nodos.



Lo primero que llama a la vista es el hecho de que, a pesar de variar la semilla de los pesos iniciales para cada red en cada una de las tres iteraciones, se obtiene la misma bondad de ajuste por lo que en este caso no parece que los mínimos locales sean un problema. En segundo lugar, vemos como los mejores resultados se obtienen para números de nodos pequeños.

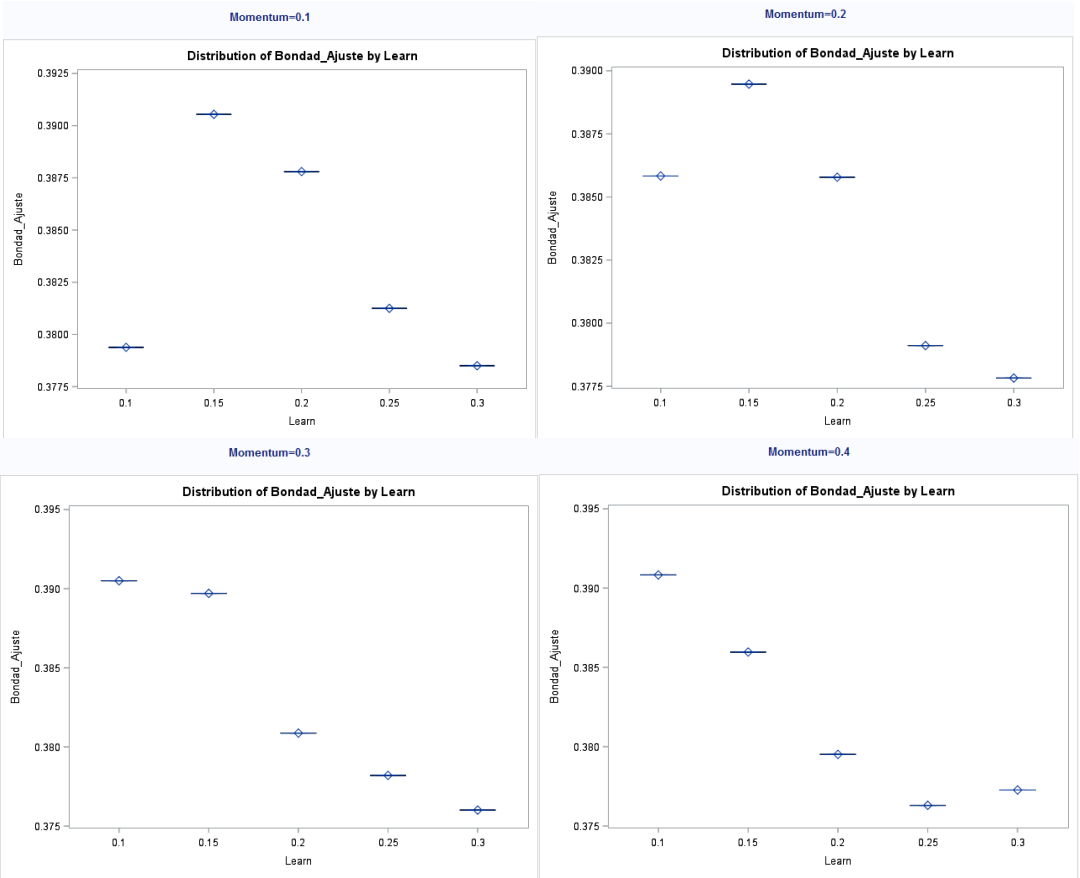
- Otro parámetro de la red neuronal es el propio **algoritmo de optimización**. Al igual que antes, en un estudio preliminar con training test repetido 3 veces y variando aleatoriamente la semilla de los pesos iniciales, se prueban el *BPROP, LEVMAR, QUANEW, TRUREG, DBLDOG* y *CONGRA*. Se establece que el número de nodos sea de 1 por lo visto en el gráfico anterior y de momento se utiliza la tangente hiperbólica como función de activación. El punto de corte se deja en 0.28 como antes.

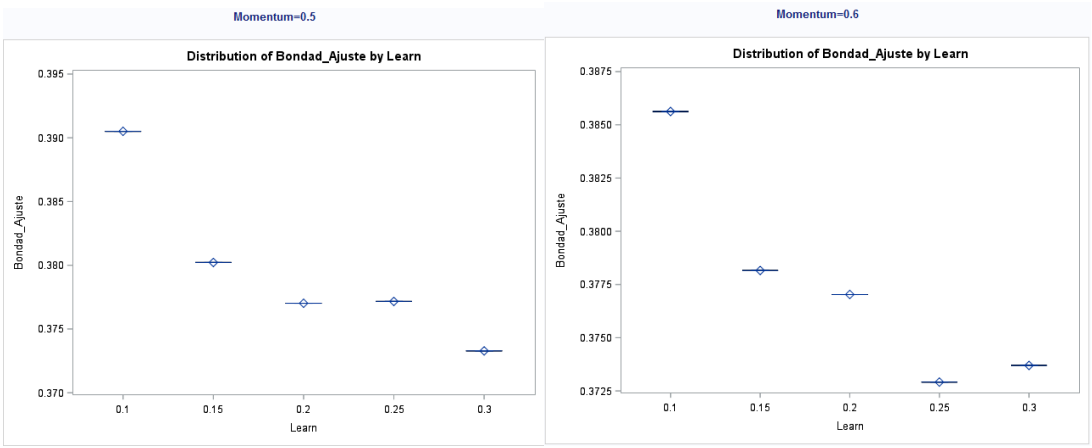
Ilustración 7.5.1.2. Distribución de la bondad de ajuste en función del algoritmo de optimización.



Vemos como los mejores resultados se obtienen con el algoritmo *BPROP*, seguido de cerca por *LEVMAR, QUANEW* y *TRURE*G. El algoritmo *CONGRA* es el que peor funciona con diferencia. Si recordamos, en el apartado 5.1.2. en el que se explicaba el funcionamiento de las redes neuronales, se explicaba a su vez como funcionaban los algoritmos de aprendizaje en 5 sencillos pasos. En el 4º paso se decía "retocar los valores de los pesos iniciales en esa dirección de decrecimiento". Pues bien, para retocar dichos pesos se tiene en cuenta un *learning rate*, de forma que los pesos se varían según ese learning rate. En el caso del *BPROP*, a parte del learning rate también influye el *momentum* (<1). Si el learning rate refleja en qué medida vamos a cambiar los pesos en cada iteración, el momentum lo que hace es que en cada iteración los pesos se vayan variando menos para acercarse *con cuidado* al mínimo de la función objetivo. Únicamente para el *BPROP* el SAS nos da la posibilidad de cambiar dichos learning rate y momentum, por lo que dado que ha sido el algoritmo que mejor ha funcionado, se varían dichos parámetros para ver si existe alguna alternativa mejor a la por defecto del SAS (learning rate=0.1 y momentum=0.9). Se utiliza la función de activación *tangente hiperbólica*.

Ilustración 7.5.1.3. Distribución de la bondad de ajuste en función del learning rate y el momentum para BPROP.





Se puede observar como a medida que aumentamos el momentum, los resultados van siendo peores para cualquier learning rate. A la vista de los gráficos y corroborado por la tabla resultado (que no se adjunta), la mejor bondad de ajuste se produce con un learning rate de 0.1 y un momentum de 0.4 que serán los valores que tomarán dichos parámetros para el algoritmo *BPROP* del siguiente punto.

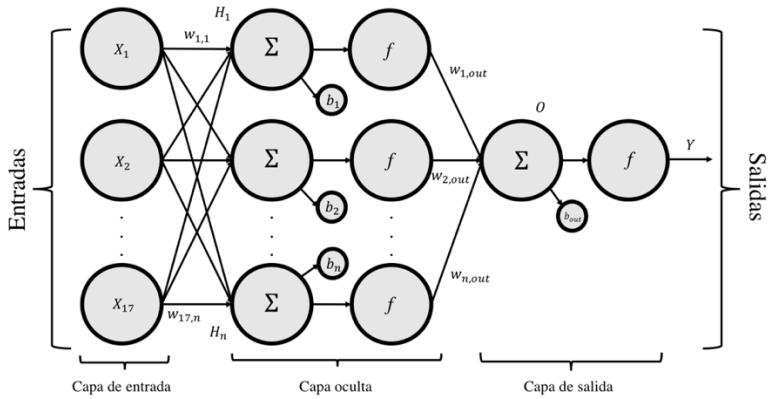
- **Función de activación**: al utilizarse por defecto la estandarización de las variables explicativas, las funciones con rangos pequeños suelen funcionar bien. En nuestro caso podrá tomar los valores: *tangente hiperbólica*, *exponencial*, *arco tangente, elliot o logística* para tener unos tiempos de ejecución razonables. Una red neuronal con la función de activación *logit* es equivalente a una regresión logística, pero en el caso de que la mejor red neuronal use como función de activación la *logit*, será preferible quedarse con la regresión logística ya que los resultados serán muy parecidos y es un modelo más robusto, estable e interpretable.

  En este caso, el punto de corte variará entre 0.2 y 0.45 con intervalos de 0.01. También es importante señalar que la **función de combinación** utilizada para todos los modelos es la función de combinación lineal, el **algoritmo de aprendizaje** o de optimización a usar podrá ser el *BPROP* (con los parámetros anteriormente señalados), *LEVMAR* o *QUANEW* y se establecen 1, 2, 3, 4 o 5 nodos en la capa oculta por lo visto en el estudio preliminar.

  Se estiman por tanto un total de **1950 modelos** de redes neuronales resultado de todas las combinaciones posibles en la que cada una tiene una semilla diferente para los pesos iniciales. Dado la gran cantidad de modelos, resulta inviable realizar training test repetido por lo que cada red solo se ajusta con una única semilla para los pesos iniciales por lo que se puede incurrir en el problema de los mínimos locales. No obstante, si recordamos, parece que para estos datos no se presenta ese problema. Además, al variar el punto de corte en intervalos tan pequeños, dos redes contiguas serán muy parecidas y cada una de ellas está estimada con una semilla de pesos iniciales diferente por lo que sí se mitiga en parte el efecto de los mínimos locales.

  La estructura de las redes construidas puede verse en la siguiente imagen.

Figura 7.5.1.1. Estructura redes neuronales construidas



Tras lo comentado anteriormente, se deduce que el **número de parámetros** a estimar en las redes neuronales que se van a construir en este trabajo dependen de la siguiente expresión:

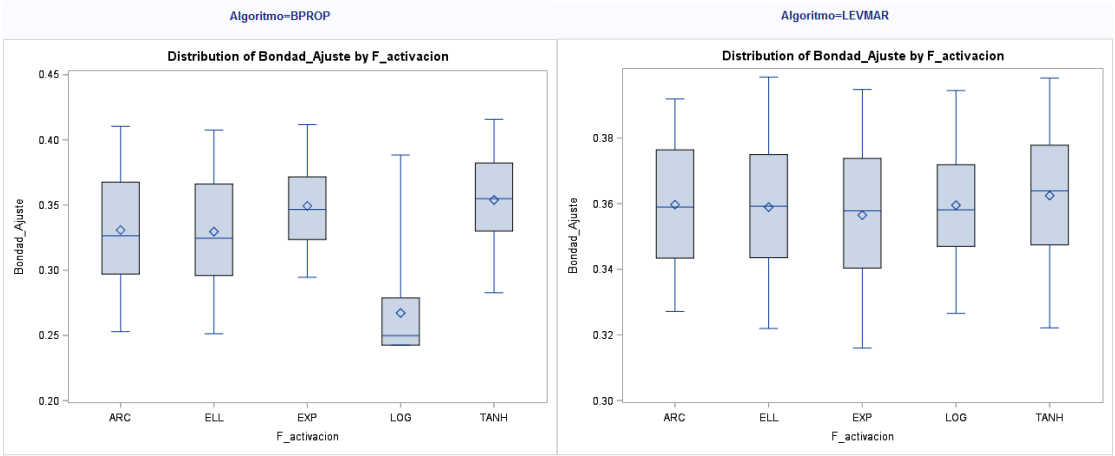
Ecuación 7.5.1.

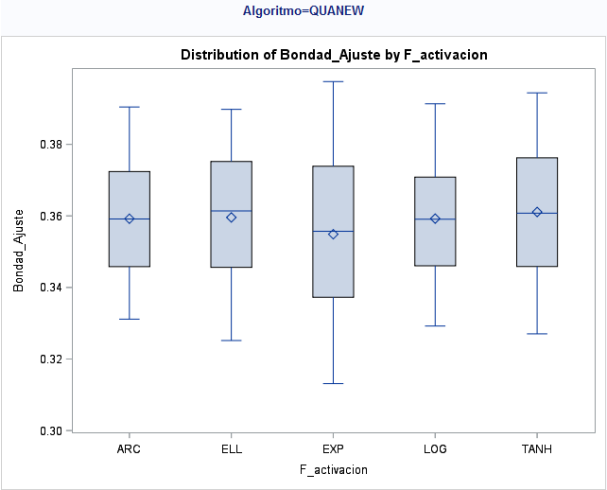
$$Nº\ parametros\ a\ estimar = (nº\ vars\ explicativas + 1) \times\ nº\ nodos\ en\ la\ capa\ oculta +\ nº\ nodos\ en\ la\ capa\ oculta\ + 1$$

7.5.2. Selección del modelo óptimo

En primera instancia se muestran los siguientes gráficos en los que se muestra la bondad de ajuste (precisión) obtenida por las distintas funciones de activación para los diferentes algoritmos de optimización (que varían en función del punto de corte y los nodos). Se han filtrado los modelos de forma que cumplan con los requisitos de visitas por ejecución.

Ilustración 7.5.2.1. Distribución de la bondad de ajuste en función de la función de activación y el algoritmo de optimización.





Como vemos, el algoritmo *BPROP* es sin duda el que mejor funciona con gran diferencia. Además, con dicho algoritmo las funciones de activación se comportan de forma más estable. Dentro del presente algoritmo, en relación acierto – varianza, la función de activación que mejor funciona es la *exponencial*, seguida de cerca por la *tangente hiperbólica*. Todo este análisis se tendrá en cuenta para elegir a continuación el mejor modelo de redes neuronales.

Al igual que con las dos técnicas multivariantes anteriores, con las redes neuronales operamos de la misma manera: se presenta a continuación una tabla construida sobre el archivo de validación a modo de resumen con los 15 mejores modelos (según la bondad del ajuste) de redes neuronales y que cumplan con las inspecciones mínimas.

Tabla 7.5.2.1. Resumen mejores modelos de redes neuronales

F activación	VP	FP	FN	VN	Visitas por ejecución	Bondad Ajuste	Nodos	P corte	Parámetros	Algoritmo
TANH	3027	4253	2245	12210	162	0.4158	1	0.31	20	BPROP
TANH	3119	4430	2153	12033	159	0.4132	3	0.31	58	BPROP
EXP	2988	4270	2284	12193	167	0.4117	3	0.31	58	BPROP
ARC	3126	4490	2146	11973	159	0.4105	2	0.33	39	BPROP
EXP	3017	4339	2255	12124	169	0.4101	1	0.31	20	BPROP
ARC	3086	4444	2186	12019	163	0.4098	1	0.3	20	BPROP
TANH	3209	4624	2063	11839	169	0.4097	1	0.3	20	BPROP
ELL	3181	4626	2091	11837	162	0.4075	2	0.33	39	BPROP
EXP	3110	4525	2162	11938	160	0.4073	4	0.31	77	BPROP
ELL	3184	4646	2088	11817	167	0.4066	1	0.3	20	BPROP
TANH	3191	4688	2081	11775	164	0.4050	2	0.35	39	BPROP
ELL	3123	4602	2149	11861	161	0.4043	3	0.3	58	BPROP
TANH	3245	4786	2027	11676	167	0.4040	3	0.3	58	BPROP

EXP	3250	4797	2022	11666	161	0.4039	5	0.31	96	BPROP
ARC	3143	4641	2129	11822	162	0.4038	3	0.3	58	BPROP

En línea con lo observado en los gráficos anteriores, vemos que los mejores modelos utilizan el algoritmo de backpropagation (*BPROP*). Vemos también que la red neuronal que presenta mayor bondad de ajuste tiene la siguiente estructura:

- Función de activación: *tangente hiperbólica*.
- Nodos en la capa oculta: 1
- Punto de corte: 0.31
- Número de parámetros a estimar: 20
- Algoritmo de optimización: *BPROP* (learning rate de 0.1 y momentum de 0.4)
- Bondad de ajuste: 0.4158

La tercera red neuronal también es interesante puesto que como vimos en los gráficos anteriores, la función *exponencial* presenta menor variabilidad que la *tangente hiperbólica* para el algoritmo *BPROP* y una bondad de ajuste muy parecida. Además, realiza más inspecciones mensuales. No obstante, presenta más nodos en la capa oculta lo que conlleva más parámetros a estimar. Puesto que la diferencia entre las funciones es mínima, dado que la función *tangente hiperbólica* ofrece mejores resultados nos quedaremos con la primera red neuronal como la mejor.

El SAS por defecto establece en 1000 el número máximo de iteraciones para que el algoritmo de aprendizaje *BPROP* optimice la función de objetivo, pero puede darse el caso de que la red empiece a sobre ajustar a partir de cierta iteración y sea conveniente realizar ***early stopping***. En el caso de la red ganadora se descubre que el algoritmo optimiza en la iteración 100. Al reducir el número de iteraciones permitidas (entre 60 y 80) se observa que la bondad de ajuste mejora (por lo que existe sobreajuste). No obstante, en dichos casos no se cumple con el número de inspecciones mínimas por ejecución por lo que la red inicial ganadora se mantiene.

Calculamos de nuevo a partir de los valores de la tabla la sensibilidad y la especificidad del modelo seleccionado. La interpretación es la misma que en el resto de técnicas.

$$Sensibilidad = \frac{3027}{3027 + 2245} = 0.5742 \qquad Especificidad = \frac{12210}{12210 + 4253} = 0.7417$$

Es decir, la probabilidad de que la red clasifique correctamente a un comercio con alguna irregularidad es de 0.5742 mientras que la probabilidad de que clasifique correctamente a un comercio sin irregularidad es de 0.7417.

7.6. Ensamble de modelos

7.6.1. Metodología y construcción de modelos

Como se comentaba en la descripción de esta técnica (ver apartado 5.1.6.), el método que vamos a utilizar para combinar los modelos es el método *stacking*. Concretamente, en un primer momento, se crea un conjunto de datos que recoge todas las predicciones para las observaciones del archivo de validación de los mejores modelos de las cuatro técnicas vistas hasta ahora, así como la variable respuesta para poder evaluar los ensamblados. Dichas predicciones/probabilidades se combinan usando el mínimo (todos los modelos considerados para el ensamblado consideran que existe fraude), máximo (al menos uno de los modelos considera que hay fraude), la mediana (la mitad de los modelos consideran que existe fraude) y la media (en media los modelos consideran que existe fraude). Para determinar si existe fraude o no, utilizamos nuevamente los puntos de corte que en este caso podrán variar entre 0.25 y 0.4 en intervalos de 0.01. Si una observación tiene una probabilidad asociada mayor al punto de corte en una determinada iteración, entonces se la clasificará como fraudulenta, y si no como no fraudulenta. Se construyen todas las combinaciones posibles (11) de orden dos, tres y cuatro que se pueden realizar con los modelos de regresión logística, redes neuronales, random forest y gradient boosting. Por tanto, en esta primera aproximación de ensamble **se estiman** un total de 11 (combinaciones) x4 (técnicas) x16 (puntos de cortes) = **704 modelos**.

En un segundo lugar, se usa nuevamente el conjunto de datos que recoge todas las predicciones para las observaciones del archivo de validación de los mejores modelos de las cuatro técnicas vistas hasta ahora, así como la variable respuesta. De dicho conjunto se eliminan las observaciones con valor 0 en la variable respuesta para crear un conjunto de entrenamiento con el que construir los modelos. Con este archivo de entrenamiento se ajustan los mejores modelos de cada técnica, pero usando como variables input las predicciones del resto de técnicas. Como archivo de validación se utiliza el mismo archivo de entrenamiento, pero sin eliminar dichas observaciones. En este caso se varía el punto de corte de 0.2 a 0.45 en intervalos de 0.01. Por tanto, **se estiman 25 modelos para cada una de las cuatro técnicas vistas**.

7.6.2. Selección del modelo óptimo

Para la primera aproximación, como siempre y hasta ahora, filtramos los 704 modelos de forma que nos quedamos con aquellos que cumplan la condición de realizar al menos 158 visitas por ejecución. Una vez hecho esto, se ordenan por orden decreciente de bondad de ajuste y a continuación se muestra una tabla con los quince mejores modelos de ensamble.

Tabla 7.6.2.1. Resumen mejores modelos de ensambles agrupando por alguna medida estadística

VP	FP	FN	VN	Visitas por ejecución	Bondad Ajuste	Combinación	Agrupación	P corte
3180	4477	2092	11986	163	0.4153	RED/GB	Mediana	0.3

3180	4477	2092	11986	163	0.4153	RED/GB	Media	0.3
3125	4408	2147	12055	165	0.4148	RED/GB	Max	0.32
3155	4571	2117	11892	158	0.4084	LOG/RED/RF/GB	Mediana	0.3
2909	4234	2363	12229	161	0.4073	LOG/RED/RF/GB	Media	0.31
3294	4811	1978	11652	165	0.4064	LOG/RED/GB	Mediana	0.3
3189	4716	2083	11747	166	0.4034	LOG/RED/RF	Mediana	0.3
3081	4559	2191	11904	163	0.4033	LOG/RED/GB	Max	0.37
2790	4130	2482	12333	160	0.4032	LOG/RED/RF	Media	0.32
3234	4789	2038	11674	161	0.4031	LOG/RF/GB	Mediana	0.29
3254	4819	2018	11644	160	0.4031	LOG/RED	Min	0.28
3039	4503	2233	11960	161	0.4029	LOG/RED	Max	0.37
3360	5010	1912	11453	177	0.4014	RED/GB	Max	0.31
3183	4752	2089	11711	161	0.4011	LOG/RED	Mediana	0.32
3183	4752	2089	11711	161	0.4011	LOG/RED	Media	0.32

A la vista de la tabla, podemos ver como la mejor combinación sin duda es aquella entre gradient boosting y redes neuronales ya que son las que presentan la mejor precisión utilizando como estadístico de agrupación indistintamente la mediana o la media.

Para la segunda opción de ensamblado se presenta la siguiente tabla en la que se muestran únicamente los mejores modelos de cada técnica que realicen las inspecciones mínimas por ejecución.

Tabla 7.6.2.2. Resumen mejores modelos de ensamble con predicciones como variables independientes

VP	FP	FN	VN	Visitas por ejecución	Bondad Ajuste	Modelo	P corte
3312	5094	1960	11369	164	0.3940	Logística	0.23
3312	5089	1960	11374	162	0.3942	Redes	0.26
2921	4263	1651	12900	161	0.4066	Random Forest	0.27
3875	5950	1397	10513	163	0.3944	Gradient Boosting	0.24

Vemos como esta segunda opción para combinar los modelos funciona un poco peor que la anterior. Se puede observar como el mejor modelo es el de random forest al que se le introducen como variables explicativas las predicciones sobre el archivo de validación de los mejores modelos de regresión logística, redes neuronales y gradient boosting. En general, se puede ver como los modelos de ensambles han funcionado de forma parecida a los modelos originales. Sin embargo, la mayor cualidad de los ensambles es que reducen la varianza, pero dicha cualidad no se puede comprobar dado el carácter temporal de los datos y la imposibilidad de utilizar técnicas de remuestreo con las que observar la varianza del modelo.

8. Elección y características del modelo definitivo

Llegamos a la fase final del trabajo y es hora de determinar cuál de los modelos óptimos estimados para cada técnica multivariante es el mejor modelo y por tanto el modelo definitivo que se usará para predecir en el futuro la probabilidad de cometer fraude de los pequeños comercios de nuestra población para que puedan ser inspeccionados por la institución y recuperar lo defraudado.

Para tomar la decisión anterior se elabora la tabla que se presenta a continuación en la que aparecen las características más importantes (que se usaran como medidas de comparación) calculadas sobre el archivo de validación de los modelos óptimos elegidos para cada técnica y en torno a las cuales se decidirá cuál es el modelo definitivo.

Tabla 8.1. Comparación de modelos óptimos

Modelo	Bondad de ajuste / Precisión	Visitas por ejecución	Parámetros
Regresión Logística	0.3776	193	18
Random Forest	0.4177	159	7721
Redes Neuronales	0.4158	162	20
Gradient Boosting	0.4093	159	7474
Ensamble	0.4153	163	-

Como se puede observar, la mayor bondad de ajuste o tasa de acierto se consigue con el modelo óptimo de random forest, siendo la diferencia respecto a la precisión del resto de modelos mínima (excepto con regresión logística). Sin embargo, hay que mencionar también que los otros modelos (excepto gradient boosting) realizan más inspecciones mensuales, pero no un número mucho mayor por el que se pueda pensar que a igualdad de inspecciones mejoren al modelo de random forest. Por todo lo expuesto, se elige el modelo de random forest como el mejor modelo para predecir el fraude sufrido por la institución. Recordemos las características principales de este modelo:

- De todos los comercios que el modelo clasifica como fraudulento, acierta un 41.53% (precisión).
- La probabilidad de estimar correctamente a un comercio fraudulento es del 62.25% (sensibilidad).
- La probabilidad de estimar correctamente a un comercio no fraudulento es del 72.20% (especificidad).

9. Post – análisis

9.1. Distribución de las inspecciones

Una vez elegido el modelo ganador de entre los innumerables estimados, para empezar este post – análisis se va a **comprobar si dicho modelo distribuye adecuadamente las inspecciones a realizar** en los datos de validación ya que puede darse el caso de que un mismo comercio sea propuesto varias veces por lo que es importante verificar que sean todos visitados al menos una vez. Para empezar, se realizará teniendo en cuenta la variable Código Alerta (ver apartado 6.1. A mayor código de alerta, mayor es la probabilidad de que el comercio sea fraudulento según la institución). Para ello se muestra la siguiente tabla de frecuencias:

Tabla 9.1.1. Distribución de inspecciones según código de alerta

Frecuencia Porcentaje Col Pct	Código Alerta	Fraude estimado		
		-1	**1**	**Total**
	2	77758 19.59 27.56	42857 10.38 37.34	120615 30.39
	4	5959 1.50 2.11	2082 0.52 1.81	8041 2.03
	3	39407 9.93 13.97	27114 6.83 23.63	66521 16.76
	1	159045 40.07 56.37	42711 10.76 37.22	201756 50.83
	Total	282169 71.09	114764 28.91	396933 100

De la tabla puede extraerse la conclusión de que el modelo visita a todos los tipos de comercio (por lo que puede considerarse correcto), pero hace más hincapié en aquellos con un código de alerta 1 y 2, los que menor riesgo de fraude tienen según la institución. En cambio, los menos inspeccionados son los que tienen códigos de alerta altos (3 y 4). Este comportamiento también sucedía en los datos de entrenamiento por lo que, como vemos, se ha trasladado a los resultados. Dicho comportamiento puede resultar paradójico pero la explicación puede ser la siguiente: para tener un código de alerta alto el comercio debe tener un historial fraudulento elevado por lo que el comercio ya es más que conocido por la institución así que al mínimo indicio de fraude por parte de estos comercios la institución toma directamente acciones legales en vez de realizar inspecciones o realiza muy pocas inspecciones solamente para corroborar los indicios. En cambio, los comercios con un nivel de alerta bajo son 'a priori menos fraudulentos' y desconocidos para la institución por lo que ésta realiza más inspecciones sobre este tipo de comercios para intentar descubrir fraude en ellos.

Acto seguido se comprobó lo más importante, la distribución de las inspecciones en función del comercio. Dada la longitud de la tabla (569 columnas dado que se tienen un total de 569 comercios) no se puede mostrar en el presente trabajo, pero los resultados arrojaron que todos los comercios fueron propuestos para al menos una visita en los dos años de validación lo que sugiere que el modelo distribuye adecuadamente las inspecciones a realizar por lo que se trata de un modelo correcto en este sentido.

9.2. Interpretación del modelo

Históricamente se ha asumido que los datos como los del presente trabajo son generados por ciertas ecuaciones o que provienen de ciertas distribuciones, pero no siempre (casi nunca) es así. Normalmente se trata de datos complejos, con relaciones no lineales, donde no se cumple la hipótesis de normalidad y un largo etcétera. En estos casos, es difícil establecer que los datos provienen de ciertas ecuaciones o distribuciones, y no es adecuado ceñirse a ello.

No obstante, existen modelos clásicos como la regresión logística que pueden funcionar como hemos visto, y son robustos. Además, presentan la ventaja de aportar información sobre la aportación de las diferentes variables input al modelo, de la significatividad del modelo y de las variables y el signo y magnitud de los parámetros correspondientes. Pero a menudo esto es poco importante en comparación con la capacidad predictiva del modelo. Las medidas clásicas de ajuste, significatividad, etc. pasan a segundo plano cuando se tiene mucha información y el objetivo es predecir, como en nuestro caso.

Aquí es donde entran el resto de técnicas empleadas en el trabajo, las cuales como ya se comentó presentan la desventaja de no poder ser interpretadas pero que, en cambio, han mejorado la capacidad predictiva sin excepción de la regresión logística. Las redes neuronales son una caja negra que no se puede interpretar de ninguna forma. Random forest y gradient boosting, al ser técnicas basadas en árboles, pueden aportar información sobre cómo de importante es una variable en función del número de veces que ha sido utilizada para dividir un nodo.

Por tanto, dado que el modelo ganador ha resultado ser random forest, la interpretación se reduce al gráfico de importancia de las variables que ya se vio en su momento. No obstante, en el siguiente apartado se analizará más en detalle el comportamiento de algunas variables para conocer como está funcionando el modelo.

9.3. Post – análisis de las variables

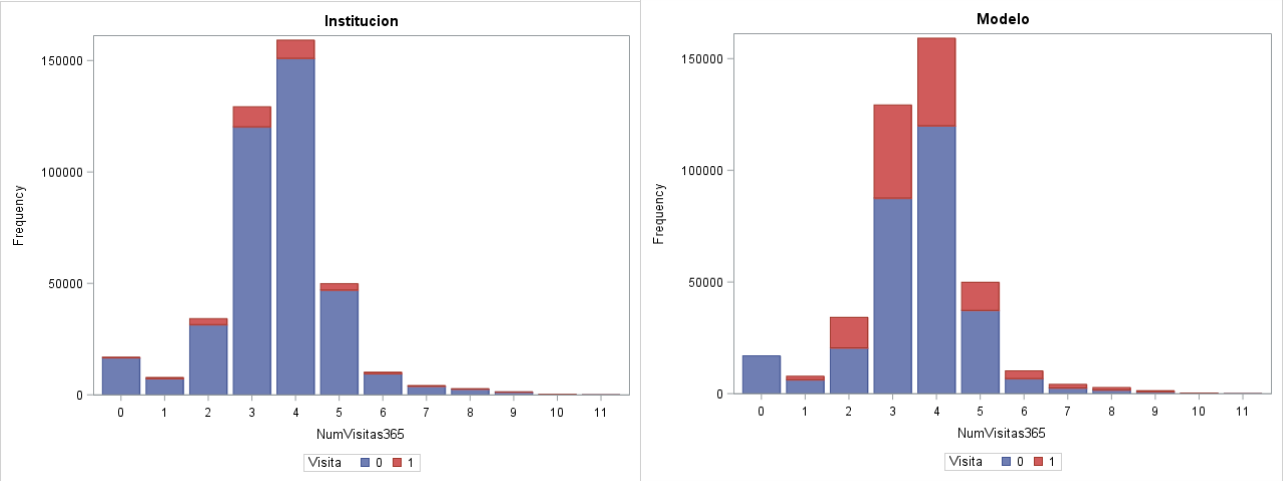
Debido a la imposibilidad de interpretar las variables, en este apartado se va a llevar a cabo un post – análisis de alguna de las variables que resultaron más importantes en el modelo ganador de random forest con el objetivo de poder observar el comportamiento de las variables y con el de conocer si la institución hace uso de la información contenida en dichas variables para diseñar las inspecciones. Todo ello para intentar esclarecer un poco mejor como está funcionando el modelo.

En este post – análisis se van a construir dos o cuatro (dependiendo del número de valores posibles de la variable input) histogramas para cada variable. El segundo (o tercero y cuarto) histograma/s se construye con los comercios visitados y no visitados por el modelo. El primer (o primero y segundo) histograma/s se construye con los comercios visitados y no visitados por la institución. En el primer histograma se podrá conocer si la institución utiliza la información de la variable correspondiente para decidir las inspecciones mientras que en el segundo se podrá observar si las visitas realizadas por el modelo son lógicas, es decir, si se visita a aquellos comercios con valores más fraudulentos a priori en la variable. De la comparación entre ambos se puede observar si existe un cambio de comportamiento entre el modelo y la institución. Se recuerda que las variables están camufladas y aunque se intentó que las variables ficticias tuvieran sentido y mismo comportamiento que las originales, pueden existir comportamientos incongruentes.

- **NumVisitas365**

Indica el número de visitas llevadas a cabo por la institución a un comercio determinado los 365 días anteriores al día para el que esté calculada la variable.

Figura 9.3.1. Distribución de las visitas por el modelo y la institución para la variable NumVisitas365

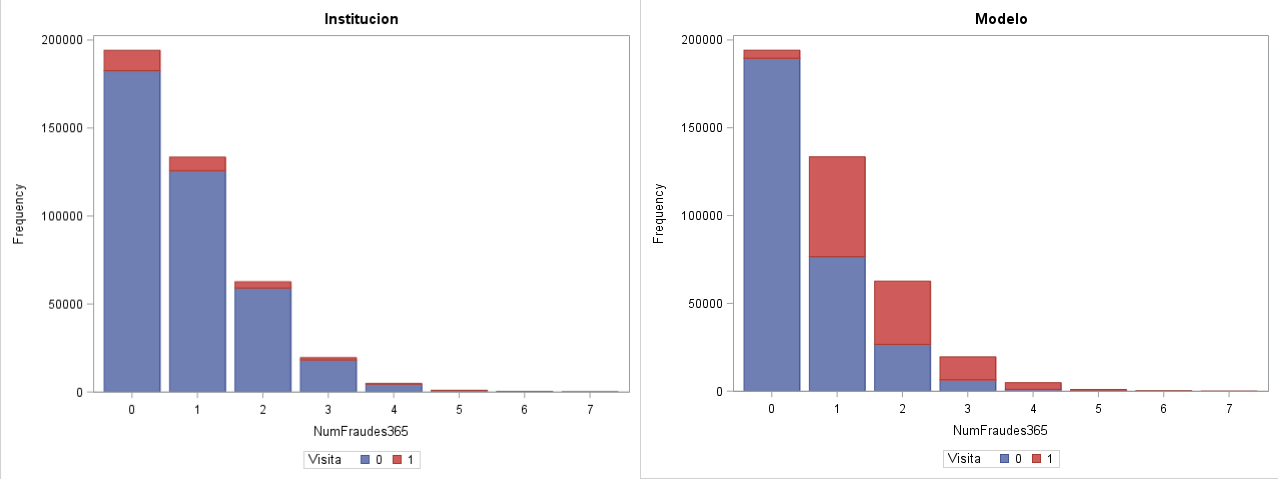


Del primer histograma podemos ver como la institución sí parece utilizar la información contenida en dicha variable para diseñar sus inspecciones, no volviendo a visitar en general a aquellos comercios que ya han sido inspeccionados en el año anterior. En el segundo histograma podemos ver como el modelo, en cambio, sí propone investigar más a los comercios que ya han sido visitados en el último año, lo que supone un cambio de comportamiento frente al actual de la institución y más lógico.

- **NumFraudes365**

Indica el número de inspecciones con anomalías encontradas en un comercio para los 365 días anteriores al día para el que esté calculada la variable.

Figura 9.3.2. Distribución de las visitas por el modelo y la institución para la variable NumFraudes365



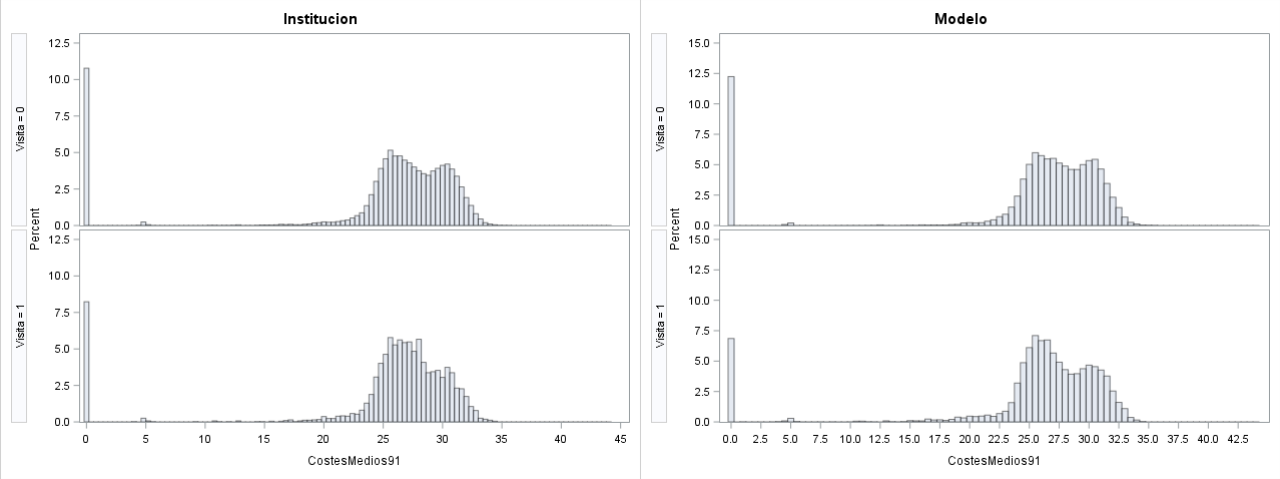
De nuevo podemos ver en el primer histograma como la institución sí parece utilizar la información contenida en dicha variable para diseñar sus inspecciones, no volviendo a visitar en general a aquellos comercios a los que se les ha encontrado fraude en el año anterior. En cambio, el modelo sí propone volver a visitarlos en función del numero de anomalías encontradas, lo cual parece más lógico. Así, para aquellos comercios con una anomalía detectada en el ultimo año, el modelo propone volver a visitar a unos pocos, pero a medida que aumentan las anomalías detectadas en el ultimo año, el modelo propone volver a visitarlos en mayor medida porque seguramente sigan reincidiendo.

Muchas de las siguientes variables más importantes vuelven a tener que ver con el numero de visitas realizadas en el pasado o el numero de inspecciones con anomalías encontradas, pero con agrupaciones temporales más pequeñas. Al ser más importantes las variables *NumFraude365* y *NumVisitas365*, la interpretación es que son importantes, pero que es más importante la información agregada en un espacio temporal mayor. No se muestran en el trabajo, pero los histogramas resultaron muy similares a los de NumFraude365 y NumVisitas365 por lo que se cree conveniente pasar a analizar variables un poco más *diferentes.*

- **CostesMedios91**

Indica los costes medios de los últimos 91 días para cada comercio.

Figura 9.3.3. Distribución de las visitas por el modelo y la institución para la variable CostesMedios91

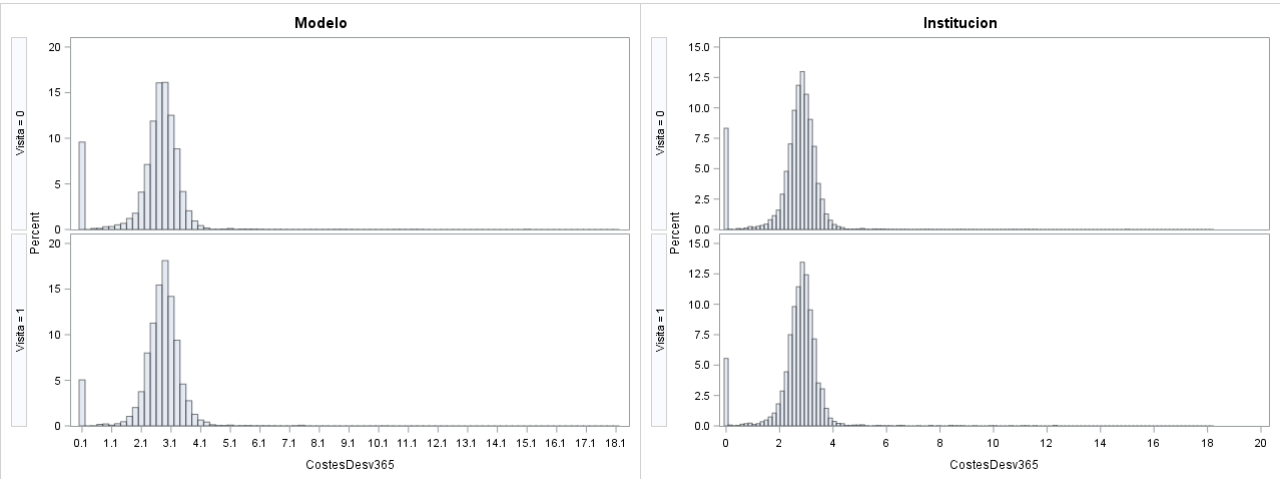


Vemos que los dos histogramas de la primera foto son muy similares lo que sugiere que la institución no está utilizando la información contenida en dicha variable para diseñar las inspecciones. Los histogramas de la segunda foto son un poco diferentes. Vemos como el modelo propone inspeccionar menos a aquellos comercios con un coste medio nulo, pero en cambio, propone investigar más a los comercios con un coste medio de entorno a 25.

- **CostesDesv365**

Indica la desviación típica de los costes para cada comercio en el último año.

Figura 9.3.4. Distribución de las visitas por el modelo y la institución para la variable CostesDesv365



De nuevo, podemos observar como los histogramas de la primera foto son muy parecidos por lo que da la impresión que la institución no está utilizando dicha información. En cambio, los dos histogramas de la segunda foto difieren un poco. El modelo visita menos a aquellos comercios con poca variabilidad entre sus valores de coste y más a aquellos con mayor variación, lo cual parece totalmente lógico.

9.4. Plan campañas de inspección

Como ya se comentó en el apartado 5.2., la temporalidad de nuestros datos (y, por ende, la de nuestros modelos) es diaria por lo que **lo óptimo sería que la institución ejecutara diariamente el modelo y mandase a inspeccionar cada día**. Si se diera este caso, como ya vimos, el número de inspecciones que los modelos tendrían que proponer por ejecución es menor (unas 5 por día en vez de 158), lo que conllevaría un mayor grado de acierto de los modelos.

Sin embargo, como también se comentó en dicho apartado, **la institución solo posee la capacidad operativa para ejecutar una vez al mes el modelo y con dicha ejecución diseñar las campañas del próximo mes**. Tal y como están definidos los modelos, en dicha ejecución el modelo ganador propondría 158 comercios a investigar los cuales podrían ser perfectamente los que la institución fuese a visitar durante el mes. Sin embargo, se propone otra alternativa que consiste en elaborar un programa que ejecute automáticamente el modelo ganador de random forest para cada uno de los días del mes anterior (30 días) al día en que se quieren diseñar las inspecciones, utilizando toda la información disponible hasta el día en cuestión. Por ejemplo, si la institución decide diseñar las inspecciones el 15 de junio, ejecutarían el programa solamente ese día (cumpliendo con la capacidad operativa de la institución), el cual ejecutará el modelo de random forest para cada día del mes anterior (desde el 15 de mayo hasta el 14 de junio). Y, por ejemplo, para el día 15 de mayo el modelo de random forest solo utilizaría la información disponible hasta el 14 de mayo.

Se trataría entonces de recoger las probabilidades de fraude estimadas por el modelo de cada comercio para cada uno de esos 30 días y agruparlas de alguna forma. Las alternativas por tanto que se proponen para diseñar las inspecciones son:

1. Mandar a inspeccionar los 158 comercios que el modelo clasifique como fraudulentos el día de la ejecución/diseño de inspecciones. Para esta opción no sería necesario desarrollar el programa.
2. Ejecutar el programa y calcular el scoring medio para cada comercio en los 30 días y visitar aquellos que presenten un scoring medio mayor que un punto de corte establecido.
3. Ejecutar el programa y calcular la mediana del scoring de cada comercio en los 30 días y visitar aquellos que presenten una mediana mayor que un punto de corte establecido.
4. Ejecutar el programa y recoger el mínimo score de cada comercio en los 30 días e inspeccionar aquellos que presenten un score mínimo en los 30 días mayor que un punto de corte establecido.
5. Ejecutar el programa y recoger el máximo score de cada comercio en los 30 días e inspeccionar aquellos que presenten un score máximo en los 30 días mayor que un punto de corte establecido.

Para conocer cual de las alternativas es la mejor se utilizaron los datos de los dos años de validación (24 meses) y las probabilidades estimadas del modelo óptimo de random forest para las observaciones de dichos datos de validación. Se dividieron por meses dichos datos y se calculó la media, mediana, mínimo y máximo de los scores de cada comercio para cada mes. A continuación, se muestra una tabla ilustrando como quedaría:

Tabla 9.4.1. Ejemplo alternativas para diseño de inspecciones

Comercio	Mes	MediaScoring	MedianaScoring	MinScoring	MaxScoring	UltScoring
1	1	0.2215	0.1959	0.0799	0.4019	0.3732
1	2	0.2885	0.2714	0.0785	0.7392	0.1454
1	3	0.4097	0.3404	0.0923	0.7811	0.5096
…	…	…	…	…	…	…
1	24	0.3439	0.2556	0.2556	0.7111	0.2145
2	1	0.2606	0.2194	0.0625	0.6459	0.2116
2	2	0.2879	0.2602	0.0843	0.7007	0.7007
2	3	0.2802	0.2612	0.0766	0.7004	0.2059
…	…	…	…	…	…	…
2	24	0.2263	0.0967	0.0710	0.7134	0.7133
…	…	…	…	…	…	…
569	1	0.2921	0.2685	0.0876	0.7156	0.0896
…	…	…	…	…	…	…
569	24	0.3451	0.2451	0.1043	0.5983	0.4328

Una vez hecho esto, se fue variando el punto de corte a raíz del cual se construyó la matriz de confusión para cada alternativa. Y con dicha matriz se calculó la bondad de ajuste. Para construir la matriz de confusión se utilizó el último valor en el mes de *Fr30* como variable dependiente (que contiene los valores reales con los que comparar los predichos), en vez de *Fr5*. Esto es así puesto que los datos están agrupados por meses y *Fr30*, por como está construida, recoge/agregada toda la información sobre fraude en el mes. En el apartado 6.3. se puede refrescar la memoria sobre como está construida *Fr30*. Entonces, si por ejemplo establecemos el punto de corte en 0.23, para el primer comercio y primer mes de la tabla, para la alternativa de la media, mediana y mínimo se clasificaría como comercio no fraudulento (-1) mientras que para las alternativas del máximo y ultimo score se clasificaría al comercio como fraudulento (1). Comparando esto con el último valor de *Fr30* en el mes, se clasificaría la observación como VP, FP, VN o FN; y así sucesivamente hasta completar la matriz de confusión con todas las observaciones.

Además, las diferentes alternativas deben cumplir la condición de clasificar como fraudulentos a 3792 (158*24) comercios en los dos años de validación para que cada vez que se ejecute mensualmente proponga en media 158 inspecciones. Operando como se describió antes y variando el punto de corte, obtenemos la siguiente tabla (filtrada para que cumplan la restricción de las visitas) en la que podemos ver los mejores resultados de las diferentes alternativas.

Tabla 9.4.2. Mejores resultados para las diferentes alternativas

VP	FP	FN	VN	Visitas por mes	Alternativa	Bondad de ajuste	Pto corte
538	753	629	2107	188	Último	0.4167	0.23
643	1070	533	1800	244	Media	0.3753	0.22
448	667	679	2203	178	Mediana	0.4017	0.29

891	1882	276	988	389	Max	0.3213	0.2
553	865	614	2005	219	Min	0.3900	0.26

Se puede observar que la alternativa con mejores resultados en los dos años de validación es la de directamente mandar a inspeccionar el próximo mes los 158 comercios que el modelo clasifique como fraudulentos el día de la ejecución. En este caso, dicha alternativa ha propuesto en media 188 inspecciones al mes por lo que la empresa podría inspeccionar los 158 comercios con mayor score. Vemos como se obtiene una bondad de ajuste muy similar a la obtenida por el modelo de random forest ganador.

Por tanto, pongamos un ejemplo del que debería ser el modus operandi de la institución a la hora de diseñar las inspecciones. Supongamos que la última campaña de inspecciones que realizó la institución fue en el mes de abril y que las diseñó/programó el 30 de marzo. Para programar las del mes de mayo, la institución decide que se haga el 30 de abril. Dado que la alternativa que mejores resultados ofrece es la de mandar a inspeccionar los comercios que el modelo proponga el mismo día de la ejecución (en este caso 30 de abril), no es necesario elaborar el programa que se describió que ejecutase el modelo para todos los días de abril, lo que sin duda supone un menor tiempo de ejecución para la institución. Por tanto, la institución debería simplemente ejecutar el modelo de random forest ganador el 30 de abril, clasificar a los comercios en función de su score y punto de corte 0.23, e inspeccionar en mayo a aquellos clasificados como fraudulentos.

9.5. Propuestas para mejorar el modelo

Una vez seleccionado el modelo final a utilizar y el plan de acción para realizar las inspecciones, se mandarían a inspeccionar los comercios correspondientes. El **post – análisis**, como su propio nombre indica, consiste en llevar a cabo un análisis con los resultados obtenidos en las visitas de campo para hacerse una idea de cómo está funcionando nuestro modelo sobre el conjunto de la población con el objetivo de entender el comportamiento del modelo sobre los distintos usuarios. De este modo, con esta información, se puede **intentar mejorar el modelo final inicial**.

En este trabajo el post análisis de los resultados de las inspecciones no ha podido realizarse debido a la falta de tiempo para la llegada de los resultados en campo. Sin embargo, sí que se ha diseñado para realizarlo en cuánto lleguen los resultados. A continuación, se explica cómo se procederá.

Con la llegada de la información sobre las visitas en campo, lo primero que se puede extraer de ella es la **efectividad total real de nuestro modelo para su comparación con la efectividad teórica esperada y con la efectividad actual de la institución**. Con esto podremos saber a grandes rasgos si nuestro modelo es correcto y si ha funcionado bien. Pero lo realmente interesante es utilizar dicha información para intentar **encontrar diferentes grupos de usuarios para los que el modelo se comporta de forma diferente**. Para ello se analizan todas las inspecciones que han resultado

fraudulentas para intentar encontrar variables comunes a todas ellas y realizar un análisis cluster[2] en función de dichas variables con los que identificar los diferentes grupos de usuarios.

Una vez se han obtenido los diferentes grupos de usuarios para los que el modelo se comporta de forma diferente, un primer paso es evaluar la efectividad del modelo sobre dichos grupos con los datos de campo. Como es lógico, habrá grupos donde la efectividad del modelo sea mayor y otros donde la efectividad sea menor. En principio, para mejorar los resultados y para conseguir que todos los grupos tengan efectividades similares, se trata de aumentar el número de visitas en aquellos grupos donde el modelo ha logrado una alta efectividad (ya que a mayor número de inspecciones menor será la efectividad puesto que se visitan a usuarios con menor *scoring*. Pero dado que la efectividad de la que se parte es alta, a pesar de vaya a bajar, la efectividad seguirá siendo buena), mantener el número de inspecciones en los grupos con una efectividad parecida a la total y dejar de visitar los grupos con una efectividad más baja o realizar menos visitas en caso de que el grupo sea de especial interés para la institución (ya que a menor número de inspecciones mayor será la efectividad puesto que se visitarán sólo a los usuarios con mayor *scoring*), pero siempre **manteniendo el mismo número de visitas totales**.

Por ejemplo, supongamos que tras realizar el análisis cluster sobre la información de campo se distinguen cuatro grupos de usuarios para los que el modelo se comporta de forma diferente y que la efectividad (ficticia) del modelo en cada uno de los grupos es la siguiente:

Tabla 9.5.1. Ejemplo modificación inspecciones por grupos

Resultado inspección	Grupo 1	Grupo 2	Grupo 3	Grupo 4	Total
Anomalía	20	20	40	20	120
Sin anomalía	80	220	50	110	440
Inspecciones realizadas	100	240	90	130	560
	17.86%	42.85%	16%	23.2%	100%
Efectividad total	20%	9.09%	44.44%	15.4%	21%

A la vista de las efectividades, considerando el tamaño de cada grupo y en línea con lo comentado, se debería proceder de la siguiente manera:

- Al grupo 1 se le debería mantener el número de visitas.
- Al grupo 2 se le debería visitar menos o directamente dejar de inspeccionar.
- Al grupo 3 se le debería aumentar el número de visitas.
- Al grupo 4 se le debería disminuir un poco el número de visitas.

[2] Conjunto de técnicas multivariantes que trata de, dado un conjunto de individuos caracterizados por una serie de variables, ser capaz de clasificarlos de manera que los individuos pertenecientes a un grupo (cluster) sean tan similares entre sí como sea posible, siendo los distintos grupos entre ellos lo más distintos posible.

Este procedimiento de aumentar o disminuir el número de visitas se debe repetir con los datos de futuras campañas de inspección con el objetivo de aumentar la efectividad y obtener una efectividad similar en todos los grupos ya que puede ir variando la concentración de fraude en los grupos.

Otro procedimiento dentro del post – análisis que puede utilizarse para mejorar la efectividad es **crear modelos independientes para cada uno de los grupos obtenidos**. De esta forma, al construir un modelo propio para cada grupo, el modelo se adaptará a las características de dicho grupo y las predicciones mejoraran.

Los pasos para construir estos modelos propios de cada grupo son similares a los utilizados para construir el modelo para el conjunto de los usuarios. Lo primero de todo consiste en dividir la base de datos en función de los grupos obtenidos y crear los archivos de entrenamiento y validación para cada grupo. Al igual que para el total de usuarios, los archivos de entrenamiento se usarán para construir los diferentes modelos de las cinco técnicas estadísticas utilizadas en este trabajo (regresión logística, redes neuronales, random forest, gradient boosting y ensambles) y los archivos de validación con los que se calcularán las medidas de comparación de modelos y con los que se evaluará la capacidad predictiva de dichos modelos. A la hora de construir los modelos se deben tener en cuenta los tamaños de los grupos y, por tanto, el número de visitas asociadas a cada grupo de forma que el número de inspecciones a realizar en cada grupo vaya acorde con su tamaño y con lo observado en los datos de campo.

Una vez construidos los diferentes modelos con las diferentes técnicas estadísticas, éstos se compararán con las medidas de comparación para seleccionar el modelo óptimo de cada técnica y de entre los cuales se elegirá el modelo final para cada grupo de forma independiente. Como ya se ha comentado, la hipótesis es que con estos modelos 'personalizados' la efectividad aumente respecto al modelo construido para el total de la población, lo cual debe comprobarse grupo a grupo.

10. Conclusiones

Como ya se comentó en la introducción, el fraude es una lacra para cualquier institución que lo sufra en todas sus modalidades. Ya sea en el ámbito fiscal o en el ámbito empresarial, el fraude acaba repercutiendo a la sociedad en todo su conjunto y no hace más que aumentar año a año, razón por la cual las instituciones públicas y privadas están realizando campañas para concienciar a la gente. Y es que la solución para erradicarlo no es fácil ya que depende básicamente de la moral de cada uno el evitar caer en la tentación de cometerlo.

A pesar de los esfuerzos, el erradicar el fraude es una tarea que se presenta harto complicada por lo que se hace necesario apelar a otro tipo de soluciones como es el hecho de intentar prevenirlo para poder combatirlo antes de que ocurra o para prevenir que se vuelva a repetir. Y es aquí donde entra la minería de datos y la estadística como herramienta para combatir el fraude y un ejemplo de ello es el presente trabajo.

El objetivo de este trabajo no era otro que el de encontrar la técnica estadística que mejor fuese capaz de estimar la probabilidad de que una observación fuese fraudulenta para que ésta fuese inspeccionada con el objetivo de aumentar la efectividad de los esfuerzos actuales para prevenir y combatir el fraude. La técnica del modelo finalmente escogido es **random forest** y la precisión teórica del modelo es de un 41.77%, con una sensibilidad del 62.25% y una especificidad del 72.2%.

Para conseguir este modelo ha sido necesario realizar un estudio exhaustivo previo de los datos originales que por motivos de confidencialidad no se ha podido mostrar de forma detallada. Este paso es esencial para conocer la información de la que se parte de cara a conocer el problema para poder crear variables explicativas relevantes y para estimar luego modelos válidos y correctos. Una vez realizado este estudio, creadas las variables explicativas y decidida la variable respuesta se procedió a estimar diferentes modelos con las diferentes técnicas estadísticas (regresión logística, redes neuronales, random forest, gradient boosting y ensambles) con los datos de entrenamiento. Estos modelos se aplicaron a los datos de validación para obtener sus grados de acierto y así poder diferenciar al mejor modelo predictivo. Cabe mencionar que los modelos estimados han sido el resultado de combinar todos los parámetros dentro de cada técnica multivariante dentro de las limitaciones del software utilizado. En este sentido, resaltar que en total **se han estimado más de 16500 modelos** en el presente trabajo (9573 regresiones logísticas, 3059 random forests, 1274 gradient boostings, 1950 redes neuronales y 804 ensambles). Una vez que se escogió el mejor modelo predictivo de entre todos ellos, se realizó un extenso post – análisis para conocer mejor el comportamiento del modelo, así como su validez y se propusieron situaciones para mejorar el modelo en el futuro.

En relación a las limitaciones del software es importante señalar que, aunque se ha procurado incluir el máximo de alternativas posibles para cada uno de los parámetros de cada técnica, la capacidad del software no es infinita por lo que han tenido que ajustarse a dichas limitaciones. En este sentido, posiblemente, con un software más potente, se podría encontrar un modelo ligeramente mejor (con mayor tasa de acierto) ya que al aumentar el número de valores a utilizar dentro de los parámetros que iban variando dentro de cada técnica se generarían más modelos entre los cuales se podría encontrar alguno mejor que el escogido finalmente. Además, hay que tener en cuenta que todos los resultados anteriores (elección del mejor modelo, post – análisis, etc.) están basados en los datos de validación. Al utilizar los datos de validación como referencia el error de predicción real puede ser mayor, pues se ha elegido el modelo que mejor funciona en los datos de validación. Si se dispusiera de más datos, lo ideal sería crear un **conjunto de datos test** sobre los que aplicar el modelo definitivo para obtener unos resultados insesgados (y más cuando debido al carácter temporal de los datos solo se ha aplicado training test, y no otras técnicas de remuestreo más potentes como validación cruzada repetida). Adicionalmente, como hemos visto, al realizar el post análisis con el que intentar identificar grupos de características similares dentro de la población para ajustar el número de visitas para cada grupo y poder estimar un modelo para cada grupo se espera también un incremento en la tasa de acierto.

Una última duda que se puede presentar es cuanto mejora nuestro modelo respecto a las técnicas utilizadas actualmente por la institución para detectar fraude. Basándonos en la tabla de frecuencia 6.1.1. de la variable *fraude* en el conjunto de datos original se deduce que el grado de acierto de la institución es de un 26.14% lo que supone que nuestro modelo realiza una **mejora relativa del 59.79%**, susceptible de mejorar ligeramente tras lo comentado en el párrafo anterior. Por tanto, dado

que el objetivo principal del presente trabajo era aumentar la efectividad de la institución a la hora de detectar el fraude, se puede decir que **se han conseguido alcanzar las metas propuestas**.

Por último, es necesario destacar que para el correcto funcionamiento del modelo en el futuro es necesario actualizar tanto los datos como los parámetros del modelo periódicamente ya que no se tratan de algoritmos estáticos y es por ello que requieren ser actualizados con el paso del tiempo. Además, de hecho, sería conveniente revisar toda la estructura del modelo cada cierto tiempo puesto que los patrones de fraude pueden cambiar y resultar óptimo otro modelo. Respecto al fraude, para terminar, también hay que mencionar que se ha trabajado con información del fraude detectado, pero existe fraude oculto que desconocemos, lo cual supone un riesgo. No obstante, se espera poder detectar dicho fraude oculto.

11. Bibliografía

[1] elEconomista.es. 2014. [sitio web]. Madrid. [consulta: junio 2017]. Disponible en: http://www.eleconomista.es/economia/noticias/6311174/12/14/Diez-datos-interesantes-sobre-la-corrupcion-a-nivel-mundial.html

Alonso, G. & Becerril, J.L. (1993). *Introducción a la inteligencia artificial*. Barcelona: ED. MULTIMEDIA EDICIONES S.A.

Brown, D. & Rothery, P. (1993). *Models in Biology*. Berlín. ED. SPRINGER-VERLAG.

Peña, D. (2002). *Análisis de Datos Multivariantes*. Madrid. S.A. MCGRAW-HILL.

Pérez, C. (2008). *Técnicas de Análisis Multivariante de Datos*. Madrid. PEARSON PRENTICE.

Escot, L. (2015). *Modelos de regresión de variables dependientes limitadas y correcciones en la selección muestral*. Madrid. Facultad Estudios Estadísticos. UCM.

Fundamentos de redes neuronales (URL: http://thales.cica.es/rd/Recursos/rd98/TecInfo/07/capitulo2.html)

Marín, J.M. *Análisis de Cluster y Árboles de Clasificación*. Madrid. UC3M.

Mansanet Sandín, J. (2012). *Técnicas de regresión para la estimación de la localización de la mirada*. Valencia. UPV.

Cárdenas – Montes, M. *Random Forest*. CIEMAT. Universidad de Extremadura.

Prieto Rodriguez, J & Sanzo Pérez, M. J. & Suárez Pandiello, J. (2006). *Análisis económico de la actitud hacia el fraude fiscal en España*. Oviedo. Revista de Economía Publica 177-(2/2006): 107.128. Instituto de Estudios Fiscales.

Communications BBVA. (2017). *Cómo el Big Data ayuda a luchar contra el fraude fiscal*. Madrid. Fintech e innovación. Comunicación corporativa.

Calviño, A. (2017). *Árboles de Clasificación y Regresión*. Madrid. Facultad Estudios Estadísticos. UCM.

Hosmer, D. & Lemeshow, S. (1989). *Applied Logistic Regression*. WILEY & SONS.

Tascón, María T. & Castaño, Francisco J. (2012). *Variables y modelos para la identificación y predicción del fracaso empresarial*. León. Universidad de León.

SAS Institute Inc 2012. *SAS Enterprise Miner and SAS Text Miner Procedures Reference for SAS 9.3*. Cary, NC: SAS Institute Inc.

Gorman, B. (2017). *A Kaggle Master Explains Gradient Boosting*. The Official Blog of Kaggle.com

E. Lopez Briega, R. *Boosting en Machine Learning con Python*. Buenos Aires. Argentina.

Portela, J. (2017). *Redes Neuronales*. Madrid. Facultad Estudios Estadísticos. UCM.

Portela, J. (2017). *Bagging, Random Forest, Gradient Boosting*. Madrid. Facultad Estudios Estadísticos. UCM.

Portela, J. (2017). *Métodos Ensemble*. Madrid. Facultad Estudios Estadísticos. UCM.

Portela, J. (2017). *Redes Neuronales para Clasificación binaria*. Madrid. Facultad Estudios Estadísticos. UCM.

12. Anexos

12.1. Análisis descriptivo de las variables originales

Tabla 12.1.1. Descriptivos variable 'Ventas'

Media	5213.145	**Máximo**	24999
Mediana	4818	**Desv. Típica**	2812.66
Mínimo	1	**Kurtosis**	5.747

Figura 12.1.1. Histograma variable 'Ventas'

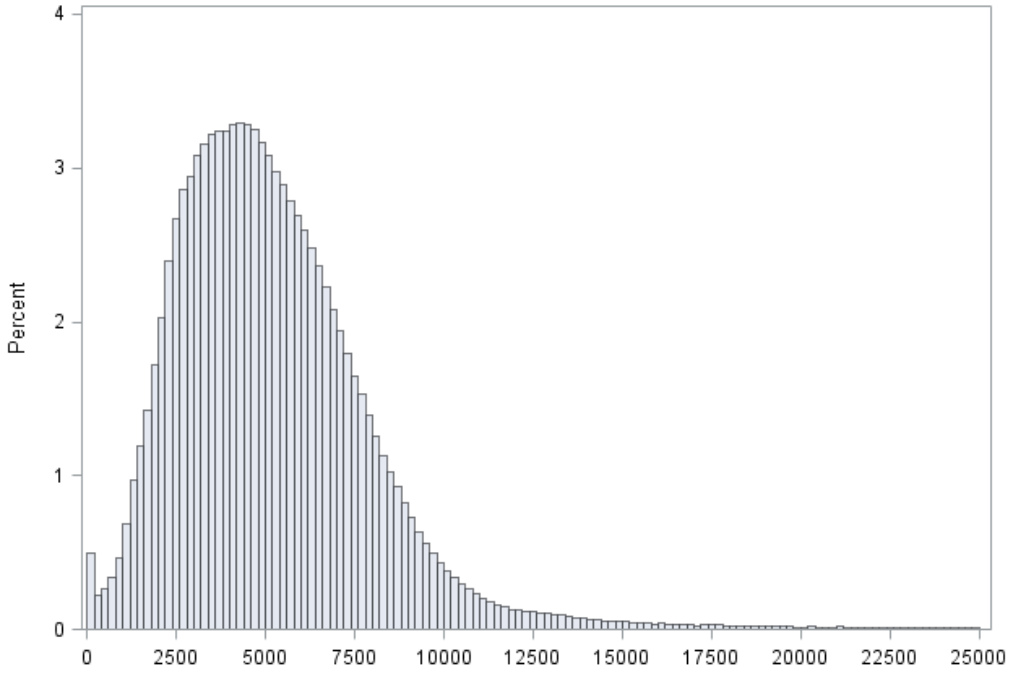


Tabla 12.1.2. Descriptivos variable 'Costes'

Media	27.180	**Máximo**	93.51
Mediana	27.08	**Desv. Típica**	3.51
Mínimo	0.09	**Kurtosis**	9.634

Figura 12.1.2. Histograma variable 'Costes'

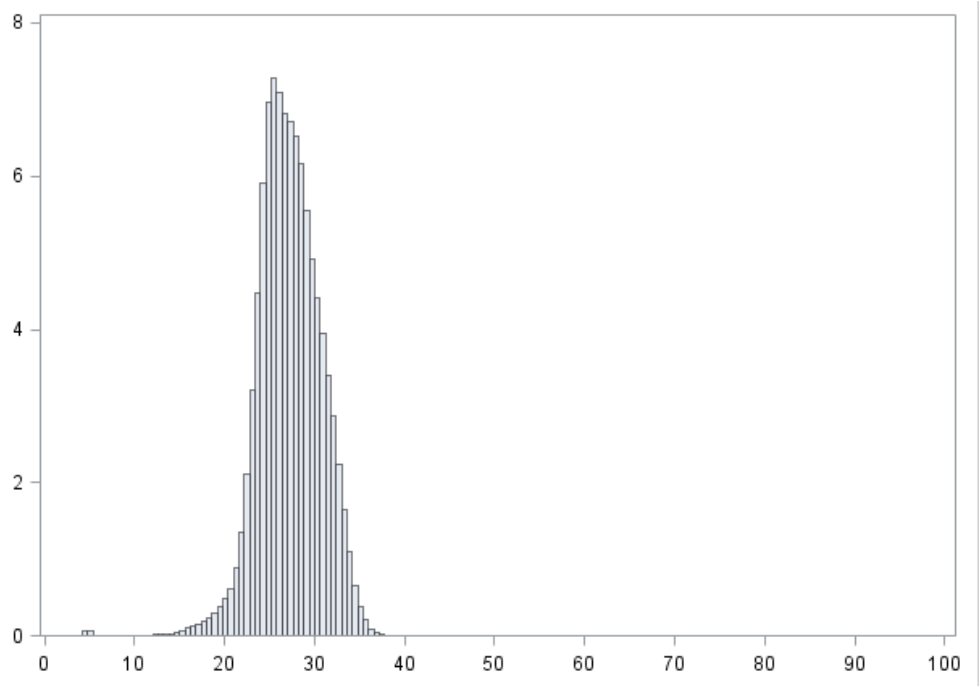


Tabla 12.1.3. Descriptivos variable 'Consumo energía'

Media	7.5277	**Máximo**	180.423
Mediana	4.9190	**Desv. Típica**	3.8322
Mínimo	0	**Kurtosis**	32.820

Figura 12.1.3. Histograma variable 'Consumo energía'

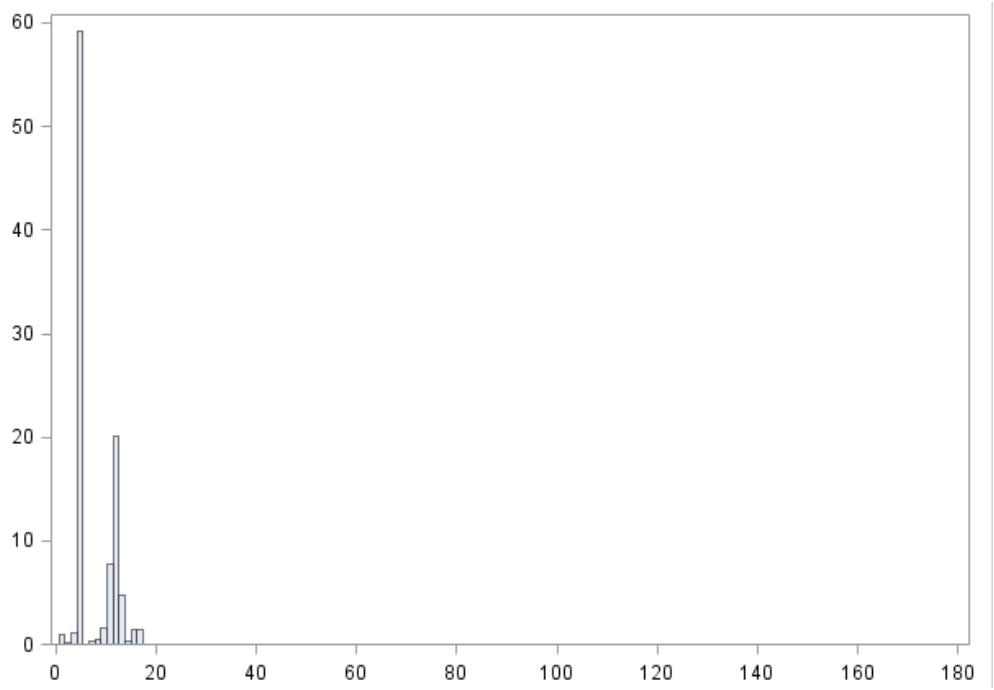


Tabla 12.1.4. Descriptivos variable 'Impuestos'

Media	7993460.42	**Máximo**	91883434
Mediana	6053481	**Desv. Típica**	7405399.59
Mínimo	3	**Kurtosis**	9.34477

Figura 12.1.4. Histograma variable 'Impuestos'

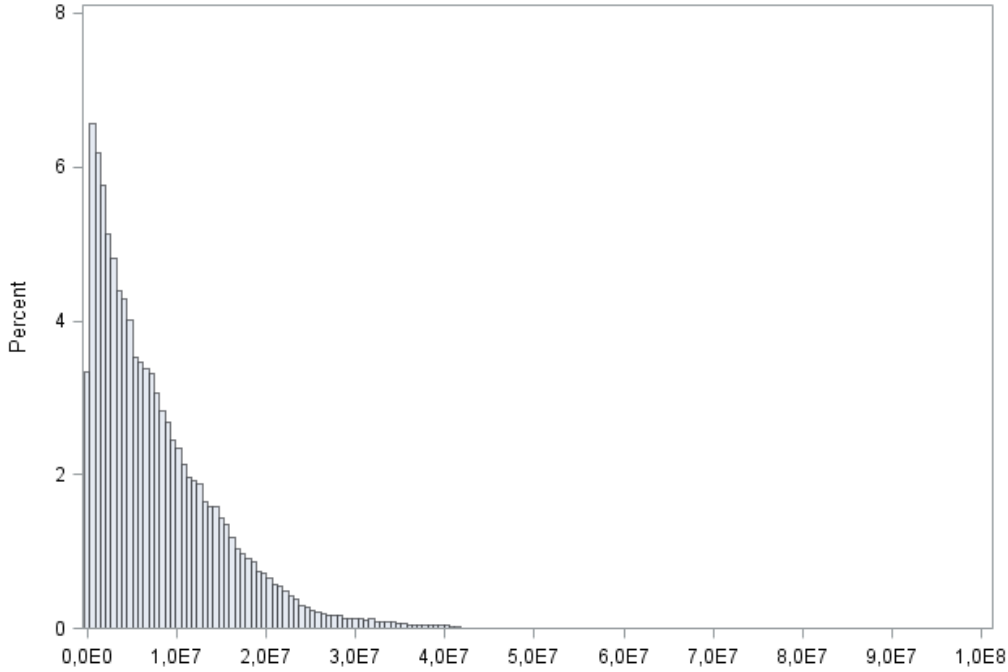


Tabla 12.1.5. Descriptivos variable 'Salario de los empleados'

Media	1491111.91	**Máximo**	42600100
Mediana	1112534	**Desv. Típica**	1392133.59
Mínimo	1	**Kurtosis**	45.0363

Figura 12.1.5. Histograma variable 'Salario de los empleados'

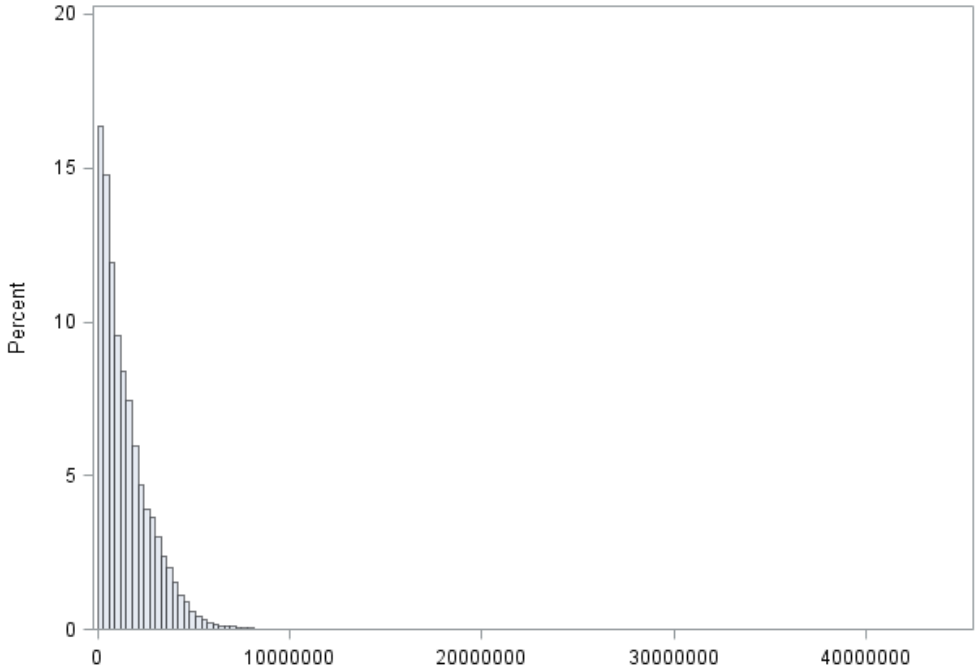


Figura 12.1.6 Gráfico de sectores variable 'Localización'

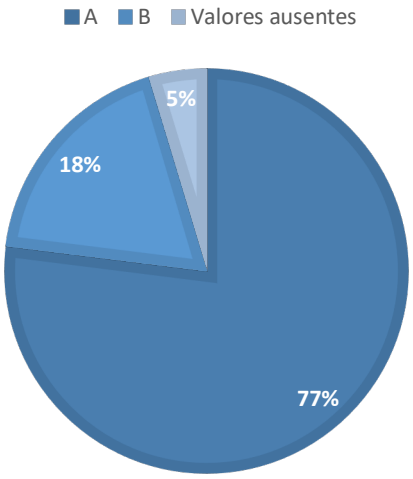


Figura 12.1.7. Gráfico de sectores variable 'Sexo'

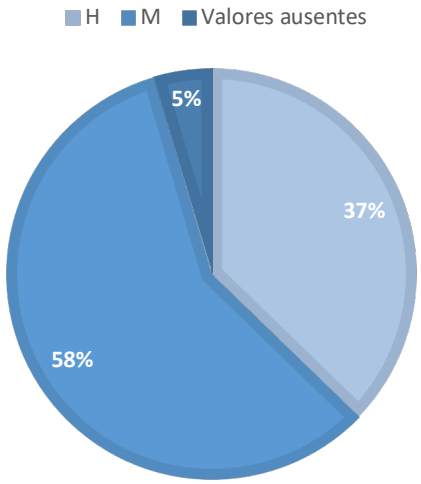


Figura 12.1.8. Gráfico de sectores variable 'Estudios'

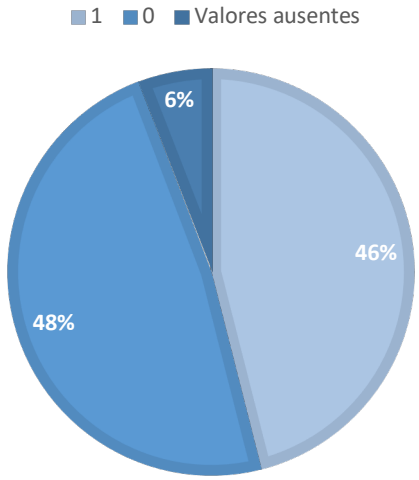
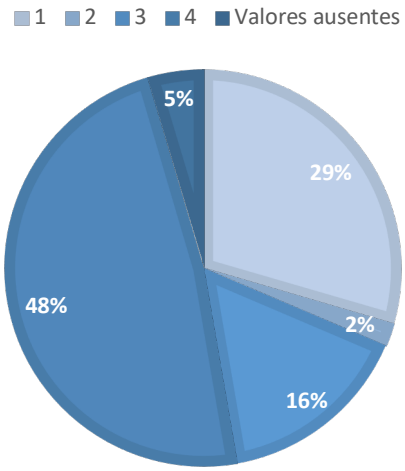


Figura 12.1.9. Gráfico de sectores variable 'Código_alerta'



12.2. Codificación de las variables construidas

Tabla 12.2.1. Codificación de las variables creadas para 'costes'

Nombre variable creada	Descripción	Nombre variable creada	Descripción
CostesMedios365	Media de los costes de los últimos 365 días	***PorcenMissCostes7***	Porcentaje de valores ausentes en los últimos 7 días
CostesMedios91	Media de los costes de los últimos 91 días	***NumMissCostes365***	Número de valores ausentes en los últimos 365 días
CostesMedios28	Media de los costes de los últimos 28 días	***NumMissCostes91***	Número de valores ausentes en los últimos 91 días
CostesMedios14	Media de los costes de los últimos 14 días	***NumMissCostes28***	Número de valores ausentes en los últimos 28 días
CostesMedios7	Media de los costes de los últimos 7 días	***NumMissCostes14***	Número de valores ausentes en los últimos 14 días
CostesDesv365	Desviación típica de los costes de los últimos 365 días	***NumMissCostes7***	Número de valores ausentes en los últimos 7 días
CostesDesv91	Desviación típica de los costes de los últimos 91 días	***NumDatosCostes365***	Número de datos válidos en los últimos 365 días
CostesDesv28	Desviación típica de los costes de los últimos 28 días	***NumDatosCostes91***	Número de datos válidos en los últimos 91 días
CostesDesv14	Desviación típica de los costes de los últimos 14 días	***NumDatosCostes28***	Número de datos válidos en los últimos 28 días
CostesDesv7	Desviación típica de los costes de los últimos 7 días	***NumDatosCostes14***	Número de datos válidos en los últimos 14 días
PorcenMissCostes365	Porcentaje de valores ausentes en los últimos 365 días	***NumDatosCostes7***	Número de datos válidos en los últimos 7 días
PorcenMissCostes91	Porcentaje de valores ausentes en los últimos 91 días	***CambiosPonderadosCostes***	Media ponderada de los cambios respecto al valor medio de los últimos 28 días

PorcenMissCostes28	Porcentaje de valores ausentes en los últimos 28 días	***MaxCambioCostes***	Número máximo de días con cambios negativos en los últimos 28 días
PorcenMissCostes14	Porcentaje de valores ausentes en los últimos 14 días	***LugarMaxCambioCostes***	Indica el lugar donde se ha producido el máximo de la variable anterior

Tabla 12.2.2. Codificación de las variables creadas para 'consumo de energía'

Nombre variable creada	Descripción	Nombre variable creada	Descripción
EnergíaMedia365	Media de la energía consumida en los últimos 365 días	***PorcenMissEnergía7***	Porcentaje de valores ausentes en los últimos 7 días
EnergíaMedia91	Media de la energía consumida en los últimos 91 días	***NumMissEnergía365***	Número de valores ausentes en los últimos 365 días
EnergíaMedia28	Media de la energía consumida en los últimos 28 días	***NumMissEnergía91***	Número de valores ausentes en los últimos 91 días
EnergíaMedia14	Media de la energía consumida en los últimos 14 días	***NumMissEnergía28***	Número de valores ausentes en los últimos 28 días
EnergíaMedia7	Media de la energía consumida en los últimos 7 días	***NumMissEnergía14***	Número de valores ausentes en los últimos 14 días
EnergíaDesv365	Desviación típica de la energía consumida en los últimos 365 días	***NumMissEnergía7***	Número de valores ausentes en los últimos 7 días
EnergíaDesv91	Desviación típica de la energía consumida en los últimos 91 días	***NumDatosEnergía365***	Número de datos válidos en los últimos 365 días
EnergíaDesv28	Desviación típica de la energía consumida en los últimos 28 días	***NumDatosEnergía91***	Número de datos válidos en los últimos 91 días
EnergíaDesv14	Desviación típica de la energía consumida en los últimos 14 días	***NumDatosEnergía28***	Número de datos válidos en los últimos 28 días

EnergíaDesv7	Desviación típica de la energía consumida en los últimos 7 días	***NumDatosEnergía14***	Número de datos válidos en los últimos 14 días
PorcenMissEnergía365	Porcentaje de valores ausentes en los últimos 365 días	***NumDatosEnergía7***	Número de datos válidos en los últimos 7 días
PorcenMissEnergía91	Porcentaje de valores ausentes en los últimos 91 días	***CambiosPonderadosEnergía***	Media ponderada de los cambios respecto al valor medio de los últimos 28 días
PorcenMissEnergía28	Porcentaje de valores ausentes en los últimos 28 días	***MaxCambioEnergía***	Número máximo de días con cambios negativos en los últimos 28 días
PorcenMissEnergía14	Porcentaje de valores ausentes en los últimos 14 días	***LugarMaxCambioEnergía***	Indica el lugar donde se ha producido el máximo de la variable anterior

Tabla 12.2.3. Codificación de las variables creadas para 'salario de los empleados'

Nombre variable creada	Descripción	Nombre variable creada	Descripción
SalariosMedios365	Media de los salarios pagados en los últimos 365 días	***PorcenMissSalarios7***	Porcentaje de valores ausentes en los últimos 7 días
SalariosMedios91	Media de los salarios pagados en los últimos 91 días	***NumMissSalarios365***	Número de valores ausentes en los últimos 365 días
SalariosMedios28	Media de los salarios pagados en los últimos 28 días	***NumMissSalarios91***	Número de valores ausentes en los últimos 91 días
SalariosMedios14	Media de los salarios pagados en los últimos 14 días	***NumMissSalarios28***	Número de valores ausentes en los últimos 28 días
SalariosMedios7	Media de los salarios pagados en los últimos 7 días	***NumMissSalarios14***	Número de valores ausentes en los últimos 14 días
SalariosDesv365	Desviación típica de los salarios pagados en los últimos 365 días	***NumMissSalarios7***	Número de valores ausentes en los últimos 7 días

SalariosDesv91	Desviación típica de los salarios pagados en los últimos 91 días	***NumDatosSalarios365***	Número de datos válidos en los últimos 365 días
SalariosDesv28	Desviación típica los salarios pagados en los últimos 28 días	***NumDatosSalarios91***	Número de datos válidos en los últimos 91 días
SalariosDesv14	Desviación típica de los salarios pagados en los últimos 14 días	***NumDatosSalarios28***	Número de datos válidos en los últimos 28 días
SalariosDesv7	Desviación típica de los salarios pagados en los últimos 7 días	***NumDatosSalarios14***	Número de datos válidos en los últimos 14 días
PorcenMissSalarios365	Porcentaje de valores ausentes en los últimos 365 días	***NumDatosSalarios7***	Número de datos válidos en los últimos 7 días
PorcenMissSalarios91	Porcentaje de valores ausentes en los últimos 91 días	***CambiosPonderadosSalarios***	Media ponderada de los cambios respecto al valor medio de los últimos 28 días
PorcenMissSalarios28	Porcentaje de valores ausentes en los últimos 28 días	***MaxCambioSalarios***	Número máximo de días con cambios negativos en los últimos 28 días
PorcenMissSalarios14	Porcentaje de valores ausentes en los últimos 14 días	***LugarMaxCambioSalarios***	Indica el lugar donde se ha producido el máximo de la variable anterior

Tabla 12.2.4. Codificación de las variables creadas para 'impuestos'

Nombre variable creada	Descripción	Nombre variable creada	Descripción
ImpuestosMedios365	Media de los impuestos pagados en los últimos 365 días	***PorcenMissImpuestos7***	Porcentaje de valores ausentes en los últimos 7 días
ImpuestosMedios91	Media de los impuestos pagados en los últimos 91 días	***NumMissImpuestos365***	Número de valores ausentes en los últimos 365 días

ImpuestosMedios28	Media de los impuestos pagados en los últimos 28 días	***NumMissImpuestos91***	Número de valores ausentes en los últimos 91 días
ImpuestosMedios14	Media de los impuestos pagados en los últimos 14 días	***NumMissImpuestos28***	Número de valores ausentes en los últimos 28 días
ImpuestosMedios7	Media de los impuestos pagados en los últimos 7 días	***NumMissImpuestos14***	Número de valores ausentes en los últimos 14 días
ImpuestosDesv365	Desviación típica de los impuestos pagados en los últimos 365 días	***NumMissImpuestos7***	Número de valores ausentes en los últimos 7 días
ImpuestosDesv91	Desviación típica de los impuestos pagados en los últimos 91 días	***NumDatosImpuestos365***	Número de datos válidos en los últimos 365 días
ImpuestosDesv28	Desviación típica de los impuestos pagados en los últimos 28 días	***NumDatosImpuestos91***	Número de datos válidos en los últimos 91 días
ImpuestosDesv14	Desviación típica de los impuestos pagados en los últimos 14 días	***NumDatosImpuestos28***	Número de datos válidos en los últimos 28 días
ImpuestosDesv7	Desviación típica de los impuestos pagados en los últimos 7 días	***NumDatosImpuestos14***	Número de datos válidos en los últimos 14 días
PorcenMissImpuestos365	Porcentaje de valores ausentes en los últimos 365 días	***NumDatosImpuestos7***	Número de datos válidos en los últimos 7 días
PorcenMissImpuestos91	Porcentaje de valores ausentes en los últimos 91 días	***CambiosPonderadosImpuestos***	Media ponderada de los cambios respecto al valor medio de los últimos 28 días

PorcenMissImpuestos28	Porcentaje de valores ausentes en los últimos 28 días	***MaxCambioImpuestos***	Número máximo de días con cambios negativos en los últimos 28 días
PorcenMissImpuestos14	Porcentaje de valores ausentes en los últimos 14 días	***LugarMaxCambioImpuestos***	Indica el lugar donde se ha producido el máximo de la variable anterior

12.3. Importancia de las variables en el modelo Random Forest óptimo

Solo se muestran 83 variables de las 173 empleadas para construir el modelo ya que el resto se utilizaban menos de 5 veces en el modelo.

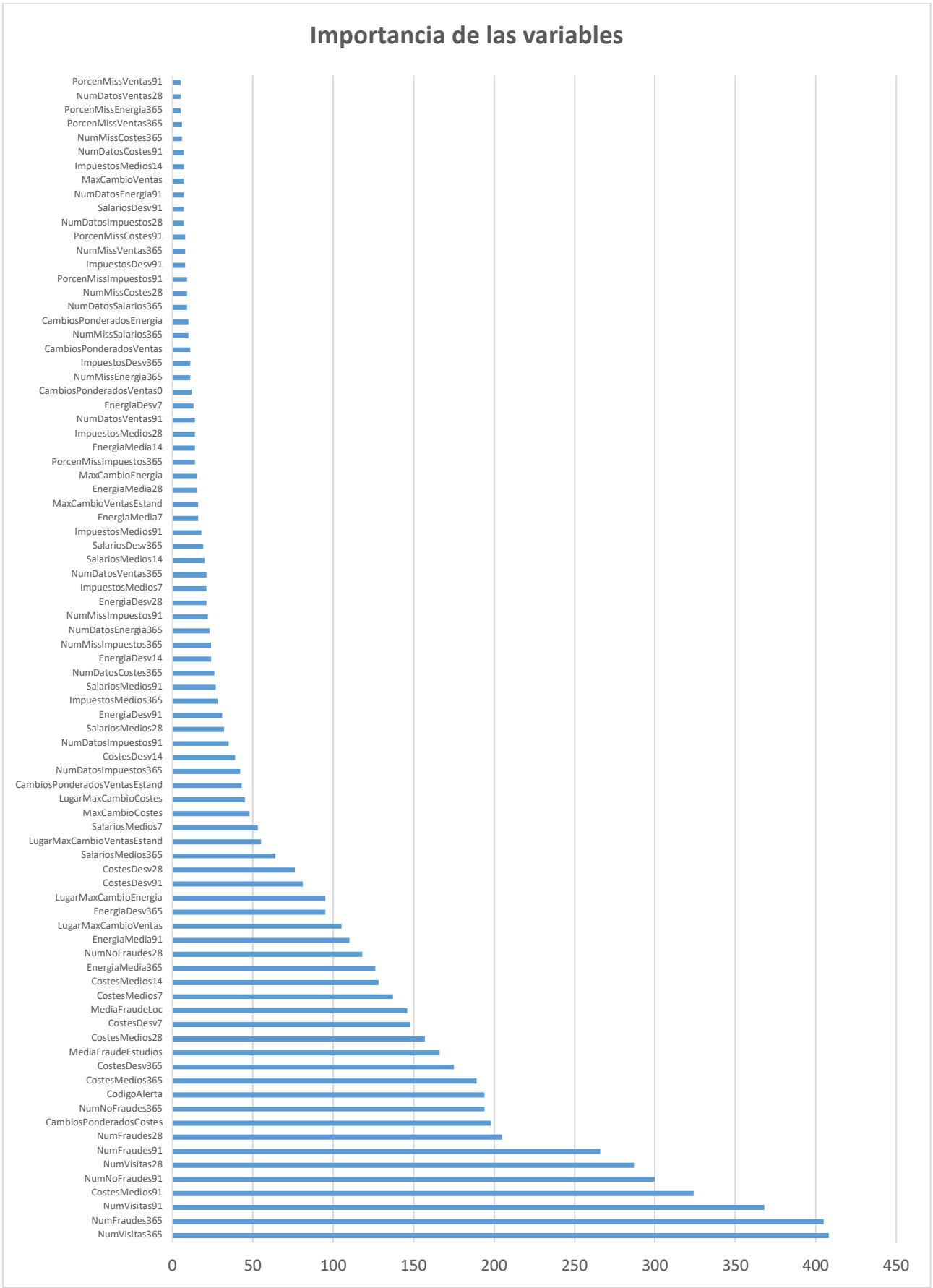
Tabla 12.3.1. Importancia variables modelo RF óptimo

Variables	Importancia
NumVisitas365	408
NumFraudes365	405
NumVisitas91	368
CostesMedios91	324
NumNoFraudes91	300
NumVisitas28	287
NumFraudes91	266
NumFraudes28	205
CambiosPonderadosCostes	198
NumNoFraudes365	194
CódigoAlerta	194
CostesMedios365	189
CostesDesv365	175
MediaFraudeEstudios	166
CostesMedios28	157
CostesDesv7	148
MediaFraudeLoc	146
CostesMedios7	137
CostesMedios14	128
EnergiaMedia365	126
NumNoFraudes28	118
EnergiaMedia91	110
LugarMaxCambioVentas	105
EnergiaDesv365	95
LugarMaxCambioEnergía	95
CostesDesv91	81
CostesDesv28	76
SalariosMedios365	64
LugarMaxCambioVentasEstand	55
SalariosMedios7	53
MaxCambioCostes	48
LugarMaxCambioCostes	45
CambiosPonderadosVentasEstand	43
NumDatosImpuestos365	42
CostesDesv14	39
NumDatosImpuestos91	35
SalariosMedios28	32
EnergiaDesv91	31
ImpuestosMedios365	28
SalariosMedios91	27
NumDatosCostes365	26
EnergiaDesv14	24
NumMissImpuestos365	24
NumDatosEnergia365	23
NumMissImpuestos91	22
EnergiaDesv28	21
ImpuestosMedios7	21
NumDatosVentas365	21
SalariosMedios14	20
SalariosDesv365	19
ImpuestosMedios91	18
EnergiaMedia7	16
MaxCambioVentasEstand	16
EnergiaMedia28	15
MaxCambioEnergía	15
PorcenMissImpuestos365	14

EnergiaMedia14	14
ImpuestosMedios28	14
NumDatosVentas91	14
EnergiaDesv7	13
CambiosPonderadosVentas0	12
NumMissEnergia365	11
ImpuestosDesv365	11
CambiosPonderadosVentas	11
NumMissSalarios365	10
CambiosPonderadosEnergía	10
NumDatosSalarios365	9
NumMissCostes28	9
PorcenMissImpuestos91	9
ImpuestosDesv91	8

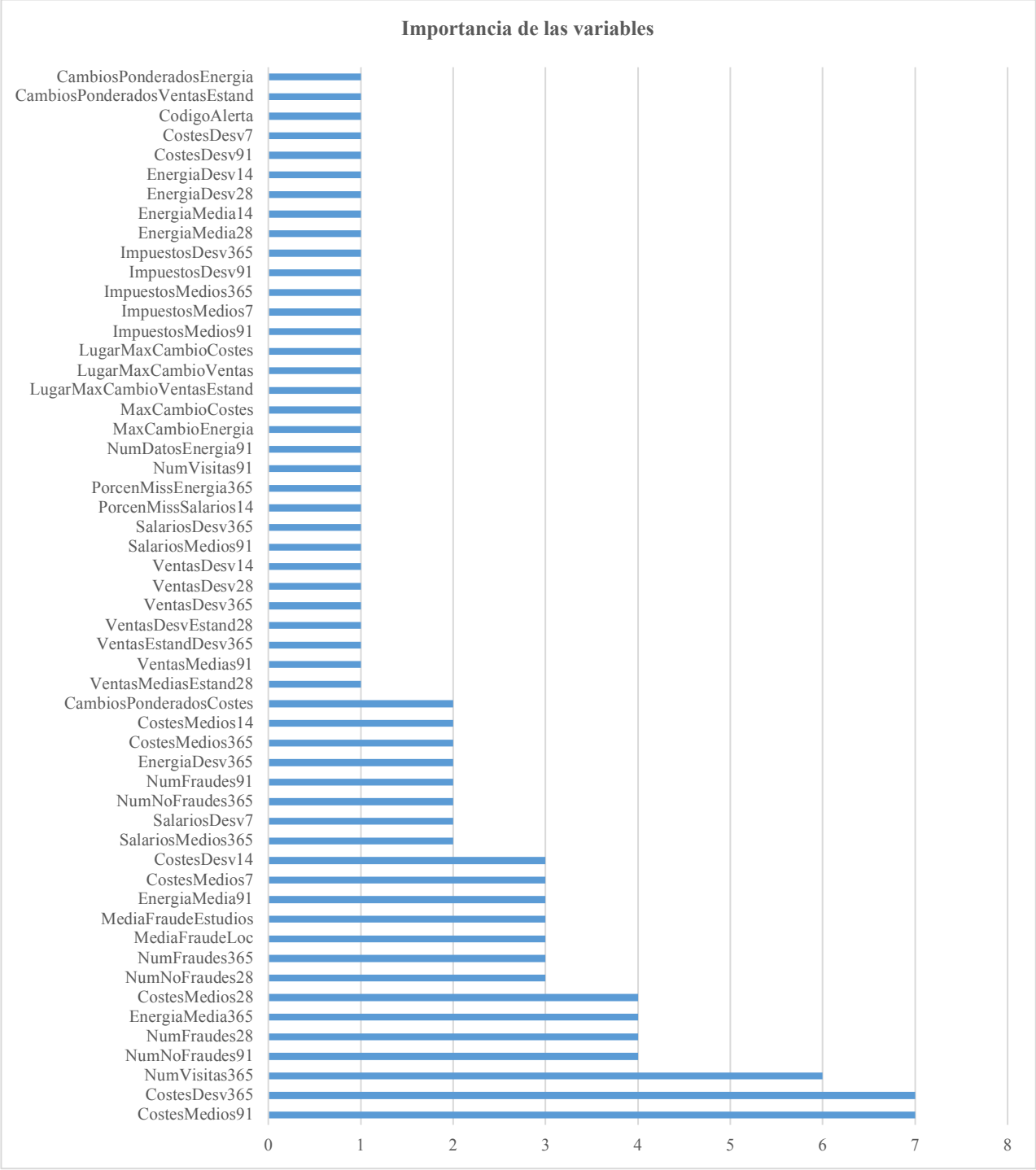
NumMissVentas365	8
PorcenMissCostes91	8
NumDatosImpuestos28	7
SalariosDesv91	7
NumDatosEnergia91	7
MaxCambioVentas	7
ImpuestosMedios14	7
NumDatosCostes91	7
NumMissCostes365	6
PorcenMissVentas365	6
PorcenMissEnergia365	5
NumDatosVentas28	5
PorcenMissVentas91	5

Figura 12.3.1. Gráfico de la importancia de las variables en el modelo óptimo de R Forest



12.4. Importancia de las variables en el modelo de Gradient Boosting óptimo

Figura 12.4.1. Gráfico de la importancia de las variables en el modelo óptimo de G Boosting



12.5. Código SAS

En este apartado del anexo se muestra el código utilizado para la construcción de las técnicas empleadas en el presente trabajo y el post – análisis. No se muestra el código de la lectura, así como

de la depuración de los datos ni el de la creación de las variables explicativas y respuestas ya que se revelaría información confidencial.

12.5.1. Código regresión logística

```
/* CARGAR LIBRERIAS */
libname Estacion 'C:\Users\Pablo Contreras\Desktop\TFM\DatosSASDepurados';
libname Original 'C:\Users\Pablo Contreras\Desktop\TFM\DatosSASOriginales';
libname DatosDia 'C:\Users\Pablo Contreras\Desktop\TFM\DatosDias';
libname Modelos 'C:\Users\Pablo Contreras\Desktop\TFM\Modelos';
libname Posana 'C:\Users\Pablo Contreras\Desktop\TFM\PosAnalisis';

data modelos.Resumen_logistica; format Metodo $10. Funcion $10.; informat Metodo Funcion $10.; put
Metodo $ Funcion $; run;

/* STEPWISE */
%macro logistica_STEPWISE;
%let lista="0.2 0.1 0.05 0.01 0.001 0.0001 0.00001 0.000001";
%let nume=8;
%do  i=1 %to &nume %by 1;
  data _null_;p_entrada=scanq(&lista,&i);call symput("p_entrada",left(p_entrada));run;
  %do  j=1 %to &nume %by 1;
   data _null_;p_salida=scanq(&lista,&j);call symput("p_salida",left(p_salida));run;
   %let lista2= "Probit Logit Cloglog";
   %let nume2=3;
   %do k=1 %to &nume2 %by 1;
      data _null_;funcion=scanq(&lista2,&k);call symput("funcion",left(funcion));run;

       proc logistic data=modelos.entrenamiento_irr5 outest=parametros;
       class /* Variables categoricas */;
       model /* Var dep */ (event="1")=

       /* Variables explicativas continuas y categoricas */

       / selection=STEPWISE Link=&funcion SLSTAY=&p_salida SLENTRY=&p_entrada;
       score data=modelos.validacion out=salpredi_logistica;
       run;
      %do pmax=8 %to 10 %by 1;
         %do pmin=1 %to 7 %by 1;
                data tabla1 (keep=P_1 irr5); set salpredi_logistica; run;
                data tabla2 (drop=P_1); set tabla1; if ((&pmin/10) < P_1 < (&pmax /10)) then
irr5_estimado=1; else irr5_estimado=-1; run;

            data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var irr5_estimado; output out=VP (keep= VP) n=VP; run;
            data tabla3; set tabla2; if (irr5=-1 and irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FP (keep= FP) n=FP; run;
            data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FN (keep= FN) n=FN; run;
            data tabla3; set tabla2; if (Irr5=-1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=VN (keep= VN) n=VN; run;
            data tabla3; set tabla2; if (Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=visitas (keep= visitas)
n=visitas; run;

            data parametros (drop=_LINK_ _TYPE_ _STATUS_ _NAME_ Intercept _LNLIKE_ _ESTTYPE_); set
parametros; run;
            proc transpose data=parametros out=parametros_tras;run;
            data parametros_tras (keep=COL1); set parametros_tras;if (COL1=.) then delete; run;
            proc means data=parametros_tras noprint; var COL1; output out=num_parametros (keep=
num_parametros) n=num_parametros; run;
            data criterio; merge VP FP FN VN visitas num_parametros; run;

            data criterio; set criterio;
            Bondad_Ajuste= VP/(VP+FP);run;

            data criterio; set criterio; format Metodo $10. Funcion $10.; informat Metodo Funcion
$10.; put Metodo $ Funcion $;run;
            data criterio; set criterio;
            Metodo="STEPWISE";
            Funcion="&funcion";
            p_entrada=&p_entrada;
```

```
            p_salida=&p_salida;
            pmax=(&pmax)/10;
            pmin=(&pmin)/10; run;
            data criterio;set criterio; format Metodo $10. Funcion $10.; informat Metodo Funcion
$10.; put Metodo $ Funcion $;run;
            data modelos.Resumen_logistica; set modelos.Resumen_logistica criterio;run;
         %end;
       %end;
      %end;
    %end;
   %end;
%mend;
%logistica_STEPWISE; run;

/* BACKWARD */
%macro logistica_BACKWARD;
%let lista="0.2 0.1 0.05 0.01 0.001 0.0001 0.00001 0.000001";
%let nume=8;
%do  i=1 %to &nume %by 1;
   data _null_;p_salida=scanq(&lista,&i);call symput("p_salida",left(p_salida));run;
    %let lista2="Probit Logit Cloglog";
    %let nume2=3;
    %do j=1 %to &nume2 %by 1;
      data _null_;funcion=scanq(&lista2,&j);call symput("funcion",left(funcion));run;
      proc logistic data=modelos.entrenamiento_Irr5 outest=parametros;

      class /* Variables categoricas */;
      model /* Var dep */  (event="1")=

        /* Variables explicativas continuas y categoricas */

      / selection=BACKWARD Link=&funcion SLSTAY=&p_salida;
      score data=modelos.validacion out=salpredi_logistica; run;

      %do pmax=8 %to 10 %by 1;
         %do pmin=1 %to 7 %by 1;
            data tabla1 (keep=P_1 Irr5); set salpredi_logistica; run;
            data tabla2 (drop=P_1); set tabla1; if ((&pmin/10) < P_1 < (&pmax /10)) then
Irr5_estimado=1; else Irr5_estimado=-1; run;

            data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=VP (keep= VP) n=VP; run;
            data tabla3; set tabla2; if (Irr5=-1 and Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FP (keep= FP) n=FP; run;
            data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FN (keep= FN) n=FN; run;
            data tabla3; set tabla2; if (Irr5=-1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=VN (keep= VN) n=VN; run;
            data tabla3; set tabla2; if (Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=visitas (keep= visitas)
n=visitas; run;

            data parametros (drop=_LINK_ _TYPE_ _STATUS_ _NAME_ Intercept _LNLIKE_ _ESTTYPE_); set
parametros; run;
            proc transpose data=parametros out=parametros_tras;run;
            data parametros_tras (keep=COL1); set parametros_tras;if (COL1=.) then delete; run;
            proc means data=parametros_tras noprint; var COL1; output out=num_parametros (keep=
num_parametros) n=num_parametros; run;
            data criterio; merge VP FP FN VN visitas num_parametros; run;

            data criterio; set criterio; Bondad_Ajuste= VP/(VP+FP); run;

            data criterio;set criterio; format Metodo $10. Funcion $10.; informat Metodo Funcion
            $10.; put Metodo $ Funcion $; run;
            data criterio;set criterio;
            Metodo="BACKWARD";
            Funcion="&funcion";
            p_entrada=0;
            p_salida=&p_salida;
            pmax=(&pmax)/10;
            pmin=(&pmin)/10;
            run;
            data criterio;set criterio; format Metodo $10. Funcion $10.; informat Metodo Funcion
            $10.; put Metodo $ Funcion $;run;
            data modelos.Resumen_logistica; set modelos.Resumen_logistica criterio;run;
         %end;
      %end;
```

```
   %end;
%end;
%mend;
%logistica_BACKWARD;

/* FORWARD */
%macro logistica_FORWARD;
%let lista="0.2 0.1 0.05 0.01 0.001 0.0001 0.00001 0.000001";
%let nume=8;
%do  i=1 %to &nume %by 1;
   data _null_;p_entrada=scanq(&lista,&i);call symput("p_entrada",left(p_entrada));run;
    %let lista2="Probit Logit Cloglog";
    %let nume2=3;
    %do j=1 %to &nume2 %by 1;
      data _null_;funcion=scanq(&lista2,&j);call symput("funcion",left(funcion));run;
      proc logistic data=modelos.entrenamiento_Irr5 outest=parametros;

      class /* Vars categoricas */;
      model /* Var dep */  (event="1")=

        /* Vars explicativas continuas y categoricas */

      / selection=FORWARD Link=&funcion SLENTRY=&p_entrada;
      score data=modelos.validacion out=salpredi_logistica; run;

      %do pmax=8 %to 10 %by 1;
         %do pmin=1 %to 7 %by 1;
            data tabla1 (keep=P_1 Irr5); set salpredi_logistica; run;
            data tabla2 (drop=P_1); set tabla1; if ((&pmin/10) < P_1 < (&pmax /10)) then
Irr5_estimado= 1; else Irr5_estimado=-1; run;

            data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=VP (keep= VP) n=VP; run;
            data tabla3; set tabla2; if (Irr5=-1 and Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FP (keep= FP) n=FP; run;
            data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FN (keep= FN) n=FN; run;
            data tabla3; set tabla2; if (Irr5=-1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=VN (keep= VN) n=VN; run;
            data tabla3; set tabla2; if (Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=visitas (keep= visitas)
n=visitas; run;

            data parametros (drop=_LINK_ _TYPE_ _STATUS_ _NAME_ Intercept _LNLIKE_ _ESTTYPE_); set
parametros; run;
            proc transpose data=parametros out=parametros_tras;run;
            data parametros_tras (keep=COL1); set parametros_tras;if (COL1=.) then delete; run;
            proc means data=parametros_tras noprint; var COL1; output out=num_parametros (keep=
num_parametros) n=num_parametros; run;
            data criterio; merge VP FP FN VN visitas num_parametros; run;
            data criterio; set criterio; Bondad_Ajuste= VP/(VP+FP);run;

            data criterio;set criterio; format Metodo $10. Funcion $10.; informat Metodo Funcion
            $10.; put Metodo $ Funcion $; run;
            data criterio;set criterio;
            Metodo="FORWARD";
            Funcion="&funcion";
            p_salida=0;
            p_entrada=&p_entrada;
            pmax=(&pmax)/10;
            pmin=(&pmin)/10;
            run;
            data criterio;set criterio; format Metodo $10. Funcion $10.; informat Metodo Funcion
            $10.; put Metodo $ Funcion $;run;
            data modelos.Resumen_logistica; set modelos.Resumen_logistica criterio;run;
         %end;
      %end;
   %end;
%end;
%mend;
%logistica_FORWARD;

proc printto;
run;

data modelos.Resumen_logistica; set modelos.Resumen_logistica; if _n_=1 then delete; run;
```

```
/* ELECCION REGRESION LOGÍSTICA ÓPTIMA */
data modelos.mejoresLogistica; set modelos.Resumen_logistica; if visitas > 113760; run;
proc sort data=modelos.mejoresLogistica; by descending Bondad_Ajuste; run;


/* EJECUCIÓN LOGISTICA ÓPTIMA */
proc logistic data=modelos.entrenamiento_irr5 outest=parametros;
class /* Vars categoricas */;
model /* Var dep */  (event="1")=

/* Vars explicativas continuas */
/ selection=STEPWISE LINK=LOGIT SLSTAY=0.001 SLENTRY=0.2;
score data=modelos.validacion out=modelos.salprediLogisticaOptima;
run;
```

12.5.2. Código redes neuronales

```
proc printto print='C:\Users\Pablo Contreras\Desktop\TFM\out_redes.txt';run;
proc printto log='C:\Users\Pablo Contreras\Desktop\TFM\log_redes.txt';run;

/* CREACION CATALOGO */
PROC DMDB DATA=modelos.entrenamiento_irr5 dmdbcat=modelos.catalogo_aprendizaje_irr5;
target /* Variable dep */; var

/* Variables explicativas continuas */;
class /* Var dep y categoricas */; run;

/* INSPECCIONAR NUMERO DE NODOS */
%macro nodos;
data modelos.nodos; run;
%do iter=1 %to 3; /* training test repetido */
   %do nodos=1 %to 30 %by 2;

      proc neural data=modelos.entrenamiento_irr5 dmdbcat=modelos.catalogo_aprendizaje_irr5;
      input
      /* Variables explicativas continuas */ /level=interval;
      input
      /* Variables explicativas categóricas */ /level=nominal;
      target /* Var dependiente */ /level=nominal;

      hidden &nodos /ACT=TANH COMBINE=LINEAR;
      train tech=LEVMAR;
      netoptions random=0; /* semilla aleatoria */
      score data=modelos.validacion out=salpredi outfit=salfit; run;

      data tabla1 (keep=P_Irr51 Irr5); set salpredi; run;
      data tabla2; set tabla1; if (P_Irr51<(28/100)) then Irr5_estimado=-1; else Irr5_estimado=1;
run;
      data tabla2 (drop=P_Irr51); set tabla2; run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
Irr5_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
Irr5_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
Irr5_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
Irr5_estimado=-1); quit;

      data criterio; merge VP FP FN VN; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);Nodos=&nodos; run;

      data modelos.nodos; set modelos.nodos criterio; run;
   %end;
%end;

data modelos.nodos; set modelos.nodos; if _n_=1 then delete; run;
proc sort data=modelos.nodos; by Nodos; run;
%mend;

%nodos; run;
proc printto; run;
data borrar; set modelos.nodos;
proc sort data=borrar nodup; by Nodos; run;
```

```
proc boxplot data=borrar; plot Bondad_Ajuste*Nodos; run;

/* INSPECCIONAR ALGORITMO */
%macro algo;
data modelos.algoritmos; run;
%do iter=1 %to 3;   /* training test repetido */
   %let lista='BPROP LEVMAR QUANEW TRUREG DBLDOG CONGRA';
   %let nume=6;
   %do  i=1 %to &nume %by 1;
      data _null_;algo=scanq(&lista,&i); call symput('algo',left(algo));run;

      proc neural data=modelos.entrenamiento_irr5 dmdbcat=modelos.catalogo_aprendizaje_irr5;
      input
      /* Variables explicativas continuas */ /level=interval;
      input
      /* Variables explicativas categóricas */ /level=nominal;
      target /* Var dependiente */ /level=nominal;

      hidden 1 /ACT=TANH COMBINE=LINEAR;
      train tech=&algo;
      netoptions random=0; /* semilla aleatoria */
      score data=modelos.validacion out=salpredi outfit=salfit; run;

      data tabla1 (keep=P_Irr51 Irr5); set salpredi; run;
      data tabla2; set tabla1; if (P_Irr51<(28/100)) then Irr5_estimado=-1; else Irr5_estimado=1;
run;
      data tabla2 (drop=P_Irr51); set tabla2; run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
Irr5_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
Irr5_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
Irr5_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
Irr5_estimado=-1); quit;

      data criterio; merge VP FP FN VN; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP); Algoritmo="&algo"; run;

      data modelos.algoritmos; set modelos.algoritmos criterio; run;
   %end;
%end;

data modelos.algoritmos; set modelos.algoritmos; if _n_=1 then delete; run;
proc sort data=modelos.algoritmos; by Algoritmo; run;
%mend;

%algo; run;
proc printto; run;
data borrar; set modelos.algoritmos;
proc sort data=borrar nodup; by Algoritmo; run;
proc boxplot data=borrar; plot Bondad_Ajuste*Algoritmo; run;

/* VARIAR LEARNING RATE Y MOMENTUM DEL ALGORITMO BACKPROPAGATION */
%macro bprop(learn=, mom=);
proc neural data=modelos.entrenamiento_irr5 dmdbcat=modelos.catalogo_aprendizaje_irr5;
input
/* Variables explicativas continuas */ /level=interval;
input
/* Variables explicativas categóricas */ /level=nominal;
target /* Var dependiente */ /level=nominal;

hidden 1 /ACT=TANH COMBINE=LINEAR;
train tech=BPROP learn=&learn momentum=&mom;
netoptions random=0; /* semilla aleatoria */
score data=modelos.validacion out=salpredi outfit=salfit; run;

data tabla1 (keep=P_Irr51 Irr5); set salpredi; run;
data tabla2; set tabla1; if (P_Irr51<(28/100)) then Irr5_estimado=-1; else Irr5_estimado=1; run;
data tabla2 (drop=P_Irr51); set tabla2; run;

proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
Irr5_estimado=1); quit;
proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
Irr5_estimado=1); quit;
```

```
proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
Irr5_estimado=-1); quit;
proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
Irr5_estimado=-1); quit;

data criterio; merge VP FP FN VN; run;
data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP); Learn=&learn; Momentum=&mom; run;
data modelos.bprop; set modelos.bprop criterio; run;

%mend;

data modelos.bprop; run;
%bprop(learn=0.1, mom=0.6);
%bprop(learn=0.1, mom=0.7);
%bprop(learn=0.1, mom=0.8);
%bprop(learn=0.1, mom=0.9);
%bprop(learn=0.15, mom=0.6);
%bprop(learn=0.15, mom=0.7);
%bprop(learn=0.15, mom=0.8);
%bprop(learn=0.15, mom=0.9);
%bprop(learn=0.2, mom=0.6);
%bprop(learn=0.2, mom=0.7);
%bprop(learn=0.2, mom=0.8);
%bprop(learn=0.2, mom=0.9);
%bprop(learn=0.25, mom=0.6);
%bprop(learn=0.25, mom=0.7);
%bprop(learn=0.25, mom=0.8);
%bprop(learn=0.25, mom=0.9);
%bprop(learn=0.3, mom=0.6);
%bprop(learn=0.3, mom=0.7);
%bprop(learn=0.3, mom=0.8);
%bprop(learn=0.3, mom=0.9);
%bprop(learn=0.1, mom=0.5);
%bprop(learn=0.15, mom=0.5);
%bprop(learn=0.2, mom=0.5);
%bprop(learn=0.25, mom=0.5);
%bprop(learn=0.3, mom=0.5);
%bprop(learn=0.1, mom=0.4);
%bprop(learn=0.15, mom=0.4);
%bprop(learn=0.2, mom=0.4);
%bprop(learn=0.25, mom=0.4);
%bprop(learn=0.3, mom=0.4);
%bprop(learn=0.1, mom=0.3);
%bprop(learn=0.15, mom=0.3);
%bprop(learn=0.2, mom=0.3);
%bprop(learn=0.25, mom=0.3);
%bprop(learn=0.3, mom=0.3);
%bprop(learn=0.1, mom=0.2);
%bprop(learn=0.15, mom=0.2);
%bprop(learn=0.2, mom=0.2);
%bprop(learn=0.25, mom=0.2);
%bprop(learn=0.3, mom=0.2);
%bprop(learn=0.1, mom=0.1);
%bprop(learn=0.15, mom=0.1);
%bprop(learn=0.2, mom=0.1);
%bprop(learn=0.25, mom=0.1);
%bprop(learn=0.3, mom=0.1);

data modelos.bprop; set modelos.bprop; if _n_=1 then delete; run;
proc sort data=modelos.bprop; by momentum learn; run;
proc boxplot data=modelos.bprop; plot Bondad_Ajuste*Learn; by momentum; run;

/* VARIAR FUNCION DE ACTIVACION, ALGORITMO, NODOS Y PUNTO DE CORTE */
%macro redes;
data modelos.Resumen_Redes; run;
%let lista='TANH EXP ARC ELL LOG';
%let nume=5;
%let lista2='BPROP LEVMAR QUANEW';
%let nume2=3;

%do  i=1 %to &nume %by 1;
   data _null_; activa=scanq(&lista,&i); call symput("activa",left(activa)); run;
   %do j=1 %to &nume2 %by 1;
      data _null_; algo=scanq(&lista2,&j); call symput("algo",left(algo)); run;
      %do nodos=1 %to 5 %by 1;

            proc neural data=modelos.entrenamiento_irr5 dmdbcat=modelos.catalogo_aprendizaje_irr5;
```

```
            input
            /* Variables explicativas continuas */ /level=interval;
            input
            /* Variables explicativas categóricas */ /level=nominal;
            target /* Var dependiente */ /level=nominal;

            hidden &nodos / act=&activa;
            train tech=&algo learn=0.1 momentum=0.4;
            netoptions random=0;
            score data=modelos.validacion out=salpredi outfit=salfit;
            run;

         %do corte=20 %to 45 %by 1;
            data tabla1 (keep=P_Irr51 Irr5); set salpredi; run;
            data tabla2; set tabla1; if (P_Irr51<(&corte/100)) then Irr5_estimado=-1; else
Irr5_estimado=1; run;
            data tabla2 (drop=P_Irr51); set tabla2; run;

            data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=VP (keep= VP) n=VP; run;
            data tabla3; set tabla2; if (Irr5=-1 and Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FP (keep= FP) n=FP; run;
            data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FN (keep= FN) n=FN; run;
            data tabla3; set tabla2; if (Irr5=-1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=VN (keep= VN) n=VN; run;
            data tabla3; set tabla2; if (Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=visitas (keep= visitas)
n=visitas; run;

            data criterio; merge VP FP FN VN visitas; run;
            data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);Nodos=&nodos;
            F_activacion="&activa"; Algoritmo="&algo"; p_corte=(&corte/100); Visitas=visitas/720;
run;

            data modelos.Resumen_Redes; set modelos.Resumen_Redes criterio;run;
            %end;
      %end;
   %end;
%end;
data modelos.Resumen_Redes; set modelos.Resumen_Redes; if _n_=1 then delete; run;
%mend;

%redes;
proc printto; run;

/* ELECCION RED ÓPTIMA */
data modelos.mejoresRedes; set modelos.Resumen_Redes; parametros=((17+1)*nodos+nodos+1); run;
data modelos.mejoresRedes; set modelos.mejoresRedes; if (visitas > 158); run;

proc sort data=modelos.mejoresRedes; by F_activacion; run;
proc boxplot data=modelos.mejoresRedes; plot Bondad_Ajuste*F_activacion; run;

proc sort data=modelos.mejoresRedes; by Algoritmo; run;
proc boxplot data=modelos.mejoresRedes; plot Bondad_Ajuste*Algoritmo; run;

proc sort data=modelos.mejoresRedes; by Algoritmo F_activacion; run;
proc boxplot data=modelos.mejoresRedes; plot Bondad_Ajuste*F_activacion; by Algoritmo; run;

proc sort data=modelos.mejoresRedes; by descending Bondad_Ajuste; run;

/* EARLY STOPPING PARA LA RED GANADORA */
%macro earlystopping (maxiter= );
   proc neural data=modelos.entrenamiento_irr5 dmdbcat=modelos.catalogo_aprendizaje_irr5;
   input
   /* Variables explicativas continuas */ /level=interval;
   input
   /* Variables explicativas categóricas */ /level=nominal;
   target /* Var dependiente */ /level=nominal;

   hidden 1 /ACT=TANH COMBINE=LINEAR;
   train tech=BPROP learn=0.1 momentum=0.4 maxiter=&maxiter;
   netoptions random=0;
   score data=modelos.validacion out=salpredival outfit=salfitval; run;

   data tabla1 (keep=P_Irr51 Irr5); set salpredival; run;
   data tabla3; set tabla1; if (P_Irr51<(31/100)) then Irr5_estimado=-1; else Irr5_estimado=1; run;
```

```
   proc sql noprint; create table VP as select count(*) as VP from tabla3 where (Irr5=1 and
Irr5_estimado=1); quit;
   proc sql noprint; create table FP as select count(*) as FP from tabla3 where (Irr5=-1 and
Irr5_estimado=1); quit;
   proc sql noprint; create table FN as select count(*) as FN from tabla3 where (Irr5=1 and
Irr5_estimado=-1); quit;
   proc sql noprint; create table VN as select count(*) as VN from tabla3 where (Irr5=-1 and
Irr5_estimado=-1); quit;
   proc sql noprint; create table Visitas as select count(*) as visitas from tabla3 where
(Irr5_estimado=1); quit;

   data criterio; merge VP FP FN VN visitas; run;
   data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);run;
%mend;

%earlystopping(maxiter=40);
%earlystopping(maxiter=50);
%earlystopping(maxiter=60);
%earlystopping(maxiter=70);
%earlystopping(maxiter=80);
%earlystopping(maxiter=90);
%earlystopping(maxiter=100);

/* EJECUCION RED OPTIMA */
proc neural data=modelos.entrenamiento_irr5 dmdbcat=modelos.catalogo_aprendizaje_irr5;
input
/* Variables explicativas continuas */ /level=interval;
input
/* Variables explicativas categóricas */ /level=nominal;
target /* Var dependiente */ /level=nominal;

hidden 1 / act=TANH COMBINE=LINEAR;
train tech=BPROP learn=0.1 momentum=0.4;
score data=modelos.validacion out=modelos.salprediRedOptima outfit=salfitRedOptima;
run;
```

12.5.3. Código random forest

```
data modelos.resumen_randomforest; run;

%macro randomforestbin(ArchivoEntrenamiento=, ArchivoValidacion=, vardep=, listconti=, listcategor=,
maxtrees=, variables=, porcenbag=, numvariables=, maxbranch=, tamhoja=, maxdepth=, pvalor=);

/* Construccion de Random Forest */
ods listing close;
proc hpforest data = &ArchivoEntrenamiento
maxtrees = &maxtrees
vars_to_try = &variables
trainfraction = 0.8
leafsize = &tamhoja
maxdepth = &maxdepth
/*maxbranch=&maxbranch*/
alpha = &pvalor
exhaustive = 5000
missing = useinsearch;
target &vardep / level = nominal;
input &listconti / level = interval;
%if (&listcategor ne) %then %do;
   input &listcategor / level = nominal;
%end;
ods output fitstatistics = estadisticas variableimportance = importancia;
save file='C:\Users\Pablo Contreras\Desktop\TFM\Modelos\modelosRF_irr5.bin';
run;

/* Validacion de los modelos */
proc hp4score data = &ArchivoValidacion;
id &listconti
   &listcategor
   &vardep;
score file='C:\Users\Pablo Contreras\Desktop\TFM\Modelos\modelosRF_irr5.bin' out=salprediRF;
run;
ods listing;
```

```
%do cortemin=20 %to 70 %by 1;

    data tabla1 (keep=IRR5 P_IRR5_1 P_IRR51 PuntoDeCorteMin); set salprediRF;
PuntoDeCorteMin=(&cortemin/100); run;
    data tabla2; set tabla1; if (P_IRR51>=PuntoDeCorteMin) then fraude_estimado=1; else
fraude_estimado=-1; run;

   data tabla3; set tabla2; if (Irr5=1 and fraude_estimado=1) then output; run;
   proc means data=tabla3 noprint; var irr5; output out=VP (keep= VP) n=VP; run;
   data tabla3; set tabla2; if (Irr5=-1 and fraude_estimado=1) then output; run;
   proc means data=tabla3 noprint; var Irr5; output out=FP (keep= FP) n=FP; run;
   data tabla3; set tabla2; if (Irr5=1 and fraude_estimado=-1) then output; run;
   proc means data=tabla3 noprint; var Irr5; output out=FN (keep= FN) n=FN; run;
   data tabla3; set tabla2; if (Irr5=-1 and fraude_estimado=-1) then output; run;
   proc means data=tabla3 noprint; var Irr5; output out=VN (keep= VN) n=VN; run;
   data tabla3; set tabla2; if (fraude_estimado=1) then output; run;
   proc means data=tabla3 noprint; var Irr5; output out=visitas (keep = visitas) n = visitas; run;
   proc means data=importancia noprint; var NRules; output out=Rules (keep= Numrules) sum=Numrules;
run;

   data criterio; merge VP FP FN VN Rules visitas; run;
   data criterio; set criterio;
   Bondad_Ajuste = VP/sum(VP, FP);
   PuntoDeCorteMin=(&cortemin/100);
   maxtrees=&maxtrees;
   vars_to_try=&variables;
   trainfraction=&porcenbag;
   leafsize=&tamhoja;
   maxdepth=&maxdepth;
   alpha=&pvalor;
   exhaustive=5000;
   numvariables=&numvariables; run;
   data modelos.resumen_RandomForest; set modelos.resumen_RandomForest criterio; run;
%end;
%mend;


options mprint=0; options notes;
%macro Forest;
%let lista2='0.2 0.1 0.05 0.01';
%let nume2=4;
%let lista3='1000 700 500 100 50';
%let nume3=5;
%do  j=1 %to &nume2 %by 1;
   data _null_;p_valor=scanq(&lista2,&j);call symput('p_valor',left(p_valor));run;
   %do  k=1 %to &nume3 %by 1;
      data _null_;tamhoja=scanq(&lista3,&k);call symput('tamhoja',left(tamhoja));run;
      /* %do maxtrees=100 %to 100 %by 100; /*era 100*/
         %do variables=20 %to 80 %by 30; /*5*/
         /*%do tamhoja=100 %to 1100 %by 500; /*5*/
            /* %do maxdepth=50 %to 50 %by 1; /* El maximo es 50 */
               /* %do maxbranch=2 %to 2 %by 2; */

%randomforestbin(
ArchivoEntrenamiento = modelos.entrenamiento_irr5,
ArchivoValidacion = modelos.validacion,
vardep = /* variable dependiente */,
listconti=
    /* variables explicativas continuas */
,
listcategor = /* variables explicativas categoricas */,
maxtrees=500,
variables=&variables.,
porcenbag=0.8,
numvariables=183,
maxbranch=2,
tamhoja=&tamhoja.,
maxdepth=50,
pvalor=&p_valor.);

   %end;
  %end;
%end;
%mend;
%forest;

/* ELECCION RANDOM FOREST ÓPTIMO */
```

```
data modelos.Resumen_randomforest; set modelos.Resumen_randomforest; if _n_=1 then delete; run;
proc printto; run;
data modelos.mejoresRandom; set modelos.resumen_RandomForest; if (visitas > 113760 ) then output;run;
proc sort data= modelos.mejoresRandom; by descending Bondad_Ajuste; run;


/* MODELO DE RANDOM FOREST OPTIMO */
ods listing close;
proc hpforest data = modelos.entrenamiento_irr5
maxtrees = 500
vars_to_try = 80
trainfraction = 0.8
leafsize = 1000
maxdepth = 50
alpha = 0.2
exhaustive = 5000
missing = useinsearch;
target /* variable dependiente */ / level = nominal;
input
   /* variables explicativas continuas */ / level = interval;
input /* variables explicativas categoricas *// level = nominal;

ods output fitstatistics = estadisticas variableimportance = modelos.importanciaRFoptimo;
save file='C:\Users\Pablo Contreras\Desktop\TFM\Modelos\modeloRF_optimo.bin';
run;

/* Validacion del modelo óptimo */
proc hp4score data = modelos.validacion;
id    /* variables explicativas continuas, categoricas y variable dependiente */
;
score file='C:\Users\Pablo Contreras\Desktop\TFM\Modelos\modeloRF_optimo.bin'
out=modelos.salprediRFoptimo;
run;
ods listing;
```

12.5.4. Código gradient boosting

```
data modelos.resumen_boosting; run;
%macro boosting;
%let lista1='1000 700 500 100 50';
%let nume1=5;
%let lista2='0.001 0.01 0.1 0.2 0.3';
%let nume2=5;
%let lista3='300';
%let nume3=1;
%do k=1 %to &nume1 %by 1;
   data _null_; tamhoja=scanq(&lista1,&k); call symput('tamhoja',left(tamhoja)); run;
   %do m=1 %to &nume2 %by 1;
      data _null_; shrink=scanq(&lista2,&m);call symput('shrink',left(shrink));run;
      %do n=1 %to &nume3 %by 1;
         data _null_;iterations=scanq(&lista3,&n);call symput('iterations',left(iterations)); run;
         proc treeboost data=modelos.entrenamiento_irr5
         shrinkage=&shrink
         maxbranch=2
         maxdepth=50
         iterations=&iterations
         missing=useinsearch
         leafsize=&tamhoja;
         input
         /* variables explicativas continuas */ /level=interval;
         input /* variables explicativas categoricas */  /level=nominal;
         target /* variable dependiente */  /level=nominal;
         SAVE FIT=estadisticas IMPORTANCE=importancia MODEL=resumen RULES=rules;
         score data= modelos.validacion out=salpredi_boosting;
         run;
         ods listing ;

         %do cortemin=20 %to 70 %by 1;

            data tabla1 (keep=Irr5 P_Irr51 PuntoDeCorteMin); set salpredi_boosting;
            PuntoDeCorteMin=(&cortemin/100); run;
            data tabla2; set tabla1;
            if (P_Irr51>=PuntoDeCorteMin) then Irr5_estimado=1; else Irr5_estimado=-1; run;
```

```
              data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var irr5_estimado; output out=VP (keep= VP) n=VP; run;
            data tabla3; set tabla2; if (irr5=-1 and irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FP (keep= FP) n=FP; run;
            data tabla3; set tabla2; if (Irr5=1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=FN (keep= FN) n=FN; run;
            data tabla3; set tabla2; if (Irr5=-1 and Irr5_estimado=-1) then output; run;
            proc means data=tabla3 noprint; var Irr5_estimado; output out=VN (keep= VN) n=VN; run;
            data tabla3; set tabla2; if (Irr5_estimado=1) then output; run;
            proc means data=tabla3 noprint; var Irr5; output out=visitas (keep = visitas) n =
visitas; run;
            proc means data=importancia noprint; var NRules; output out=Rules (keep= Numrules)
sum=Numrules; run;

            data criterio; merge VP FP FN VN Rules visitas; run;
            data criterio; set criterio;
            Bondad_Ajuste = VP/sum(VP, FP);
            Visitas = visitas/720;
            PuntoDeCorteMin=(&cortemin/100);
            leafsize=&tamhoja;
            maxdepth=50;
            exhaustive=5000;
            shrinkage=&shrink;
            iterations=&iterations;
            maxbranch=2;
            run;

            data modelos.resumen_boosting; set modelos.resumen_boosting criterio; run;
          %end;
       %end;
    %end;
%end;
%mend;
%boosting;

proc printto;
run;

data modelos.Resumen_boosting; set modelos.Resumen_boosting; if _n_=1 then delete; run;

/* ELECCION MEJOR BOOSTING */
data modelos.mejoresBoosting; set modelos.Resumen_boosting;
if visitas > 158; run;
proc sort data=modelos.mejoresBoosting; by descending Bondad_Ajuste; run;

/* MODELO DE GRADIENTO BOOSTING OPTIMO */
proc treeboost data=modelos.entrenamiento_irr5
shrinkage=0.001
maxbranch=2
maxdepth=50
iterations=300
missing=useinsearch
leafsize=500;
input
/* variables explicativas continuas */  /level=interval;
input /* variables explicativas categoricas */ /level=nominal;
target /* variable dependiente */  /level=nominal;
SAVE FIT=estadisticas IMPORTANCE=modelos.importanciaGBoptimo MODEL=resumen RULES=rules;
score data= modelos.validacion out=modelos.salprediGBoptimo;
run;
ods listing;
quit;
```

12.5.5. Código ensamble de modelos

```
/* PREPARACION ARCHIVOS */
data fraude (keep= /* variable dependiente */); set modelos.validacion; run;

data salprediLogisticaOptima (keep= P_1); set modelos.salprediLogisticaOptima; run;
data salprediLogisticaOptima (rename=(P_1= P1_Log)); set salprediLogisticaOptima; run;

data salprediRedOptima (keep= P_/* variable dependiente */1); set modelos.salprediRedOptima; run;
data salprediRedOptima (rename= (P_/* variable dependiente */1=P1_Red)); set salprediRedOptima; run;
```

```
data salprediRFOptimo (keep= P_/* variable dependiente */1); set modelos.salprediRFOptimo; run;
data salprediRFOptimo (rename= (P_/* variable dependiente */1=P1_RF)); set salprediRFOptimo; run;

data salprediGBOptimo (keep= P_/* variable dependiente */1); set modelos.salprediGBOptimo; run;
data salprediGBOptimo (rename= (P_/* variable dependiente */1=P1_GB)); set salprediGBOptimo; run;

data modelos.Predicciones; merge salprediLogisticaOptima salprediRedOptima salprediRFOptimo
salprediGBOptimo fraude; run;

/* ENSAMBLE LOGISTICA Y REDES */
%macro ensambleLOGRED;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_LOG P1_RED /* variable dependiente */ P1); set modelos.predicciones; P1=
&stad(P1_LOG, P1_RED); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=1; RED=1; RF=0; GB=0;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE LOGISTICA Y RANDOM FOREST */
%macro ensambleLOGRF;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_LOG P1_RF /* variable dependiente */ P1); set modelos.predicciones; P1=
&stad(P1_LOG, P1_RF); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=1; RED=0; RF=1; GB=0;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE LOGISTICA Y GRADIENT BOOSTING */
```

```
%macro ensambleLOGGB;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_LOG P1_GB /* variable dependiente */ P1); set modelos.predicciones; P1=
&stad(P1_LOG, P1_GB); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=1; RED=0; RF=0; GB=1;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE REDES Y RANDOM FOREST */
%macro ensambleREDRF;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_RED P1_RF /* variable dependiente */ P1); set modelos.predicciones; P1=
&stad(P1_RED, P1_RF); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=0; RED=1; RF=1; GB=0;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE REDES Y GRADIENT BOOSTING */
%macro ensambleREDGB;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_RED P1_GB /* variable dependiente */ P1); set modelos.predicciones; P1=
&stad(P1_RED, P1_GB); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;
```

```
      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=0; RED=1; RF=0; GB=1;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE RANDOM FOREST Y GRADIENT BOOSTING */
%macro ensambleRFGB;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_RF P1_GB /* variable dependiente */ P1); set modelos.predicciones; P1=
&stad(P1_RF, P1_GB); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=0; RED=0; RF=1; GB=1;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE LOGISTICA, REDES Y RANDOM FOREST */
%macro ensambleLOGREDRF;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_LOG P1_RED P1_RF /* variable dependiente */ P1); set modelos.predicciones;
P1= &stad(P1_LOG, P1_RED, P1_RF); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;
```

```
      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=1; RED=1; RF=1; GB=0;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE LOGISTICA, REDES Y GRADIENT BOOSTING */
%macro ensambleLOGREDGB;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_LOG P1_RED P1_GB /* variable dependiente */ P1); set modelos.predicciones;
P1= &stad(P1_LOG, P1_RED, P1_GB); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=1; RED=1; RF=0; GB=1;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE REDES, RANDOM FOREST Y GRADIENT BOOSTING */
%macro ensambleREDRFGB;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_RF P1_RED P1_GB /* variable dependiente */ P1); set modelos.predicciones;
P1= &stad(P1_RF, P1_RED, P1_GB); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=0; RED=1; RF=1; GB=1;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE LOGISTICA, RANDOM FOREST Y GRADIENT BOOSTING */
```

```
%macro ensambleLOGRFGB;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_RF P1_LOG P1_GB /* variable dependiente */ P1); set modelos.predicciones;
P1= &stad(P1_RF, P1_LOG, P1_GB); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=1; RED=0; RF=1; GB=1;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* ENSAMBLE LOGISTICA, REDES, RANDOM FOREST Y GRADIENT BOOSTING */
%macro ensambleLOGREDRFGB;
%let lista1= 'min max median mean';
%let num1= 4;
%do i=1 %to &num1;
   data _null_; stad=scanq(&lista1,&i); call symput("stad",left(stad)); run;
   data tabla1 (keep= P1_RF P1_LOG P1_GB P1_RED /* variable dependiente */ P1); set
modelos.predicciones; P1= &stad(P1_RF, P1_LOG, P1_GB, P1_RED); run;

   %do corte=25 %to 40;
      data tabla2; set tabla1; if (P1>&corte/100) then fraude_estimado=1; else fraude_estimado= -1;
run;

      proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
fraude_estimado=1); quit;
      proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
fraude_estimado=-1); quit;
      proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(fraude_estimado=1); quit;

      data criterio; merge VP FP FN VN visitas; run;
      data criterio; set criterio; Bondad_Ajuste= VP/sum(VP, FP);LOG=1; RED=1; RF=1; GB=1;
      Agregacion="&stad"; P_corte=&corte/100; visitas=(visitas/720); run;

      data modelos.resumen_ensambles; set modelos.resumen_ensambles criterio; run;
   %end;
%end;
%mend;

/* EJECUCION */
data modelos.resumen_ensambles; run;
%ensambleLOGRED;
%ensambleLOGRF;
%ensambleLOGGB;
%ensambleREDRF;
%ensambleREDGB;
%ensambleRFGB;
%ensambleLOGREDRF;
%ensambleLOGREDGB;
%ensambleREDRFGB;
%ensambleLOGRFGB;
```

```
%ensambleLOGREDRFGB;
data modelos.resumen_ensambles; set modelos.resumen_ensambles; if _n_=1 then delete; run;
proc printto; run;

/* ELECCION ENSAMBLE OPTIMO */
data modelos.mejoresEnsamble; set modelos.resumen_ensambles; if (visitas > 158); run;
proc sort data=modelos.mejoresEnsamble; by descending Bondad_Ajuste; run;

/* ENSAMBLE VIA PREDICCIONES COMO VARS INDEPENDIENTES */
data modelos.entrenaensambles; set modelos.predicciones; if Irr5=-1 or Irr5=1 then output; run;

/* LOGISTICA */
%macro ensambleLOG;
data modelos.Resumen_EnsambleLOGIT; run;
proc logistic data=modelos.entrenaensambles;
model Irr5 (event="1")=
   P1_Red
   P1_RF
   P1_GB
/ selection=STEPWISE Link=LOGIT SLSTAY=0.001 SLENTRY=0.2;
score data=modelos.predicciones out=salpredi_ensamble; run;
%do ptocorte=20 %to 45 %by 1;
   data tabla1 (keep=P_1 Irr5); set salpredi_ensamble; run;
   data tabla2 (drop=P_1); set tabla1; if P_1 > (&ptocorte/100) then irr5_estimado=1; else
irr5_estimado=-1; run;

   proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
Irr5_estimado=1);quit;
    proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
Irr5_estimado=1);quit;
    proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
Irr5_estimado=-1);quit;
    proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
Irr5_estimado=-1);quit;
    proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(Irr5_estimado=1); quit;

   data criterio; merge VP FP FN VN visitas; run;
    data criterio; set criterio; Bondad_Ajuste= VP/sum(VP,FP);
   Metodo="LOGIT";
   ptocorte=(&ptocorte/100); run;

   data modelos.Resumen_EnsambleLOGIT; set modelos.Resumen_EnsambleLOGIT criterio; run;
%end;
%mend;
%ensambleLOG;

data modelos.Resumen_EnsambleLOGIT; set modelos.Resumen_EnsambleLOGIT; if _n_=1 then delete; run;
data modelos.mejoresEnsambleLOGIT; set modelos.Resumen_EnsambleLOGIT; if (visitas > 113760); run;
proc sort data=modelos.mejoresEnsambleLOGIT; by descending Bondad_Ajuste; run;

/* REDES NEURONALES */
proc dmdb data=modelos.entrenaensambles dmdbcat=catalogo_aprendizaje_ensamble;
target Irr5; var
   P1_Log
   P1_RF
   P1_GB;
class Irr5; run;

%macro ensambleRED;
data modelos.Resumen_EnsambleRED; run;
proc neural data=modelos.entrenaensambles dmdbcat=catalogo_aprendizaje_ensamble;
input
   P1_Log
   P1_RF
   P1_GB /level=interval;
target Irr5 /level=nominal;

hidden 1 / act=TANH;
train tech=BPROP learn=0.1 momentum=0.4;
netoptions random=0;
score data=modelos.predicciones out=salpredi outfit=salfit; run;

%do ptocorte=20 %to 45 %by 1;
   data tabla1 (keep=P_Irr51 Irr5); set salpredi; run;
   data tabla2 (drop=P_Irr51); set tabla1; if P_Irr51 > (&ptocorte/100) then irr5_estimado=1; else
irr5_estimado=-1; run;
```

```
   proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
Irr5_estimado=1);quit;
    proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
Irr5_estimado=1);quit;
    proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
Irr5_estimado=-1);quit;
    proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
Irr5_estimado=-1);quit;
    proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(Irr5_estimado=1); quit;

   data criterio; merge VP FP FN VN visitas; run;
    data criterio; set criterio; Bondad_Ajuste= VP/sum(VP,FP);
   Metodo="REDES";
   ptocorte=(&ptocorte/100); run;

   data modelos.Resumen_EnsambleRED; set modelos.Resumen_EnsambleRED criterio; run;
%end;
%mend;
%ensambleRED;

data modelos.Resumen_EnsambleRED; set modelos.Resumen_EnsambleRED; if _n_=1 then delete; run;
data modelos.mejoresEnsambleRED; set modelos.Resumen_EnsambleRED; if (visitas > 113760); run;
proc sort data=modelos.mejoresEnsambleRED; by descending Bondad_Ajuste; run;

/* RANDOM FOREST */
%macro ensambleRFO;
data modelos.Resumen_EnsambleRFO; run;
proc hpforest data = modelos.entrenaensambles
maxtrees = 500
vars_to_try = 1
trainfraction = 0.8
leafsize = 100
maxdepth = 50
alpha = 0.2
exhaustive = 5000
missing = useinsearch;
target Irr5 / level = nominal;
input P1_Log P1_Red P1_GB / level = interval;
save file='C:\Users\Pablo Contreras\Desktop\TFM\Modelos\modelosRFensamble.bin';

proc hp4score data = modelos.predicciones;
id P1_Log P1_Red P1_GB Irr5;
score file='C:\Users\Pablo Contreras\Desktop\TFM\Modelos\modelosRFensamble.bin' out=salRF; run;

%do ptocorte=20 %to 45 %by 1;
   data tabla1 (keep=P_Irr51 Irr5); set salRF; run;
   data tabla2 (drop=P_Irr51); set tabla1; if P_Irr51 > (&ptocorte/100) then irr5_estimado=1; else
irr5_estimado=-1; run;

   proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
Irr5_estimado=1);quit;
    proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
Irr5_estimado=1);quit;
    proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
Irr5_estimado=-1);quit;
    proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
Irr5_estimado=-1);quit;
    proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(Irr5_estimado=1); quit;

   data criterio; merge VP FP FN VN visitas; run;
    data criterio; set criterio; Bondad_Ajuste= VP/sum(VP,FP);
   Metodo="RFO";
   ptocorte=(&ptocorte/100); run;

   data modelos.Resumen_EnsambleRFO; set modelos.Resumen_EnsambleRFO criterio; run;
%end;
%mend;
%ensambleRFO;

data modelos.Resumen_EnsambleRFO; set modelos.Resumen_EnsambleRFO; if _n_=1 then delete; run;
data modelos.mejoresEnsambleRFO; set modelos.Resumen_EnsambleRFO; if (visitas > 113760); run;
proc sort data=modelos.mejoresEnsambleRFO; by descending Bondad_Ajuste; run;

/* GRADIENT BOOSTING */
```

```
%macro ensambleGB;
data modelos.Resumen_EnsambleGBO; run;
proc treeboost data=modelos.entrenaensambles
shrinkage=0.001
maxbranch=2
maxdepth=50
iterations=200
missing=useinsearch
leafsize=250;
input P1_Log P1_Red P1_RF / level=interval;
target Irr5 /level=nominal;
score data= modelos.predicciones out=salgbo; run;

%do ptocorte=20 %to 45 %by 1;
   data tabla1 (keep=P_Irr51 Irr5); set salgbo; run;
   data tabla2 (drop=P_Irr51); set tabla1; if P_Irr51 > (&ptocorte/100) then irr5_estimado=1; else
irr5_estimado=-1; run;

   proc sql noprint; create table VP as select count(*) as VP from tabla2 where (Irr5=1 and
Irr5_estimado=1);quit;
    proc sql noprint; create table FP as select count(*) as FP from tabla2 where (Irr5=-1 and
Irr5_estimado=1);quit;
    proc sql noprint; create table FN as select count(*) as FN from tabla2 where (Irr5=1 and
Irr5_estimado=-1);quit;
    proc sql noprint; create table VN as select count(*) as VN from tabla2 where (Irr5=-1 and
Irr5_estimado=-1);quit;
    proc sql noprint; create table Visitas as select count(*) as visitas from tabla2 where
(Irr5_estimado=1); quit;

   data criterio; merge VP FP FN VN visitas; run;
    data criterio; set criterio; Bondad_Ajuste= VP/sum(VP,FP);
   Metodo="GBO";
   ptocorte=(&ptocorte/100); run;

   data modelos.Resumen_EnsambleGBO; set modelos.Resumen_EnsambleGBO criterio; run;
%end;
%mend;
%ensambleGB;

data modelos.Resumen_EnsambleGBO; set modelos.Resumen_EnsambleGBO; if _n_=1 then delete; run;
data modelos.mejoresEnsambleGBO; set modelos.Resumen_EnsambleGBO; if (visitas > 113760); run;
proc sort data=modelos.mejoresEnsambleGBO; by descending Bondad_Ajuste; run;
```

12.5.6. Código post – análisis

```
/* DISTRIBUCION DE INSPECCIONES */
data malo (keep=/* variable dependiente */  fraude color_Estacion); set modelos.salprediRFoptimo
/**/; if P_/* variable dependiente */ 1>0.32 /**/ then fraude=1; else fraude=0; run;
proc freq data=malo; table fraude*estacion; run;
proc freq data=malo; table color_estacion*fraude; run;

/* AGREGACION DEL SCORING */
data borrar (keep=DiaAcum Estacion Irr30); set modelos.validacion; run;
data modelos.salprediRFoptimo; merge borrar modelos.salprediRFoptimo; run;

data posanalisis (keep= Estacion DiaAcum P_/* variable dependiente */ 1 /**/ Irr30 cuenta30); set
modelos.salprediRFoptimo /**/;
dia_ini = 1461;
do i=1 to 25;
if dia_ini =< DiaAcum < dia_ini+30 then cuenta30=i;
dia_ini = dia_ini+30;                                  /* Para saber en que mes estamos */
end;
drop i dia_ini;
run;

/* Nos quedamos con el ultimo scoring de cada mes para cada estacion */
data auxiliar; set posanalisis; run;
proc sort data=auxiliar; by estacion cuenta30 descending diaacum; run;
proc sort data=auxiliar nodupkey; by estacion cuenta30; run;
data auxiliar (rename=(P_/* variable dependiente */ 1=Ultimo_Scoring)); set auxiliar; run;
data auxiliar; set auxiliar; label Ultimo_Scoring=Ultimo_Scoring; run;

/* Media, min, max y mediana del scoring de cada estacion en cada mes */
```

```
proc sort data=posanalisis; by Estacion DiaAcum; run;
proc means data=posanalisis noprint; var P_/* variable dependiente */ 1; by estacion cuenta30;
output out=agregados (keep= Estacion cuenta30 Media_Scoring Max_Scoring Min_Scoring Mediana_Scoring)
mean=Media_Scoring max=Max_Scoring min=Min_Scoring median=Mediana_Scoring; run;
data agregados; set agregados; label Media_Scoring=Media_Scoring Max_Scoring=Max_Scoring
Min_Scoring=Min_Scoring Mediana_Scoring=Mediana_Scoring; run;

/* Nos quedamos con el ultimo valor de Irr30 para cada cuenta30 (mes) ya que por como esta construida
Irr30 agrupa toda la info del mes (30 dias) anterior */
data auxiliarIrr30; set posanalisis; run;
proc sort data=auxiliarIrr30; by estacion cuenta30 descending diaacum; run;
proc sort data=auxiliarIrr30 nodupkey; by estacion cuenta30; run;

/* Union */
data modelos.agregadosDEF; merge agregados auxiliarIrr30 auxiliar; by Estacion cuenta30; run;
proc freq data=modelos.agregadosDEF; table Irr30; run;

data modelos.resumen_Agregacion; run;
%macro resumenAgregacion;
%do p_corte=15 %to 65;

   data aux1; set modelos.agregadosDEF;
   if media_scoring > (&p_corte/100) then fraude_estimadoMedia = 1; else fraude_estimadoMedia = -1;
   if mediana_scoring > (&p_corte/100) then fraude_estimadoMediana = 1; else fraude_estimadoMediana =
   -1;
   if max_scoring > (&p_corte/100) then fraude_estimadoMax = 1; else fraude_estimadoMax = -1;
   if min_scoring > (&p_corte/100) then fraude_estimadoMin = 1; else fraude_estimadoMin = -1;
   if ultimo_scoring > (&p_corte/100) then fraude_estimadoUlt = 1; else fraude_estimadoUlt = -1; run;

   proc sql noprint; create table VPMedia as select count(*) as VP from aux1 where (irr30=1 and
   fraude_estimadoMedia=1);quit;
   proc sql noprint; create table FPMedia as select count(*) as FP from aux1 where (irr30=-1 and
   fraude_estimadoMedia=1);quit;
   proc sql noprint; create table FNMedia as select count(*) as FN from aux1 where (irr30=1 and
   fraude_estimadoMedia=-1);quit;
   proc sql noprint; create table VNMedia as select count(*) as VN from aux1 where (irr30=-1 and
   fraude_estimadoMedia=-1);quit;
   proc sql noprint; create table VisitasMedia as select count(*) as visitas from aux1 where
   (fraude_estimadoMedia=1); quit;

   proc sql noprint; create table VPMediana as select count(*) as VP from aux1 where (irr30=1 and
   fraude_estimadoMediana=1);quit;
   proc sql noprint; create table FPMediana as select count(*) as FP from aux1 where (irr30=-1 and
   fraude_estimadoMediana=1);quit;
   proc sql noprint; create table FNMediana as select count(*) as FN from aux1 where (irr30=1 and
   fraude_estimadoMediana=-1);quit;
   proc sql noprint; create table VNMediana as select count(*) as VN from aux1 where (irr30=-1 and
   fraude_estimadoMediana=-1);quit;
   proc sql noprint; create table VisitasMediana as select count(*) as visitas from aux1 where
   (fraude_estimadoMediana=1); quit;

   proc sql noprint; create table VPMax as select count(*) as VP from aux1 where (irr30=1 and
   fraude_estimadoMax=1);quit;
   proc sql noprint; create table FPMax as select count(*) as FP from aux1 where (irr30=-1 and
   fraude_estimadoMax=1);quit;
   proc sql noprint; create table FNMax as select count(*) as FN from aux1 where (irr30=1 and
   fraude_estimadoMax=-1);quit;
   proc sql noprint; create table VNMax as select count(*) as VN from aux1 where (irr30=-1 and
   fraude_estimadoMax=-1);quit;
   proc sql noprint; create table VisitasMax as select count(*) as visitas from aux1 where
   (fraude_estimadoMax=1); quit;

   proc sql noprint; create table VPMin as select count(*) as VP from aux1 where (irr30=1 and
   fraude_estimadoMin=1);quit;
   proc sql noprint; create table FPMin as select count(*) as FP from aux1 where (irr30=-1 and
   fraude_estimadoMin=1);quit;
   proc sql noprint; create table FNMin as select count(*) as FN from aux1 where (irr30=1 and
   fraude_estimadoMin=-1);quit;
   proc sql noprint; create table VNMin as select count(*) as VN from aux1 where (irr30=-1 and
   fraude_estimadoMin=-1);quit;
   proc sql noprint; create table VisitasMin as select count(*) as visitas from aux1 where
   (fraude_estimadoMin=1); quit;

   proc sql noprint; create table VPUlt as select count(*) as VP from aux1 where (irr30=1 and
   fraude_estimadoUlt=1);quit;
```

```
   proc sql noprint; create table FPUlt as select count(*) as FP from aux1 where (irr30=-1 and
   fraude_estimadoUlt=1);quit;
   proc sql noprint; create table FNUlt as select count(*) as FN from aux1 where (irr30=1 and
   fraude_estimadoUlt=-1);quit;
   proc sql noprint; create table VNUlt as select count(*) as VN from aux1 where (irr30=-1 and
   fraude_estimadoUlt=-1);quit;
   proc sql noprint; create table VisitasUlt as select count(*) as visitas from aux1 where
   (fraude_estimadoUlt=1); quit;

   data criterio2; merge VPMediana FPMediana FNMediana VNMediana visitasMediana; Medida =
   'Mediana';run;
   data criterio1; merge VPMedia FPMedia FNMedia VNMedia visitasMedia; Medida = 'Media'; run;
   data criterio3; merge VPMax FPMax FNMax VNMax visitasMax; Medida = 'Max'; run;
   data criterio4; merge VPMin FPMin FNMin VNMin visitasMin; Medida = 'Min'; run;
   data criterio5; merge VPUlt FPUlt FNUlt VNUlt visitasUlt; Medida = 'Ult'; run;

   data criterio; set criterio2 criterio1 criterio3 criterio4 criterio5;
   Bondad_Ajuste= VP/sum(VP,FP);
   Punto_Corte=(&p_corte/100); run;

   data modelos.resumen_Agregacion; set modelos.resumen_Agregacion criterio; run;
%end;
%mend;
%resumenAgregacion;

/* ELECCIÓN AGREGACIÓN ÓPTIMA */
data modelos.resumen_Agregacion; set modelos.resumen_Agregacion; if _n_=1 then delete; run;
data modelos.resumen_Agregacion; set modelos.resumen_Agregacion; label VP=VP FP=FP FN=FN VN=VN
visitas=visitas; run;
data resumen_Agregacion; set modelos.resumen_Agregacion; if Visitas > 158*24; run;
proc sort data=resumen_Agregacion; by descending Bondad_Ajuste; run;

/* POST - ANALISIS CON GRAFICOS DE LAS VARIABLES */
data visitasModelo; set modelos.salpredirfoptimo /**/; if P_/* variable dependiente */1>0.32 /**/
then Visita=1; else Visita=0; run;
data visitasGNF; set modelos.validacion; if resultado="" then Visita=0; else Visita=1; run;

proc sgplot data=visitasModelo;
   xaxis label='NumVisitas365'; title "Modelo";
   vbar /* variable explicativa */ /group=Visita; run;

proc sgplot data=visitasGNF;
   xaxis label='NumVisitas365'; title "Institucion";
   vbar /* variable explicativa */ /group=Visita; run;

proc sgplot data=visitasModelo;
   xaxis label='NumFraudes365'; title "Modelo";
   vbar /* variable explicativa */ /group=Visita; run;

proc sgplot data=visitasGNF;
   xaxis label='NumFraudes365'; title "Institucion";
   vbar /* variable explicativa */ /group=Visita; run;

proc univariate data=visitasGNF;
  class Visita;
  var /* variable explicativa */; label /* variable explicativa */ = 'CostesMedios91';
  histogram /* variable explicativa */ / vscale=percent nrows=2 odstitle="Institucion";
  ods select histogram;
run;

proc univariate data=visitasModelo;
  class Visita;
  var /* variable explicativa */; label /* variable explicativa */ = 'CostesMedios91';
  histogram /* variable explicativa */ / vscale=percent nrows=2 odstitle="Modelo";
  ods select histogram;
run;

proc univariate data=visitasGNF;
  class Visita;
  var /* variable explicativa */; label /* variable explicativa */ = 'CostesDesv365';
  histogram /* variable explicativa */ / vscale=percent nrows=2 odstitle="Institucion";
  ods select histogram;
run;

proc univariate data=visitasModelo;
  class Visita;
  var /* variable explicativa */; label /* variable explicativa */ = 'CostesDesv365';
```

```
  histogram /* variable explicativa */ / vscale=percent nrows=2 odstitle="Modelo";
  ods select histogram;
run;
```