Santiago de Cali, junio 13 de 2023

Doctor

Andrés Felipe Amador Rodríguez

Director de la carrera de Matemáticas Aplicadas

PONTIFICIA UNIVERSIDAD JAVERIANA CALI

Cordial Saludo,

Por medio de la presente me permito hacer la entrega a usted mi proyecto de grado titulado **"EVALUACIÓN DE MODELOS DE CLASIFICACIÓN PARA LA PREDICCIÓN DE RIESGO CREDITICIO DE CLIENTES EN EL SECTOR FINANCIERO"**, para ser evaluado por la facultad.

Espero que este proyecto cumpla con los requisitos estipulados para su aprobación.

Atentamente,

_____

Korin Alexa Idárraga Ospina
CC. 1143874831
Código estudiantil: 8913116

Santiago de Cali, junio 13 de 2023

Doctor

Andrés Felipe Amador Rodríguez

Director de la carrera de Matemáticas Aplicadas

PONTIFICIA UNIVERSIDAD JAVERIANA CALI

Cordial Saludo,

Por medio de la presente me permito informarle que la estudiante del programa de Matemáticas Aplicadas, Korin Alexa Idárraga Ospina con código 8913116, trabajó y finalizó bajo mi dirección, el proyecto de grado denominado **"EVALUACIÓN DE MODELOS DE CLASIFICACIÓN PARA LA PREDICCIÓN DE RIESGO CREDITICIO DE CLIENTES EN EL SECTOR FINANCIERO"** , el cual considero se encuentra en condiciones para ser sometido a evaluación.

Atentamente,

Isabel C. García A.

_____

Isabel Cristina García Arboleda
Profesor
Departamento de Ciencias Naturales y Matemáticas
Pontifica Universidad Javeriana Cali

EVALUACIÓN DE MODELOS DE CLASIFICACIÓN PARA LA PREDICCIÓN DE RIESGO CREDITICIO DE CLIENTES EN EL SECTOR FINANCIERO

KORIN ALEXA IDARRAGA OSPINA

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA Y CIENCIAS
SANTIAGO DE CALI
2023

EVALUACIÓN DE MODELOS DE CLASIFICACIÓN PARA LA PREDICCIÓN DE RIESGO CREDITICIO DE CLIENTES EN EL SECTOR FINANCIERO

KORIN ALEXA IDARRAGA OSPINA

Trabajo de grado para optar por el título de matemático aplicado

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA Y CIENCIAS
SANTIAGO DE CALI
2023

Índice general

Capítulo 1

Introducción

El riesgo se refiere a las posibles consecuencias negativas asociadas a una situación peligrosa o no deseada. En el ámbito empresarial, nos enfrentamos a diversos tipos de riesgos, como el económico y el financiero. Sin embargo, en el sector de las entidades financieras, existen riesgos adicionales inherentes a sus actividades, tales como el riesgo de liquidez, operacional, legal y crediticio. En este estudio, nos centraremos específicamente en el riesgo crediticio, que se refiere al posible impacto que enfrenta una empresa cuando sus deudores incumplen, lo cual puede desencadenar crisis financieras y poner en peligro la estabilidad de la entidad[20].

El enfoque en el riesgo crediticio se justifica debido a la incertidumbre generada por los activos de los clientes. La medición de este riesgo se basa en el conocimiento que tenemos del cliente y su operación. Para ello, se recopila información del cliente a través del sistema financiero y otras entidades, con el objetivo de obtener las variables necesarias para clasificar o calificar al cliente como riesgoso o no riesgoso. En el pasado, la medición de este riesgo dependía principalmente de los datos y la intuición del encargado, ya que no se contaba con suficiente capacidad tecnológica para implementar nuevas estrategias. Sin embargo, en la actualidad, gracias al avance tecnológico y al uso de modelos estadísticos, es posible evaluar de manera más precisa y objetiva si los clientes representan un riesgo para la entidad financiera

En Colombia, la medición del riesgo de crédito se lleva a cabo mediante un conjunto de políticas, normas y metodologías establecidas en el Sistema de Administración de Riesgo Crediticio (SARC), supervisado por la Superintendencia Financiera de Colombia. Estas regulaciones se aplican en el proceso de otorgamiento de créditos, lo que permite a las entidades financieras asegurar una gestión adecuada del riesgo crediticio y establecer modelos internos para su evaluación [5].

En este contexto, los datos desempeñan un papel crucial, ya que proporcionan información valiosa sobre los clientes, como sus datos personales, reportes en centrales de riesgo y movimientos bancarios. Estos datos son utilizados en la construcción de modelos que permiten comprender y predecir la deserción de clientes, también conocido como “churn”. El churn de clientes en el sector financiero puede tener un impacto significativo en el riesgo crediticio, ya que la pérdida de clientes puede afectar nega-

tivamente los ingresos y la estabilidad financiera de una entidad [7].

En particular, utilizaremos los modelos estadísticos ampliamente aplicados en la clasificación: regresión logística, random forest y xgboost. Estos modelos nos permitirán clasificar a un cliente como riesgoso o no riesgoso, lo que contribuirá a reducir el riesgo financiero asociado con la deserción dentro de una entidad financiera. Previamente, llevamos a cabo un análisis descriptivo de las variables a utilizar, descartando los valores nulos y aplicando la codificación de variables para obtener una base de datos completamente numérica. Todo el proceso de modelado se realizó utilizando Python 3.0, un lenguaje de programación ampliamente utilizado en el ámbito de la ciencia de datos.

Es importante destacar que existen diversos modelos de clasificación disponibles, cada uno con sus propias ventajas y limitaciones. La elección del modelo más adecuado dependerá de las características específicas de la entidad financiera y la disponibilidad de datos. Al combinar la gestión adecuada del riesgo crediticio basada en políticas y regulaciones, junto con la aplicación de modelos de churn, las entidades financieras pueden fortalecer su capacidad para anticipar y abordar los riesgos asociados con la deserción de clientes, protegiendo así su salud financiera y manteniendo relaciones sólidas con sus clientes.

Capítulo 2

Marco teórico

Se presentarán los conceptos teóricos de tres técnicas de analítica sobre las cuales se sustentará la construcción y validación del modelo para la detección del riesgo crediticio. Se hablará sobre regresión logística, random forest y multilayer perceptrón. Finalmente, se describirá la metodología y los criterios para la selección de las variables cuantitativas y cualitativas, que permitirán optimizar el proceso de clasificación del modelo.

2.1 Modelo de regresión logística

Para introduccir la regresión logística, primero vamos a definir qué es la regresión en general con el fin de involucrar al lector en la comprensión de esta.

Definición: *La regresión es una técnica que examina las relaciones funcionales entre variables. El objetivo principal es predecir el valor de una variable para cierto valor dado de otra u otras. La variable a estimar se denomina variable dependiente o respuesta, se denota como Y, las que usamos para predecirla las denominamos independientes o predictoras, y las denotamos como $\boldsymbol{X} = X_1, X_2, ...X_k$.*

Las técnicas que vamos a seguir en el estudio de la regresión logística están influidas por las que se usan en regresión lineal, por lo tanto, daremos una breve explicación de ello, resumiendo sus características más importantes.[16] El **modelo de regresión lineal simple** estudia la relación entre dos variables, una variable independiente Y y una variable dependiente X, a través de una línea recta. La ecuación del modelo de regresión lineal es:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i. \tag{2.1}$$

Extrapolando la regresión lineal simple nos encontramos con el **modelo de regresión lineal multiple** cuyo modelo matemático que se caracteriza por:

$$Y_i = \beta_0 + \sum_{j=1}^{k} \beta_j X_{ij} + \varepsilon_i. \tag{2.2}$$

La descripción de las variables para ambos modelos es:

- β_0 y β_1, son los coeficientes de regresión.
- ε_i , es el error debido al azar y refleja la variabilidad de la variable dependiente Y en cada observación i .

Las principales características de la regresión lineal son: la variable Y es cuantitativa continua entre $(-\infty, \infty)$ y tiene una distribución normal.

Ahora bien, la regresión logística es una herramienta importante en el campo de la inteligencia artificial y el machine learning debido a sus múltiples beneficios. En comparación con otras técnicas de machine learning, los modelos de regresión logística son más simples, rápidos y flexibles, y ofrecen una mayor visibilidad lo que facilita la solución de problemas y la corrección de errores[1]. Además, la regresión logística tiene diversas aplicaciones en el mundo real en sectores como la industria, las finanzas y el marketing, donde se utiliza para estimar la probabilidad de fallos, detectar fraudes y predecir la interacción de los usuarios con los anuncios.

En resumen, la regresión logística es una poderosa herramienta para analizar grandes volúmenes de datos y tomar decisiones basadas en la información obtenida. La regresión logística se deriva de la regresión lineal ya que esta se utiliza para modelar y predecir variables continuas, mientras que la regresión logística se aplica cuando la variable dependiente es categórica. Sin embargo, la regresión logística se basa en una transformación no lineal de las variables independientes utilizando la función logit, lo que permite estimar la probabilidad de pertenencia a una categoría específica [11]. De esta manera, la regresión logística amplía el alcance de la regresión lineal al abordar variables categóricas y permitir la clasificación en lugar de la estimación de valores continuos. La elección de la metodología adecuada en la regresión logística dependerá del tipo de variable dependiente, ya sea binaria, ordinal o multinomial.

2.1.1 Regresión logística binaria

La regresión logística simple es útil en frecuentes situaciones prácticas de investigación en que la respuesta puede tomar únicamente dos valores que representan la ocurrencia o no de un determinado suceso de interés, es decir, la variable respuesta Y no es continua sino dicotómica. Definimos por convenio que la variable de categórica es Y=1 representa que el éxito e Y=0 como el fracaso del suceso, permitiendo que no se asuma la linealidad de la relación entre las variables dependientes e independientes.

Este modelo tiene un objetivo principal calcular la probabilidad de ocurrencia de un evento P y el valor o nivel del mismo, así como identificar las variables predictoras significativas para el modelo. Como el valor que queremos obtener va a estar entre 0 y 1, pues es una probabilidad; un valor cerca de 0 indicara que el suceso Y sea muy poco probable, mientras que un valor próximo a 1 indicara que Y es bastante probable de que ocurra. Teniendo en cuenta este hecho y recurriendo a la idea inicial de regresión lineal podríamos intentar modelar esta situación de la siguiente manera

$$P(Y = 1|X) = \beta_0 + \beta_1 X.$$

Vamos a aplicar una restricción para asegurarnos de que los valores predichos de probabilidad siempre se encuentren dentro del rango de 0 a 1, sin importar los valores que pueda tomar la variable independiente. Este proceso permitirá la verificación de la relación sigmoidal, la cual se ilustra mediante una curva en forma de "S" en la figura 2.1. Dicha función es capaz de tomar cualquier valor real entre 0 y 1, lo que posibilita la clasificación de éxito si la probabilidad supera el umbral de 0.5, y de fracaso en caso contrario.

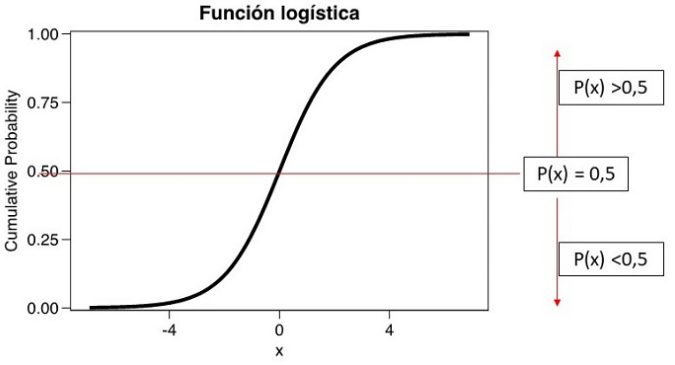


Figura 2.1: Tomada de evidenciasenpediatria.es

Por medio de la transformación logística, que consiste en aplicar el logaritmo al Odds el cual se define como la probabilidad de que cierto evento Y ocurra, dado un valor de la variable independiente X [18].
Sea $P(Y = 1|X)$ que por facilidad lo denotaremos como p:

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X. \tag{2.3}$$

donde β_0 y β_1 son los coeficientes de regresión y X la variable independiente. Notemos que la ecuación (2.3), está bien definida, ya que el logaritmo es continuo y su intervalo es $(-\infty, \infty)$. Ademas el Odds tiene un intervalo de $(0, \infty)$. Ahora aplicando e podemos escribir la ecuación (2.3) como:

$$\begin{aligned}
\frac{p}{1-p} &= e^{\beta_0+\beta_1 X} \\
p &= e^{\beta_0+\beta_1 X} - pe^{\beta_0+\beta_1 X} \\
p &= \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}} \\
p &= \frac{1}{(1+e^{\beta_0+\beta_1 X})(e^{\beta_0+\beta_1 X})^{-1}} \\
p &= \frac{1}{1+e^{-\beta_0-\beta_1 X}}.
\end{aligned} \tag{2.4}$$

Si representamos los valores de p en función de los valores de X, obtendremos la forma de S alargada presentada en la Figura 2.1. En conclusión, tenemos que la ecuación para el modelo de regresión logística binomial esta dado por:

$$p = \frac{1}{1 + e^{-\beta_0 - \beta_1 X}}. \tag{2.5}$$

2.1.2 Regresión logística multinomial

La regresión logística multinomial es un modelo estadístico que se utiliza para analizar situaciones en las que la variable dependiente Y, tiene tres o más categorías, es una extensión del modelo de regresión logística binomial y se basa en la suposición de que Y tiene una distribución multinomial. El objetivo de la regresión logística multinomial es estimar la probabilidad de que un individuo presente o no un evento de interés, dadas un conjunto de variables que explican características particulares de los individuos. Estas variables independientes pueden ser tanto cuantitativas como cualitativas, por ejemplo, el sexo, nivel de educación, tipo de empleo, entre otros [18].

Cuando se evalúa la validez y calidad de un modelo de regresión logística múltiple, se analiza tanto el modelo en su conjunto como los predictores individuales que lo componen. Se considera que el modelo es útil si es capaz de mostrar una mejora significativa con respecto al modelo nulo, es decir, el modelo sin ningún predictor.

La regresión logística multinomial se basa en los mismos principios que la regresión logística simple, que se explicaron anteriormente. Sin embargo, este modelo es capaz de manejar un mayor número de predictores que el modelo simple. Esto permite analizar y evaluar el impacto de múltiples variables independientes en la variable dependiente, lo que puede ser útil en diversas aplicaciones.

La variable respuesta en un modelo de regresión logística multinomial es una variable aleatoria con una distribución multinomial que representa el número de éxitos en cada una de las j categorías. Para incorporar múltiples variables independientes, se extiende el modelo de regresión logística simple antes mencionado. El conjunto de variables independientes se representa como $\mathbb{X} = X_1, X_2, ..., X_j$ y se asume la linealidad del logaritmo del Odds, siguiendo el mismo proceso que en el modelo de regresión logística simple.

Haciendo una extencion de la ecuación (2.3) , tenemos que:

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2, ..., +\beta_j X_j. \tag{2.6}$$

donde β_j son los parámetros del modelo y p es la probabilidad $P(Y = 1|X)$ que se puede encontrar igualando y despejando dicha probabilidad. Podemos escribir, como en el caso simple, la ecuación del modelo de regresión logística múltiple de la siguiente

manera:

$$P = \frac{\epsilon^{\beta_0+\beta_1 X_1+\beta_2 X_2,...,+\beta_j X_j}}{1+\epsilon^{\beta_0+\beta_1 X_1+\beta_2 X_2,...,+\beta_j X_j}} \Rightarrow \frac{1}{1+\epsilon^{-(\beta_0+\beta_1 X_1+\beta_2 X_2,...,+\beta_j X_j)}}.$$

2.1.3 Regresión logística ordinal

La regresión logística ordinal es una técnica de análisis de regresión utilizada para modelar relaciones entre una variable ordinal y una o más variables predictoras. Se utiliza cuando la variable dependiente o de respuesta tiene un orden específico, es decir, cuando se pueden establecer diferentes niveles o categorías de acuerdo a su grado de intensidad o magnitud.

En la regresión logística ordinal, la variable dependiente es transformada en una variable latente continua y subyacente, la cual se modela utilizando una función logística. A partir de esta función, se estima la probabilidad de que la variable dependiente tome un valor igual o menor a un cierto umbral de corte. Esta probabilidad se utiliza para estimar los coeficientes de regresión[10].

El análisis de regresión lineal ordinario consiste en encontrar la combinación de variables predictoras (independientes) que mejor se ajusta a una variable de respuesta (dependiente) y minimizar la diferencia entre la suma de los cuadrados. Los coeficientes estimados en la regresión indican la forma en que los cambios en las variables predictoras influyen en la variable de respuesta, esta lógica tambien se aplica a los apartados anteriores IBM [13].

La regresión logística ordinal se puede modelar mediante la siguiente ecuación:

$$\ln(P(Y \leq k)) = \ln\left(\frac{p}{1-p}\right) = \alpha_k - \beta_1 X_1 - \beta_2 X_2, ..., -\beta_j X_j. \tag{2.7}$$

- $P(Y \leq k)$ es la probabilidad de que la variable dependiente tome un valor igual o menor a k.
- $\beta_1 ... \beta_j$, son los coeficientes de regresión de las variables predictoras.
- α_k es el parámetro de intercepción para la categoría k de la variable dependiente $X_1, X_2, ..., X_j$ son los valores observados de las variables predictoras.

En conclusión, la regresión logística es una técnica ampliamente utilizada en el análisis de datos para modelar la relación entre variables independientes y una variable binaria o categórica ordinal. Cada una de estas variantes de la regresión logística tiene sus propias ventajas y aplicaciones. La regresión logística binaria es sencilla de interpretar y se puede aplicar en problemas de clasificación binaria comunes. La regresión logística multinomial amplía el enfoque a problemas de clasificación con múltiples categorías. Por su parte, la regresión logística ordinal considera el orden de las categorías y es útil cuando existe un ordenamiento claro en las respuestas.

2.2 Modelo Random Forest

Para definir un bosque aleatorio (random forest) primero debemos definir que es un árbol de decisión. Los árboles de decisión son una herramienta fundamental en el campo del aprendizaje automático y la inteligencia artificial, utilizados para resolver problemas de clasificación.

Un árbol de decisión es una estructura jerárquica que se compone de nodos y ramas. Cada nodo representa una característica o atributo del conjunto de datos, y las ramas representan las diferentes opciones o resultados posibles basados en dicha característica. Siguiendo el recorrido por las ramas, se llega a las hojas del árbol, que contienen las etiquetas o clasificaciones finales.

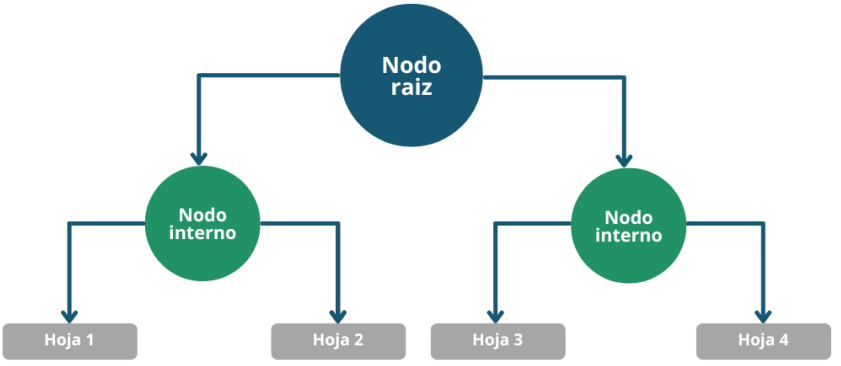


Figura 2.2: Elaboración propia.

Definición: *Un árbol de decisión es un clasificador -en forma de árbol- tal que:*

- *En cada nodo se prueban las features.*
- *Hay una rama por cada valor de las features probado.*
- *Las hojas simbolizan las categorías.*

Los árboles de decisión son altamente interpretables y permiten visualizar de manera clara el proceso de toma de decisiones. Además, pueden manejar datos de diferentes tipos, como variables categóricas y numéricas, es más fácil de interpretar y flexible con la preparación de los datos. Sin embargo, los árboles de decisión individuales pueden tener limitaciones en términos de sobreajuste (overfitting), precisión y puede ser más costoso de entrenar en comparación con otros algoritmos [2].

Por lo anterior, Breiman (2001) introduce el algoritmo de random forest, el cual planteó una relación entre la estadística y las ciencias de la computación [3]. El algoritmo de Random Forest es una técnica de aprendizaje supervisado que genera un conjunto de árboles de decisión sobre un conjunto de datos de entrenamiento que contienen

un grupo de observaciones aleatorias, donde la combinación de las salidas de estos árboles proporciona un único y mejor resultado en comparación con las salidas individuales de cada uno de ellos. Luego, las predicciones de todos los árboles se combinan mediante votación o promedio para obtener una clasificación final más confiable y robusta [19].

Definición: *Un bosque aleatorio es un clasificador que consta de una colección de clasificadores con estructura de árbol* $\{h(\boldsymbol{x}, \theta_k), k = 1, ...\}$ *donde* θ_k *son vectores aleatorios independientes distribuidos de forma idéntica y cada árbol emite un voto unitario para la clase más popular en la entrada* $\boldsymbol{x}$ [3].

Sea:

- θ_k : vector que genera el k-ésimo árbol.
- $\vec{x}$: vector de características en $\mathbb{R}^p$.
- *voto*: etiqueta de la clase.

Es uno de los métodos que puede manejar una gran cantidad de datos y que busca el equilibrio entre el sesgo y la varianza. Se puede decir que es una extensión de los árboles de decisión y que al no necesitar de supuestos para su funcionamiento se convierte en un método no paramétrico. Los bosques aleatorios no se sobreajustan a medida que se agregan más árboles, pero producen un valor límite del error de generalización[19].

2.2.1 Construcción del modelo Random Forest

Este proceso de construcción se puede visualizar en la siguiente Figura: 2.3

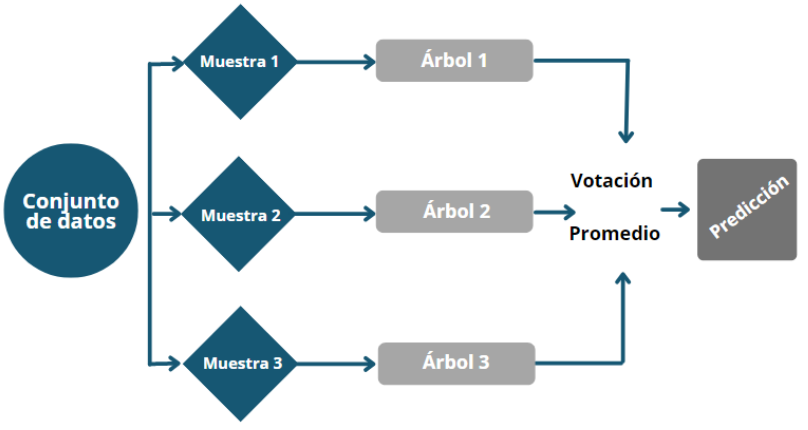


Figura 2.3: Elaboración propia.

Adicionalmente, cada árbol se construye de la siguiente forma: [17].

- Seleccionamos k features (columnas o variables) de las **m** totales (k<m) y creamos un árbol de decisión con esas k características.

- Creamos **n** árboles variando siempre la cantidad de k características y también es posible variar la cantidad de muestras que pasamos a esos árboles.
- Tomamos cada uno de los **n** árboles y le pedimos que hagan una misma clasificación. Cada árbol crece hasta su máxima extensión posible y no hay proceso de poda.
- Calculamos los votos obtenidos para cada "clase" seleccionada y consideraremos a la más votada como la clasificación final de nuestro "bosque".

2.2.2 Hyperparámetros

Los hiperparámetros desempeñan un papel crucial en el ajuste de los modelos de aprendizaje automático, y uno de ellos es el número de variables candidatas que se evalúan en cada ramificación. Sin embargo, además de este hiperparámetro, hay otros aspectos adicionales que también deben tenerse en cuenta. Aunque los nombres pueden variar entre las diferentes bibliotecas y herramientas, al menos los siguientes hiperparámetros deberían estar presentes y considerarse durante el proceso de ajuste del modelo [12].

- **ntree**: número de árboles en el bosque, usar demasiados árboles puede ser ineficiente.
- **mtry**: número de variables aleatorias como candidatas en cada ramifiación.
- **sampsize**: el número de muestras sobre las cuales entrenar.Generalmente se mantiene un rango 60-80 %
- **nodesize**: mínimo número de muestras dentro de los nodos terminales.
- **maxnodes**: máximo número de nodos terminales.

2.2.3 Ventajas y desventajas de Random Forest

Random forest tiene características que son interesantes de conocer dado que éste algoritmo permite tener unas ventajas y desventajas estadísticas sobre otros métodos. A continuación se enumeraran algunas ventajas [4].

1. Puede usarse para clasificación o regresión: En el primer caso, mediante votos y en el segundo caso, el resultado del modelo es el promedio de las salidas.
2. El modelo es simple de entender y tiene un buen desempeño.
3. Se usa para reducción dimensionalidad, ya que logra estimar cuáles son los predictores más importantes.
4. Mantiene su precisión con proporciones grandes de datos faltantes.
5. Existen muy pocas suposiciones y por lo tanto la preparación de los datos es mínima.

Por otra parte, sus principales desventajas son:

1. Visualmente puede ser difícil de interpretar.
2. Puede sobre ajustar ciertos grupos de datos en presencia de ruido.
3. En regresión no son de naturaleza continua y no puede predecir más allá del rango de valores del conjunto de datos usado para entrenar el modelo. En el caso de predictores categóricos con diferente número de niveles, los resultados pueden sesgarse hacia los predictores con más niveles.
4. Se tiene poco control sobre lo que hace el modelo.
5. En regresión, no puede predecir más allá del rango de valores del conjunto de entrenamiento.

2.2.4 Aplicaciones

El modelo de Random Forest tiene diversas aplicaciones en varios campos debido a su capacidad para realizar predicciones precisas y su flexibilidad en el manejo de diferentes tipos de datos. Algunas de las aplicaciones más comunes del modelo de Random Forest incluyen:

- **Análisis de riesgo** : se utiliza para evaluar y gestionar el riesgo en diversas industrias. Puede aplicarse en la identificación de riesgos crediticios, evaluación de riesgos en seguros, detección de fraudes financieros, análisis de riesgos ambientales, entre otros.
- **Optimización** : El Random Forest puede utilizarse para la optimización de procesos y la toma de decisiones. Puede aplicarse en la optimización de rutas de transporte, asignación de recursos, planificación de la cadena de suministro, selección de características relevantes, entre otros.
- **Predicción y regresión**: El Random Forest puede utilizarse para predecir valores numéricos o continuos. Esto se aplica en la predicción de precios de acciones, pronóstico del clima, estimación de la demanda de productos, análisis financiero, entre otros.

2.3 Modelo XGBoost

Es importante comprender que es el gradient boosting pues tiene un enorme impacto en el modelo que será analizado. El gradient boosting es un método de aprendizaje automático ampliamente utilizado en problemas de regresión y clasificación [9]. Se basa en la construcción gradual de un modelo predictivo, similar a otros métodos de boosting.

XGBoost o Extreme Gradient Boosting, es una técnica de aprendizaje automático supervisado que se basa en árboles de decisión, al igual que el método de Random

Forest. Sin embargo, xgboost presenta mejoras significativas sobre estos y otros métodos similares, como Gradient Boosting. De hecho, xgboost combina características de ambos algoritmos.

Al igual que Gradient Boosting, XGBoost construye un modelo predictivo en etapas, donde cada etapa se enfoca en corregir los errores cometidos por los modelos anteriores. Sin embargo, XGBoost introduce mejoras en la forma en que se construyen y combinan los árboles de decisión. Durante cada etapa, se calcula el gradiente de la función de pérdida con respecto a las predicciones actuales. Este gradiente indica la dirección y magnitud del ajuste necesario para minimizar la pérdida. Luego, se ajusta un nuevo árbol de decisión para capturar los errores residuales y se agrega al modelo existente [6].

2.3.1 Algoritmo

Suponiendo un un total de K arboles y sea F el modelo de árboles básico

$$\widehat{y_i} = \sum_{k=1}^{K} f_k(x_i), f_k \in F. \tag{2.8}$$

La función objetivo:

$$L = \sum_i l(\widehat{y_i}, y_i) + \sum_k \Omega(f_k), \tag{2.9}$$

Donde l es la función de pérdida la cual representa el error entre el valor predictivo y el valor real. Ω es la función utilizada para la regularización, es decir, para evitar el sobreajuste

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \|w\|^2, \tag{2.10}$$

Además T representa el número de hojas por árbol, y w representa el peso de las hojas de cada árbol.

Luego de la expansión de Taylor de segundo orden sobre la función objetivo y otros cálculos, podemos obtener finalmente la ganancia de información de la función objetivo después de cada división es:

$$G = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} + \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma. \tag{2.11}$$

En (2.11) se añade γ como umbral de division. Lo que permite reajustar el árbol mientras se optimiza la función objetivo[15], xgboost aplica dos técnicas para evitar el overfitting:

- Si todos los pesos de muestra de los nodos hoja son inferiores al umbral, se detiene la división.
- Muestrea las características aleatoriamente al construir cada árbol.

2.3.2 Ventajas y desventajas de XGBoost

XGBoost presenta varias ventajas que lo hacen una opción popular en el aprendizaje automático [8]. Estas ventajas incluyen:

1. Trabajar directamente con valores faltantes, lo que simplifica el proceso y evita la introducción de sesgos en los resultados. Esto significa que no es necesario realizar una imputación previa de los datos.

2. Adaptarse a problemas más complejos donde las relaciones entre las características y la variable objetivo no son lineales.

3. Ofrece una excelente velocidad de ejecución, lo que significa que puede entrenar modelos y realizar predicciones de manera eficiente.

4. XGBoost proporciona una medida de importancia de las características, que ayuda a comprender cómo las características afectan las predicciones del modelo.

Por otra parte, sus principales desventajas son:

1. XGBoost puede requerir una cantidad significativa de recursos computacionales, especialmente cuando se trabaja con conjuntos de datos grandes. Esto se debe a que la construcción y combinación de múltiples árboles de decisión puede ser intensiva en términos de memoria y tiempo de procesamiento.

2. Tiene varios parámetros que deben ajustarse correctamente para obtener los mejores resultados. Esto implica encontrar un equilibrio entre el rendimiento del modelo y la precisión de las predicciones, evitando el sobreajuste.

3. XGBoost no trabaja con variables categóricas. Esto significa que, es necesario realizar una conversión previa de los tipos de datos no numéricos a numéricos.

2.3.3 Aplicaciones

El modelo xgboost tiene una amplia gama de aplicaciones en diferentes áreas. Algunas de las aplicaciones más comunes incluyen:

- **Clasificación**: se utiliza con frecuencia en problemas de clasificación, como detección de fraudes, clasificación de textos web. análisis de sentimientos, diagnóstico médico y detección de spam.

- **Regresión**:se aplica en problemas de regresión, donde se busca predecir valores numéricos continuos. Por ejemplo, se utiliza en la predicción de la tasa de abandono en cursos en línea ,precios de viviendas,comportamiento del cliente y demanda de productos.

- **Ranking**: se ha utilizado en aplicaciones que implican clasificación y ranking, como en motores de búsqueda y sistemas de recomendación. Puede ayudar a ordenar y clasificar elementos en función de su relevancia o puntuación, lo que resulta útil en situaciones donde se necesita priorizar o seleccionar elementos de acuerdo a ciertos criterios.

Capítulo 3

Descripción del problema

A continuación, se presenta el análisis que se realizó sobre la base de datos tomada de la plataforma de competencia de ciencia de datos y comunidad de científicos donde se encuentran herramientas y repositorios de datos a nivel público. La base titulada Analise de Crédito tiene información financiera sobre el consumo y manejo de los productos del cliente. Posteriormente se implementó los modelos de regresión logistica, random forest y xgboots sobre la base de datos. Finalmente, se muestra un comparativo entre los resultados de los modelos.

3.1 Procesamiento de los datos

En este trabajo, llevamos a cabo un análisis exploratorio de una base de datos que consta de 10.127 registros. Estos registros contienen 16 columnas, donde 15 de ellas son variables independientes que capturan diferentes aspectos relacionados con los clientes y una es la variable objetivo que indica si el cliente es moroso o no. En el cuadro 3.1, se proporciona una descripción detallada de las variables incluidas en la base de datos, junto con su tipo de dato correspondiente.

Variable	**Descripción**	**Tipo**
Id	Número de cuenta	Numérica
Edad	Edad de los clientes	Numérica
Sexo	M(masculino 1) F(femenino 0)	Numérica
Personas a cargo	Número de personas que el titular tiene a cargo	Numérica
Escolaridad	Escolaridad de los clientes	Categórica
Estado civil	Estado civil de los cliente	Categórica
Salario anual	Eonto total recibido en el año por el cliente	Categórica
Tipo tarjeta	Tipo de tarjeta del cliente	Categórica
Meses con tarjeta	Número de meses que el cliente ha tenido una cuenta	Numérica
Num productos	Número de productos que tiene el cliente con el banco	Numérica
Iteraciones 12m	Número de iteraciones del cliente con el banco	Numérica
Meses inactivos 12m	Número de meses que el cliente ha estado inactivo	Numérica
Valor disponible	Saldo que el cliente tiene disponible con el banco	Numérica
Valor transación 12m	Promedio de las transferencias desde la cuenta del titular	Numérica
Num transaciones 12m	Número de transferencias en los últimos 12 meses	Numérica
Default	Clasificación de los clientes	Numérica

Cuadro 3.1: Estructura de la base de datos

Además, contamos con la presencia de valores faltantes y es relevante destacar que esta base de datos contiene tanto información numérica como categórica. Para proteger la confidencialidad de los datos, se realizó un proceso de anonimización de variables y algunos de sus valores. De esta manera, se garantiza la privacidad de los individuos representados en la base de datos.

Exploración variable objetivo (default)
Al realizar una exploración gráfica de la variable objetivo podemos observar que el 83.93 % de los clientes cuenta con un score negativo para las personas categorizadas como morosas, en relación a un 16.07 % con un score positivo para las personas categorizadas como morosas.

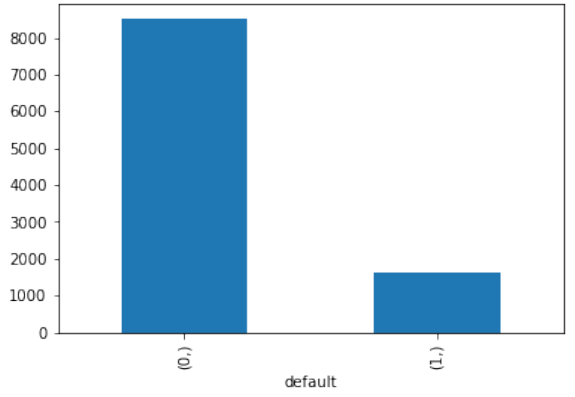


Figura 3.1: Distribución variable objetivo.

Exploración variables independientes
En una primera etapa, se llevó a cabo un análisis de las variables independientes con el fin de determinar la proporción de datos faltantes en relación a la base de datos completa, para así mismo determinar si era necesario el uso de la imputación, que entre lo métodos de imputación podemos encontrar la media, moda, mediana, por regresión ,por knn (K-Nearest-Neighbors),Multiple Imputation Chained Equations, entre otros.

En general, la base contiene pocos registros nulos, en la figura 3.2 se refleja el porcentaje de nulos por variable sobre el total de registros. Las variables con mayor volumen de vacíos son: escolaridad (15 %), salario anual (10.98 %) y estado civil (7.4 %). Esta evaluación nos permitió tomar la decisión de eliminar aquellos registros que presentaban valores faltantes, dado que su porcentaje es menor al 20 %. Este enfoque evitó la necesidad de imputar valores con datos artificiales o estimaciones.

En la segunda etapa, se realizó la codificación de las variables categóricas a numéricas con el objetivo de lograr una base de datos más consistente y uniforme. Esta transformación permitió manejar toda la información en un mismo tipo de dato, lo cual resulta fundamental para la implementación de las variables en los modelos utilizados en el análisis.

escolaridad	15.0000
salario_anual	10.9800
estado_civil	7.4000
id	0.0000
default	0.0000
edad	0.0000
sexo	0.0000
personas_a_cargo	0.0000
tipo_tarjeta	0.0000
meses_con_tarjeta	0.0000
num_productos	0.0000
iteraciones_12m	0.0000
meses_inactivos_12m	0.0000
valor_disponible	0.0000
valor_transacion_12m	0.0000
num_transaciones_12m	0.0000

Figura 3.2: Proporción de registros nulls

Es importante destacar que algunos modelos, como la regresión logística y xgboots, requieren que todas las variables sean numéricas. Por lo tanto, la codificación de las variables categóricas fue necesaria para poder utilizarlas de manera efectiva en el estudio.

- Codificación ordinaria: en esta técnica se asigna a cada categoría un valor numérico ordinal basado en algún criterio, como el orden natural de las categorías. Esta codificación se utilizó en las variables de salario anual y escolaridad.
- Codificación de frecuencia: en esta técnica las categorícas se codifican según su frecuencia en la muestra. Se aplicó a las variables de tipo de tarjeta y estado civil.

Asimismo, tomamos la variable sexo como un dummy, permitiendo pasar de ser categórica a númerica, facilidado su análisis. Por ultimo, se observarón datos atipicos en algunas variables.

3.2 Análisis descriptivo

A continuación, se mostrarán algunas descripciones de las variables tanto categóricas como númericas.

Variables categóricas
A continuación, se presenta un análisis descriptivo de algunas de las variables categóricas y su frecuencia en la base, dichas variables posteriormente fueron codificadas como se menciono en la sección anterior.

En las figuras 3.3 y 3.4 logramos identificar que dentro de los registros hay más clientes categorizados por el sexo femenino, igualmente esta población usa más la tarjeta de blue. Por útlimo, la mayoría de los clientes se distribuyen entre personas que cuentan con algún estudio profesional en la figura 3.6.

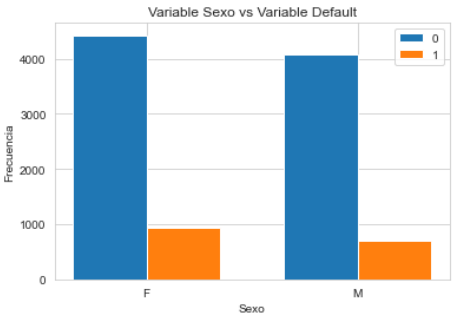


Figura 3.3: Distribución por sexo

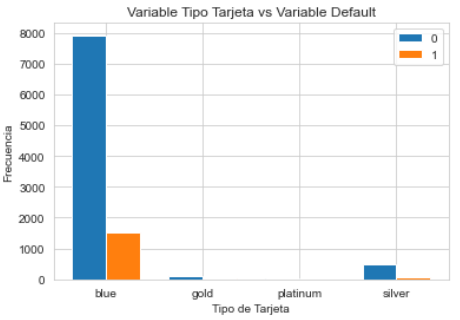


Figura 3.4: Distribución tipo tarjeta

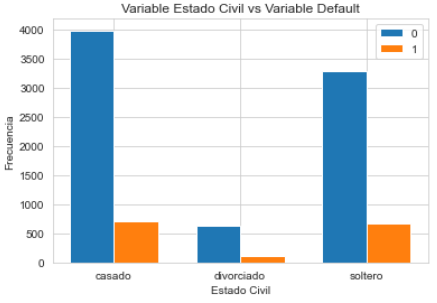


Figura 3.5: Distribución estado civil

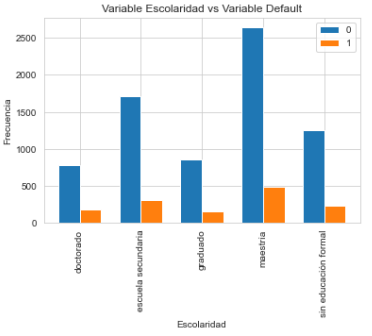


Figura 3.6: Distribución escolaridad

Variables numéricas

También, se realizó un análisis descriptivo de algunas de las variables numéricas y su distribución frente al comportamiento de la variable objetivo.

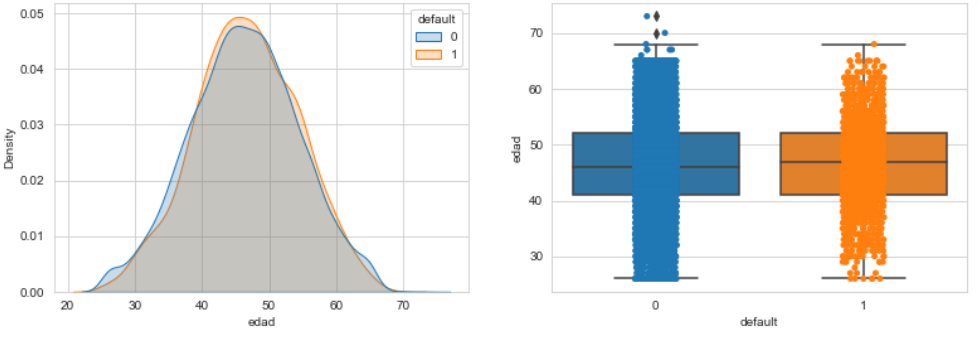


Figura 3.7: Distribución de la edad respecto al default

En la figura:3.7 Logramos ver algunos datos atipicos, adicional las personas entre los 40 y 50 años logran tener inconvenientes con el cumplimiento de sus obligaciones. Los clientes que duran entre 3 y 4 meses sin usar sus productos, son mas propensos a ser morosos, esto se puede observar en la figura:3.8. Por el contrario, las personas que mantienen activas sus cuentas se caracterizan por pagar a tiempo sus productos.

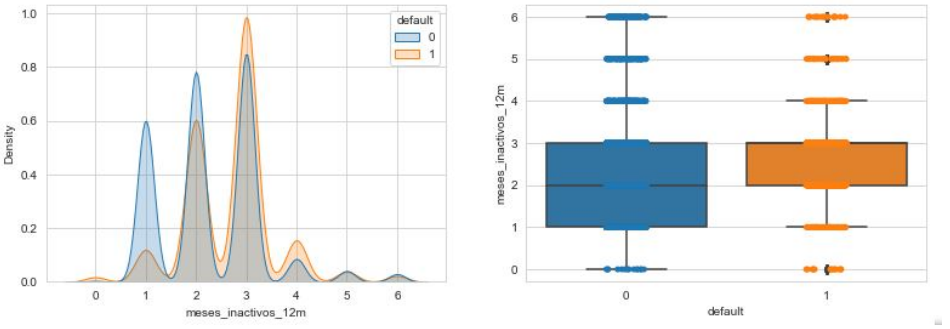


Figura 3.8: Comportamiento de los meses inactivos

Sea la Figura:3.9 Las personas que realizan menos transacciones en el año son mas propensas a ser clientes morosas, por el contrario quienes manejan mas sus tarjetas tienden a ser clientes que son cumplidos con sus pagos.

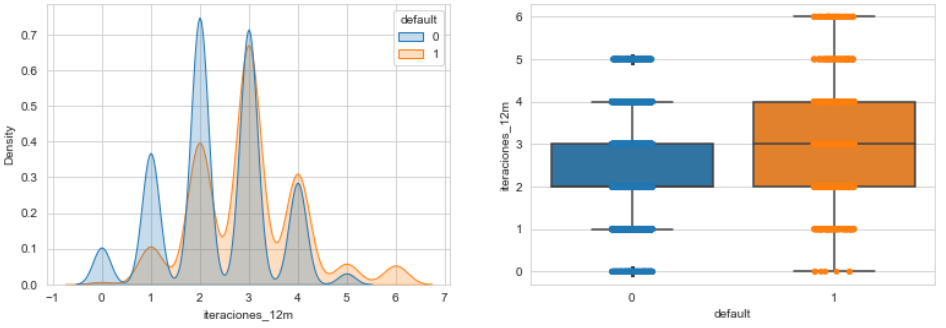


Figura 3.9: Distribución segun los movimientos de los productos

En la Figura:3.10 Cuando los montos de las transaciones son bajos son propensos a ser clientes que no son cumplidos con los pagos. Entre mayor es el monto en las transacciones mas efectivo es el cumplimiento en los pagos.

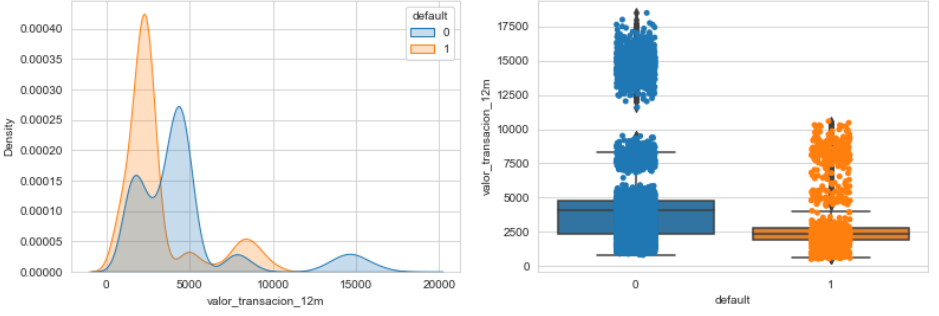


Figura 3.10: Comportamiento según el valor de los movimientos

3.3 Implementación de los modelos

Luego del procesamiento realizado en las variables, se divide la base en dos partes para la etapa de modelado la cual se usará para probar y validar los modelos. Esta división de los datos se hace usando la librería sklearn de python.

1. Base de entrenamiento: Con esta base se "entrenarán" los tres modelos (regresión logística, random forest y xgboost) y tendrá 70 % del total de registros elegidos aleatoriamente (4956 registros).

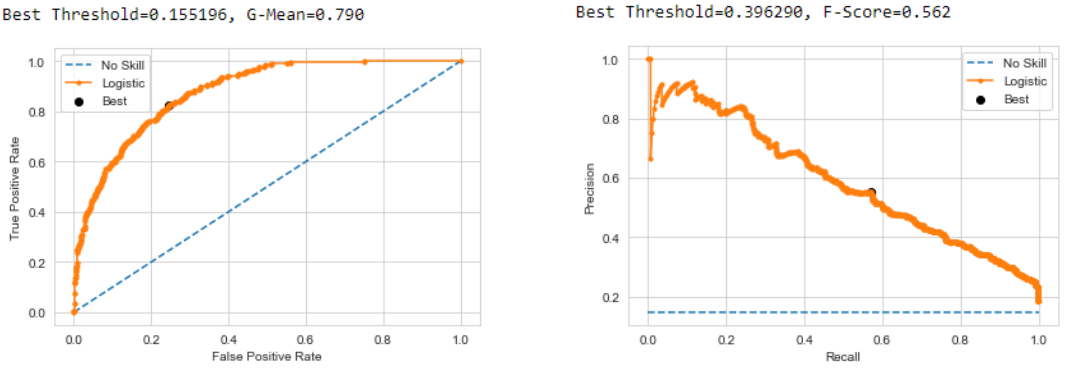
2. Base de validación: Con esta base se validarán los resultados de los modelos y contendrá 30 % restante del total de registros (2125 registros).

Se aplicaron estas divisiones a los tres algoritmos, usando la base de entrenamiento a fin de "entrenar" a los tres correspondientes modelos.

Para encontrar el punto óptimo, utilizamos la curva ROC (Receiver Operating Characteristic) que muestra la relación entre la sensibilidad y la especificidad de un modelo de clasificación, mostrando la tasa de verdaderos positivos (sensibilidad) en el eje y y la tasa de falsos positivos (1 - especificidad) en el eje x. También, la curva f1-score que maximizar el rendimiento de un modelo de clasificación en términos de la precisión y el rendimiento del modelo, teniendo en cuenta tanto los falsos positivos como los falsos negativos; ambos métodos se implementaron en los tres modelos.

A continuación, se observa las curvas de los modelos y su respectico punto de corte.

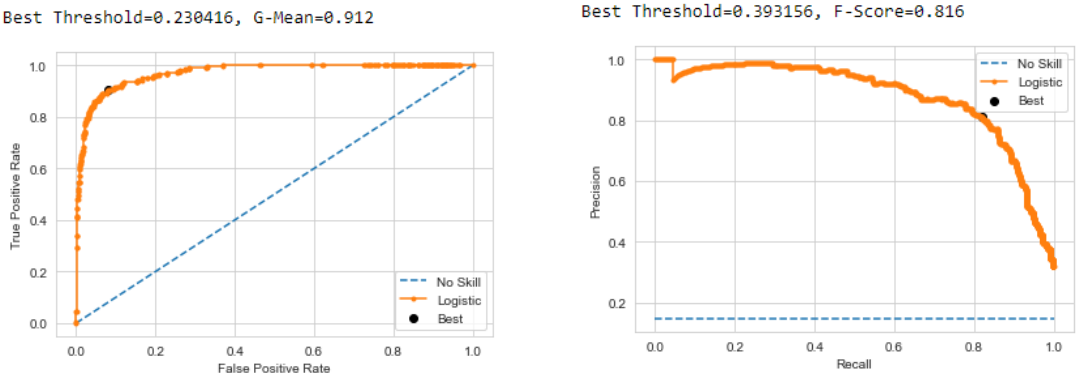
Regresión Logisitca



ROC F1-score

Figura 3.11: Punto de corte de Regresión Logisitca

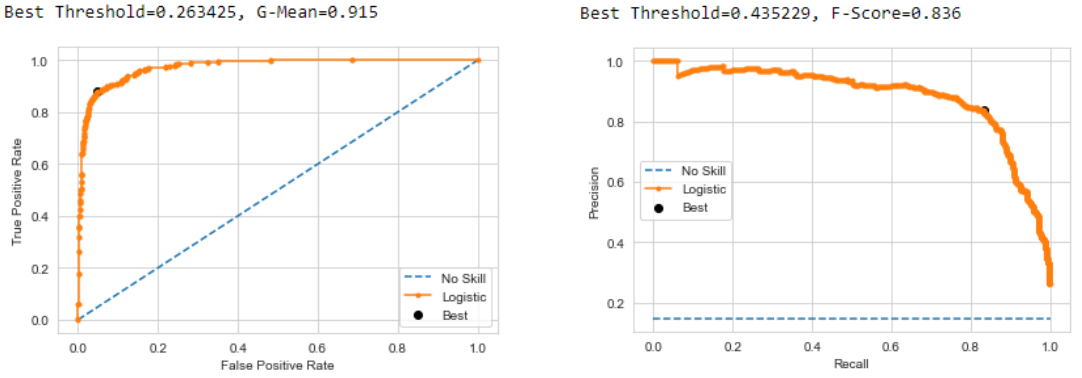
Random Forest



ROC F1-score

Figura 3.12: Punto de corte de Random Forest

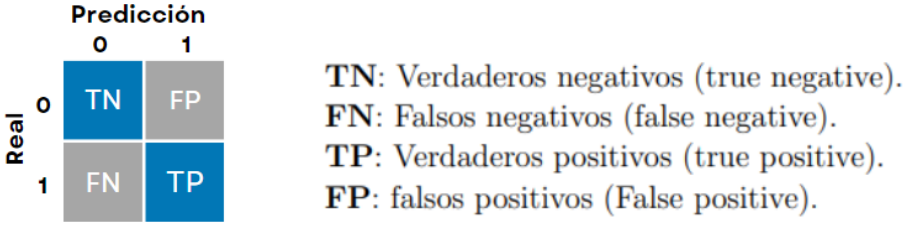
XGBoots



ROC F1-score

Figura 3.13: Punto de corte de XGBoots

Adicionalmente, se hizo uso de la matriz de confusión la cual permite analizar los resultados de cómo trabajan los moelos. Esta matriz se presenta en forma de tabla, de manera que relaciona las predicciones realizadas y los resultados correctos que debería haber mostrado el modelo.



TN: Verdaderos negativos (true negative).
FN: Falsos negativos (false negative).
TP: Verdaderos positivos (true positive).
FP: falsos positivos (False positive).

Figura 3.14: Matriz de confusión

- **Positivo o negativo**: se refiere a la predicción. Si el modelo predice 1 entonces será positivo, y se predice 0 será negativo.
- **Verdadero o falso**: se refiere si la predicción es correcta o no.

Las métricas son las que ayudan interpreta dicha matriz, estas pueden definirse como:

- **Precision**: podemos medir la calidad del modelo, que tan bien puede clasificar los verdaderos positivos.

$$\frac{TP}{TP + FP} \tag{3.1}$$

- **Recall**: informa sobre la cantidad que el modelo es capaz de identificar los verdaderos negativos.

$$\frac{TP}{TP + FN} \tag{3.2}$$

- **Accuracy**: mide el porcentaje de casos que el modelo ha acertado, es decir, la fuerza que tiene el modelo para predecir.

$$\frac{TP + TN}{TN + TP + FN + FP} \tag{3.3}$$

Para los tres modelos se seleccionó el punto óptimo que brinda el precision-recall, ya que, presenta de forma más homogénea las métricas por modelo y cada dato de la variable objetivo, esta afirmación se confirma en el siguiente apartado.

3.4 Análisis de resultados

La curva ROC, la curva de precisión-recallbtenidas en la sección anterior y la matriz de confusión dan el sustento técnico para comparar los modelos, sin embargo, presentar solo estos resultados no es suficiente. Uno de los mayores retos en minería de datos es presentar los resultados de un modelo, por lo tanto, también se hizo uso de métricas para comprobar y comparar el desempeño de los modelos permitiendo un mayor entendimiento de los resultados.

Despues de aplicar los respectivos modelos regresión logística, random forest y xgboost a la base de entrenamiento, y se validaron con base a los datos de prueba, de esta manera se obtuvieron los siguientes resultados:

	Regresión Logistica		Random Forest		XGBoots	
	0	1	0	1	0	1
Precision	93 %	55 %	97 %	81 %	97 %	84 %
Recall	92 %	57 %	97 %	82 %	97 %	83 %
F1-score	92 %	56 %	97 %	81 %	97 %	84 %
Accuracy		87 %		94 %		95 %

El modelo de regresión logística presenta una exactitud del 87 % , para random forest del 94 %, mientras que el accuracy del modelo xgboost es de 95 %. Al realizar la comparación entre las otras métricas se observa que al modelo de regresión logística se le dificultad la clasificación de los verdaderos positivos, caso que no ocurre en los otros dos.

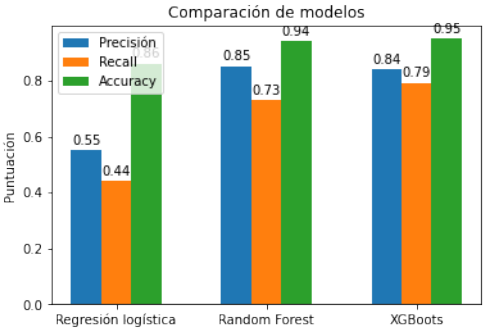


Figura 3.15: Comparación de modelos

Al comparar los modelos usando las curvas y las métricas que se utilizan para conocer el rendimiento global de las pruebas realizadas se obtuvo que el modelo de mayor rendimiento ha sido el modelo xgboost, pues bien, el valor de las métricas es el más alto respecto a los otros modelos, mostrando la relación entre los valores verdaderos negativos y verdaderos positivos , esto se puede observar en la Figura:3.15.

Matrix de Confusión

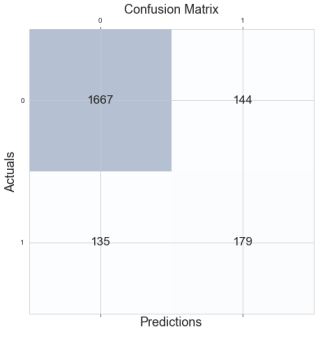


Figura 3.16: Regresión logística

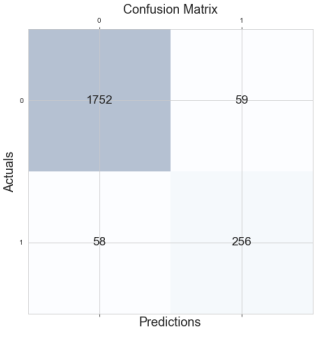


Figura 3.17: Random Forest

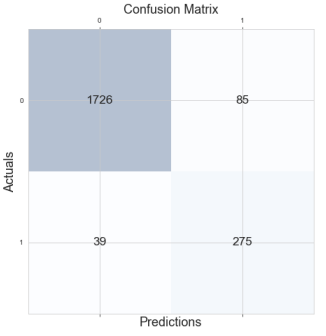


Figura 3.18: XGBoots

Finalmente, los resultados obtenidos por medio de las metricas se pueden ver reflejados en la matriz de confusión de cada modelo. Para nuestro análisis fue importante que el modelo seleccionado (xgboost) fuera mas preciso a la hora de clasificar los verdaderos positivos,es decir, los clientes morosos ya que nuestro objetivo principal es detectar estos clientes propensos a la deserción para generar un plan de fidelidad y evitar este suceso. Por otro lado, el modelo de regresión logistica genera una clasificación de verdaderos negativos bastante alta, lo cual podría traer como consecuencia un análisis del cliente innecesario y costoso para la entidad.

Capítulo 4

Conclusiones

De acuerdo con los resultados obtenidos, se concluye que:

- En este estudio, se compararon tres modelos de aprendizaje automático: xgboost, random forest y regresión logística. Los resultados mostraron que XGBoost destacó como el mejor modelo en términos de desempeño. Este resultado se fundamenta en la evaluación de diversas métricas, como precision, recall y accuracy.

- Al comparar directamente los modelos random forest y xgboost, se encontró que random forest logró una precisión ligeramente superior, superando al modelo xgboost en un 1 %. Aunque esta diferencia puede parecer pequeña, es relevante en casos donde la precisión es crucial para la toma de decisiones. Sin embargo, cabe destacar que xgboost presentó mejor relación al clasificar los verdaderos positivos y verdaderos negativos.

- Un paso fundamental para la implementación exitosa de los modelos de regresión logística y xgboost fue la codificación adecuada de las variables. La transformación y codificación de los datos permitieron que estos algoritmos pudieran trabajar con las características del conjunto de datos de manera efectiva. Esta etapa de procesamiento garantizó una interpretación correcta y coherente de las características, contribuyendo al rendimiento óptimo de los modelos.

- Durante el análisis, se encontraron algunos registros con valores faltantes en el conjunto de datos. Sin embargo, se determinó que estos valores no tenían un impacto significativo en el resultado final del estudio y su presencia no afectaba la validez de las conclusiones. Por lo tanto, se optó por eliminar estos registros vacíos, lo que simplificó el proceso de análisis y evitó la necesidad de imputar valores.

Apéndice A

Códigos

A continuación anexamos el código utilizado para el desarrollo del proyecto.

A.1 Código en Python

```
import numpy as np #operaciones matriciales y con vectores
import pandas as pd #tratamiento de datos
# Y otras librerias de sklearn

#Lectura y cambio de idioma
df=pd.read_excel('Base de datos.xlsx')
df=df.rename(columns={'idade':'edad','dependentes':'personas_a_cargo',
'escolaridade':'escolaridad','tipo_cartao':'tipo_tarjeta',
'meses_de_relacionamento':'meses_con_tarjeta','qtd_produtos':'num_productos',
'iteracoes_12m':'iteraciones_12m','meses_inativo_12m':'meses_inactivos_12m',
'limite_credito':'valor_disponible','valor_transacoes_12m':'valor_transacion_12m',
'qtd_transacoes_12m':'num_transaciones_12m' })
df.escolaridad=df.escolaridad.replace('mestrado', 'maestría')
.replace('ensino medio', 'escuela secundaria')
.replace('sem educacao formal', 'sin educación formal')
.replace('graduacao', 'graduado')
.replace('doutorado', 'doctorado')
df.estado_civil=df.estado_civil.replace('solteiro', 'soltero')

# Analisis descriptivo
df[["default"]].value_counts().plot.bar()
display(round(df.isnull().sum() / df.shape[0] * 100, 2)
.sort_values(ascending=False))

fig, axes = plt.subplots(16, 2,figsize=(13,80))
var = ['edad', 'sexo', 'personas_a_cargo', 'escolaridad',
       'estado_civil_mod', 'salario_anual', 'tipo_tarjeta_mod',
       'meses_con_tarjeta',
       'num_productos', 'iteraciones_12m', 'meses_inactivos_12m',
```

```
        'valor_disponible', 'valor_transacion_12m', 'num_transaciones_12m',
        'estado_civil_mod', 'tipo_tarjeta_mod']
for i in range (len(var)):
  sns.set_style('whitegrid')
  ax = sns.kdeplot(data=df, x=var[i], hue='default',
  shade=True, common_norm=False, ax=axes[i,0])
  sns.set_style('whitegrid')
  ax= sns.boxplot(x='default',y=var[i],data=df,ax=axes[i,1])
  ax = sns.stripplot(x="default", y=var[i],data=df,ax=axes[i,1])


  #Codificación
  df.sexo=df.sexo.replace('M', 1).replace('F', 0)
  #salario anual
df.salario_anual=df.salario_anual.replace('$120K +', 5)
.replace('$60K - $80K', 3).replace('$80K - $120K', 4)
.replace('$40K - $60K', 2).replace('menos que $40K', 1)
#escolaridad con cod ordinaria
df.escolaridad=df.escolaridad.replace('doctorado', 5).replace('graduado', 3)
.replace('maestría', 4).replace('escuela secundaria', 2)
.replace('sin educación formal', 1)
#Estado civil - Tipo de tarjeta
frecuencias = df.estado_civil.value_counts(normalize=True)
frecuencias2 = df.tipo_tarjeta.value_counts(normalize=True)
# Asignamos un valor numérico a cada categoría basado en su frecuencia
codificacion_frecuencia = {categoria: i for i, categoria in
enumerate(frecuencias.index)}
codificacion_frecuencia2 = {categoria: i for i, categoria in
enumerate(frecuencias2.index)}
# Creamos una nueva columna 'modificado' que contiene la codificación basada en frec
df['estado_civil_mod'] = df['estado_civil'].map(codificacion_frecuencia)
df['tipo_tarjeta_mod'] = df['tipo_tarjeta'].map(codificacion_frecuencia2)

df_vacios=df.dropna()

## MODELOS

#Regresion Logistica
#Variables dependientes
X = df_vacios[['id', 'edad', 'sexo', 'personas_a_cargo', 'escolaridad',
'estado_civil_mod', 'salario_anual', 'tipo_tarjeta_mod',
'meses_con_tarjeta', 'num_productos', 'iteraciones_12m',
'meses_inactivos_12m',valor_disponible', 'valor_transacion_12m',
'num_transaciones_12m']]
#Variable objetivo
```

```
y = df_vacios['default']
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    train_size   = 0.7,
                                                    random_state = 37)



# Creación del modelo utilizando matrices como en scikitlearn
X_train = sm.add_constant(X_train, prepend=True)
modelo = sm.Logit(endog=y_train, exog=X_train,)
modelo = modelo.fit()
#print(modelo.summary())


# ================================================================================
X_test = sm.add_constant(X_test, prepend=True)
predicciones = modelo.predict(exog = X_test)
clasificacion = np.where(predicciones<0.5, 0, 1)
accuracy = accuracy_score(
            y_true    = y_test,
            y_pred    = clasificacion,
            normalize = True
           )


#Punto optimo con ROC
yhat = modelo.predict(X_test)
# yhat = yhat[:, 1]
# calculate roc curves
fpr, tpr, thresholds = roc_curve(y_test, yhat)

# calculate the g-mean for each threshold
gmeans = np.sqrt(tpr * (1-fpr))
# locate the index of the largest g-mean
ix = np.argmax(gmeans)
print('Best Threshold=%f, G-Mean=%.3f' % (thresholds[ix], gmeans[ix]))
# plot the roc curve for the model
pyplot.plot([0,1], [0,1], linestyle='--', label='No Skill')
pyplot.plot(fpr, tpr, marker='.', label='Logistic')
pyplot.scatter(fpr[ix], tpr[ix], marker='o', color='black', label='Best')
# axis labels
pyplot.xlabel('False Positive Rate')
pyplot.ylabel('True Positive Rate')
pyplot.legend()
# show the plot
```

```
pyplot.show()

#Punto optimo con F1-SCORE
# Calcular la precisión, recall y umbrales
# utilizando la curva de precisión-recall
precision, recall, thresholds = precision_recall_curve(y_test, yhat)

# Calcular el F1-score para cada par (precisión, recall)
f1_scores = 2 * (precision * recall) / (precision + recall)

# Encontrar el umbral óptimo que maximiza el F1-score
optimal_threshold = thresholds[np.argmax(f1_scores)]

# Calcular las predicciones utilizando el umbral óptimo
y_pred_optimal = np.where(yhat >= optimal_threshold, 1, 0)

# Calcular el F1-score utilizando el umbral óptimo
f1_optimal = f1_score(y_test, y_pred_optimal)

# Graficar la curva de precisión-recall
plt.figure(figsize=(8, 6))
plt.plot(recall, precision, color='b', lw=2, label='Curva de Precisión-Recall')
plt.xlabel('Recall')
plt.ylabel('Precisión')
plt.title('Curva de Precisión-Recall')
plt.legend(loc='lower left')
plt.grid(True)
plt.show()

# Imprimir el umbral óptimo y el F1-score correspondiente
print("Umbral óptimo:", optimal_threshold)
print("F1-score óptimo:", f1_optimal)

#Comparación de puntos
puntos_corte=[0.155196,0.39629]
for i in puntos_corte:
    predicciones=np.where(yhat >= i, 1, 0)
    print("Métricas con punto de corte:",i)


    f1 = f1_score(y_test, predicciones)
    print('f1 score: ',f1)


    # Calcular la precisión (accuracy) del modelo
```

```
165     accuracy = accuracy_score( y_test, predicciones )
166     print("Accuracy:", accuracy)
167 
168     # Calcular la precisión (precision) del modelo
169     precision = precision_score(y_test, predicciones)
170     print('Precisión del modelo:',precision)
171 
172     sensibilidad = recall_score(y_test, clasificacion)
173     print('Sensibilidad del modelo:',sensibilidad)
174 
175 
176     print(
177     classification_report(
178         y_true = y_test,
179         y_pred = predicciones
180     ))
181     # Calculate the confusion matrix
182     #
183     conf_matrix = confusion_matrix(y_true=y_test, y_pred=predicciones)
184     #
185     # Print the confusion matrix using Matplotlib
186     #
187     fig, ax = plt.subplots(figsize=(7.5, 7.5))
188     ax.matshow(conf_matrix, cmap=plt.cm.Blues, alpha=0.3)
189     for i in range(conf_matrix.shape[0]):
190         for j in range(conf_matrix.shape[1]):
191             ax.text(x=j, y=i,s=conf_matrix[i, j], va='center',
192             ha='center', size='xx-large')
193 
194     plt.xlabel('Predictions', fontsize=18)
195     plt.ylabel('Actuals', fontsize=18)
196     plt.title('Confusion Matrix', fontsize=18)
197     plt.show()
198     print("\n")
199 
200 
201 #Random forest
202 # Definir los hiperparámetros a probar con distribuciones de probabilidad
203 param_dist = {
204     'n_estimators': randint(100, 600),  # Número de árboles en el bosque
205     'max_depth': [None, 5, 10],  # Profundidad máxima de cada árbol
206     'min_samples_split': randint(2, 10),
207     # Número mínimo de muestras requeridas para dividir un nodo interno
208     'min_samples_leaf': randint(1, 5)
209     # Número mínimo de muestras requeridas en cada hoja del árbol
```

```
}

# Crear el clasificador de Random Forest
rf_classifier = RandomForestClassifier(random_state=42)

# Definir la función de puntuación (score) como el F1-score
scorer = make_scorer(roc_auc_score)

#Realizar la búsqueda aleatoria de los mejores hiperparámetros por RandomizedSearchC
random_search = RandomizedSearchCV(estimator=rf_classifier,
param_distributions=param_dist, scoring=scorer, cv=3, n_iter=10)
random_search.fit(X_train, y_train)

# Obtener los mejores hiperparámetros y el mejor resultado
best_params = random_search.best_params_
best_score = random_search.best_score_

print("Mejores hiperparámetros encontrados:", best_params)

# Entrenar el modelo con los mejores hiperparámetros
# utilizando todos los datos de entrenamiento
best_rf_classifier = RandomForestClassifier(random_state=42, **best_params)
best_rf_classifier.fit(X_train, y_train)

# Se hace el mismo proceso que en regresión logistica, se buscan los puntos
# y se comparan,con la misma estructura de codigo,
# puntos optimos generados.

# XGBoots
from scipy.stats import randint

# Definir los hiperparámetros a probar con distribuciones de probabilidad
param_dist = {
    'n_estimators': randint(100, 500),  # Número de árboles
    'max_depth': randint(3, 10),  # Profundidad máxima del árbol
    'learning_rate': [0.01, 0.1, 0.2],  # Tasa de aprendizaje
    'subsample': [0.6, 0.8, 1.0],  # Submuestra de las instancias
    'colsample_bytree': [0.6, 0.8, 1.0],  # Submuestra de las columnas
    'gamma': [0, 1, 5]  # Parámetro de regularización gamma
}

# Crear el clasificador XGBoost
xgb_classifier = xgb.XGBClassifier(random_state=42)

# Definir la función de puntuación (score) como el ROC AUC
```

```
255 scorer = make_scorer(roc_auc_score)
256
257 random_search = RandomizedSearchCV(estimator=xgb_classifier, param_distributions=par
258 random_search.fit(X_train, y_train)
259
260 # Obtener los mejores hiperparámetros y el mejor resultado
261 best_params = random_search.best_params_
262 best_score = random_search.best_score_
263 print("Mejores hiperparámetros encontrados:", best_params)
264
265 best_xgb_classifier = xgb.XGBClassifier(random_state=42, **best_params)
266 best_xgb_classifier.fit(X_train, y_train)
267
268 #Ocurre igual que random forest.
269
270 #Comparacion modelos
271 import matplotlib.pyplot as plt
272
273 # Datos de ejemplo
274 modelos = ['Regresión logística', 'Random Forest', 'XGBoots']
275 precision = [0.55, 0.85, 0.84]
276 recall = [0.44, 0.73, 0.79]
277 accuracy = [0.86, 0.94, 0.95]
278
279 # Configuración del gráfico
280 fig, ax = plt.subplots()
281 ind = range(len(modelos))
282 width = 0.2
283
284 # Graficar barras para la precisión, recuperación y exactitud
285 rects1 = ax.bar(ind, precision, width, label='Precisión')
286 rects2 = ax.bar([i + width for i in ind], recall, width, label='Recall')
287 rects3 = ax.bar([i + 2 * width for i in ind], accuracy, width, label='Accuracy')
288
289 # Configuración de ejes y etiquetas
290 ax.set_ylabel('Puntuación')
291 ax.set_title('Comparación de modelos')
292 ax.set_xticks([i + width for i in ind])
293 ax.set_xticklabels(modelos)
294 ax.legend()
295
296 # Añadir etiquetas de valores a las barras
297 def autolabel(rects):
298     for rect in rects:
299         height = rect.get_height()
```

```
300         ax.annotate('{}'.format(height),
301                     xy=(rect.get_x() + rect.get_width() / 2, height),
302                     xytext=(0, 3),  # 3 puntos de desplazamiento vertical
303                     textcoords="offset points",
304                     ha='center', va='bottom')
305
306 autolabel(rects1)
307 autolabel(rects2)
308 autolabel(rects3)
309
310 plt.show()
311
```

Repositorio de Github[14]

Bibliografía

Libros

[11] David W Hosmer Jr, Stanley Lemeshow y Rodney X Sturdivant. *Applied logistic regression*. Vol. 398. John Wiley & Sons, 2013.

[18] Peter McCullagh. *Generalized linear models*. Routledge, 2019.

Artículos

[2] R Araya. «nduction of Decision Trees when Examples are Describedwith Noisy Measurements and with Fuzzy Class Membership». En: (1994).

[3] Leo Breiman. «Random forests». En: *Machine learning* 45.1 (2001), págs. 5-32.

[4] Fulgencio Cánovas-Garcıa et al. «Modification of the random forest algorithm to avoid statistical dependence problems when classifying remote sensing imagery». En: *Computers & Geosciences* 103 (2017), págs. 1-11.

[5] Luis Ángel Meneses Cerón y Ronald Alejandro Macuacé Otero. «Valoración y riesgo crediticio en Colombia». En: *Revista Finanzas y Polıtica Económica* 3.2 (2011), págs. 65-82.

[7] Andrés Felipe Echeverri Giraldo. «Modelo predictivo de Churn de clientes para el negocio de Telecomunicaciones». En: (2019).

[8] Javier Jesús Espinosa-Zúñiga. «Aplicación de algoritmos Random Forest y XGBoost en una base de solicitudes de tarjetas de crédito». En: *Ingenierıa, investigación y tecnologıa* 21.3 (2020).

[9] Jerome H Friedman. «Greedy function approximation: a gradient boosting machine». En: *Annals of statistics* (2001), págs. 1189-1232.

[10] Jobany J Heredia, Aida G Rodrıguez y José A Vilalta. «Predicción del rendimiento en una asignatura empleando la regresión logıstica ordinal». En: *Estudios pedagógicos (Valdivia)* 40.1 (2014), págs. 145-162.

[15] Wei Li et al. «Gene expression value prediction based on XGBoost algorithm». En: *Frontiers in genetics* 10 (2019), pág. 1077.

[17] Tamahı Constanza Martınez Fernández et al. «Comparación de modelos machine learning aplicados al riesgo de crédito.» En: (2022).

[20] Maria Jesús Mures Quintana, Ana Garcia Gallego y Maria Eva Vallejo Pascual. «Aplicación del análisis discriminante y regresión logistica en el estudio de la morosidad en las entidades financieras. Comparación de resultados». En: *Pecunia: revista de la Facultad de Ciencias Económicas y Empresariales* 1 (2005), págs. 175-199.

Tesis

[16] Marina Dominguez Martin. «Regresión Logistica y Técnicas de Aprendizaje. Aplicaciones.» B.S. thesis. 2021.

[19] Edith Andrea Pérez Tatamués. «Algoritmo de random forest aplicado a la detección de fraude en el sistema bancario ecuatoriano». B.S. thesis. 2019.

Bases de datos

[1] URL: https://aws.amazon.com/es/what-is/logistic-regression/.

[12] IBM. *Random forest*. URL: https://bookdown.org/content/2031/ensambladores-random-forest-parte-i.html.

[13] IBM. *Regresión ordinal*. URL: www.ibm.com/docs/es/spss-statistics/25.0.0?topic=SSLVMB_25.0.0/spss/advanced/idh_plum.htm.

[14] Korin Alexa Idarraga. *Repositorio-Proyecto de grado*. URL: https://github.com/korinidarraga9/Proyecto-de-Grado.