Modelos recientes de la Estadística y el aprendizaje máquina para la valoración del riesgo de incumplimiento crediticio

Pedro Pinedo Borobio

Trabajo de fin de máster en Modelización e Investigación matemática, estadística y computación
Universidad de Zaragoza

Directores del trabajo : José Tomás Alcalá Nalvaiz y Vanesa Cortés Utrillas

27 de Noviembre de 2019

Abstract

These days, when hundreds of people apply for credit from different banks, banks don't have very reliable methods that allow them to determine whether a person is going to repay the credit, assuming a considerable loss of money in the event the credit is not repaid. Could new technologies, such as machine learning techniques, more accurately establish whether a person is going to repay a credit? This would save banks a lot of money.

Risk analysis is of greatest interest to banks and they are making significant efforts to study and understand the different techniques that allow them to determine trends and behaviors. Some of the most important researches are based on the study of risk calculation of credit default through the probability that a person will return a credit or loan to the bank and the expected money that will be lost in the event that the credit is not returned.

In the history of economics there have arisen many changes marked by the circular economy and its different crises. Due to this, the Basel Accords have been defined. They are the recommendations that the Bank of Europe makes so that financial institutions have a certain minimum capital and it establishes the different methods that can be used for a better calculation of the credit risk that allows to determine if a person is going to pay back their credit or not.

Another important part of the Basel Accords is assuring the interpretability of the data, since it is necessary that it can be explained in a clear and simple way to other banks or supervisors.

For this it is necessary to store personal, economic and market data. This requires an infrastructure that enables banks to organize the credits that they have given to their borrowers, from which different models will be trained to establish the variables that are most important when forecasting whether a Credit will be Default or Fully Paid. Due to technical advancements and the development of the different machine learning techniques, there is a high number of statistical approaches.

In this paper, the problem of calculating credit risk will be studied and modeled mathematically through different statistical techniques, ranging from the oldest such as the logistic regression model, to some of the most modern like XGBoost. A study of both techniques, more thoroughly in XGBoost, will be carried out. Once done, the analyses will be compared through a metric, in this case it will be the AUC, which is the area under the ROC curve, defined by the relation of the sensitivity or rate of true positives and the complementary of the specificity or false positive rate. The goal is to maximize the area enclosed under the curve, therefore, the model with the highest AUC will be better.

Agradecimiento

Agradecimiento a Fundación Ibercaja por la financiación del proyecto a través de la "Catedra Ibercaja de Innovación Bancaria".

Índice general

Capítulo 1

Introducción

1.1. Motivación

El desarrollo tecnológico y científico de las últimas décadas ha permitido estudiar problemas complejos con unas técnicas más óptimas y eficientes, por ejemplo la inteligencia artificial, entre las que se incluyen las distintas técnicas de machine learning.

Para poder aplicar estas técnicas es necesario tener un mínimo conocimiento de matemáticas y estadística. Otro problema que se encuentran al calcular un modelo es la necesidad de realizar una depuración de los datos para que funcione correctamente, esto implica un estudio detallado del conjunto de datos que se va a tratar y de las variables y las posibles relaciones que se pueden apreciar entre ellas.

Actualmente, estas técnicas se usan en la mayoría de los grandes problemas que se presentan. Por ejemplo, la incertidumbre a la que se enfrentan los bancos o páginas o aplicaciones entre inversionistas y prestatarios cuando una persona pide un crédito. Este problema es objeto de estudio de los distintos bancos y aplicaciones, ya que si existiese una técnica fiable que permitiese establecer de una manera segura cuando un crédito será reembolsado, el ahorro sería muy grande. Para intentar resolver este problema, se calcula la probabilidad de endeudamiento o probability default. Otros de los grandes problemas que se plantean en los bancos, pero que no se va a tratar, es intentar determinar la cantidad de dinero que se perdería si el prestatario no pudiese asumir los pagos. Este caso cada vez se está investigando más, definiendo nuevos modelos matemáticos que permitan determinar de una manera fiable el dinero perdido en caso de que el prestatario entre en bancarrota.

Una vez calculada la probabilidad de que una persona te devuelva un crédito, el banco tiene definidos los umbrales en los que si la probabilidad es menor, se concede el crédito. La probabilidad de endeudamiento pueden tener una relación directa con el interés del préstamo en caso de que sea aceptado, ya que a mayor probabilidad, mayores intereses. Esto implica que el precio a devolver será mayor, ya que el prestatario paga el capital + Intereses (pueden ser de distintos tipos) en el número de periodos acordado en cada préstamo.

A la hora de tomar una decisión, el banco o entidad financiera solicita una cantidad de información importante. A veces se encuentra relacionada con la solvencia del cliente , pero también sobre aspectos menos evidentes.

Una vez estos datos son analizados, existen métodos que permiten calcular la probabilidad de endeudamiento. El más habitual y usado por los bancos es el de regresión logística, se

caracteriza por ser una metodología bien asentada y que ofrece resultados satisfactorios en la mayoría de los casos. En la actualidad, se están estudiando distintas técnicas para aplicarlas a este problema, una de las que mejor resultado han dado es la basada en la utilización de árboles con un método de ensamblaje por medio de una técnica boosting; por ejemplo, Gradient Tree Boosting ó Newton Tree Boosting (en particular XGBoost ' Extreme Gradient Boosting').

1.2. Objetivos

El objetivo principal de este trabajo es aplicar algunas de las técnicas de machine learning más modernas al calculo del riesgo de incumplimiento crediticio, e intentar implementar alguna mejora o modificación. Pero para ello, ha sido necesario conseguir varios objetivos adyacentes :

- Estudiar y entender el problema de la probabilidad de endeudamiento o valoración del riesgo de incumplimiento crediticio. Para ello ha sido necesario estudiar la terminología económica utilizada.

- Estudiar los diferentes términos estadísticos : Aprendizaje supervisado, árboles de regresión, técnicas boosting, los métodos del descenso del gradiente, de Newton y de ensamblaje para obtener un modelo más consistente.

- Diseñar un estudio que permita hacer una comparativa entre diversos métodos de estimación del riesgo de incumplimiento crediticio. Una de las herramientas utilizadas para comparar ha sido el cálculo de las curvas ROC y de los respectivos AUC (área encerrada bajo la curva por sus siglas del inglés)

- Aplicación a una base de datos real. Necesidad de un análisis detallado de casos y variables; depuración de la base y ajuste del modelo evitando el sobreajuste.

A continuación se va a exponer algunos de los conceptos que se encontrarán implícitos a lo largo del trabajo.

1.3. Aprendizaje supervisado

El aprendizaje máquina, de acuerdo con [1], se basa en la definición de algoritmos que permitan a los ordenadores o máquinas aprender de un conjunto de datos, que se utilizará para el entrenamiento del modelo, de manera que posteriormente responda a las preguntas que se plantean con una precisión alta. A la hora de desarrollar estos algoritmos, existen dos formas : **aprendizaje supervisado** o **aprendizaje no supervisado**.

En el **aprendizaje supervisado**, los algoritmos trabajan a partir de datos etiquetados, intentado estimar una función que, dadas las variables imput o independientes, les asigne la etiqueta o la probabilidad, a partir de la cual se obtendrá. El algoritmo se entrena con un conjunto de datos a partir del cual aprende y posteriormente asigna una etiqueta, es decir, predice el valor de la variable respuesta. Por lo tanto, se trata de encontrar una relación entre la variable a predecir **y** y las variables independientes. Los dos grandes problemas de la tarea de aprendizaje supervisados son los métodos de regresión y de clasificación.

La ausencia de conocimiento de la variable respuesta marca la diferencia entre estos dos tipos de aprendizaje. En el **aprendizaje no supervisado** sólo se tienen las características que definen la variable a predecir, debido a que no se encuentra la etiqueta o el valor de la respuesta disponible en el conjunto de datos. Una de las principales funciones del aprendizaje no supervisado es la agrupación de las observaciones, es decir, el algoritmo debería ser capaz de encontrar similitudes e ir generando grupos en función de las distintas características. Se encargan de encontrar patrones en las características con el fin de determinar una estructura en el conjunto de variables que lo definen.

El trabajo se va a centrar en el aprendizaje supervisado, donde se tiene un conjunto de datos *lending club* que se va a dividir en: entrenamiento y validación. En ambos conjuntos de datos se encuentra el mismo número de características o variables.

En el conjunto de entrenamiento se encuentran las variables $\mathbf{x_i}$, características de cada individuo, que se utilizarán para definir un modelo que permita calcular la variable respuesta $\mathbf{y_i}$, $\forall i = 1, \ldots, n$, siendo n el número de observaciones del conjunto de datos utilizado.

Por lo tanto, se trata de definir un modelo predictivo que estime las etiquetas o valores para finalmente compararlas con los resultados reales. Para construir un modelo probabilístico es necesario realizar una estimación de los parámetros que lo definen. Una vez calculado el modelo, es necesario ver cuánto de bueno es, para ello se va a definir una función que mida la precisión, se le llamará función objetivo. Este proceso a elegir puede variar, según el tipo de variable a predecir: categórica o continua.

En nuestro caso, la función objetivo será el riesgo empírico (se va a calcular el modelo que menor riesgo presente dentro del conjunto de datos, no de manera global). A continuación, se va a definir los conceptos de función de pérdida y riesgo.

1.4. Función de pérdida

En esta sección se va a introducir la función de pérdida. Es un indicador que mide la discrepancia entre la variable respuesta y el valor estimado a partir de cualquier función o modelo definidos a partir de las variables independientes que tome un valor comparable con la respuesta. Otro concepto subyacente a este es el riesgo, viene definido por la esperanza de la función de pérdida, que es la que se va a utilizar.

El objetivo es obtener la mejor función a través de la estimación de los parámetros que definen el modelo. Existen distintas funciones de pérdida que varían en función de si la variable a predecir es categórica o numérica.

Algunas de las funciones de perdida más habituales son

- Función de pérdida cuadrática

$$L(y, \hat{f}(\mathbf{x})) = \frac{1}{2}(y - \hat{f}(\mathbf{x}))^2$$

- Función de pérdida absoluta

$$L(y, \hat{f}(\mathbf{x})) = |y - \hat{f}(\mathbf{x})|$$

- Cuando la variable a predecir es categórica, se utiliza la función de pérdida 0-1

$$L(y, \hat{f}(\mathbf{x})) = 1(y \neq \hat{f}(\mathbf{x})),$$

  se trata de un indicador que toma valor 1 si está mal clasificada y 0 en caso contrario.

- Si se desea puede ver más información acerca de funciones de pérdidas en [2].

El **riesgo de un modelo** f está relacionado con la función de pérdidas, ya que es la esperanza de la función de pérdida,

$$R(f) = E[L(Y, f(X))], \tag{1.1}$$

El modelo óptimo será el que minimice (1.1)

$$\hat{f} = \arg\min_f R(f) = \arg\min_f E[L(Y, f(X))] = \arg\min_f E[L(Y, f(X))|X]$$

El riesgo de las funciones de pérdida que se han explicado anteriormente serían:

- Función de riesgo cuadrático

$$R(f) = E[(y - f(\mathbf{x}))^2 | X = \mathbf{x}]$$

- Función de riesgo absoluta

$$R(f) = E[|y - f(\mathbf{x})||X = \mathbf{x}]$$

- Función de riesgo 0-1

$$R(f) = Prob[y \neq f(\mathbf{x})|X = \mathbf{x}],$$

- Si se desea más información acerca de funciones de riesgos se puede encontrar en [2].

Debido a de que en numerosas ocasiones no es posible obtener la función de riesgo teórica. Por ejemplo, cuando no se conoce con exactitud el modelo de probabilidad conjunta que relaciona $\mathbf{y}$ y $\mathbb{X}$. Es la mayor parte de las ocasiones. Por lo tanto, se va a estudiar la **función de riesgo empírico**.

1.5. Función de riesgo empírico

Es necesario definir este concepto ya que no se tiene conocimiento de la distribución de probabilidad de la variable respuesta en función del resto de variables. En el caso empírico, se va a considerar una distribución en la que cada una de las observaciones tiene la misma importancia, dicha probabilidad será $\frac{1}{n}$, donde n = "Número de observaciones".

El riesgo empírico es una aproximación a la función riesgo, con la particularidad que sólo se calcula en las observaciones del conjunto de datos, viene dado por

$$\hat{R}(f) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) \tag{1.2}$$

Se va a tratar de buscar un modelo f que nos permita minimizar el riesgo empírico, que se denotará por $\hat{f}$,

$$\hat{f} = \arg\min_{f} \hat{R}(f) \tag{1.3}$$

donde f, son las funciones que definen el espacio de modelos , pueden ser combinaciones lineales de funciones lineales, polinomios, splines, etc.

Una vez definido la minimización de riesgo empírico y la clase de modelo a elegir, se tiene que estudiar el problema de optimización para poder estimar los parámetros que lo definen.

Durante bastantes años el cálculo del riesgo crediticio ha sido calculado por los banco a partir de un modelo de regresión logística o probit.

El modelo de regresión logística, fue introducido por Joseph Berkson (1944) y es una generalización de [17], se formula cuando la variable respuesta es cualitativa, es igual que el modelo de regresión lineal múltiple con la excepción de que la variable a predecir es categórica. En la mayoría de los estudios se trata de una variable dicotómica o binaria. El objetivo es modelar como influye diversas variables en la probabilidad de un suceso, por ejemplo, determinar cuando una persona va a ser capaz de afrontar todos los pagos de un crédito o no.

Por ser una variable dicotómica, se le va a asociar una distribución Bernoulli que verifica

$$\mathbb{E}[y|\vec{X}] = P(y=1|\vec{X}) = f(\beta_0 + \beta_1\mathbf{x}_1 + \ldots + \beta_p\mathbf{x}_p). \tag{1.4}$$

En el caso de que f sea la función logística se tiene el modelo logístico. Si por el contrario f es la función de distribución de probabilidad de una variable normal, entonces se trata de un modelo probit.

Modelo logístico:

La función logística unidimensional viene definida por $f(x) = \frac{e^x}{1+e^x} = \frac{1}{1+e^{-x}}$, $0 < f(x) \leq 1$.

El modelo para cada observación es de la forma

$$P(y=1|\mathbf{x}) = \frac{e^{\mathbf{x}^T\beta}}{1+e^{\mathbf{x}^T\beta}} = \frac{1}{1+e^{\mathbf{x}^T\beta}}$$

entonces

$$Logit(\mathbf{x}) = log\left(\frac{P(y=1|\mathbf{x})}{1-P(y=1|\mathbf{x})}\right) = \beta_0 + \ldots + \beta_k \mathbf{x}_k$$

Los valores a predecir de un modelo logístico se calculan a partir del $Logit(\mathbf{x})$. La predicción de la probabilidad viene dada por la función logística:

$$\hat{p} = \frac{1}{1+e^{-\hat{Logit}(\mathbf{x})}}.$$

En caso de que $\hat{p} > 0{,}5$, se considera que Y se predice a través del modelo con el valor 1 y viceversa en el caso contrario.

Algunos de los conceptos importantes en la regresión logística son

- **odds=**$\frac{P(y=1)}{1-P(y=1)} \in (0,+\infty)$ viene asociado al cociente de la probabilidad de que un suceso ocurra frente a la no ocurrencia. En el caso en que $odds > 1$ entonces la probabilidad de que ocurra es mayor que la de no ocurrencia.

- Sea (1.4). Construyendo su odds y aplicando el logaritmo a la función f,

  $$Logit = log\left(\frac{f(\mathbf{x})}{1-f(\mathbf{x})}\right) = \beta_0 + \beta_1 \mathbf{x}_1 + \ldots + \beta_k \mathbf{x}_k \in (-\infty,\infty),$$

  se trata de un modelo lineal. Por lo tanto, se pueden utilizar técnicas similares a la regresión lineal múltiple.

- **odds-ratio** = $\frac{\frac{P(y=1|F)}{1-P(y=1|F)}}{\frac{P(y=1|F^c)}{1-P(y=1|F^c)}}$. Es el cociente de dos odds en conjuntos complementarios de manera que $\mathbb{X} = F \cup F^c$

Interpretación de los coeficientes β_i:

Son los parámetros que nos van a determinar el valor de Logit. El cociente de los odds $\mathbf{x} + (0,\ldots,0,1,0,\ldots,0)$ y $\mathbf{x}$ correspondientes, cuando $x_i = x+1$ y $x_i = x$, es por definición $odds - ratio_i = e^{\beta_i}, \quad i = 1,\ldots,n$. Por lo tanto, en el modelo de regresión logística los $\beta_i = \log(odds - ratio)_i$.

Para la estimación de los parámetros se va a utilizar el criterio de log(maxima-verosimilitud).

Este método, permite calcular la probabilidad con una fiabilidad aceptable, además no es muy costoso de programar.

Algunas de las **ventajas** que presenta el modelo de regresión logística es la estabilidad, esto quiere decir que estos modelos presentan coeficientes y estimaciones similares aunque el conjunto de datos cambie ligeramente. Por otro lado, es fácil de interpretar en comparación a otros modelos económicos o algunas de las técnicas empleadas, únicamente es necesario entender el funcionamiento de los odds-ratio. Algunos **inconvenientes** que presenta es que

no es capaz de resolver problemas no lineales, en estos casos es mejor utilizar otros modelos como los árboles de decisión. Ademas, cada vez se necesitan modelos más eficientes como consecuencia de que el volumen de variables decisorias es cada vez mayor.

Otros modelos paramétricos lineales que se han utilizado para el problema del riesgo de incumplimiento crediticio es SVM (máquina de vector soporte por sus sigles en inglés). En nuestro caso se va a centrar en modelos lineales no paramétricos, en los basados en árboles de decisión, que fueron introducidos por Breiman (1984) [9].

Capítulo 2

Árboles y boosting.

2.1. Modelos de árbol.

Los árboles de decisión es una modelización de tipo no paramétrico. El procedimiento de construcción de árbol que se va a emplear es CART ('Classification And Regression Trees'), que fue introducido por Breiman, L., H. Friedman, J., A. Olshen, R., and J. Stone, C (1984)[9]. Los árboles de clasificación y regresión son métodos intuitivos que se pueden observar y entender por medio de gráficos de manera sencilla.

El conjunto de datos inicial que se encuentra en el **nodo padre** se divide en dos subconjuntos, en los que se ajustará un valor predicho distinto. Estos a su vez se dividen en otros dos nodos, en cada uno se toma una decisión sobre la dirección a seguir en función del valor de una de las variables independientes. Puede ocurrir que la función de riesgo asociada al árbol disminuya de manera progresiva, consiguiendo en cada división un modelo que clasifique mejor. Como consecuencia es necesario introducir un criterio de parada, por ejemplo, el número máximo de nodos terminales T_{max} ó el mínimo número de observaciones o datos n_{min} en cada nodo terminal. Los nodos finales de un árbol se les llaman **hojas**, al resto que no son el nodo raíz o las hojas se les denomina **nodos internos**. A la hora de estudiar un modelo de árbol, es muy importante optimizar la **profundidad**, que es el número de niveles de partición. En los modelos de árbol es muy frecuente realizar una penalización, de manera que los que presentan mayor profundidad no sean los mejores.

El objetivo principal de los árboles es segmentar la población inicial en grupos homogéneos según la variable respuesta, es decir, intentar dividir el conjunto de variables independientes en conjuntos disjuntos y lo menos correlacionados posibles.

Para la construcción de los árboles es necesario determinar qué variables se utilizarán y cuando se definirá la ramificación. Para ello se va a definir reglas de partición y podado para determinar el tamaño adecuado del árbol. También es necesario tener en cuenta el coste asociado a la complejidad del modelo.

División del árbol:

Antes de aplicar esta técnica, todos los datos se encuentran en el nodo raíz, a partir de un estudio detallado de las características se va a establecer qué variable es la más importante para predecir el valor de **y**. Para ello, es necesario determinar un algoritmo que defina donde debe ramificarse un árbol. Existen distintos métodos para conseguir una ramificación óptima, por ejemplo, el índice de Gini (utilizado en CART) y la entropía o ganancia de información.

Como criterio de partición se va a utilizar el concepto de impureza, servirá para medir la calidad del nodo. En el caso de variables binarias, la máxima impureza se alcanzará cuando $\mathbb{P}(Y=0)=\mathbb{P}(Y=1)=\frac{1}{2}$.

Medidas de impureza

- Índice de Gini: Supongamos que en la variable predictora existen K categorias o clases, se va a relacionar con el índice k. Sea p_{jk} la proporcion de observaciones con etiqueta k en el nodo j. El índice de Gini viene dado por

$$\sum_{k=1}^{K} p_{jk}(1-p_{jk}), \forall j \geq 1.$$

- Entropía : Es la ganancia de pureza asociada a la división. Se expresa como

$$-\sum_{k=1}^{K} p_{jk}\log_2(1-p_{jk}), \forall j \geq 1.$$

En el caso del **índice de Gini** y **entropía**, este proceso se realizará con todas las variables y sus posibles divisiones, de manera que cuando la variable sea continua, el conjunto de posibles divisiones es muy grande. Esto hace que calcular la división óptima sea un problema complejo y con un coste computacional grande, esto ocurre también cuando se tiene variables categóricas que no están ordenadas, de manera que en cada ramificación se tienen que mirar 2^k grupos, siendo k el número de categorias. Se eligirá la ramificación que presente el menor valor de impureza o entropía cuando sea menor que la impureza o entropía del nodo padre.

Los modelos de árbol definen el conjunto generado por las variables independientes X en M rectángulos que permiten predecir el valor de la variable $\mathbf{y}$ por medio de un ajuste sencillo en cada uno de ellos. Las regiones $R_1,\ldots,R_M$ se caracterizan por ser disjuntas, con un número similar de observaciones $\mathbf{x_i}$ en cada grupo y porque en cada una se va a ajustar un modelo simple, es decir, a cada individuo del nodo terminal se le asigna el valor medio de la variable respuesta en este nodo o la etiqueta más frecuente en dicho nodo, de manera que con la suma se obtiene el modelo que predice $\hat{\mathbf{y}}$. En el trabajo se va a hablar únicamente de **árboles binarios**, cada nodo se puede ramificar como máximo en dos nodos.

Algunos de los modelos de árbol más simples son

$$f(\mathbf{x}) = \sum_{j=1}^{M} w_j I(\mathbf{x} \in R_j),$$

siendo w_j el valor predicho para el dato o input $\mathbf{x_i} \in R_j$, en nuestro caso será la etiqueta más común dentro del conjunto R_j.

Por lo tanto, al realizar la suma como son conjuntos disjuntos, el dato se encontrará en uno de los R_j y tendrá asignado el valor o etiqueta w_j asociada a su nodo terminal.

Algoritmo de aprendizaje

Para la construcción de un modelo de árbol es necesario estimar los parámetros para obtener el árbol óptimo. Primero hay que definir la complejidad del modelo, que viene asociada a la profundidad ó al número de nodos terminales. Una forma de reducirla es estableciendo una cota inferior en el mínimo número de observaciones en los nodos terminales, de esta manera se limitará el número de particiones disjuntas del conjunto de características. Otra técnica utilizada es la diferencia del peso de los nodos terminales dos a dos.

La función objetivo a minimizar es la suma del riesgo empírico $\hat{R}(f)$ definida en (1.2) y la penalización asociada al número máximo de nodos terminales,

$$J(f) = \hat{R}(f) + \gamma T_{\text{máx}} = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \sum_{j=1}^{T_{max}} w_j 1(\mathbf{x_i} \in R_j)) + \gamma T_{\text{máx}},$$

siendo γ el parámetro de complejidad del modelo. El árbol crece a lo sumo hasta que el número de nodos terminales sea T_{max}, posteriormente se realiza una poda eliminando los nodos en los que la ramificación no ha minimizado el riesgo empírico.

Se puede observar que es difícil encontrar una solución a este problema, por lo tanto, se va a calcular una solución aproximada que minimice la función del riesgo empírico, sin la penalización. Para ello, primeramente se estimarán los pesos, suponiendo que las regiones R_1, $\ldots$, $R_{T_{\text{máx}}}$ son conocidas. Una vez estimados, se calculará la estructura asociada al árbol.

Dadas las regiones R_j, j=1,$\ldots$, $T_{\text{máx}}$, la estimación de los valores o etiquetas en cada una de ellas es igual a

$$\hat{w}_j = \arg\min_{w} \sum_{\mathbf{x_i} \in R_j} L(y_i, w).$$

Esto es debido a la división del conjunto de características en subconjuntos, permitiendo calcular w_j en cada uno de los nodos terminales.

Una vez estimados, el riesgo empiríco viene dado por

$$\hat{R}(\hat{f}) = \sum_{j=1}^{T_{\text{máx}}} \sum_{\mathbf{x_i} \in R_j} L(y_i, \hat{w}_j) = \sum_{j=1}^{T_{\text{máx}}} \hat{L}_j,$$

siendo $\hat{f}$ definida en (1.3).

Criterio de poda

A partir del modelo estimado, se va a realizar un podado del árbol, yendo desde los nodos terminales hasta el nodo raíz, eliminando las divisiones en las que no se minimiza el riesgo empírico. Supongamos sin perdida de generalidad la división definida en el nodo k, el riesgo empírico pasa de la forma

$$\hat{R}(\hat{f}_{before}) = \sum_{j \neq k} \hat{L}_j + \hat{L}_k$$

a ser

$$\hat{R}(\hat{f}_{after}) = \sum_{j \neq k} \hat{L}_j + \hat{L}_{kI} + \hat{L}_{kD}. \tag{2.1}$$

Si la diferencia entre los riesgo empíricos en el nodo k,denominada **ganancia** asociada a la división del nodo,

$$Ganancia = \hat{R}(\hat{f}_{before}) - \hat{R}(\hat{f}_{after}) = \hat{L}_k - (\hat{L}_{kI} + \hat{L}_{kD}),$$

es mayor que 0, no se poda. Se va a podar por un nodo si la probabilidad de mala clasificación del subárbol descendiente del nodo no mejora la probabilidad de malclasificación existente en el nodo.

Algunas de las **ventajas** que presentan los árboles son

- Son muy fáciles de interpretar y de programar.
- Puede entenderlo cualquier persona que no tenga un conocimiento de estadística. Es una forma de pensar similar a la que utilizan los seres humanos.
- Se puede utilizar para variables categóricas o continuas indistintamente.
- Las ramas del árbol definen las diferentes reglas de asignación.
- Sabe tratar los valores ausentes durante el entrenamiento a partir de la división secundaria, de manera que si algún dato no tiene definido el valor en la variable decisora, existan características, denominadas reglas secundarias, de manera que se le asigne a un subnodo. Es decir, esto permite que si una observación no tiene la variable de decisión se pueda seguir clasificando, ver más detalladamente en [7] (Página 180) y [9]. Además, se caracteriza por ser robusto frente a valores atípicos.
- Invariante frente a transformaciones de los datos o variables. Además puede capturar las relaciones no lineales de los datos. Así como una gran correlación entre las variables.

Inconvenientes

- Poder predictivo limitado. Para solucionar este problema se puede mezclar con algún método, por ejemplo Boosting, Random Forest ó Bagging.
- Tiende a seleccionar variables con un alto número de valores distintos.
- Tiende a sobreajustar en casos donde hay predictores con muchas categorías.
- Inestable y alta varianza, esto indica que pequeñas modificaciones en los datos implican árboles y reglas muy diferentes.

Las ventajas e inconvenientes se pueden consultar más detalladamente en [9] y [7].

2.2. Metaclasificadores

Para mejorar algunos de los inconvenientes que presentan los árboles, se va a aplicar metaclasificadores. En el trabajo se va a utilizar el método Boosting aunque existen otros métodos con los que se obtienen unos resultados bastante buenos, por ejemplo, **Bagging** ó **Random Forest**.

2.2.1. Bagging

Los árboles de decisión estudiados hasta ahora son propensos a sufrir cambios importantes en función del conjunto de entrenamiento que se toma, se dice entonces que el clasificador presenta mucha variabilidad. Una idea muy utilizada para reducir dicha variabilidad es utilizar técnicas Bootstrap (de remuestreo).

Fruto de estos conceptos nace el Bagging o Bootstrap Aggregation, fue introducido por Breiman, L. (1996)[11], como una técnica de recombinación de árboles que surge para tratar de disminuir la variabilidad. La idea es simple, se trata de ajustar diferentes árboles mediante distintos muestreos y después se toma la clase más frecuente para la predicción de un nuevo individuo. Es decir, clasificar un individuo en la clase más predicha por una serie de árboles generados sobre conjuntos de entrenamiento diferentes. Cabe añadir que los árboles usados nunca están podados, pues nos interesa reducir el sesgo en cada árbol, ya que el hecho de que la variabilidad aumente (si no se poda) no resultará un problema al ponderar muchos árboles, ya que esto hará que se reduzca la varianza.

Algunas de las desventajas que presentan la técnica Bagging es que se ajusta una serie de árboles que pueden ser similares, especialmente cuando existen variables cuya importancia a la hora de clasificar son muy superior a otras. Esto hace que las predicciones de los árboles estén fuertemente correladas y en consecuencia que la técnica Bagging podría resultar más lenta a la hora de reducir la variabilidad de las predicciones.

2.2.2. Random Forest

Aunque la técnica Bagging introduce una componente aleatoria cuando se eligen las observaciones para definir el árbol, no se consigue que los árboles calculados sean independientes entre sí como consecuencia de que siempre se utilizan las mismas variables y la estructura será similar. En relación con posibles mejoras, se realizó un estudio detallado, a partir del cual se definió la técnica Random Forest, que fue introducida por Breiman, L. (2001) [12]. Este método se basa en coger N submuestras del conjunto de datos de manera aleatoria con reemplazamiento, permitiendo la existencia de variabilidad. Estas remuestras presentan el mismo número de observaciones que el conjunto de datos original. Para reducir la correlación entre los árboles calculados, en cada división se va a incluir aleatoriedad para la elección del conjunto de predictores. En el método Random Forest, cada árbol crece hasta la máxima profundidad que se ha establecido como límite y se caracteriza por no tener proceso de poda.

Por lo tanto, se trata de una técnica de remuestreo en las variables y observaciones, haciendo que los árboles sean independientes, esto hace que se reduzca el riesgo de sobreajuste.

Además, por tratarse de un método de ensamblaje, al promediar las predicciones de los distintos árboles, se mantiene estable con nuevas observaciones o datos. También presenta algún inconveniente, por ejemplo es muy difícil de interpretar (más que el Bagging) y más complejo, siendo necesario estimar dos parámetros adicionales en lugar de uno (número máximo de árboles y el porcentaje de variables predictoras que se eligirán en cada división).

2.2.3. Boosting

La idea principal de este método coincide con los anteriores métodos de ensamblaje o consenso al combinar distintos clasificadores débiles para obtener un clasificador que predijese de manera más precisa, sin embargo presenta algunas diferencias:

- Se ponderan las muestras para concentrar el aprendizaje en las observaciones más difíciles de clasificar.
- Las observaciones que se encuentran más cercanas a la frontera son más difíciles de clasificar, esto implica que los pesos asociados serán altos.
- Se ponderan los pesos asociados a los clasificadores débiles, de manera que no todos tienen la misma importancia.

Uno de los métodos Boosting más conocidos es **AdaBoost**, fue introducido por Freund, Y. and Schapire, R. E. (1996) [13], en cada paso se utiliza un modelo simple que ajuste al conjunto de datos con el que se está trabajando. Inicialmente, se le da la misma importancia a cada una de las observaciones, modificándola tras cada paso por la ponderación de los pesos, de esta forma se le da más importancia a las observaciones mal clasificadas hasta obtener un ajuste aceptable. Para finalizar, el clasificador es la media ponderada de los pesos con el correspondiente modelo definido. El funcionamiento puede observarse en la Figura 2.1.

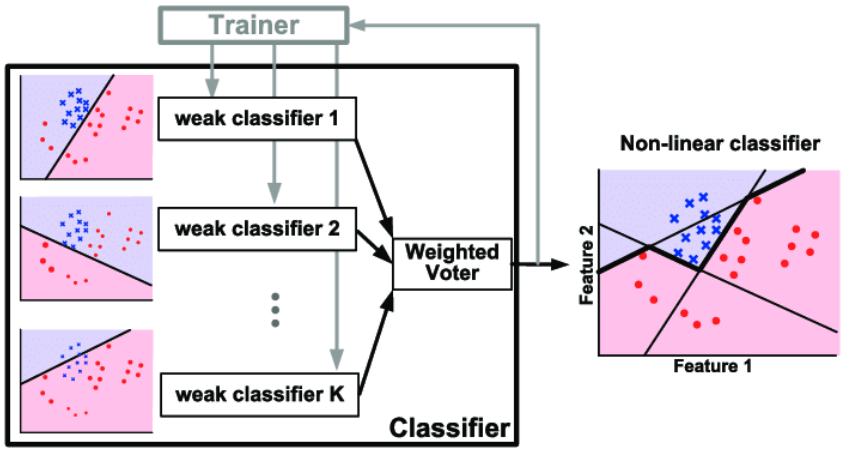

Figura 2.1: Funcionamiento del AdaBoost con k pasos. Se puede observar que en cada paso se da mayor importancia a las observaciones mal clasificadas. Por último se hace una media ponderada de todos los modelos. La foto se ha obtenido de [19].

El modelo matemático final viene dado por

$$\hat{f}(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m \hat{f}_m(\mathbf{x}),$$

siendo una combinación lineal de los distintos modelos que se han ajustado compensados con el valor de los pesos α_m, como consecuencia de elegir ese modelo, $m = 1, \ldots, M$.

El algoritmo paso a paso del AdaBoost se puede ver en Algoritmo 1.

Entrada:

Conjunto de datos

Número máximo de modelos a predecir M

Número de datos n.

Inicialización: Sea el conjunto de puntos $y_i \in \{-1, +1\}$ y sus pesos asociados $w_i = \frac{1}{n}$, $i = 1, \ldots, n$;

para $m = 1, \ldots, M$ **hacer**

Ajuste del modelo $\hat{f}_m(\mathbf{x}) \in \{-1, +1\}$

Calculo de la suma de los pesos de las observaciones mal clasificadas: $e_m = \sum_{i=1}^{n} w_i^m 1(y_i \neq \hat{f}_m(\mathbf{x_i}))$

Calculo de los nuevos pesos: $\alpha_m = \log\left(\frac{1-e_m}{e_m}\right)$

Definir los pesos para el siguiente paso: $\mathbf{w}_i^{m+1} = \mathbf{w}_i^m \exp(\alpha_m 1(y_i \neq \hat{f}_m(\mathbf{x_i})))$, normalizado de forma que $\sum_{i=1}^{n} w_i^{m+1} = 1$

fin

Resultado: $\hat{f}(\mathbf{x}) = signo(\sum_{m=1}^{M} \alpha_m \hat{f}_m(\mathbf{x}))$

Algoritmo 1: Algoritmo AdaBoost

Este procedimiento o método equivale a minimizar una función de riesgo empírico de la forma

$$\hat{R}(f) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) = \frac{1}{n}\sum_{i=1}^{n} \exp(-y_i f(\mathbf{x}_i)),$$

como se puede ver en [2] (páginas 34-35).

Por lo tanto, en cada etapa se trata de calcular los pesos α_m y $\mathbf{w}^{m+1}$ que minimizan el riesgo empírico. Para ello, se elige ajustar un clasificador débil para compensar los puntos mal clasificados en los pasos anteriores que están caracterizados por tener pesos grandes.

Inconveniente: Para problemas que no son sencillos, se requieren demasiados clasificadores débiles para conseguir un modelo final aceptable. Esto implica que el entrenamiento de estas técnicas puede tener un coste computacional grande. Es un algoritmo secuencial, hasta que no se tiene el submodelo f_m no se puede comenzar a estimar el submodelo f_{m+1}. La principal **ventaja**: Es robusto frente al sobreajuste.

Boosting avanzado (Gradient y Newton)

En el trabajo, se va a estudiar más detalladamente las técnicas de Gradient y Newton boosting. Primeramente, se hablará de los método iterativos, gradiente descendiente y Newton, que se van a aplicar. En cada paso, se trata de ajustar un modelo $f(\mathbf{x}) = f(\mathbf{x};\mathbf{w})$ a través de la parametrización de $\mathbf{w}$. La función de riesgo a minimizar se puede escribir a partir de los parámetros a estimar como

$$\hat{R}(\mathbf{w}) = \mathbb{E}(L(\mathbf{y}, f(\mathbb{X};\mathbf{w}))).$$

Como $\hat{R}$ viene definida por el riesgo empírico es diferenciable respecto $\mathbf{w}$, por lo tanto, se puede utilizar un método numérico de orden 1 como el gradiente descendiente. En el caso del método de Newton es necesario que $\hat{R} \in \mathscr{C}^2$, es decir, dos veces diferenciable.

En cada iteración se van actualizando los valores de $\mathbf{w}$. Los pesos en la i-ésima iteracción viene dados por

$$\begin{cases} \mathbf{w}^{(i)} = \mathbf{w}^{(i-1)} + \mathbf{w}_i, \ i \geq 1 \\ \mathbf{w}^{(0)} = \mathbf{w}_0 \end{cases}$$

siendo $\mathbf{w}_i$ el paso dado en la iteración i-ésima, con $i = 1, \ldots, M$.

Descenso de gradiente o de máxima pendiente

El descenso de gradiente es un algoritmo numérico de optimización iterativo de primer orden que se emplea para encontrar el mínimo de una función, fue propuesto inicialmente por Cauchy, A-L. (1847) [2] (36). Para reducir el riesgo empírico, se van a coger pasos proporcionales al negativo del gradiente

$$-\hat{\mathbf{g}}_i = -\nabla_{\mathbf{w}} \hat{R}(\mathbf{w})|_{\mathbf{w}=\mathbf{w}^{(i-1)}}, \quad i \geq 1, \tag{2.2}$$

y se obtiene un nuevo punto dado por

$$\mathbf{w}^{(i)} = \mathbf{w}^{(i-1)} - \rho \hat{\mathbf{g}}_i, i \geq 1,$$

siendo ρ la **amplitud del paso**, se trata de una tasa para compensar el efecto del gradiente negativo.

Hay que elegir un valor de ρ que garantice disminuir el riesgo empírico $\hat{R}(\mathbf{w}^{(i+1)})$ de una iteracción a la siguiente, la tasa de paso óptima se denotará por ρ_i , $\forall i \in \mathbb{N}$ (descenso monótomo de la función de riesgo empírico).

El paso en la iteración i-ésima es igual a

$$\mathbf{w}_i = -\rho_i \hat{\mathbf{g}}_i$$

Método de Newton

El método de Newton se puede utilizar para calcular los máximos o mínimos de una función calculando los valores en los que la primera derivada se anula. En condiciones generales el método de Newton es de orden 2. En el trabajo, el método está basado en encontrar las raíces de la primera derivada del riesgo empírico respecto $\mathbf{w_i}$

$$\nabla_{\mathbf{w}_i}\hat{R}(\mathbf{w}^{(i-1)}+\mathbf{w}_i)=0,\ i\geq 1 \tag{2.3}$$

Hay que minimizar en cada iteracción el riesgo en función de $\mathbf{w}_i$. Haciendo un desarrollo de Taylor de orden 2 en el punto $\mathbf{w}^{(i-1)}$, $\forall i \geq 1$ y definiendo la matriz Hessiana

$$\hat{\mathbb{H}}_i=\nabla^2_{\mathbf{w}}\hat{R}(\mathbf{w})|_{\mathbf{w}=\mathbf{w}^{(i-1)}}$$

se obtiene que

$$\hat{R}(\mathbf{w}^{(i-1)}+\mathbf{w}_i)\approx\hat{R}(\mathbf{w}^{(i-1)})+\hat{\mathbf{g}}_i^T\mathbf{w}_i+\frac{1}{2}\mathbf{w}_i^T\hat{\mathbb{H}}_i\mathbf{w}_i,\ i\geq 1. \tag{2.4}$$

Juntando (2.3) y (2.4)

$$\nabla_{\mathbf{w}_i}\hat{R}(\mathbf{w}^{(i-1)}+\mathbf{w}_i)\approx\mathbf{g}_i^T+\hat{\mathbb{H}}_i\mathbf{w}_i=0,\ i\geq 1$$

consecuentemente se tiene que

$$\mathbf{w}_i=-\mathbb{H}_i^{-1}\mathbf{g}_i^T. \tag{2.5}$$

Para obtener el valor de $\mathbf{w}_i$ es necesario que exista $\mathbb{H}_i^{-1}$, es decir, que la función riesgo $\hat{R}$ sea de clase $\mathscr{C}^2$.

El paso natural del método de Newton es 1. En algunos casos, el gradiente descendiente y Newton son equivalentes; por ejemplo, en el caso donde la función de pérdidas es el error cuadrático. Algunos de los inconvenientes que presenta el método de Newton es que existen funciones que no son dos veces derivables y por lo tanto, no podría aplicarse la técnica asociada.

Se puede realizar el método de gradiente descendiente y de Newton en espacio de funciones, puede verse detalladamente en [2].

A continuación se va a combinar estas técnicas de optimización con el método Boosting [14].

Gradient Boosting

Gradient Boosting o potenciación del gradiente fue introducido por H.Friedman, J. (1999) [6]. Al igual que AdaBoost, Gradient Boosting es una media ponderada de los modelos secuenciales caculados en cada iteración, es decir, se trata de construir un modelo paso a paso que minimice los valores residuales generados en las iteraciones previas.

Por lo tanto, hay que minimizar el riesgo empírico en función del gradiente descendiente (2.2),

$$-\hat{g}_i(\mathbf{x}) = -\left[\frac{\partial \hat{R}(f(\mathbf{x}))}{\partial f(\mathbf{x})}\right]_{f(\mathbf{x})=\hat{f}^{(i-1)}(\mathbf{x})}, \quad \mathbf{x} \in \mathbb{X}, \quad i \geq 1.$$

Debido a que el riesgo es empírico y que $\hat{g}_i$ está definido para cada uno de los datos $\{\mathbf{x_j}\}_{j=1}^n$, es necesario realizar un ajuste global del gradiente negativo para generalizarlo en todo el espacio de características $\mathbb{X}$. Se va a restringir las posibles soluciones a la base de un espacio de funciones Φ, que tendrá que minimizar el error cuadrático de la diferencia empírica del gradiente negativo y una combinación lineal de las funciones de la base del espacio:

$$\hat{\phi}_i = \arg_{\phi \in \Phi, \beta} \text{mín} \sum_{j=1}^{n} [(-\hat{\mathbf{g}}_i(\mathbf{x_j})) - \beta \phi(\mathbf{x}_j)]^2, \; i \geq 1$$

Se puede ver que $\hat{\phi}_i$ es la mejor aproximación a $-g_i(\mathbf{x_j})$ en Φ, combinación lineal de las funciones de la base, por lo tanto se espera una alta correlación.

Una vez definida la dirección del paso en cada iteración, se va a calcular la amplitud del paso. Para ello, es necesario minimizar la función

$$\hat{R}(\hat{f}^{(i-1)} + \rho \hat{\phi}_i), \; i \geq 1$$

en función de ρ. Al tamaño del paso óptimo se le denotará con ρ_i,$i \geq 1$.

Friedman, J. H.(1999) [6] introdujo la constante de contracción $0 \leq \eta \leq 1$, parámetro que multiplica la longitud del paso para regularlo. La constante η se define como la **tasa de aprendizaje** de manera que cuanto menor es el valor, más lento es el aprendizaje de manera que evita que el algoritmo de optimización caiga en ciclos que no converjan.

Combinando la teoría del gradiente descendiente y la tasa de contracción se obtiene que el paso en la iteracción i-ésima es

$$\hat{f}_i(\mathbf{x}) = \eta \rho_i \hat{\phi}_i(\mathbf{x}), \; i \geq 1.$$

El modelo que se obtiene a partir de la media ponderada de los clasificadores débiles pasadas K iteraciones, $\hat{f}^{(K)}$ (es una aproximación a $\hat{f}$), viene dado por

$$\hat{f}^{(K)}(\mathbf{x}) = \sum_{i=0}^{K} \hat{f}_i(\mathbf{x}).$$

El algoritmo paso a paso del Gradient Boosting

Entrada:

Conjunto de datos

Función de pérdidas L.

Número máximo de iteraciones K.

Tasa de contracción η.

Inicialización : $\hat{f}^{(0)}(\mathbf{x}) = \omega_0 = \arg_{\omega} \min \sum_{i=1}^{n} L(y_i; \omega)$

para $j = 1, \ldots, K$ **hacer**

$\hat{g}_j(\mathbf{x_i}) = \left[\frac{\partial \hat{R}(f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)}\right]_{f(\mathbf{x})=\hat{f}^{(j-1)}(\mathbf{x})}$

$\hat{\phi}_j = \arg_{\phi \in \Phi, \beta} \min \sum_{i=1}^{n} [(-\hat{\mathbf{g}}_j(\mathbf{x_i})) - \beta \phi(\mathbf{x}_i)]^2$

$\hat{\rho}_j = \arg_{\rho} \min \sum_{i=1}^{n} L(y_i, \hat{f}^{(j)}(\mathbf{x}_i) + \rho \hat{\phi}_j(\mathbf{x}_i))$

$\hat{f}_j(\mathbf{x}) = \eta \hat{\rho}_j \hat{\phi}_j(\mathbf{x})$

$\hat{f}^{(j)}(\mathbf{x}) = \hat{f}^{(j-1)}(\mathbf{x}) + \hat{f}_j(\mathbf{x})$

fin

Resultado: $\hat{f}(\mathbf{x}) \equiv \hat{f}^{(K)}(\mathbf{x}) = \sum_{j=0}^{K} \hat{f}_j(\mathbf{x}) = \omega_0 + \eta \sum_{j=1}^{K} \hat{\rho}_j \hat{\phi}_j(\mathbf{x})$

Algoritmo 2: Algoritmo Gradient Boosting

Newton Boosting

Al igual que en Gradient Boosting se define el gradiente particular asociada a cada una de las observaciones o datos, en el hessiano ocurre lo mismo. Se va a definir de la siguiente forma

$$\hat{\mathbb{H}}_j(\mathbf{x}) = \left[\frac{\partial^2 \hat{R}(f(\mathbf{x}))}{\partial^2 f(\mathbf{x})}\right]_{f(\mathbf{x})=\hat{f}^{(j-1)}(\mathbf{x})}.$$

En general, el hessiano se va a considerar únicamente en el caso unidimensional

Por lo tanto, es necesario un conjunto de funciones que realicen un ajuste global del gradiente y la hessiana de manera conjunta. Para ello, es necesario tener en cuenta el desarrollo de Taylor mostrado en (2.4) y que se trata del riesgo empírico. Una aproximación al paso obtenido por el método de Newton, teniendo en cuenta que $\hat{R}(\hat{f}^{(j-1)})$ es constante, viene dado por

$$\hat{\phi}_j = \arg_{\phi \in \Phi} \min \sum_{i=1}^{n} [\hat{g}_j(\mathbf{x}_i)\phi(\mathbf{x_i}) + \frac{1}{2}\hat{\mathbb{H}}_j(\mathbf{x}_i)\phi(\mathbf{x_i})^2] \tag{2.6}$$

.

Completando cuadrados y como se está minimizando respecto de ϕ, $\frac{\hat{g}_j(\mathbf{x_i})}{\hat{\mathbb{H}}_j(\mathbf{x_i})}$ es una constante y no se puede minimizar. En consecuencia, la aproximación al paso de Newton es

$$\hat{\phi}_j = \arg_{\phi\in\Phi} \min \sum_{i=1}^{n} \hat{\mathbb{H}}_j(\mathbf{x}_i) \left[(-\frac{\hat{g}_j(\mathbf{x_i})}{\hat{\mathbb{H}}_j(\mathbf{x}_i)}) - \phi(\mathbf{x_i}) \right]^2$$

Consecuentemente, en cada iteración hay que encontrar la función que minimiza el cuadrado de la diferencia $(-\frac{\hat{g}_j(\mathbf{x_i})}{\hat{\mathbb{H}}_j(\mathbf{x}_i)}) - \phi(\mathbf{x_i})$ ponderado por la hessiana. Equivalentemente, se trata de encontrar un ajuste global del gradiente negativo multiplicado por $\frac{1}{\hat{\mathbb{H}}_j(\mathbf{x}_i)}$.

El paso en la iteración j-ésima viene dado por

$$\hat{f}_j(\mathbf{x}) = \eta \hat{\phi}_j(\mathbf{x}),$$

siendo $\eta \in (0,1)$ la tasa de contracción o aprendizaje. Repitiendo este proceso, tras K iteraciones se obtiene que la aproximación a $\hat{f}(\mathbf{x})$ es $\hat{f}^{(K)}(\mathbf{x})$,

$$\hat{f}(\mathbf{x}) \equiv \hat{f}^{(K)}(\mathbf{x}) = \sum_{j=0}^{K} \hat{f}_j(\mathbf{x}).$$

El algoritmo de Newton Boosting es similar a Gradient Boosting

Entrada:

Conjunto de datos

Función de pérdidas L.

Número máximo de iteraciones K.

Tasa de aprendizaje η.

Inicialización : $\hat{f}^{(0)}(\mathbf{x}) = \omega_0 = \arg_\omega \min \sum_{i=1}^{n} L(y_i; \omega)$

para $j = 1, \ldots, K$ **hacer**

$\hat{g}_j(\mathbf{x}) = \left[\frac{\partial \hat{R}(f(\mathbf{x}))}{\partial f(\mathbf{x})} \right]_{f(\mathbf{x})=\hat{f}^{(j-1)}(\mathbf{x})}$

$\hat{\mathbb{H}}_j(\mathbf{x}) = \left[\frac{\partial^2 \hat{R}(f(\mathbf{x}))}{\partial^2 f(\mathbf{x_i})} \right]_{f(\mathbf{x})=\hat{f}^{(j-1)}(\mathbf{x})}$

$\hat{\phi}_j = \arg_{\phi\in\Phi} \min \sum_{i=1}^{n} \hat{\mathbb{H}}_j(\mathbf{x}_i) \left[(-\frac{\hat{g}_j(\mathbf{x_i})}{\hat{\mathbb{H}}_j(\mathbf{x}_i)}) - \phi(\mathbf{x_i}) \right]^2$

$\hat{f}_j(\mathbf{x}) = \eta \hat{\phi}_j(\mathbf{x})$

$\hat{f}^{(j)}(\mathbf{x}) = \hat{f}^{(j-1)}(\mathbf{x}) + \hat{f}_j(\mathbf{x})$

fin

Resultado: $\hat{f}(\mathbf{x}) \equiv \hat{f}^{(K)}(\mathbf{x}) = \sum_{j=0}^{K} \hat{f}_j(\mathbf{x}) = \omega_0 + \sum_{j=1}^{K} \eta \hat{\phi}_j(\mathbf{x})$

Algoritmo 3: Algoritmo Newton Boosting

Newton Boosting es la técnica que se aplica en el paquete de R, denominado XGBoost.

Parámetros a optimizar en los distintos métodos.

En cada iteración, tanto en el método de Newton como el del gradiente, se calcula el mejor clasificador débil a través del aprendizaje del modelo en el conjunto de entrenamiento. La mejor función se elige de manera que sea lo más simple posible, es decir, la que pueda tener un sesgo alto pero presente poca variabilidad. Como se ha dicho antes, estas funciones tienen que minimizar el error cuadrático medio ponderado. Algunas de las funciones utilizadas para minimizar son los árboles de regresión o decisión.

La dificultad de estos problemas se halla en calcular el mejor valor de los parámetros de manera que capte la tendencia y no se ajuste demasiado a los datos particulares con los que se está entrenando el modelo, en ese caso será incapaz de reconocer y predecir correctamente datos nuevos, este concepto está relacionado con el sobreajuste. Además, es necesario determinar el número máximo de iteraciones, ya que cuanto mayor sea el número, mayor será el conjunto de funciones a optimizar y el coste computacional, así como la complejidad del modelo. Por otro lado, hay que optimizar la tasa de aprendizaje η.

Debido a que estas cantidades no son independientes, se va a optimizar el número máximo de iteraciones suponiendo que η es un valor fijo y a continuación se optimizará. El valor óptimo se va a calcular con una técnica de validación cruzada de 5 subconjuntos elegidos aleatoriamente y con reemplazamiento, de esta forma se obtendrá un número máximo de iteracciones más fiable. En cuanto a η se observa que a menor valor, mejor comportamiento general tiene y mayor es el número de iteraciones. Hay que elegir la tasa de aprendizaje realizando un estudio del coste computacional asociado al número de iteraciones. Entre ellas, se eligirá la tasa η que tenga menor valor, siendo aceptable el coste computacional.

2.3. Combinación de clasificadores tipo árbol y técnicas Boosting.

La idea principal de combinar los modelos de tipo árbol y boosting es crear un modelo aditivo de manera que las funciones utilizadas para modelizar en cada paso son árboles de decisión. Se trata de un modelo aditivo donde la suma de clasificadores débiles permite obtener un clasificador más consistente. Además en boosting, los árboles dependen de los árboles óptimos de las etapas anteriores, de manera que todos contribuyen al modelo final.

Algunas de las ventajas que presentan es que heredan las características de los árboles, aumentando la capacidad predictiva de manera considerable y permitiendo obtener unas predicciones más reales. También presenta algunos inconvenientes, por ejemplo, los resultados obtenidos no son fácilmente interpretables, el modelo es más complejo y supone un mayor coste computacional ya que es necesario ajustas más parámetros que en Bagging o Random Forest.

2.3.1. Estructura general del algoritmo

Antes de definir la estructura del algoritmo de aprendizaje del Newton Boosting Tree, se va a introducir algunos conceptos de notación a partir de los capítulos teóricos desarrollados. En este apartado, se va a utilizar como funciones base $\phi_j(\mathbf{x})$, $\forall j \geq 1$, los árboles de decisión

$$\phi_j(\mathbf{x}) = \sum_{k=1}^{T_j} w_{kj} I(\mathbf{x} \in R_{kj}),$$

siendo $R_{kj} \subset \mathbb{R}^d$, $\forall k = 1, \ldots, T_j$ (d y T_j son el número de variables predictoras y nodos terminales en la iteracción j respectivamente) los rectángulos en los que se divide el conjunto de variables predictoras y w_{kj} el valor predicho para el dato o input $\mathbf{x_i} \in R_{kj}$, en nuestro caso será la etiqueta más común dentro del conjunto R_{kj} en la iteración j-ésima.

Los conjuntos R_{kj} se caracterizan por cumplir las siguientes propiedades :

$$\begin{cases} R_{1j} \cup R_{2j} \cup R_{3j} \ldots \cup R_{T_{\text{máx}}j} = \mathbb{X} \\ \\ R_{1j} \cap R_{2j} \cap R_{3j} \ldots \cap R_{T_{\text{máx}}j} = \emptyset \end{cases}$$

Se cumple que $\mathbf{x_i} \in R_{kj} \longleftrightarrow i \in I_{kj}$, donde I_{kj} es el conjunto de índices de los datos en R_{kj}, es decir,

$$I_{kj} = \{i \in 1, \ldots, n | \mathbf{x_i} \in R_{kj}\}.$$

La función de riesgo empírico dada por (2.6) cuando ϕ es un árbol de decisión, es decir,

$$\hat{R}_j(\phi_j) = \sum_{i=1}^{n} [\hat{g}_j(\mathbf{x}_i) \sum_{k=1}^{T_j} w_{kj} 1(\mathbf{x}_i \in \hat{R}_{kj}) + \frac{1}{2} \hat{\mathbb{H}}_j(\mathbf{x}_i) (\sum_{k=1}^{T_j} w_{kj} 1(\mathbf{x}_i \in \hat{R}_{kj}))^2]$$

$$= \sum_{k=1}^{T_j} \sum_{i \in I_{kj}} [\hat{g}_j(\mathbf{x}_i) w_{kj} + \tfrac{1}{2} \hat{\mathbb{H}}_j(\mathbf{x}_i) w_{kj}^2], \forall j = 1, \ldots, M.$$

De manera que denotando por $G_{kj} = \sum_{i \in I_{kj}} \hat{g}_j(\mathbf{x}_i)$ y $H_{kj} = \sum_{i \in I_{kj}} \hat{\mathbb{H}}_j(\mathbf{x}_i)$ se tiene que

$$\hat{R}_j(\phi_j) = \sum_{j=1}^{T_j} [G_{kj} w_{kj} + \frac{1}{2} H_{kj} w_{kj}^2]$$

En cada iteracción, la etiqueta donde se obtiene el riesgo empírico mínimo viene dado por

$$\hat{w}_{kj} = -\frac{G_{jk}}{H_{jk}}. \tag{2.7}$$

El riesgo empírico asociado a la función ϕ_j es igual a

$$\hat{R}_j(\phi_j) = \frac{1}{2} \sum_{k=1}^{T_j} \frac{G_{kj}^2}{H_{kj}}. \tag{2.8}$$

En cada división se busca minimizar (2.8), lo que equivale a maximizar la ganancia, que en el nodo k y la iteracción j-ésima viene dada por

$$Ganancia = \frac{1}{2} \left[\frac{G_{kjL}^2}{H_{kjL}} + \frac{G_{kjR}^2}{H_{kjR}} - \frac{G_{kj}^2}{H_{kj}} \right]$$

Se puede ver de manera más detallada en [2],(55-63).

Se va a establecer una enumeración del procedimiento algorítmico de XGBoost

1. Fijar la estructura del árbol y de las distintas regiones R_{kj} para calcular los pesos asociados a cada uno de los nodos terminales $\hat{w}_{kj}$.

2. Determinar la estructura del árbol y las regiones $\hat{R}_{kj}$, $\forall k = 1, \ldots, T$ a partir de los pesos $\hat{w}_{kj}$ calculados anteriormente.

3. Finalmente, una vez establecida la estructura de árbol, se estiman los pesos finales $\hat{w}_{kj}$ en cada uno de los nodos terminales.

Uno de lo métodos de Boosting con árboles más utilizados es XGBoost, se trata de la implementación estocástica del Gradient Boosting más general y que aporta mejores resultados. Entre los modelos más destacados se encuentran: Newton Boosting Tree y Gradient Boosting Tree. En el trabajo se va a centrar en el Newton Boosting Tree ya que es el utilizado a la hora de programar el modelo.

Newton Boosting Tree

A continuación, se va presentar el algoritmo de Newton Boosting para árboles, fue introducido por Chen y Guestrin (2014) [15].

Entrada:

Conjunto de datos

Función de pérdidas L.

Número máximo de iteraciones K.

Tasa de aprendizaje η.

Número de nodos terminales $T_{\text{máx}}$.

Inicialización : $\hat{f}^{(0)}(\mathbf{x}) = \omega_0 = \arg_\omega \min \sum_{i=1}^{n} L(y_i;\omega)$

para $j = 1,\ldots,K$ **hacer**

$\hat{g}_j(\mathbf{x_i}) = \left[\frac{\partial \hat{R}(f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)}\right]_{f(\mathbf{x})=\hat{f}^{(j-1)}(\mathbf{x})}$

$\hat{\mathbb{H}}_j(\mathbf{x_i}) = \left[\frac{\partial^2 \hat{R}(f(\mathbf{x_i}))}{\partial^2 f(\mathbf{x_i})}\right]_{f(\mathbf{x})=\hat{f}^{(j-1)}(\mathbf{x})}$

Encontrar la región $\{\hat{R}_{kj}\}_{k=1}^{T_{\text{máx}}}$ seleccionando las ramificaciones que maximizan la ganancia, $\max Ganancia = \max \frac{1}{2}\left[\frac{G_{kjL}^2}{H_{kjL}} + \frac{G_{kjR}^2}{H_{kjR}} - \frac{G_{kj}^2}{H_{kj}}\right]$

$\hat{w}_{kj} = -\frac{\sum_{i\in I_{kj}} \hat{g}_j}{\sum_{i\in I_{kj}} \hat{\mathbb{H}}_j} = -\frac{G_{kj}}{H_{kj}}$

$\hat{f}_j(\mathbf{x}) = \eta \sum_{k=1}^{T_j} \hat{w}_{kj} I(\mathbf{x} \in \hat{R}_{kj})$

$\hat{f}^{(j)}(\mathbf{x}) = \hat{f}^{(j-1)}(\mathbf{x}) + \hat{f}_j(\mathbf{x})$

fin

Resultado: $\hat{f}(\mathbf{x}) \equiv \hat{f}^{(K)}(\mathbf{x}) = \sum_{j=0}^{K} \hat{f}_j(\mathbf{x}) = \omega_0 + \eta \sum_{j=1}^{K} \sum_{k=1}^{T_j} \hat{w}_{kj} I(\mathbf{x} \in \hat{R}_{kj})$

Algoritmo 4: Algoritmo Newton Boosting Tree

Uno de los inconvenientes es que para saber el tamaño o número de observaciones de las submuestras es necesario estimar dos parámetros más en cada iteracción.

Algunas de las **ventajas** que presenta este algoritmo son

- Existen **paquetes específicos** en algunos de los software estadísticos más importantes, como Python y R. En el trabajo, únicamente se va a emplear el paquete xgboost [16] de R, está basado en la técnica del método de Newton Boosting con árboles[15].

- **Computación en paralelo**, esto permite realizar cálculos de manera más rápida y tratar conjuntos de datos con un número de observaciones elevado. Si no se indica nada, se va a utilizar todos los núcleos del ordenador o de la máquina virtual.

- Se trata de un método muy **flexible** debido a que admite numerosas funciones objetivo, además de las especificadas. También se puede elegir entre distintas métricas para ver la calidad del modelo estimado.

- Es un modelo **iterativo**, por lo tanto, en cada paso modifica las funciones base. Además, en cada iteracción se eligen aleatoriamente las variables y observaciones que definirán el modelo. De esta manera se evita el sobreajuste y la similitud de los modelos.

- **Alta capacidad predictiva**: Mejor que la de los árboles, esto hará que las predicciones sean más fiables.

- Admite **regularización**, basada en las penalizaciones de los pesos de las hojas, y se trata de una técnica utilizada para evitar el sobreajuste. Incluye una penalización relacionada con el número de nodos terminales.

- El algoritmo sabe estudiar los modelos con **valores perdidos**, si existe una tendencia entre los datos ausentes lo calcula y la indica.

- Permite obtener en cada iteracción el mejor conjunto de parámetros a través de la **validación cruzada**.

También presenta algunos **inconvenientes**,

- Hay muchos parámetros para estimar. Por lo tanto, el coste computacional es muy grande y el tiempo de calculo elevado. Además, si el modelo aprende lentamente (pequeños valores de η), el número de iteracciones y funciones a calcular es muy grande haciendo que la complejidad del modelo aumente. Esto hace que no sea sencillo usarlo y obtener un modelo óptimo.

- Es más difícil de interpretar al carecer de unas reglas de decisión expresadas de forma sencilla en función de las características de los datos.

En la siguiente sección se detallan algunos de los parámetros que es necesario optimizar en un modelo de XGBoost.

2.3.2. Parámetros a modelizar

Antes de introducir los parámetros que son necesarios estimar se va a hablar del sobreajuste. El sobreajuste ocurre cuando un modelo captura ciertos patrones que no son reproducibles, así pues el modelo no solo capta la tendencia, si no que también capta el ruido. Consecuentemente, este modelo no se podrá aplicar a nuevas muestras de datos y de hecho, las predicciones que se obtendrán con el conjunto de validación no serán buenas. Para evitarlo se va a considerar la regularización de los distintos parámetros que caracterizan XGBoost.

La regularización permite evitar el sobreajuste ya que una elección específica de los parámetros puede cambiar significativamente el ajuste del modelo. Por tanto, a la hora de elegir los distintos parámetros es necesario tener un criterio que permitan establecer el óptimo, por ejemplo, validación cruzada. La regularización viene determinada por la complejidad del modelo, de las restricciones, de la penalización y la aleatoriedad.

Algunas de las formas en las que se puede observar que un modelo está sobreajustado son

- La precisión del modelo en el conjunto de validación difiere mucho a la del entrenamiento.

- Las predicciones del modelo son muy variables, de manera que se obtienen unos resultados inestables.

La complejidad del modelo viene definida por el número de árboles y la profundidad máxima de cada uno. Por lo tanto para reducirla es necesario restringir los parámetros indicados. Otra de las variables que se podría optimizar es el mínimo número de observaciones en cada nodo terminal, esto está relacionado con él número de nodos terminales.

Otros de los elementos de regularización a considerar es la penalización de la función objetivo. Se escribe como:

$$\Omega(f) = \sum_{k=1}^{K} \left[\gamma T_k + \frac{1}{2}\lambda ||\mathbf{w_k}||_2^2 + \alpha ||\mathbf{w_k}||_1 \right], \tag{2.9}$$

donde K es el número máximo de iteraciones, T_k el número máximo de nodos terminales y $\mathbf{w_k}$ es el vector de pesos en los nodos terminales en la iteración k-ésima.

Por otra parte, γ es la penalización del número de nodos terminales en cada uno de los árboles que definen el modelo, λ y α la penalización del peso de los nodos terminales en norma cuadrática y en el espacio l^1, respectivamente.

Mejora el procedimiento de aprendizaje y se evita el sobreajuste eligiendo aleatoriamente sin reemplazamiento las observaciones y conjunto de variables que definen cada submuestra.

Debido a que se realizarán submuestras aleatorias tanto en las observaciones como en el conjunto de características es necesario calcular en cada división el porcentaje de observaciones y características ω_r y ω_c elegidas para estimar el mejor modelo, se trata de parámetros que oscilan entre 0 y 1. Si el valor de ω_r o/y ω_c es 1, la estimación del modelo se realiza en el conjunto de datos original o/y se emplean todas las características.

A continuación, se va a realizar un resumen de los parámetros más importantes a estimar para obtener un modelo XGBoost óptimo

- η : Es el parámetro asociado al aprendizaje que se encuentra entre 0 y 1. El valor por defecto es 0.3, pero para conjuntos de datos con mucho ruido es recomendable utilizar valores pequeños, por ejemplo, 0.1. Cuanto más próximo esté a 1, más rápido será su aprendizaje y mayor será el riesgo de sobreajuste.

- γ : Se encarga de controlar o prevenir el sobreajuste. Cuanto mayor es el valor de γ, mayor es la penalización de los grandes coeficientes que no mejoran el rendimiento del modelo. El valor por defecto es 0 (Aparece en (2.9)). Se puede encontrar una información más detallada en [2] (75-76).

- **nrounds** : Se corresponde con el número de iteracciones. Existe una relación inversa entre η y nrounds, de manera que pequeños valores de η, implican un número alto de

nrounds para que el algorimo pueda aprender lentamente. El número de iteracciones por defecto es 100.

- **max_depth** : Indica la máxima profundidad de cada árbol. En el caso de Boosting, los árboles se caracterizan por ser poco profundos. Además cuanto más profundo es el árbol más complejo es el modelo. El valor por defecto es 6.

- **subsample** y **colsample_bytree** : Submuestras de las observaciones o de las características que se cogen para realizar los modelos en cada iteracción de Boosting, se trata de un muestreo sin reemplazamiento. Se corresponden con los parámetros ω_c y ω_r que se han explicado anteriormente y que se utilizan para eliminar el sobreajuste en los modelos. En el caso de subsample, los valores se encuentran entre 0.5 y 0.8 . Mientras que en colsample_bytree se encuentran en el intervalo (0.5,0.9)

- λ y α son los parámetros que se utilizan para controlar el sobreajuste mediante las normas l^1 y l^2 de los pesos en los nodos terminales. Los valores por defecto son 0 y 1 respectivamente.

- **min_child_weigth** : Peso mínimo o número de observaciones en el caso de que todas tengan la misma importancia. Un valor pequeño del parámetro implicará un modelo más complejo y con mayor tendencia al sobreajuste. Mientras que un valor grande puede hacer que no se puedan observar las principales características que define el conjunto de datos.

- Existen otros parámetros menos usuales como : **tree_method**, **sketch_eps** que es el número de variables para obtener un algoritmo estable, **max_delta_step** y **scale_pos_weight** que se usan cuando la diferencia de observaciones entre las distintas clases es muy grande, es decir, cuando se trabaja con datos no balanceados [7].

El objective que se va a considerar es **binary:logistic** que se trata de un modelo que permita determinar las probabilidades de las dos clases que se definirán en el conjunto de datos : *creditos pagados* y *no pagados*. El parámetro **eval_metric** se utiliza para medir la precisión del modelo en un conjunto de validación. La función de error que se va a considerar en el trabajo es AUC(Area Under Curve).

Capítulo 3

Probabilidad de endeudamiento. Conceptos teóricos y definición.

Durante años, se han utilizado distintas técnicas para determinar con la mayor precisión el riesgo de impago en un crédito, esto permite a los bancos conseguir una gestión más eficiente de sus decisiones. Para este estudio, es necesario definir los términos económicos correctamente y entender el funcionamiento de cada uno de ellos.

En la historia del cálculo del riesgo de incumplimiento crediticio ha jugado un papel importante los Acuerdos de Basilea, el primero de ellos se remonta a 1988. En Basilea II, una parte importante de las modificaciones se centran en el cálculo de incumplimiento crediticio, en él establece que un crédito está determinado por las características económicas, personales, la vida crediticia de la persona y la situación económica del país. Por lo tanto, los datos son una parte fundamental a la hora de determinar si se le concede o no un crédito a una persona. Consecuentemente, es necesario establecer una estructura que sea capaz de soportar y almacenar todos los datos, siendo importante tener a personas capacitadas para analizar y realizar una modelización estadística de los datos.

El algoritmo XGBoost se va a utilizar en un contexto bancario con el objetivo de detectar el riesgo de impago. Además, se va a realizar una comparación con algunos de los métodos tradicionales, como la regresión logística.

Se puede obtener una información más detallada de los conceptos y definiciones que se presentan en la siguiente sección en [18].

3.1. Definiciones

Crédito: Es un contrato por el cual una entidad financiera da una cierta cantidad de dinero a una persona. El prestatario debe devolver esa cantidad de dinero más unos intereses, estos varían en función del número de plazos para realizar un pago y de las condiciones económicas y personales. Existen algunas diferencias respecto del préstamo, ya que los créditos pueden ser de interés fijo o variable en función de las condiciones preferidas por el prestatario.

El **riesgo** viene definido por la incertidumbre generada por la evolución de un activo, esta relacionado con la posibilidad de que una inversión tenga un rendimiento diferente al que se había planteado. El **riesgo de crédito** es un tipo de riesgo generado por la incertidumbre de que una persona no haga frente a la totalidad de un pago, bien sea en el vencimiento o en instantes

posteriores, entendido como el periodo de tiempo que transcurre desde que una persona pide el crédito hasta que finaliza el pago, es decir, su longitud o profundidad. Existen distintos tipos de riesgos de crédito y varían en función de los tipos pedidos por los prestatarios. También existen otros tipos de crédito económicos, por ejemplo, el de mercado, liquidez e interés.

Para las entidades financieras es muy importante el cálculo del riesgo crediticio, ya que dar un crédito y que finalmente no se devuelva puede implicar un pérdida de dinero grande. Para la valoración del riesgo de incumplimiento crediticio se considerará la **probabilidad de endeudamiento** (que denotaremos por PD por sus siglas en inglés), indica las posibilidades que hay de que una persona no devuelva el crédito. Antes de la aceptación y cesión de un crédito es necesario calcular su probabilidad de incumplimiento, en función del valor obtenido se modificarán los intereses generados por el dinero recibido. Está estrictamente relacionado con el riesgo y depende no sólo de los características monetarias de la persona, si no de las condiciones económicas del país en ese momento. Existen distintos ratios en función si el crédito es para particulares o empresas.

- Para consumidores se tiene la calificación FICO.
- En las empresas, la probabilidad de endeudamiento está implicita en la calificación crediticia.

En el trabajo se va a calcular la probabilidad de endeudamiento a partir de la modelización estadística del conjunto de datos definidos por el histórico de la persona. Existen otros indicadores adyacentes, por ejemplo, **la pérdida esperada** (LGD) en caso de que el particular no cumpla con sus obligaciones y **exposición en caso de incumplimiento** (EAD), importe de deuda pendiente en el momento de incumplimiento del cliente. Se trata de dos cantidades asociadas al riesgo para determinar las posibles pérdidas relacionadas con el incumplimiento. La **perdida esperada** viene definida por el producto

$$PE = PDxLGDxEAD.$$

3.2. Acuerdos de Basilea

Los distintos conceptos se encuentran regulados en los acuerdos de Basilea, son recomendaciones que han sido elaboradas por el Comité de Basilea que se encargan de establecer una garantías mínimas sobre los créditos. Se encarga de fijar los niveles de riesgos que son asumibles para los distintos bancos basandose en el capital mínimo que deben disponer cada una de las entidades, teniendo en cuenta el número de créditos que conceden y el riesgo ó perdida esperada por cada operación cuando el crédito no se pague.

Los acuerdos de Basilea se han ido modificando con el objetivo de reducir lo máximo posible el número de personas que no pagan los créditos y garantizar la capacidad de respuesta ante el cálculo del riesgo crediticio por medio de la probabilidad de endeudamiento.

- **Acuerdo de Basilea I** (1988) : Se establecieron algunos de los principios básicos para la actividad bancaria. El capital de cualquier entidad tiene que ser superior que el dinero

necesario para hacer frente a riesgos de créditos, de mercado y de tipo de cambio por la devaluación de las monedas y la pérdida asociada. En este acuerdo se establecía que el capital mínimo de la entidad debería constituir el 8 % del total de los riesgos en todo momento.

- **Acuerdo de Basilea II** (2004) : Esta recomendación tiene como objetivo establecer una base más solida para el capital, teniendo en cuenta el dinero para hacer frente al riesgo. Se acepta que cada una de las entidades bancarias realice sus modelos internos para obtener el cálculo del riesgo crediticio a partir de la probabilidad de endeudamiento o distintos indicadores siempre que dicho modelo sea aceptado por el supervisor. Esto hace que sea necesario realizar una clasificación de los créditos en función de la dificultad que tiene una persona para hacer frente a las obligaciones. En todos los modelos internos se tiene en cuenta las bases legales de las metodologías internas, la revisión de supervisión y las distintas características de la situación económica del país.

- **Acuerdo de Basilea III** (2010) : Surgío a partir de la gran crisis económica. Se establecieron distintas recomendaciones al Acuerdo de Basilea II como endurecer los criterio de manera que se disminuya las pérdidas, todo ello se consigue modificando los criterios de cálculo y los umbrales a la hora de determinar si un crédito va a ser Default o Fully Paid. También se estableció la necesidad de tener un mínima cantidad de dinero para hacer frente a cualquier cambio de ciclo económico, asociado por una economía circular. Otra técnica utilizada es utilizar el ratio de apalancamiento como medida complementaria a los distintos indicadores establecidos, como el ratio de solvencia.

En nuestro caso se va a considerar de manera más detallada el Acuerdo de Basilea II. Servirá para determinar algunos de los conceptos más importantes que se van a tratar a lo largo de la parte final de la memoria. En él se busca crear unos ideas sólidas que permitan gestionar mejor los riesgos y el capital asociado a cada entidad bancaria. Los supervisores tienen un papel muy importante, son los encargados de aceptar o rechazar los modelos estimados para calcular el riesgo de incumplimiento crediticio con el objetivo de regular el capital. De esta manera se pretenden conseguir unos bancos más solidos y fiables.

Características y condiciones de Basilea II

Basilea II ha implicado modificaciones jurídicas y de regularización. Antes de aplicar Basilea II es necesario realizar un estudio de los ingresos y los gastos para compararlos con el resto de entidades nacionales, para ello se tiene que observar la efectividad de los servicios centrales ya que son los encargados de la seguridad del sistema bancario. Con este estudio, los supervisores serán capaces de determinar si la entidad es capaz de aceptar Basilea II a corto o largo plazo.

Primeramente hay que determinar los bancos que permanecerán sujetos al nuevo marco definido por este acuerdo. La aceptación de estas, supondrá que el estudio del cálculo asociado al riesgo crediticio se realizará por medio de métodos iterativos cualitativos o cuantitativos.

Cada país cuenta con numerosas entidades bancarias, cada una con un tamaño, nivel de desarrollo y capital diferentes. Por lo tanto, cada una tendrá una metodología diferente para el calculo del riesgo, con distintos niveles de complejidad y consolidación en las técnicas. Debido a que el número de técnicas para el cálculo es muy grande, los requisitos mínimos de capital

en cada una de las entidades son diferentes para que se le acepte un crédito o préstamo a un particular o una empresa. El supervisor tiene que trabajar para obtener una estrategia particular que permita conseguir los objetivos propuestos, siendo consciente del capital de las entidades bancarias. Además tiene que tener en cuenta si se trata de un banco nacional o internacional, la complejidad de los modelos bancarios y la cualificación del supervisor para poder revisar métodos más complejos y especializados. También son importantes las revisiones estrátegicas para alcanzar los objetivos del capital mínimo por medio de nuevos métodos que permitan determinar los riesgos asociados. Se pueden encontrar más detalladamente en [8] (páginas 11-12).

Bancos que pueden adoptar el acuerdo de Basilea II

A la hora de determinar los bancos que tienen las características para implantar el acuerdo de Basilea II, se tiene en cuenta el tamaño, la complejidad de las operaciones realizadas, la presencia e interacción internacional, la capacidad para gestionar el riesgo y el estudio previo para observar si un modelo más complejo aumentará el capital inicial.

Se establecerán distintos umbrales para que las entidades puedan implantar el acuerdo, estos tendrán que estar previamente aceptados por los supervisores para garantizar el buen funcionamiento.

Basilea II establece que además de los requerimientos mínimos, el supervisor tiene que realizar un examen riguroso centrado en el segundo y tercer pilar que se explicarán a continuación.

Pilares fundamentales de Basilea II

En el acuerdo de Basilea I no se tenía en cuenta la capacidad de una persona o empresa para hacer frente a los créditos concedidos. Tampoco se estudiaba el dinero ni el tiempo de recuperación que podría suponer el incumplimiento crediticio. A partir de esto, surgen los tres pilares en los que se basa el acuerdo de Basilea.

Pilar I: Disponer de unos requisitos mínimos de capital

Es necesario evaluar el riesgo crediticio clasificando a cada particular o empresa dentro de grupos , esta es una de las grandes diferencias respecto de Basilea I. Exige que el coeficiente de recursos propios en los bancos sea superior al 8 %, además añade algunos requisitos más en el capital asociados al riesgo. En este pilar se tiene en cuenta la probabilidad de inclumplimiento (PD) y la perdida esperada si el prestatario no cumple con sus obligaciones (que se denotará por LGD por sus siglas en inglés) que se calcularán a partir de diferentes métodos y se compararán con las calificaciones de los riesgos emitidas por las distintas empresas especializadas. Otro de los factores que se estudia es la exposición al riesgo (que se denotará por EAD por sus siglas en inglés), entendido como el dinero que se perderá cuando el cliente no puede hacerse cargo de la deuda (Una vez pasados 60 días del momento de pago, este valor varía en función de la entidad bancaria).

Por tanto, se trata de determinar el riesgo crediticio a través de distintas técnicas de estadística avanzada. Por su parte, los bancos son responsables de establecer y mejorar los sistemas de gestión del riesgo, pero los supervisores juegan un papel muy importante a la hora de indicar los métodos más sofisticados.

Para definir y crear nuevos métodos es necesario la recopilación de los datos para determinar que características están asociadas para que una personas finalmente te pague el crédito. Estos datos deben cogerse de una fuente fiable, permitiendo realizar pruebas para determinar si los métodos son buenos y presentan diferencias significativas respecto de las técnicas más usuales. Además, será necesario realizar una limpieza de los datos para obtener una información fiable y esta debe estar aceptada dentro de los marcos de Basilea II.

Estos cambios implican invertir una gran cantidad de dinero, sería necesario modificar las tecnlogías utilizadas. Además, se tiene que contar con un almacen de datos para tener un histórico que permita obtener las características más influyentes de los créditos no pagados y guardar las predicciones realizadas una vez el método esté consolidado.

Pilar II: Supervisar la gestión de los fondos propios

Es necesario que en cada país exista un banco encargado de que las entidades mantengan un capital suficiente para hacer frente a los riesgos acumulados a través de la estrategia establecida. En el caso de España, es el Banco de España. Es necesario que los bancos tengan la información de los créditos generados en los últimos 5 o 7 años para que sirvan como test para modelos internos.

Los bancos deben contar con un modelo que permita identificar los riesgos de manera que asigne el capital objetiva y sistemáticamente. El segundo Pilar abarca los distintos riesgos que no se especifican en el Pilar I (interés, liquidez, etc). También se recogen algunos factores como el ciclo económico, son muy importantes a la hora de determinar un modelo, debido a que la técnica habitual puede que no funcione para estos casos. (Habría que realizar un nuevo estudio si se modifican mucho las condiciones). Se trata de un proceso para la evaluación de la suficiencia de capital.

Los métodos usados para el cálculo del coeficiente de solvencia varían en función del tamaño de la entidad financiera, cuanto más grande sea, más complejo será el modelo definido. Los supervisores se encargarán de realizar una evaluación de los métodos utilizados para conseguir la suficiencia del capital. En la evaluación es necesario que los bancos hayan incorporado todos los riesgos y los casos de uso de los modelos y que el capital inicial sea correcto. Los criterios de evaluación de los supervisores tienen que ser claros y transparentes, los bancos tienen que tener conocimiento de ellos y de los posibles cambios relacionados con las supervisiones realizadas.

Los supervisores tiene que asegurarse que los bancos operen por encima de los coeficientes mínimos de capital regulador y deberán ser capaces de obligarles a que mantengan un capital superior al mínimo. Además, tienen que asegurar que todos los riesgos que no se encuentran en el primer Pilar se encuentren cubiertos. Existen distintas formas de cumplir este segundo pilar: algunas generales, haciendo que todo banco tenga un capital 8 % mayor al mínimo y otras particulares, estableciendo un coeficiente específico para cada banco calculado por los riesgos del banco y su manera de gestionarlos o por el objetivo propuesto.

Cuando se produce un descenso del capital, el supervisor tiene que actuar antes de que disminuya más de los mínimos establecidos, aunque es conveniente que el banco realice una supervisión cuando no mantiene el nivel esperado. Si el capital se aproximar al mínimo, los supervisores tienen que aclarar todas las medidas adoptadas, asegurandose si la caída es consecuencia de una tendencia negativa o está relacionada con causas subyacentes.

Pilar III: Disciplina del mercado

En este pilar se busca que los bancos sean capaces de explicar la información de manera clara para que los agentes del mercado puedan comprenderla. Es necesario que los bancos sean capaces de interpretar la información obtenida. Existen distintas formas de publicarla, por ejemplo, a través de páginas en internet o mediante informes.

Cada supervisor se encargará de distintos aspectos : tamaño del sistema bancario, si los sistemas son sotisficados, el desarrollo y solidez de las normas contables, la importancia de la función de supervisión y las distintas opciones que ofrece Basilea II para que las entidades bancarias se ajusten a ella. El tercer Pilar está definido a partir de las características anteriores. Además, se tienen que encargar de la observación y la identificación de los distintos problemas asociados a ellas, publicándolas de manera clara y precisa a las distintas entidades.

La aceptación de Basilea II implicará cambios en los sistemas jurídicos y reguladores. Se tendrá que realizar un estudio previo de los cambios, procedimientos y tiempos necesario,etc. Además, para la supervisión se requieres personas altamente cualificadas.

Capítulo 4

Diferentes técnicas de estadística aplicada al problema de probability default. Lending Club dataset.

4.1. Estudio de los datos

Para este estudio se ha contado con un conjunto de datos de creditos "loan"que se ha obtenido de la página abierta

https://www.lendingclub.com/info/download-data.action.

Se van a considerar los datos de los créditos que han sido aceptados durante los años 2014, 2015 y los dos primeros cuatrimestres de 2016. El conjunto consta de 146 variables y 888.473 observaciones. Se va a realizar un estudio detallado del conjunto de datos para obtener una depuración óptima y conseguir unos modelos estables. En este caso se trata de predecir el estado final del crédito a través de algunas características, por ejemplo, personales y profesionales, vida prestamista, entre otras. En el conjunto de datos cada fila representa un préstamo.

Primeramente, se va a aplicar ingeniería de características y posteriormente se va a explicar una a una todas las variables que definirán el modelo final. Los pasos realizados para la depuración de la base de datos son

- **Eliminación de las observaciones duplicadas**. Hay 7 filas que se encuentran repetidas totalmente.

- Se van a eliminar todas las variables que tengan un cantidad de **valores omitidos** mayor al 40 % de los datos. En este caso se eliminan 58 variables.

- Como se quiere calcular la variable *loan_status* se van a eliminar todos las observaciones en las que está omitido ya que no se tiene conocimiento del estado del crédito. Solamente hay un préstamo del cual no se conoce el estado, será un posible error cuando introdujeron los datos.

- **Eliminación de variables con características constantes**. Esto quiere decir que se van a eliminar las variables que sólo toman un único valor, ya que la varianza es cero y no nos dan ninguna información. Se trata de una variable.

- Se van a eliminar las variables numéricas que no son constantes pero que su variabilidad es próxima a 0, **variables near-zero**. El umbral que se ha establecido es 0,05 y se eliminan 5 variables del conjunto de datos. En el caso de las variables categóricas se ha realizado un estudio detallado eliminando las variables que tienen pocas clases y se encuentran desequilibradas, es decir, variables con dos clases muy desproporcionadas.

- **Eliminación de la variable correlacionadas**: Se va a realizar una separación del conjunto de datos en variables numéricas y categóricas y se calculará la matriz de correlaciones. El corte para eliminar variables es de 0,75 y se corresponden con 23.

- **Eliminación de la variable correlacionadas**: Las características *grado* y *subgrado* al igual que *interés* con *longitud del crédito* y *subgrado*.

- Las **variables que filtran datos de prestamos pasados** se van a eliminar, ya que no nos aportan información para los préstamos actuales y además pueden sesgar los datos. Se van a eliminar 8 variables.

- Existen **variables cuya variabilidad es grande** debido al elevado número de clases, por lo tanto, recodificar esta variable es bastante complicado y además no aportan mucho valor. Es el caso de la variable *title* que indica el título de la persona. Además no aporta gran información, ya que estudiar una carrera no quiere decir que tu vida laboral sea de lo mismo. Esto mismo ocurre con el tipo de aplicación ya que hay 1998 valores distintos.

- Se van a realizar algunas operaciones entre los datos de tipo fecha, por ejemplo, una variable que mida la diferencia de tiempo entre el primer y el último crédito.

- Las variables que no aportan información a la hora de determinar el estado final de un crédito se van a eliminar, por ejemplo, el codigo postal.

- A los valores numéricos omitidos se les va a asignar la media, mientras que en los categóricos se les va a crear una etiqueta nueva que será SinInfo.

Conjunto de datos final

Una vez realizada la ingeniería de características se tiene un conjunto de 888.465 observaciones y 41 variables. A continuación se va a realizar un diccionario explicando la variable predictora ”loan_status” y las variables independientes que definirán el modelo que la va a estimar.

Variable dependiente

Loan_status : Estado del crédito.

Hay 7 estados de préstamo : Con carga desactivada, actual, default, totalmente pagado, en periodo de gracia, tardía (16-30 días) y tardía (31-120 días) . En este caso, interesan únicamente

los créditos cuyo estado es totalmente pagado, default, carga desactivada y tardía(31-120 días) ya que en el resto de casos se va a obtener una probabilidad pero no se tiene la conclusión del crédito. Se va a recodificar en dos clases

1. **Default** : Carga desactivada, default y tardía(31-120 días).

2. **Fully Paid** : Totalmente pagado.

El resto de clases se van a eliminar del conjunto de datos, se trata de las clases minoritarias que representan menos del 10 %.

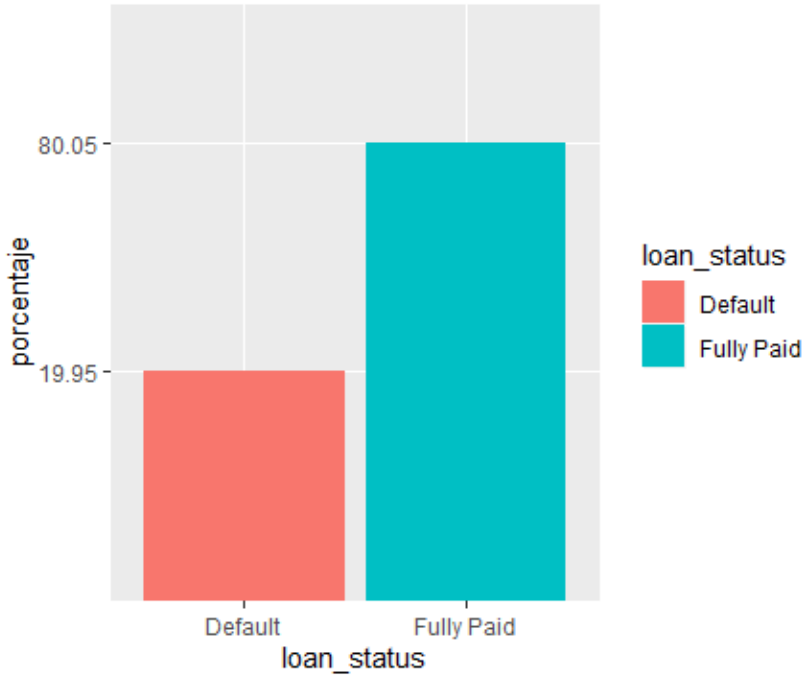

Figura 4.1: Histograma de la variable Loan_status

Se observa una relación clara de 1:4 entre las dos clases. Se va a ver la distribución de los estados crediticios en función del año.

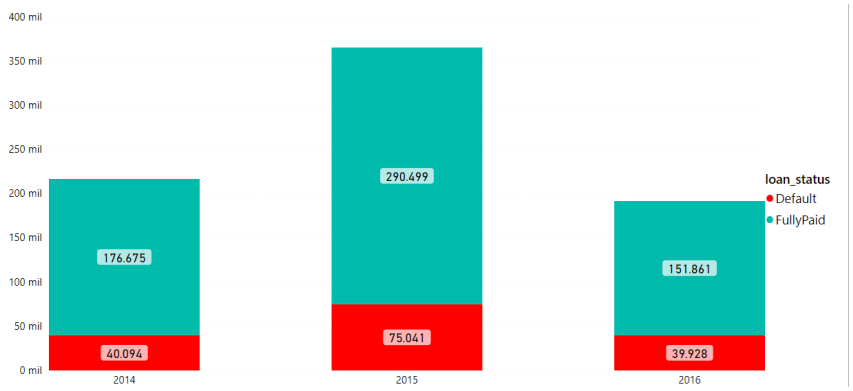

Figura 4.2: Distribución de los estados finales de los créditos en función de los años estudiados.

Variables independientes

• Variables categóricas

Term : Número de pagos del préstamo, pueden ser créditos de 36 o 60 meses.

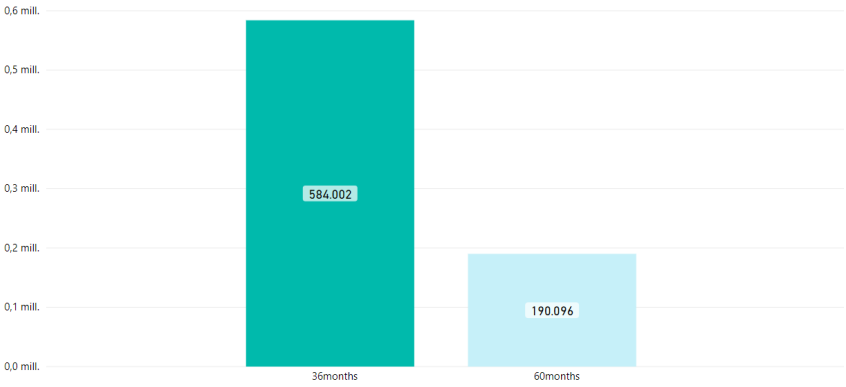

Figura 4.3: Distribución de los créditos en función de la duración.

Subgrade : Subgrado del préstamo. Se tienen 7 grados y cada grado está dividido en 5 subgrados, por lo tanto, existen 35 subgrados como se puede observar en

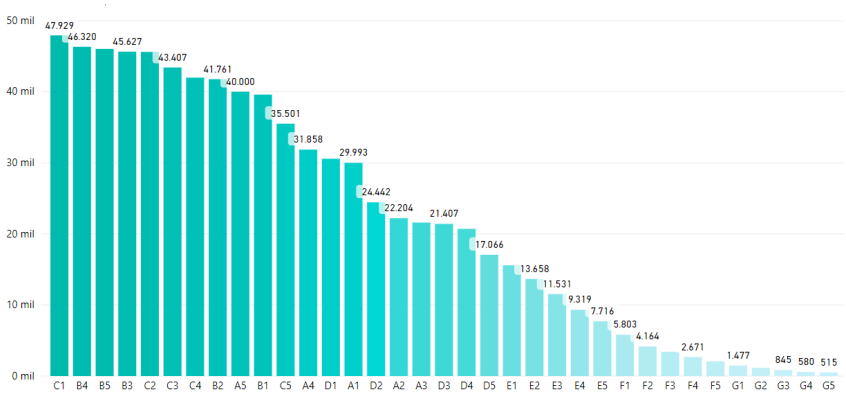

Figura 4.4: Distribución de los créditos en función de los subgrados asignados a los tomadores a partir de los datos recogidos.

Emp_length: Años trabajados de la persona que se le ha concedido el crédito. Se ha recodificado la variable en 4 clases : <=3Years, Between4y7Years, MoreorEqual8Years y WithoutDesk.

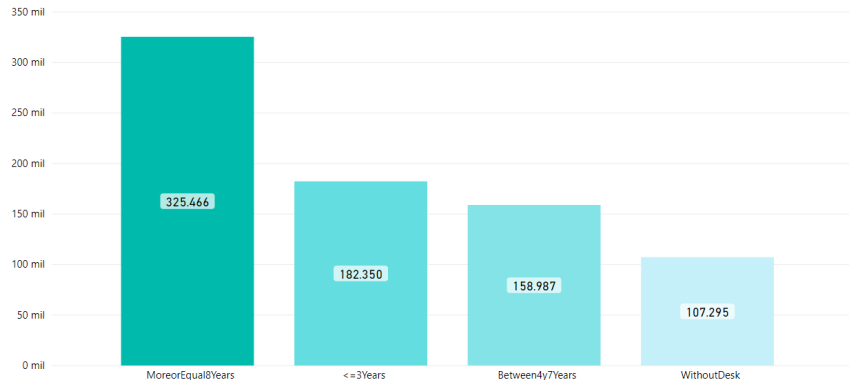

Figura 4.5: Distribución de los créditos en función de los años trabajados.

Home_ownership: El estado de propiedad de la vivienda proporcionado por el prestatario durante el registro. Los valores de la variable son: ANY, MORTGAGE, OWN y RENT. Sólamente hay 3 observaciones en las que esta variable es ANY, por lo tanto, no se van a considerar ya que está muy desequilibradas.

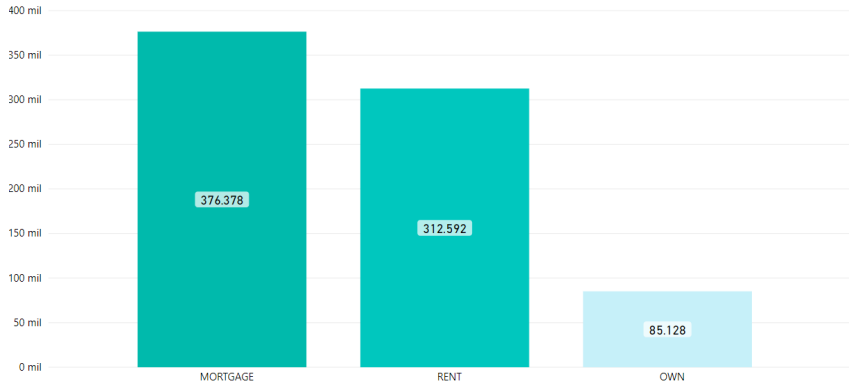

Figura 4.6: Distribución de los créditos en función del estado de propiedad de la vivienda en el que encuentra el prestatario.

Verification_status:Indica si los datos han sido o no verificados por alguna fuente externa. Existen dos tipos : Not verified, source verified y verified.

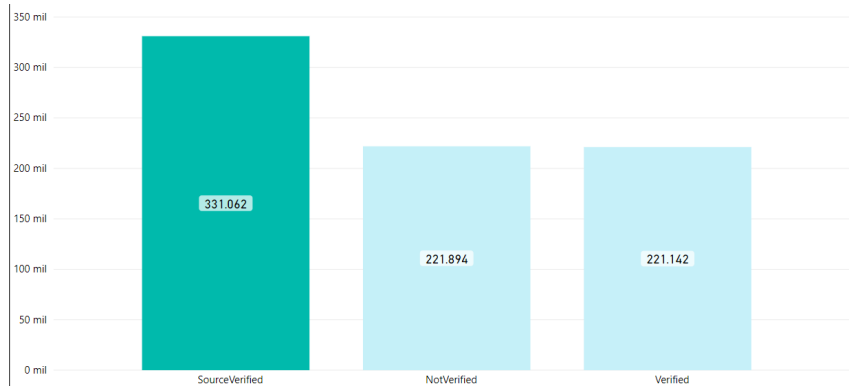

Figura 4.7: Distribución de los créditos en función de la veracidad de los datos.

Purpose: Indica cual es el propósito para el que se ha pedido el crédito. Existen 14 clases, se puede observar que la mayoritaria es la de debt_consolidation.

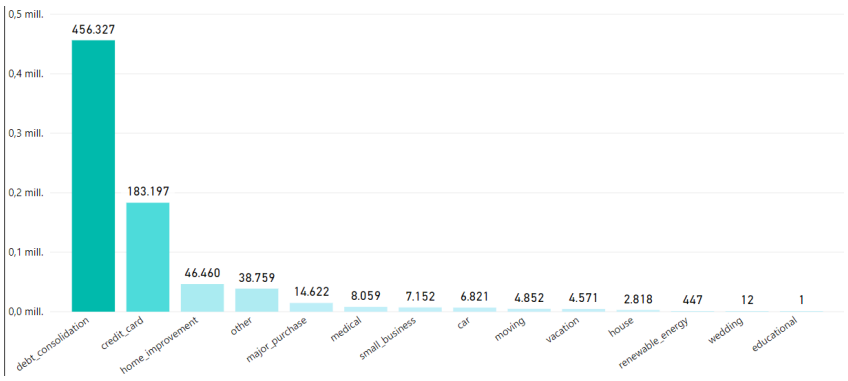

Figura 4.8: Distribución de los créditos en función de la finalidad para la que se ha pedido el crédito.

Initial_list_status: Estado inicial del préstamo. Existen dos valores f y w que se distribuyen de la siguiente forma

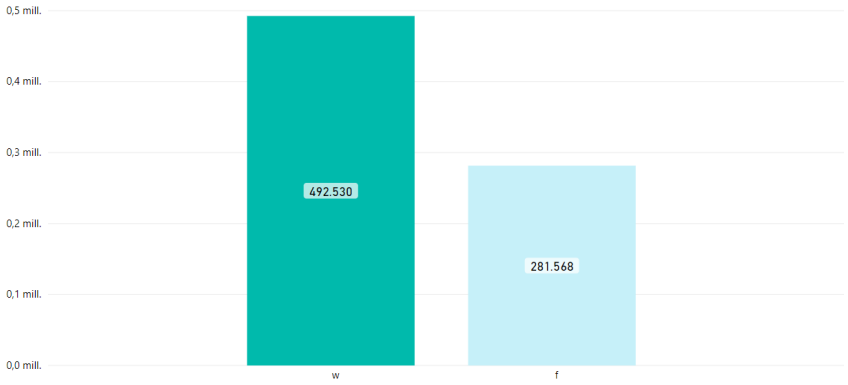

Figura 4.9: Distribución de los créditos en función del estado inicial en el que se encontraba.

• Variables numéricas

Installment: El pago mensual que tiene que depositar el tomador al prestamista.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
14.01  255.60  379.76  442.11  586.49 1569.11
```

Annual_inc: Salario anual del prestatario en el momento en el que se registra para pedir un crédito. Tiene una distribución que no es muy simétrica como se puede observar a continuación

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0    45760    65000   76392   90000 9573072
```

Dti: Relación entre el total de los pagos mensuales de la deuda del prestatario y los ingresos, excluyendo como gastos la hipoteca y el prestamo solicitado. Se expresa en porcentajes, consecuentemente los valores que no se encuentren en el intervalo [0,100] son que se han tomado de manera errónea. La variable se distribuye como

- Si dti > 100 % implica que el monto a pagar por mes es superior a la ganancia mensual que se tiene.
- Si dti < 0 % se trata de un dato erróneo, ya que se trata de personas que tienen deudas al registrarse para solicitar el crédito.
- Si dti = 0 % implica que el gasto mensual a aportar es 0, por lo tanto, no se le da ningún crédito.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-1.00   12.24   18.13   18.74   24.75  999.00
```

Únicamente existen 41 valores de manera que se van a eliminar ya que son datos atípicos.

Delinq_2yrs: El número de pagos con retraso de más de 30 días del prestatario durante los dos últimos años. La distribución viene dada por

```
0      1      2      3      4      5      6
638784 109032  32768  12216   5537   2831   1568
7      8      9     10     11     12     13
883    523    320    225    127    101     74
14     15     16     17     18     19     20
57     30     23     12     10      9      4
21     22     26     27     30     39
4      4      2      1      1      1
```

de manera que el resumen numérico es

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.000   0.000   0.000   0.346   0.000  39.000
```

Se puede observar que en la mayoría de los prestatarios el número de delinq_2yrs es 0, se trata de una variable muy apuntalada a la izquierda. Debido a que existe mucha variabilidad, se va a eliminar las observaciones en las que esta variable es mayor que 19.

Inq_last_6mths: Número de consultas en los últimos 6 meses, se excluyen las consultas por automóviles e hipotecas. La distribución es de la forma

```
     0      1      2      3      4      5      6
470455 217093  77135  27827   8922   3160    538
```

de manera que el resumen numérico es igual a

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000  0.0000  0.0000  0.6329  1.0000  6.0000
```

Pub_rec: Número de registros públicos despectivos. La distribución de esta variable es

```
0      1      2      3      4      5      6
658300 119957  17532   5446   1919    939    472
7      8      9     10     11     12     13
214    129     61     47     33     21     14
14     15     16     17     18     19     20
2      8      6      2      5      3      2
21     22     23     28     31     34     37
5      2      1      1      1      1      1
40     46     47     49     63     86
1      1      1      1      1      1
```

de manera que el resumen numérico es

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000  0.0000  0.0000  0.2385  0.0000 86.0000
```

Se puede destacar que la mayoría de personas se encuentran en el intervalo [0,13], el resto representa un 0.006 % de las observaciones totales, por lo tanto, se van a eliminar.

Revol_bal: Crédito total de balance rotativo. Un ejemplo es la tarjeta de crédito. El intervalo de esta variable es

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0    6175   11450   16938   20431 2904836
```

Avg_cur_bal: Saldo promedio actual de todas las cuentas. La distribución de esta variable se puede observar a continuación

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0    3106    7180   13116   18070  555925
```

Bc_open_to_buy: Cantidad total de dinero disponible para la compra con tarjetas bancarias rotativas.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0    1371    4375    9537   11381  559912
```

Delinq_amnt: Suma de la cantidad de dinero atrasada en cada una de las cuentas en las que está en mora.

```
Min.   1st Qu.    Median      Mean   3rd Qu.
0.00      0.00      0.00     16.06  0.00
Max.
130778.00
```

Mo_sin_old_il_acct: Años desde que se abrió la cuenta bancaria a plazos más antigua.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.00    8.50   10.83   10.59   12.67   60.33
```

Mo_sin_old_rev_tl_op: Años desde la apertura de la cuenta rotatoria más.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.250   9.917  13.917  15.395  19.500  71.000
```

Mo_sin_rcnt_rev_tl_op: Años desde la apertura de la cuenta rotatoria más reciente.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000  0.3333  0.6667  1.0865  1.3333 36.5000
```

Mo_sin_rcnt_tl: Años desde la apertura de la cuenta más reciente. El resumen numérico es

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000  0.2500  0.4167  0.6546  0.8333 24.0833
```

Mort_acc: Número de cuentas asociadas a hipotecas. El porcentaje de prestatarios en función del número de cuentas hipotecarias es igual a

```
0     1     2     3     4     5     6     7
40.09 17.00 14.48 10.91  7.55  4.62  2.62  1.33
8     9    10    11    12    13    14    15
0.66  0.33  0.17  0.10  0.05  0.03  0.02  0.01
16    17    18    19    20    21    22    23
0.01  0.01  0.00  0.00  0.00  0.00  0.00  0.00
24    25    26    27    28    29    30    31
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
32    34    35    36    37    47    51
0.00  0.00  0.00  0.00  0.00  0.00  0.00
```

de manera que el resumen numérico viene dado por

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.000   0.000   1.000   1.701   3.000  51.000
```

Se trata de una variable muy dispersas, cuando la variable es mayor que 7 el número de apariciones es menor que 1 %. Se van a considerar como datos atípicos.

Mths_since_recent_bc: Años desde la apertura de la cuenta bancaria más reciente. El resumen numérico es

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.000   0.500   1.167   1.992   2.333  53.250
```

Mths_since_recent_inq: Años desde la última investigación.El resumen numérico es

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000  0.1667  0.4167  0.5505  0.7500  2.0833
```

Num_accts_ever_120_pd: Número de cuentas en las que presenta 120 o más días de atraso en los pagos.El porcentaje de prestatarios en función del número de cuentas con demora superior o igual a 120 es

```
0     1     2     3     4     5     6     7
75.62 12.95  5.31  2.42  1.41  0.84  0.52  0.33
8     9    10    11    12    13    14    15
0.20  0.13  0.09  0.05  0.04  0.02  0.02  0.01
16    17    18    19    20    21    22    23
0.01  0.01  0.00  0.00  0.00  0.00  0.00  0.00
24    25    26    27    28    29    30    31
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
32    35    38
0.00  0.00  0.00
```

de manera que el resumen numérico viene dado por

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000  0.0000  0.0000  0.5211  0.0000 38.0000
```

Se van a considerar como datos atípicos las observaciones con valores mayores a 4.

Num_actv_bc_tl: Número de cuentas con tarjeta bancaria actualmente activas.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0     2.0     3.0     3.7     5.0    35.0
```

Num_bc_tl: Número de cuentas bancarias. El summary viene dado por

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.00    5.00    7.00    8.09   10.00   70.00
```

Num_il_tl: Número de cuentas a plazos.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.000   3.000   7.000   8.498  11.000 159.000
```

Num_tl_30dpd: Número de cuentas actualmente con 30 días de atraso (actualizado en los últimos 2 meses).

```
Min.  1st Qu.   Median     Mean  3rd Qu.
0.000000 0.000000 0.000000 0.004084 0.000000
Max.
4.000000
```

Num_tl_90g_dpd_24m: Número de cuentas con 90 o más días de atraso en los últimos 24 meses.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.000   0.000   0.000   0.078   0.000  18.000
```

Num_tl_op_past_12m: Número de cuentas abiertas en los últimos 12 meses. Los porcentajes de número de cuentas abiertas por los prestatarios son

```
0     1     2     3     4     5     6     7
16.80 24.76 22.72 16.65  9.83  4.58  2.15  1.15
8     9    10    11    12    13    14    15
0.61  0.33  0.17  0.10  0.06  0.04  0.02  0.01
16    17    18    19    20    21    22    23
0.01  0.01  0.00  0.00  0.00  0.00  0.00  0.00
24    25    26    27    28    30    32
0.00  0.00  0.00  0.00  0.00  0.00  0.00
```

de manera que el resumen numérico es

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.000   1.000   2.000   2.161   3.000  32.000
```

Cuando la variable toma valores superiores a 11, el porcentaje es bajo, en torno a un 0.17 %. Por lo tanto, se trata de una distribución muy dispersa, de manera que dichos valores nos van a generar más ruido que la información que nos aporta.

Pct_tl_nvr_dlq: Porcentaje de operaciones realizas de manera corrrecta, sin ningún periodo de demora. El resumen numérico es

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.00   91.70   97.90   94.51  100.00  100.00
```

Percent_bc_gt_75:Porcentaje de todas las cuentas bancarias > 75 % del límite.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.00   16.70   50.00   47.09   75.00  100.00
```

Pub_rec_bankruptcies: Número de veces en estado de bancarrota. La distribución de la variable es

```
    0      1      2      3      4      5      6
674991  90927   5398   1114    274    100     31
7      8      9     11     12
13      6      2      1      1
```

cuyo resumen numérico es

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000  0.0000  0.0000  0.1385  0.0000 12.0000
```

Tax_liens: Número de gravámenes fiscales. La distribución es igual a

```
     0      1      2      3      4      5      6
742921  20317   5587   2042    914    494    269
7      8      9     10     11     12     13
125     71     46     37     23     12      6
14
1
```

de manera que el resumen numérico es igual a

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000  0.0000  0.0000  0.0622  0.0000 14.0000
```

Se puede ver que los datos se encuentran muy dispersos, de manera que se van a eliminar las observaciones donde tax_liens>=13.

Total_il_high_credit_limit: Límite superior del crédito.

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0   14868   31625   41928   56236 2101913
```

Issue_d: El mes en el que se fundo el crédito. Sólamente se van a considerar dos años y medio que se corresponden con 2014, 2015 y mitad de 2016. El número de créditos en estos años se distribuye

```
2014   2015   2016
216612 364886 191367
```

Time_first_credit: Años transcurridos desde el primer crédito hasta el actual. El resumen numérico es

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-54.95   11.34   14.93   16.27   20.27   47.45
```

No es posible que el crédito más reciente sea más antiguo que el primero que se ha pedido. Por lo tanto, se van a eliminar los valores en los que es menor que 0. Se trata únicamente de 41 observaciones que se han tomado de manera errónea.

4.2. Estimación del modelo

La métrica utilizada en el trabajo para conseguir el mejor modelo es el AUC (Area under curve), existen otros indicadores como el error, rmse, logloss, aucpr, merror, etc.

Una vez realizada la explicación detallada de las variables, se va a transformar las categóricas a numéricas, a partir de variables dummies. Esta transformación se realizará de manera que si una variable tiene n categorías, se van a definir $n-1$ variables dummies para evitar multicolinealidad. De esta manera si al transformar la variable en dummies todas toman el valor 0, indica que dicha observación se encuentra en la categoría restante. Esto permitirá hacer un estudio más rápido en los modelos.

A continuación, se va a dividir el conjunto de datos en **conjunto de entrenamiento** y de **validación**. En el de entrenamiento figuran los créditos que se han aceptado durante los años 2014 y 2015, consta de 579.951 observaciones y 87 variables. Mientras que el de validación figuran los créditos de los dos primeros cuatrimestres de 2016 y consta de 190.912 observaciones y 87 variables. Esta división se realiza de manera que en el conjunto de entrenamiento se calcule el mejor modelo del extreme gradient boosting y en el de validación a partir de los datos se va a predecir el estado del crédito.

Como se trata de datos no balanceados, se observa una relación de 1:4, se va a establecer que el parámetro scale_pos_weight=3,9. Esto permite controlar el balance de pesos positivos y negativos, es muy útil para clases desequilibradas. Esto significa que clasificar erróneamente un crédito con estado default es aproximadamente cuatro veces peor que en el caso de hacerlo con uno fully paid. Por otra parte, se va a trabajar con matrices xgbDmatrix que son las propias del modelo

El procedimiento que se va a emplear para calcular el modelo más óptima será

1. Se va a calcular el AUC con los parámetros por defecto que nos ofrece R

   • **Parámetros generales**: booster = "gbtree".

   • **Parámetros del tree boosting**: scale_pos_weight=3,9, eta=0,3, gamma=0, max_depth=6, min_child_weight=1, subsample=1, colsample_bytree=1.

   • **Parámetros asociados a la tarea de aprendizaje**: objective = "binary:logistic".

2. Fijar todos los parámetros que forman el modelo y aplicar validación cruzada, se trata de una técnica para calcular el mejor valor de la métrica elegida a partir de la estimación de los parámetros nrounds, η, min_child_weight y max_depth conjuntamente. En nuestro caso, se va a utilizar una **validación cruzada** con 5 iteracciones. El conjunto de datos se va a dividir en 5 subconjuntos. En cada iteracción se ajusta un modelo sobre los 4 subconjuntos (submuestra de validación) y se predice con el subconjunto restante (submuestra de validación). En cada submuestra de entrenamiento y validación se mide el AUC que se ha tomado como parámetro para la tarea del aprendizaje, se cogerá el modelo que tenga un valor más alto, por lo tanto, se trata de maximizar la métrica elegida para comparar modelos. Se repite este proceso circularmente sobre cada iteracción $k = 1, \ldots, 5$. Por último, se combinan y ponderan los resultados.

η	nround	max_depth	min_child_weigth	AUC
0,1	**413**	**4**	**3**	**0,732347**
0,1	430	4	1	0,73228
0,7	999	2	1	0,73220
0,5	999	2	1	0,73220
0,1	1000	2	1	0,73212
0,9	998	2	1	0,73208
0,3	1000	2	1	0,73207
0,1	204	6	5	0,73136
0,1	165	6	3	0,73127
0,3	154	4	1	0,73058
0,3	109	4	3	0,73033
0,9	71	4	3	0,72824
0,5	89	4	3	0,72821
0,7	68	4	1	0,72816
0,3	66	6	5	0,72814
0,3	52	6	3	0,72807
0,9	62	4	1	0,72792
0,5	61	4	1	0,72789
0,7	58	4	3	0,72789
0,5	30	6	3	0,72517
0,7	30	6	3	0,72500
0,5	32	6	5	0,72496
0,9	27	6	5	0,72491
0,9	28	6	3	0,72456
0,7	28	6	5	0,72455

Cuadro 4.1: Estudio del AUC en función de los parámetros nrounds, η, min_child_weight y max_depth a través de un criterio de validación cruzada. En rojo se encuentra el conjunto de parámetros para los que se ha conseguido el mejor AUC.

En el caso de los árboles de decisión stump, o de profundidad 2 se puede observar que no son los mejores y que el coste computacional es muy elevado debido al gran número de iteracciones para obtener una respuesta fiable. Por lo tanto, se va a proceder a calcular el modelo partiendo de los valores que han maximizado el AUC.

3. Una vez se han estimado los valores nrounds, η, min_child_weight y max_depth óptimos, se van a fijar y se estimarán los parámetros del boosting restantes : subsample, colsample_bytree, λ y α. A consecuencia de que la diferencia de AUC entre el conjunto de entrenamiento y validación no es muy grande, el valor de γ será igual a 0. Como se ha establecido anteriormente, subsample y colsample_bytree son valores que se encuentran entre 0,5 y 0,8, por lo tanto, los parámetros en los que se va a realizar el estudio son 0,5 y 0,7. En el caso de λ y α se van a considerar los valores 0 y 5 para ver como se modifica el AUC. En función de los valores de AUC obtenidos, se va a realizar una partición más fina del conjunto de los parámetros a estimar.

subsample	colsample_bytree	λ	α	AUC
0,5	**0,5**	5	0	**0,735678**
0,7	0,5	5	5	0,73540
0,5	0,7	5	0	0,73539
0,5	0,7	5	5	0,73534
0,7	0,5	5	0	0,73516
0,7	0,7	5	0	0,73515
0,7	0,7	0	0	0,73499
0,5	0,5	5	5	0,73487
0,7	0,7	5	5	0,73473
0,7	0,5	0	0	0,73471
0,5	0,7	0	0	0,73466
0,7	0,5	0	5	0,73459
0,5	0,5	0	0	0,73449
0,7	0,7	0	5	0,73438
0,5	0,5	0	5	0,73426
0,5	0,7	0	5	0,73418

Cuadro 4.2: Estudio del AUC a través de la función xgb.train de [16] de los parámetros subsample, colsample_bytree, λ y α a partir de los valores óptimos que se han obtenido en el proceso anterior. El conjunto de parámetros que nos dan el valor de AUC óptimo está marcado en rojo.

El mejor AUC se obtiene cuando los valores de subsample y colsample_bytree son los más bajos, de manera que, subsample = 0,5 y colsample_bytree=0,5. Se puede observar que los valores de λ y α no quedan muy claros. Por lo tanto, se va a realizar un mallado de manera que permita obtener el modelo final que mejor se ajusta a los datos, a través del AUC.

λ	α	AUC
10	20	**0.73656**
15	15	0,73647
20	10	0,73641
20	20	0,73629
15	10	0,73624
10	15	0,73621
10	10	0,73608
20	15	0,73585
15	20	0,73583

Cuadro 4.3: Estudio del AUC a través de los parámetros λ y α a partir de los valores óptimos que se han obtenido anteriormente. El conjunto de parámetros que nos dan el valor de AUC óptimo está marcado en rojo.

Se puede observar que el mejor modelo se obtiene cuando $\lambda = 10$ y $\alpha = 20$. Debido a que el AUC óptimo se obtiene cuando λ toma el menor valor y es mejor que el obtenido

anteriormente, se va a fijar y estimar el mejor parámetro de α. Se va a realizar un estudio detallado del AUC cuando α toma los valores 25, 30, 40 y 50.

α	AUC
25	**0.73658**
20	0,73655
50	0,73654
40	0,73635
30	0,73628
15	0,73621
10	0,73608

Cuadro 4.4: Estudio del AUC fijando todas las variables y modificando el valor de α El parámetro α óptimo está marcado en rojo.

Observando los resultados obtenidos y teniendo en cuenta que valores grandes de α generan modelos más conservativos, se tiene que el valor de α que da el modelo más óptimo es 25. El máximo AUC es 0,73658 y se alcanza cuando $nround = 406$. Por lo tanto, el modelo estimado es mejor que el obtenido a partir de los parámetros por defecto.

4.3. Resultados

Resultados obtenidos a partir del modelo de regresión logística

Se va a realizar un breve estudio para ver como se comporta el conjunto de datos con un modelo de regresión logística, es el más utilizado junto con probit.

Se ajusta un modelo de regresión logística y se calcula la matriz de confusión sobre el conjunto de validación. En la siguiente tabla se pueden ver los casos que se han clasificado correcta e incorrectamente con este modelo. En estas matrices, las columnas son las etiquetas predichas por el modelo y las filas se corresponden con las etiquetas reales asociadas a los datos.

Predicciones	**Fully Paid**	**Default**
Fully Paid	149.019	2.229
Default	36.283	3.381

Cuadro 4.5: Matriz de confusión para el modelo de regresión logística asociado

El modelo predice como Fully Paid 149.019 de los 151.248 créditos Fully Paid que tenemos definidos en la base de datos, de manera que la sensibilidad es igual a 0,98526. En el caso del Default el modelo predice bien 3.381 de los 39.664, por lo tanto la especificidad es 0,08524. La precisión y el AUC del modelo es 0,79830 y 0,53525. Esto implica que la precisión es elevada,

aunque no es capaz de captar los créditos que son Default, esto es consecuencia de que las clases no están balanceadas.

La **curva ROC** asociada al modelo de regresión logística para estos datos se puede ver en la Figura 4.10.

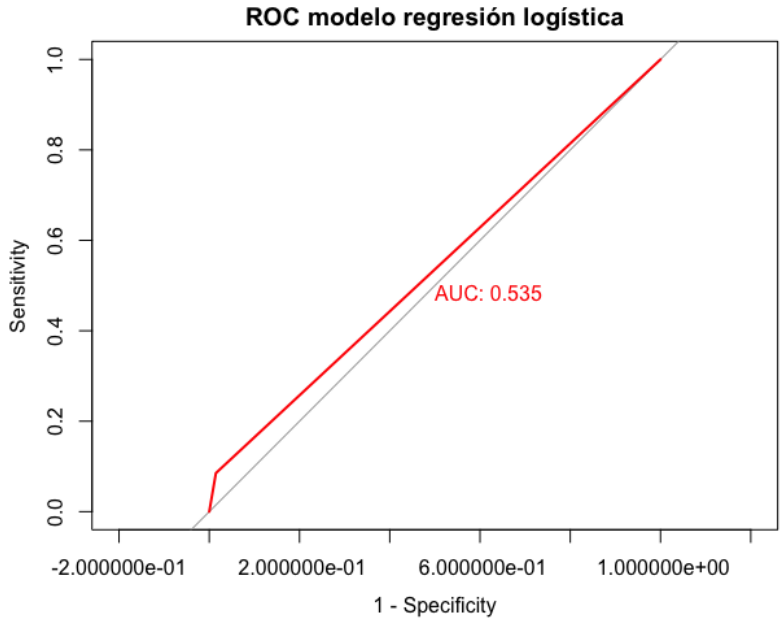

Figura 4.10: Curva ROC generada a partir de la especificidad y la sensibilidad del modelo de regresión logística para el conjunto de datos.

Debido a que el AUC es bajo, se va a intentar buscar un modelo que maximice dicho valor, para ello se va a obtener los resultados de los distintos casos teóricos que se han mencionado.

Resultados obtenidos a partir de los parámetros por defecto de R

En el primer caso, se calculó que el número de iteracciones para obtener el mejor modelo con los parámetros por defecto es 60. El AUC es igual a 0,73210.

La matriz de confusión obtenida a partir del modelo dado por los parámetros

- scale_pos_weight=3,9
- eta=0,3
- gamma=0
- max_depth=6
- min_child_weight=1
- subsample=1

- colsample_bytree=1.

viene definida por

Predicciones	**Fully Paid**	**Default**
Fully Paid	112.821	38.427
Default	16.587	23.077

Cuadro 4.6: Matriz de confusión para el modelo XGBoost ajustado con los valores por defecto.

El modelo predice como Fully Paid 112.821 de los 151.248 créditos Fully Paid que tenemos definidos en la base de datos, de manera que la sensibilidad es igual a 0,74590. En el caso del Default el modelo predice bien 23.077 de los 39.664, por lo tanto la especificidad es 0,58180. La precisión es 0,71180, esto implica que es elevada, aunque inferior a la de la regresión logística. Se ha conseguido una mejoría considerable en los créditos Default que se han clasificado correctamente y en el valor de AUC que es igual a 0,732.

La **curva ROC** asociada al modelo de XGBoost con los parámetros por defecto puede verse en la Figura 4.11.

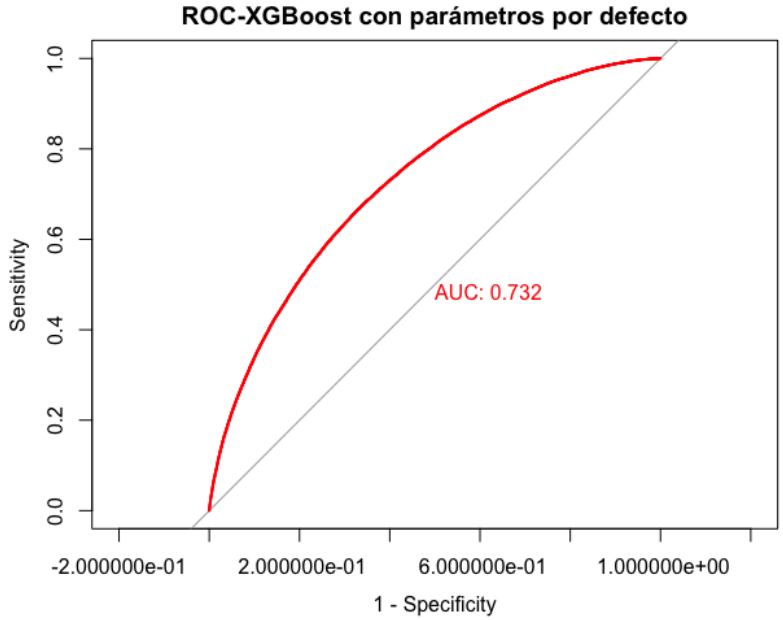

Figura 4.11: Curva ROC generada a partir de la especificidad y la sensibilidad del modelo de XGBoost con los parámetros por defecto de R.

Resultados obtenidos a partir de los parámetros estimados

Una vez realizada la estimación de los parámetros, se determinarán los resultados más característicos e interesante. Primeramente, se va a calcular la matriz de confusión para ver como se distribuyen los aciertos y errores en cada una de los factores, en nuestro caso *Fully Paid* y *Default*.

La matriz de confusión obtenida a partir del modelo dado por los parámetros

- scale_pos_weight=3,9
- eta=0,1
- gamma=0
- max_depth=4
- min_child_weight=3
- subsample=0,5
- colsample_bytree=0,5
- $\lambda = 10$
- $\alpha = 25$

viene definida por

Predicciones	**Fully Paid**	**Default**
Fully Paid	111.593	39.655
Default	15.940	23.274

Cuadro 4.7: Matriz de confusión para el modelo XGBoost con parámetros optimizados a partir de validación cruzada.

El modelo predice como Fully Paid 111.593 de los 151.248 créditos Fully Paid que tenemos definidos en la base de datos, de manera que la sensibilidad es igual a 0,73780. En el caso del Default el modelo predice bien 23.274 de los 39.664, por lo tanto la especificidad es 0,59810. Se puede ver que el modelo permite clasificar correctamente aproximadamente el 60 % en los casos de crédito Default. La precisión es 0,70880, esto implica que es elevada, aunque inferior a la del modelo con los parámetros por defecto. A pesar de empeorar en la precisión, se ha obtenido un AUC y sensibilidad mayor, consiguiendo neutralizar que las clases están desbalanceadas.

La **curva ROC** asociada al modelo XGBoost con los parámetros estimados puede verse en la figura 4.12.

Para terminar, se va a indicar la importancia de las variables en el modelo en función de la ganancia que aporta cada una de ellas. El gráfico representa cada variable con una barra horizontal de longitud proporcional a la importancia de una característica, se encuentran ordenadas por la ganancia y se trata de una importancia relativa. Esto quiere decir que la variable con mayor ganancia, tendrá como importancia relativa un 1 y así sucesivamente.

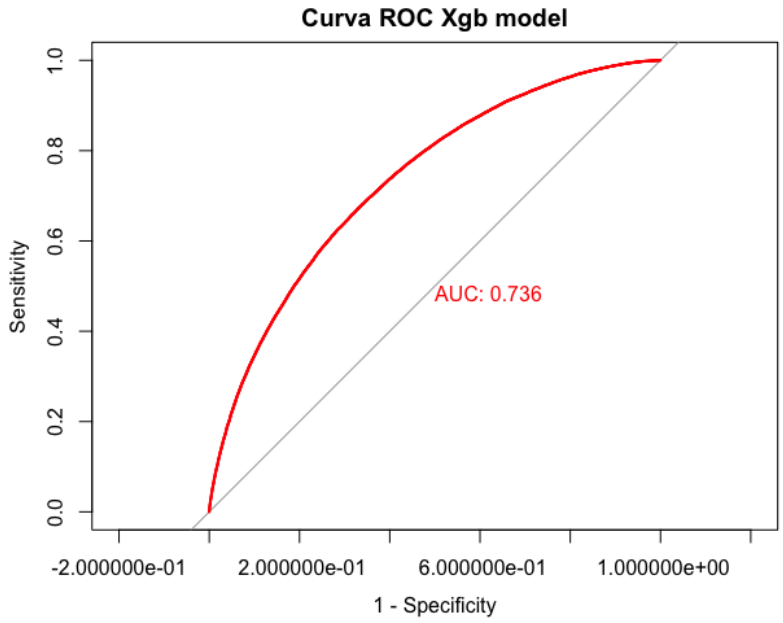

Figura 4.12: Curva ROC generada a partir de la especificidad y la sensibilidad del modelo de XGBoost con los parámetros que se han estimado maximizando el AUC. Se observa que mejora claramente los dos anteriores.

Debido al gran número de variables en nuestro conjunto de datos (87), se ha realizado un top 10 de las características que más influyen a la hora de determinar si un crédito es Fully Paid o Default. Se pueden encontrar las variables más importantes en la Figura 4.13.

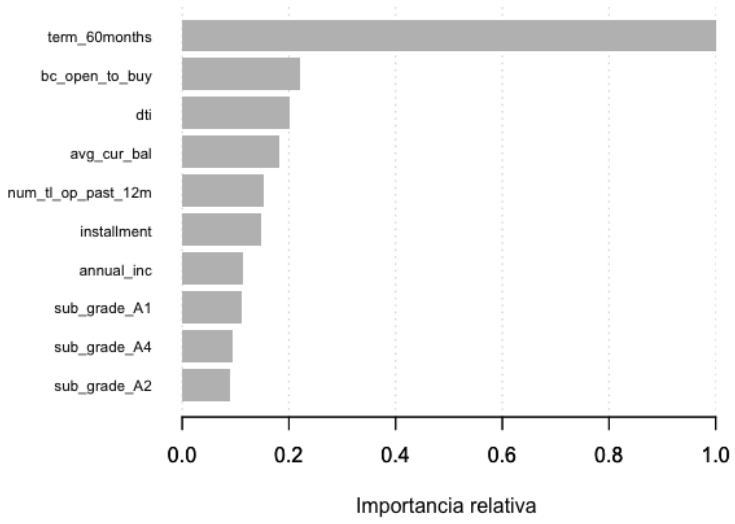

Figura 4.13: Top 10 de las variables según la importancia en el modelo.

Lo que más determina el estado final de un crédito es si la duración es de 30 ó 60 meses, la variable bc_open_to_buy, el ratio de endeudamiento, el saldo actual y el número de cuentas

abiertas en los últimos 12 meses, el salario anual en el momento de la solicitud del crédito, el deposito a pagar en cada mensualidad, así como si el subgrado es A1, A4 y A2.

Se han determinado 413 árboles, uno para cada una de las nrounds en las que se ha realizado el método iterativo, el segundo árbol que se ha obtenido a partir del método empleado es de la forma

```
0:[f0<0.5] yes=1,no=2,missing=1,gain=29083.4414,cover=113839.289"
"1:[f64<5549.5] yes=3,no=4,missing=3,gain=3342.21875,cover=75582.6016"
"3:[f63<13631.5] yes=7,no=8,missing=7,gain=1716.61108,cover=46715.4922"
"7:[f58<20.1549988] yes=15,no=16,missing=15,gain=1002.08081,cover=34797.957"
"15:leaf=-0.0269545373,cover=18765.5215"
"16:leaf=0.0069863922,cover=16032.4365"
"8:[f72<0.375] yes=17,no=18,missing=17,gain=335.970703,cover=11917.5342"
"17:leaf=-0.0350885391,cover=5171.18994"
"18:leaf=-0.0704478547,cover=6746.34375"
"4:[f24<0.5] yes=9,no=10,missing=9,gain=1727.25879,cover=28867.1074"
"9:[f63<7392.5] yes=19,no=20,missing=19,gain=949.792969,cover=26900.2832"
"19:leaf=-0.0416985713,cover=14742.749"
"20:leaf=-0.0800908282,cover=12157.5352"
"10:[f56<57.5100021] yes=21,no=22,missing=21,gain=12.7753906,cover=1966.82263"
"21:leaf=-0,cover=3.19808149"
"22:leaf=-0.15738669,cover=1963.62451"
"2:[f63<10674.5] yes=5,no=6,missing=5,gain=2229.29297,cover=38256.6914"
"5:[f58<23.6150017] yes=11,no=12,missing=11,gain=426.986328,cover=21705.8848"
"11:[f70<1.5] yes=23,no=24,missing=23,gain=227.446289,cover=13052.0508"
"23:leaf=0.0851539448,cover=9585.71875"
"24:leaf=0.052528996,cover=3466.33154"
"12:[f64<13104.5] yes=25,no=26,missing=25,gain=61.2363281,cover=8653.83398"
"25:leaf=0.111950137,cover=7259.73193"
"26:leaf=0.0792178139,cover=1394.10205"
"6:[f58<19.1549988] yes=13,no=14,missing=13,gain=579.8479,cover=16550.8086"
"13:[f64<5532] yes=27,no=28,missing=27,gain=316.183533,cover=8137.82031"
"27:leaf=0.0370634645,cover=4725.8999"
"28:leaf=-0.00220815954,cover=3411.92065"
"14:[f64<8455] yes=29,no=30,missing=29,gain=190.728271,cover=8412.9873"
"29:leaf=0.0681676194,cover=6070.26709"
"30:leaf=0.031891223,cover=2342.7207"
```

4.4. Conclusiones finales

En este trabajo, a partir de los datos de LendingClub se ha realizado una depuración para posteriormente utilizar distintas técnicas para estimar los modelos que nos permitiesen determinar el estado del crédito, *Default* y *Fully Paid*. Para ello se han estudiado diferentes técnicas estadísticas, yendo desde las más antiguas y comunes como la **regresión logística** hasta algunas de las más modernas como **XGBoost**.

Para realizar este trabajo, ha sido necesario el estudio de los distintos métodos de manera teórica, permitiendo entender cual era el funcionamiento del algoritmo y los parámetros a estimar. Para determinar el mejor modelo a elegir, se ha considerado como métrica el AUC. Esto permite establecer un orden en la bondad de los modelos. Una comparación de los tres métodos estudiados se puede ver en la Figura 4.14.

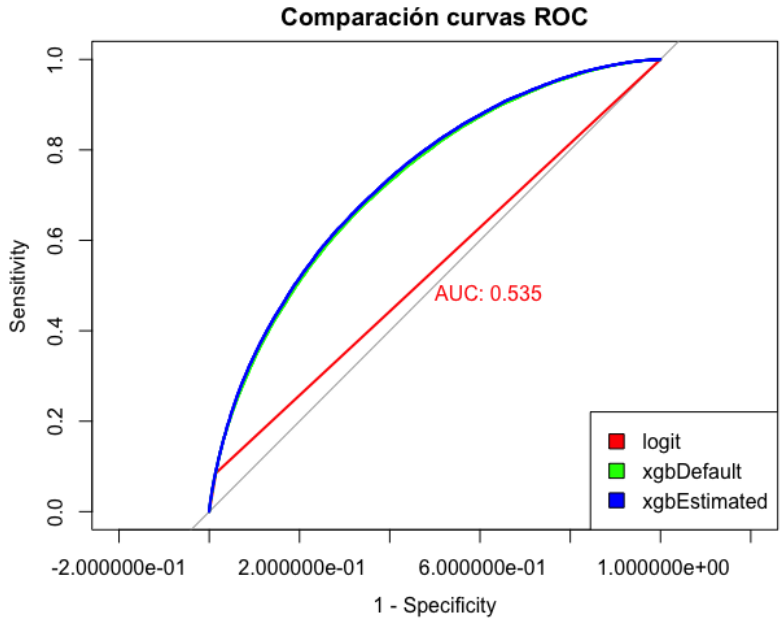

Figura 4.14: Modificación de los tres modelos estudiados para el calculo del riesgo de incumplimiento crediticio a partir de la probabilidad de endeudamiento y con un úmbral igual a 0.5.

El modelo que presenta un AUC mayor es el que se ha estimado los parámetros paso a paso, seguido por el modelo de XGBoost con los parámetros por defecto. Para terminar se encuentra el modelo de regresión logística (en rojo), que presenta un valor de AUC muy bajo, como consecuencia de su baja especificidad.

Se ha conseguido un método en el que el AUC es mayor que el obtenido con los métodos tradicionales. Esto implica que el porcentaje de que clasifique correctamente el estado final de un crédito a partir de las características es superior que al resto de modelos.

4.5. Lineas Futuras

Algunas de los casos que se podrían estudiar a partir de este trabajo son

- Establecer una interpretación del modelo para facilitar que todas las personas sin conocimientos estadísticas puedan entenderlo.

- Replicar el estudio realizado anteriormente en un conjunto de datos con un número de observaciones más grande, de manera que se pueda estudiar el modelo mejor y poder observar si predicciones las son mejores . Esto sería posible con máquinas virtuales con gran potencia. No se ha podido debido a las limitaciones presentadas en los ordenadores portátiles, de memoria RAM y potencia.

Bibliografía

[1] BISHOP, C. M., *Pattern Recognition and Machine Learning*, New York, 2006.

[2] NIELSEN,D , *Tree Boosting With XGBoost. Why Does XGBoost Win "Every" Machine Learning Competition?*, Master of Science in Physics and Mathematics. Department of Mathematical Science. Norwegian University of Science and Technology. 2016.

[3] BRETT POWELL, *Mastering Microsoft Power BI : Expert techniques for effective data analytics and bussiness intelligence*, Editorial Packt.2018.

[4] BRUCE,P. & BRUCE,A. , *Practical Statistics for Data Science*, 50 essential concepts, Editorial O'Reilly. 2017.

[5] BEATE SICK, *Advanced Studies in Applied Statistics (WBL)*, ETHZ Applied Multivariate Statistics, 2018.

[6] FRIEDMAN, J. H., *Greedy Function Approximation: A gradient Boosting Machine*, IMS 1999 reitz lecture, 1999 (modified 2000 and 2001)

[7] KUHN,M. & JOHNSON,J., *Applied Predictive Modeling*, Springer. New York. 2013.

[8] BANCO DE PAGOS INTERNACIONALES, *Comité de Supervisión Bancaria de Basilea. Aplicación de Basilea II: aspectos prácticos*, Press & Communications. Julio 2004.

[9] BREIMAN, L., H. FRIEDMAN, J., A. OLSHEN, R., AND J. STONE, C. , *Classification and Regression Trees*, Chapman Hall, New York, 1984

[10] MARTÍNEZ, I., *Árboles de decisión*, consultada el día 18/06/2019. https://www.uv.es/mlejarza/actuariales/tam/arbolesdecision.pdf

[11] BREIMAN, L., *Bagging predictors*, Machine Learning, 24,123-140, 1996.Statistics. KluwerAcademicPublishers.

[12] BREIMAN, L., *Random Forest*, Machine Learning, 45, 5-32, 2001.Statistics. KluwerAcademicPublishers.

[13] FREUND, Y. AND SCHAPIRE, R. , *Experiments with a new boosting algorithm*, In Saitta, L., editor, Proceedings of the Thirteenth International Conference on Machine Learning (ICML 1996), pages 148-156. Morgan Kaufmann

[14] E. SIGRIST, F., *Gradient and Newton Boosting for Classification and Regression*, Lucerne University of Applied Sciences and Arts. March 4, 2019

[15] CHEN, T. AND GUESTRIN, C., *XGBoost: A Scalable Tree Boosting System*, 2016, Proceedings of the 22Nd ACM SIGKDD International Conference on Knowl- edge Discovery and Data Mining, KDD'16, pages 785 - 794, New York, NY, USA. ACM.

[16] CHEN, T. , HE ,T. , BENESTY, M. , KHOTILOVICH, V. , TANG, Y. , CHO, H. , CHEN, K. , MITCHELL, R. , CANO, I. , ZHOU, T. , LI, M. , XIE, J. , LIN, M. , GENG, Y. AND LI, Y., *XGBoost : Extreme Gradient Boosting*, Version 0.90.0.2, (2019-08-01)

[17] JOANNE , C. -Y. J., LEE, K.L. AND INGERSOLL, G. M. , *An Introduction to Logistic Regression Analysis and Reporting*, (2002).Indiana University-Bloomington. The Journal of Educational Research, 96(1), 3-14.

[18] BANCO DE ESPAÑA, *Implantación y validación de enfoques avanzado de Basilea II en España*, 2006.

[19] WANG, Z. , ZHANG, J. AND VERMA, N. , *Realizing Low-Energy Classification Systems by Implementing Matrix Multiplication Directly Within an ADC*, 2015, pages 1-1, IEE Transactions on Biomedical Circuits and Systems.

Anexo

```
#Librerías
library(dplyr)
library(caret)
library(stringr)
library(lubridate)
library(plyr)
library(fastDummies)
library(dplyr)
library(bestglm)
library(ROCR)
library(pROC)
library(Rcmdr)
library(xgboost)
library(data.table)

#Carga del conjunto de datos

setwd("~/Google Drive/plantillaTFM/Programacio?n")
load("loan.Rdata")

#Eliminación de valores duplicados.

loan<-loan %>% distinct
N= dim(loan)[2]
p= dim(loan)[1]

#Creación de la variable deuda, definida por

loan$deuda<-loan$funded_amnt-loan$total_pymnt

#Eliminación de las variables que tienen un número de observaciones omitidas
mayor al 40%
eliminar<-vector("numeric",N)

mifun=function(dataframe){
NaValor<-vector("numeric",dim(dataframe)[2])
for(i in 1:dim(dataframe)[2]){
NaValor[i]<-sum(is.na(dataframe[,i]))*100/dim(dataframe)[1]
}
```

```
return(NaValor)
}

v<-mifun(loan)
cd1<-v<=40
loan<-loan[,cd1]


#Elimina las observaciones que tengan valores omitidos en la variable
 loan_status

loan <- loan[!is.na(loan$loan_status),]

#Eliminanación de variables con características constantes
#varianza es cero y no nos dan ninguna información

loan<-Filter(function(x)(length(unique(x))>1), loan)

#Eliminación de las variables con varianza próxima a cero

varianza<-vector("numeric")
for(columna in colnames(loan)){
varianza[columna] <- var(loan[[columna]],na.rm=TRUE)
}
which(varianza<0.05)

#Eliminación de las variables anteriores

loan[,c("collections_12_mths_ex_med","acc_now_delinq",
"chargeoff_within_12_mths","num_tl_120dpd_2m")]<- NULL

#Transformación de la variable rev_util a numérica

loan$revol_util = (as.numeric(gsub(pattern = "%",replacement = "",
x = loan$revol_util)))

#Análisis de correlaciones de las variables cuantitativas.

continuas<-sapply(loan,class)=="numeric"
categoricas<-sapply(loan,class)=="character"
loancat<-loan[,categoricas]
loancont<-loan[,continuas]

#Los valores omitidos se van a reemplazar por la media asociada a cada
 variable numérica.

median_impute_model <- preProcess(loancont,method="medianImpute")
```

```
loancont <- predict(median_impute_model,loancont)

matrizCorrelacion<- cor(loancont,use="na.or.complete")
varCor <- findCorrelation(matrizCorrelacion, cutoff = .75)

#Eliminan 25 variables

loanCor<-loancont[,-varCor]
names<-colnames(loanCor)

loan<- cbind(loancat,loanCor)

#Transformación de fechas a partir del paquete lubridate

loan$earliest_cr_line<-mdy(loan$earliest_cr_line)
loan$last_pymnt_d<-mdy(loan$last_pymnt_d)
loan$last_credit_pull_d<-mdy(loan$last_credit_pull_d)
loan$issue_d<-mdy(loan$issue_d)

#Tiempo pasado desde el primer al último crédito

loan$time_first_credit = loan$issue_d - loan$earliest_cr_line
loan$time_first_credit = loan$time_first_credit/365

#Paso de medir en días a años, debido a que se trata de medidas
muy grandes

loan$mo_sin_old_il_acct<- loan$mo_sin_old_il_acct/12
loan$mo_sin_rcnt_rev_tl_op<- loan$mo_sin_rcnt_rev_tl_op/12
loan$mo_sin_rcnt_tl<- loan$mo_sin_rcnt_tl/12
loan$mo_sin_old_il_acct<-loan$mo_sin_old_il_acct/12
loan$mo_sin_old_rev_tl_op<-loan$mo_sin_old_rev_tl_op/12
loan$mths_since_recent_bc<-loan$mths_since_recent_bc/12
loan$mths_since_recent_inq<-loan$mths_since_recent_inq/12

#Estudio de variables una a una.

length(unique(loan$title)) #Muchas clases y no aportan mucha información

dim(loan[loan$hardship_flag=="Y",])
dim(loan[loan$debt_settlement_flag=="Y",])
#Variable con variabilidad próxima a cero.

#Eliminación de variables que no aportan información y son dispersas.
#Al igual las variables que son combinación de otras.
#Se evita multicolinealidad

loan[,c("application_type", "emp_title", "grade", "title", "addr_state",
```

```
"pymnt_plan","zip_code","hardship_flag","debt_settlement_flag",
"int_rate","earliest_cr_line")] <- NULL


#Existencia de variables que dependen de prestamos pasados
#No nos aportan información para este préstamo y filtra información de
créditos pasados.

#Eliminación de estas variables

loan[,c("total_rec_int","total_rec_late_fee","recoveries","last_pymnt_d",
"out_prncp","last_credit_pull_d","last_pymnt_amnt","tot_coll_amt",
"num_tl_120dpd_2m")]<-NULL


#Estadística descriptiva

#Variable destino : loan_status

table(loan$loan_status)
loan<-loan[loan$loan_status == "Charged Off" |
loan$loan_status == "Default" | loan$loan_status == "Fully Paid" |
loan$loan_status == "Late (31-120 days)",]
loan$loan_status[loan$loan_status == "Charged Off"] <- "Default"
loan$loan_status[loan$loan_status == "Late (31-120 days)"] <- "Default"
table(loan$loan_status)
table(loan$loan_status)/dim(loan)[1]
#El estudio sería mejor si el número de Default y Fully Paid fuese
#similar.Clases desbalanceadas.

porcentaje<-round(table(loan$loan_status)/dim(loan)[1]*100,2)
medida<-data.frame(cbind(loan_status = c("Default","Fully Paid"),
porcentaje))
ggplot(medida,aes(x=loan_status,y=porcentaje,fill=loan_status))
+ geom_bar(stat="identity") #Relación clara  de 1:4
loan$loan_status<-ifelse(loan$loan_status=="Default","Default","FullyPaid")

colnames(loan)

#Estudio detallado de las variables
##Variables categóricas
#Term
loan$term<-ifelse(loan$term=="36 months","36months","60months")
#Subgrade
table(loan$sub_grade)
#Emp_length, se va categorizar
loan$emp_length<-ifelse(loan$emp_length %in% c("<1 year","1 year","2 years"
,"3 years"),"<=3Years",ifelse(loan$emp_length %in% c("4 years","5 years",
```

```
"6 years","7 years"),"Between4y7Years",ifelse(loan$emp_length  %in%
c("8 years","9 years","10+ years"),"MoreorEqual8Years","WithoutDesk" )))

#home_ownership
table(loan$home_ownership)
#Se va a eliminar la clase "ANY" ya que es irrelevante comparado con el total
loan<-loan[!loan$home_ownership=="ANY", ]

#verification_status
loan$verification_status<-ifelse(loan$verification_status=="Not Verified",
"NotVerified",ifelse(loan$verification_status=="Source Verified",
"SourceVerified","Verified"))

#Purpose
#Distribución de la variable purpose
table(loan$purpose)
#initial_list_status
table(loan$initial_list_status)
#Posibles relaciones
count(loan[loan$initial_list_status == "w" & loan$loan_status=="Default",])
count(loan[loan$initial_list_status == "f" & loan$loan_status=="Default",])

##Variables numéricas

#Installment
summary(loan$installment)

#annual_inc (Resumen numérico y distribución)
summary(loan$annual_inc)
table(loan$annual_inc)

#dti
summary(loan$dti)
#Cantidad en el intervalo [0,100]
dim(loan[0<loan$dti & loan$dti<100,])
loan<-loan[0<loan$dti & loan$dti<100,]
summary(loan$dti)

#delinq_2yrs
table(loan$delinq_2yrs)
summary(loan$delinq_2yrs) #Variable muy apuntalada a la izquierda
loan<-loan[loan$delinq_2yrs <= 19,]
summary(loan$delinq_2yrs)

#inq_last_6mths
table(loan$inq_last_6mths)
summary(loan$inq_last_6mths)
```

```
#pub_rec
table(loan$pub_rec)
summary(loan$pub_rec)
loan<-loan[loan$pub_rec<=14,]
summary(loan$pub_rec)

#revol_bal
table(loan$revol_bal)
summary(loan$revol_bal)

#avg_cur_bal
summary(loan$avg_cur_bal)

#bc_open_to_buy
summary(loan$bc_open_to_buy)

#delinq_amnt
table(loan$delinq_amnt) #Muy distribuido, mayor parte es 0
summary(loan$delinq_amnt)

#mo_sin_old_il_acct
summary(loan$mo_sin_old_il_acct)

#mo_sin_old_rev_tl_op
summary(loan$mo_sin_old_rev_tl_op)

#mo_sin_rcnt_rev_tl_op
summary(loan$mo_sin_rcnt_rev_tl_op)

#mo_sin_rcnt_tl
summary(loan$mo_sin_rcnt_tl)

#mort_acc
porcentaje1<-round(table(loan$mort_acc)/dim(loan)[1]*100,2)
porcentaje1
summary(loan$mort_acc)
loan<-loan[loan$mort_acc<=7,]

#mths_since_recent_bc
summary(loan$mths_since_recent_bc)

#mths_since_recent_inq
summary(loan$mths_since_recent_inq)
table(loan$mths_since_recent_inq)

#num_accts_ever_120_pd
summary(loan$num_accts_ever_120_pd)
porcentaje2<-round(table(loan$num_accts_ever_120_pd)/dim(loan)[1]*100,2)
```

```
porcentaje2
#se van a considerar valores atípicos cuando sean > 4
loan<-loan[loan$num_accts_ever_120_pd <= 4,]

#num_actv_bc_tl
summary(loan$num_actv_bc_tl)

#num_bc_tl
summary(loan$num_bc_tl) #Valores próximos a 0.

#num_il_tl
summary(loan$num_il_tl)

#num_tl_30dpd
summary(loan$num_tl_30dpd)
table(loan$num_tl_30dpd)

#num_tl_90g_dpd_24m
summary(loan$num_tl_90g_dpd_24m)
porcentaje3<-round(table(loan$num_tl_90g_dpd_24m)/dim(loan)[1]*100,2)
porcentaje3
#Cuando num_tl_90g_dpd_24m > que 3, pocos valores (3.65%)
loan<-loan[loan$num_tl_90g_dpd_24m <=3,]

#num_tl_op_past_12m
summary(loan$num_tl_op_past_12m)
porcentaje4<-round(table(loan$num_tl_op_past_12m)/dim(loan)[1]*100,2)
porcentaje4
#Gran variabilidad, no pueda predecir correctamente
loan<-loan[loan$num_tl_op_past_12m <=11,] #Se eliminan el 0.17% de los datos

#pct_tl_nvr_dlq
summary(loan$pct_tl_nvr_dlq)

#percent_bc_gt_75
summary(loan$percent_bc_gt_75)

#pub_rec_bankruptcies
summary(loan$pub_rec_bankruptcies,na.rm=TRUE )
table(loan$pub_rec_bankruptcies )

#tax_liens
summary(loan$tax_liens)
table(loan$tax_liens )

#total_il_high_credit_limit
summary(loan$total_il_high_credit_limit)
table(loan$total_il_high_credit_limit)
```

```
#issue_y
#Años en los que se encuentran los créditos
table(loan$issue_y)

#time_first_credit
loan$time_first_credit<-as.numeric(loan$time_first_credit)
summary(loan$time_first_credit)
dim(loan[loan$time_first_credit<0,])
#Se van a eliminar los valores en los que esta variable es menor que 0.
#Representa el 0.26% de los datos
loan<-loan[loan$time_first_credit>=0,]

#Exportación del archivo de datos
write.csv(loan,file="loan.csv",row.names = FALSE)

loan$issue_d<-as.character(loan$issue_d)
continuas1<-sapply(loan,class)=="numeric"
categoricas1<-sapply(loan,class)=="character"
loancont1<-loan[,continuas1]
loancat1<-data.frame(loan[,categoricas1])
#Transformación de variables categóricas en dummies
loancat1<-select(loan[,categoricas1],-c(loan_status,issue_d))
loancat2<-select(loan[,categoricas1],c(loan_status,issue_d))
loancat1 <- dummy_cols(loancat1,remove_first_dummy = TRUE,ignore_na=TRUE)
loan<-cbind(loanpcat1,loancat2,loancont1)

#Eliminación de las variables convertidas a variables numéricas.
loan$term<- NULL
loan$sub_grade<- NULL
loan$emp_length<- NULL
loan$home_ownership<- NULL
loan$verification_status<- NULL
loan$purpose<- NULL
loan$initial_list_status<- NULL

#Separación en conjunto de validación y entrenamiento
entrenamiento<-filter(loan,loan$issue_y==2014 | loan$issue_y==2015 )
validacion<-filter(loan,loan$issue_y==2016)

entrenamiento$issue_d <- NULL
validacion$issue_d <- NULL
entrenamiento$issue_y <- NULL
validacion$issue_y <- NULL
#Eliminación de la variable deuda debido a que
#Deudas < 0 implican Default y >0 Fully Paid
#Nos hace un ajuste perfecto
entrenamiento$deuda <- NULL
```

```
validacion$deuda <- NULL

#Transformación de loan_status en una variable binaria
entrenamiento$loan_status_bin <- ifelse (
entrenamiento$loan_status == "Default",1,0)
entrenamiento$loan_status<- NULL
validacion$loan_status_bin <- ifelse (validacion$loan_status == "Default",1,0)
validacion$loan_status<-NULL

#Modelo de regresión logística
GLM <- glm(loan_status_bin ~ .,family=binomial(logit),data=entrenamiento)
summary(GLM)
round(exp(coef(GLM),3))
prediccion<-predict(GLM,select(validacion,-c(loan_status_bin)),type="response")
predLoanStatus<- ifelse(prediccion > 0.5,1,0)
validacion$predLoanStatus<-ifelse(prediccion > 0.5,1,0)
validacion$loan_status_bin<- as.factor(validacion$loan_status_bin)
predLoanStatus<-as.factor(predLoanStatus)
#Matriz de confusión
confusionMatrix(predLoanStatus,validacion$loan_status_bin)

#Curva ROC
ROC_logit = roc(response = validacion$loan_status_bin,
predictor = validacion$predLoanStatus)
auc_ROC = auc(ROC_logit)
plot(ROC_logit,legacy.axes = TRUE,print.auc = TRUE,col="red",
main="ROC modelo regresión logística")

#Eliminación de la variable generada en el modelo
validacion$predLoanStatus<- NULL

#Modelo XGBoost
set.seed(123)
xgbEntrenamiento<- xgb.DMatrix(
data = as.matrix(select(entrenamiento,-c(loan_status_bin))),
label=entrenamiento$loan_status_bin)
xgbValidacion<- xgb.DMatrix(
data = as.matrix(select(validacion,-c(loan_status_bin))),
label=validacion$loan_status_bin)

#default parameters
paramsDefault <- list(booster = "gbtree", objective = "binary:logistic",
scale_pos_weight=3.9, eta=0.3, gamma=0, max_depth=6,
min_child_weight=1, subsample=1, colsample_bytree=1)
xgbcvDefault <- xgb.cv( params = paramsDefault, data = xgbEntrenamiento,
nrounds = 100, nfold = 5, showsd = T, stratified = T, early.stop.round = 10,
metrics = "auc",maximize = T)
watchlist<-list(train=xgbEntrenamiento,eval=xgbValidacion)
```

```
xgb1 <- xgb.train (params = paramsDefault, data = xgbEntrenamiento,
nthread = 1,silent = 0, verbose =1,nrounds = 60 , watchlist = watchlist,
early.stop.round = 10, maximize = T , eval_metric = "auc")

#Predicción
xgbDefaultpred <- predict (xgb1,xgbValidacion)
xgbpredDefault <- ifelse (xgbDefaultpred > 0.5,1,0)
#Curva ROC
ROC_xgbDefault = roc(response = validacion$loan_status_bin,
predictor = xgbDefaultpred)
auc_curveDefault = auc(ROC_xgbDefault);auc_curveDefault
plot(ROC_xgbDefault,legacy.axes = TRUE,print.auc = TRUE,col="red",
main="ROC-XGBoost con parámetros por defecto")
validacion$loan_status_bin<- as.factor(validacion$loan_status_bin)
xgbpredDefault<-as.factor(xgbpredDefault)
#Matriz de confusión
confusionMatrix(xgbpredDefault,validacion$loan_status_bin)
validacionDefault<-cbind(validacion,xgbpredDefault)

etas<-c(0.1,0.3,0.5,0.7,0.9)

#max_depth = 6
min_child_weigths<-c(3,5)

params <- list(booster = "gbtree", objective = "binary:logistic",eta=etas
,scale_pos_weight=3.9, gamma=0, max_depth=6,
min_child_weight= min_child_weigths, subsample=1, colsample_bytree=1)

saveData1<-data.frame(eta=numeric(0),iteraccion=numeric(0),
min_child_weigth=numeric(0),auc=numeric(0))
dim(saveData1)
for(eta in etas){
for(min_child_weigth in min_child_weigths){
xgbcv <- xgb.cv( params = params, data = xgbEntrenamiento,
nrounds = 100, nfold = 5,metrics = "auc", stratified = T,
early.stop.round = 20, maximize = TRUE)
saveData1[nrow(saveData1)+1,]<-c(0.3,xgbcv$best_iteration,min_child_weigth,
xgbcv$evaluation_log[xgbcv$best_iteration,4])

}
}

save(saveData1,file="saveData1.Rdata")

#max_depth = 4
min_child_weigths1<-c(1,3)
params2 <- list(booster = "gbtree", objective = "binary:logistic",
eta=etas,scale_pos_weight=3.9, gamma=0, max_depth=4,
```

```
min_child_weight= min_child_weigths1, subsample=1, colsample_bytree=1)

saveData2<-data.frame(eta=numeric(0),iteraccion=numeric(0),
min_child_weigth=numeric(0),auc=numeric(0))

dim(saveData2)
for(eta in etas){
for(min_child_weigth in min_child_weigths1){
xgbcv2 <- xgb.cv( params = params2, data = xgbEntrenamiento,
nrounds = 250,nfold = 5,metrics = "auc", stratified = T,
early.stop.round = 20, maximize = TRUE)
saveData2[nrow(saveData2)+1,]<-c(0.3,xgbcv2$best_iteration,min_child_weigth,
xgbcv2$evaluation_log[xgbcv2$best_iteration,4])
}
}
save(saveData2,file="saveData2.Rdata")

#max_depth = 2

params3 <- list(booster = "gbtree", objective = "binary:logistic", eta=etas,
scale_pos_weight=3.9, gamma=0, max_depth=2, min_child_weight=1,
 subsample=1, colsample_bytree=1)
saveData3<-data.frame(eta=numeric(0),iteraccion=numeric(0),
min_child_weigth=numeric(0),max_depth=numeric(0),auc=numeric(0))
dim(saveData3)
for(eta in etas){
xgbcv3<- xgb.cv( params = params3, data = xgbEntrenamiento, nrounds = 350,
nfold = 5,metrics = "auc", early.stop.round = 10, maximize = TRUE)
saveData3[nrow(saveData3) +1,] <-c(eta,xgbcv3$best_iteration,1,2,
xgbcv3$evaluation_log[xgbcv3$best_iteration,4])
}
save(saveData3,file="saveData3.Rdata")

#calcular 4 variables: subsample, colsample_bytree,lambda y alpha

subsamples=c(0.5,0.7)
colsample_bytrees=c(0.5,0.7)
lambdas=c(0,5)
alphas=c(0,5)

paramstrain <- list(booster = "gbtree", objective = "binary:logistic",
eta=0.1,scale_pos_weight=3.9, gamma=0, max_depth=4,
min_child_weight=3, alpha=alphas,subsample=subsamples,
colsample_bytree=colsample_bytrees,lambda=lambdas)

watchlist<-list(train=xgbEntrenamiento,eval=xgbValidacion)
saveEntrenamiento<-data.frame(subsample=numeric(0),
colsample_bytree = numeric(0),lambda=numeric(0),alpha=numeric(0),
```

```
auc=numeric(0))
dim(saveEntrenamiento)

for(subsample in subsamples){
for(colsample_bytree in colsample_bytrees){
for(lambda in lambdas){
for(alpha in alphas){

xgbtrain <- xgb.train (params = paramstrain, data = xgbEntrenamiento,
nrounds = 423, watchlist = watchlist,early.stop.round = 10,
maximize = T , eval_metric = "auc")
saveEntrenamiento[nrow(saveEntrenamiento)+1,]<-c(subsample,
colsample_bytree,lambda,alpha,xgbtrain$best_score)

}
}
}
}

saveEntrenamiento<-saveEntrenamiento%>% arrange(desc(auc))
save(saveEntrenamiento,file="saveEntrenamiento.Rdata")

#Calculo del mejor alpha y lambda
lambdas1=c(10,15,20)
alphas1=c(10,15,20)
paramstrain1 <- list(booster = "gbtree", objective = "binary:logistic",
eta=0.1,scale_pos_weight=3.9, gamma=0, max_depth=4,
min_child_weight=3, subsample=0.5, colsample_bytree=0.5,
lambda=lambdas1,alpha=alphas1)
watchlist<-list(train=xgbEntrenamiento,eval=xgbValidacion)
saveEntrenamiento1<-data.frame(lambda=numeric(0),alpha=numeric(0),
iteraccion=numeric(0),auc=numeric(0))
dim(saveEntrenamiento1)
for(lambda in lambdas1){
for(alpha in alphas1){
xgbtrain1 <- xgb.train (params = paramstrain1, data = xgbEntrenamiento,
nrounds = 423, watchlist = watchlist,early.stop.round = 10,
maximize = T , eval_metric = "auc")
saveEntrenamiento1[nrow(saveEntrenamiento1)+1,]<-c(lambda,alpha,
xgbtrain1$best_iteration,xgbtrain1$best_score)
}
}
saveEntrenamiento1<-saveEntrenamiento1%>% arrange(desc(auc))
save(saveEntrenamiento1,file="saveEntrenamiento1.Rdata")

#Calculo del mejor valor de aplha
alphas2=c(25,30,40,50)
paramstraincomp <- list(booster = "gbtree", objective = "binary:logistic",
```

```
eta=0.1,scale_pos_weight=3.9, gamma=0, max_depth=4, min_child_weight=3,
subsample=0.5, colsample_bytree=0.5,lambda=10,alpha=alphas2)
watchlist<-list(train=xgbEntrenamiento,eval=xgbValidacion)
saveEntrenamientocomp<-data.frame(lambda=numeric(0),alpha=numeric(0),
iteraccion=numeric(0),auc=numeric(0))
dim(saveEntrenamientocomp)
for(alpha in alphas2){
xgbtraincomp <- xgb.train (params = paramstraincomp, nfold = 5,
data = xgbEntrenamiento,nrounds = 423, watchlist = watchlist,
early.stop.round = 10,maximize = T , eval_metric = "auc")
saveEntrenamientocomp[nrow(saveEntrenamientocomp)+1,]<-c(10,alpha,
xgbtraincomp$best_iteration,xgbtraincomp$best_score)
}
saveEntrenamientocomp<-saveEntrenamientocomp%>% arrange(desc(auc))
save(saveEntrenamientocomp,file="saveEntrenamientocomp.Rdata")

#Modelo estimado

paramstrainFinal<-list(booster = "gbtree", objective = "binary:logistic",
eta=0.1,scale_pos_weight=3.9, gamma=0, max_depth=4, min_child_weight=3,
subsample=0.5, colsample_bytree=0.5,lambda=10,alpha=25)
watchlist<-list(train=xgbEntrenamiento,eval=xgbValidacion)
saveEntrenamientoFinal<-data.frame(iteraccion=numeric(0),auc=numeric(0))
dim(saveEntrenamientoFinal)
xgbtrainFinal <- xgb.train (params = paramstrainFinal,
data = xgbEntrenamiento, nrounds = 423, watchlist = watchlist,
early.stop.round = 10,maximize = T , eval_metric = "auc")
saveEntrenamientoFinal[nrow(saveEntrenamientoFinal)+1,]<-
c(xgbtrainFinal$best_iteration,xgbtrainFinal$best_score)
save(saveEntrenamientoFinal,file="saveEntrenamientoFinal.Rdata")

#Predicción
xgbFinalpred <- predict (xgbtrainFinal,xgbValidacion)
xgbpredFinal <- ifelse (xgbFinalpred > 0.5,1,0)

#Curva ROC
ROC_xgb = roc(response = validacion$loan_status_bin,
predictor = xgbFinalpred)
auc_curve = auc(ROC_xgb)
plot(ROC_xgb,legacy.axes = TRUE,print.auc = TRUE,col="red",
main="Curva ROC Xgb model")
validacion$loan_status_bin<- as.factor(validacion$loan_status_bin)
xgbpredFinal<-as.factor(xgbpredFinal)
#Matriz de confusión del modelo estimado
confusionMatrix(xgbpredFinal,validacion$loan_status_bin)

#Importancia de las variables
importancia<-xgb.importance(colnames(xgbEntrenamiento),
```

```
model = xgbtrainFinal)
print(importancia)

#Gráfico de la importancia de las variables.
#TOP 10 de las variables.
xgb.plot.importance(importance_matrix = importancia,rel_to_first = T,
top_n=10,xlab ="Importancia relativa")

#Árboles que definen el modelo
xgb.dump(xgbtrainFinal,with_stats = T)
xgb.plot.tree(model=xgbtrainFinal)

#Dibujo de las tres curvas ROC en el mismo gráfico.
plot(ROC_logit,legacy.axes = TRUE,print.auc = TRUE,col="red",
main="Comparación curvas ROC")
plot(ROC_xgbDefault,legacy.axes = TRUE,col="green",add=TRUE)
plot(ROC_xgb,legacy.axes = TRUE,col="blue",add=TRUE)
legend("bottomright",legend=c("logit","xgbDefault","xgbEstimated"),
fill=c("red","green","blue"))
```