Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN CIENCIA E ENXEÑARÍA DE DATOS

Técnicas de tratamiento de datos faltantes y aplicación en problema de detección de fraude bancario

Estudante: Alexandre Zas Pérez

Dirección: José Jorge García Romarís

Manuel Oviedo de la Fuente

Rubén Fernádez Casal

A Coruña, junio de 2023.

Resumen

En este documento se realiza una revisión de las principales técnicas de tratamiento de valores faltantes para aplicarlas en un problema de detección de transacciones fraudulentas. Se realiza un análisis inicial de algunas de las 134 variables iniciales y se realiza un preprocesado de los datos incluyendo una selección de variables, creación de variables e imputación de datos faltantes con distintas técnicas. Los datos son divididos en transacciones de banca electrónica y de banca móvil. Para el primer conjunto de datos los mejores resultados se obtienen con el algoritmo XGBoost aplicando missForest como técnica de imputación; mientras que para el segundo conjunto el mejor modelo es un CatBoost con missForest como técnica de imputación.

Abstract

In this document, a review of the main techniques for handling missing values is conducted in order to apply them to a fraudulent transaction detection problem. An initial analysis is performed on some of the 134 initial variables, and data preprocessing is carried out, including variable selection, variable creation, and imputation of missing data using different techniques. The data is divided into electronic banking transactions and mobile banking transactions. For the first dataset, the best results are obtained with the XGBoost algorithm using missForest as the imputation technique, while for the second dataset, the best model is a CatBoost with missForest as the imputation technique.

Palabras clave:

- Datos faltantes
- Imputación de datos
- Transacciones fraudulentas
- XGBoost
- CatBoost
- MissForest

Keywords:

- Missing data
- Data imputation
- Fraudulent transactions
- XGBoost
- CatBoost
- MissForest

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introdución

Este trabajo se realiza en torno a las prácticas realizadas en el departamento de Data, Analytics y Rtech de Abanca. El objetivo es realizar una revisión de las principales técnicas de datos faltantes al mismo tiempo que se aplican en un problema de detección de transacciones bancarias fraudulentas.

Con la aparición de las nuevas tecnologías y el cambio de hábitos debido a la pandemia cada vez es menos común el uso de dinero en efectivo, lo que supone un auge de las operaciones con tarjeta, bizum o transferencias entre otras. Según el estudio de los medios de pago del Banco de España [1] el 36% de las personas usan bizum, el 37% transferencias, el 86% tarjetas y el 99% dinero en efectivo. Según este mismo estudio el dinero en efectivo se usa más habitualmente en operaciones de importes reducidos, mientras que el importe medio de las transferencias es superior.

De todas las operaciones realizadas de forma electrónica, hay un pequeño porcentaje de operaciones que son fraudulentas y es necesario detectarlas para no incurrir en costes inncecesarios. Para esto es necesario tener un modelo capaz de detectar el fraude intentando reducir los falsos negativos y falsos positivos. Un falso positivo sería una operación que el modelo indica como fraudulenta siendo una operación no fraudulenta, tendría el coste asociado de que un cliente ve su operación cancelada y llamará al banco donde tendrá que atenderlo una persona. En cambio, un falso negativo será una operación fraudulenta que no es detectada como tal; su coste asociado sería el coste de la operación que caería en el cliente o el banco. En ambos casos tendríamos un cliente insatisfecho.

En este problema nos vamos a centrar en las operaciones realizadas con banca electrónica (transacciones a través de la web de Abanca) y banca móvil (transacciones a través de la aplicación de Abanca). Las operaciones que se pueden realizar son transferencias, bizums o creación de tarjetas virtuales. En nuestro conjunto de datos contamos con 973000 observaciones de las cuales 1031 son fraudulentas. Separaremos las observaciones en dos conjuntos de datos en función de sí fueron realizadas en banca electrónica (32.15%) o mediante la banca

móvil (67.85%).

En ambos conjuntos de datos contamos con más de 25 variables con valores faltantes, algunas de ellas superando el 50% de observaciones nulas; por lo que será necesario realizar un correcto tratamiento de estas variables.

En el segundo capítulo nos introducimos en los valores faltantes, explicando diversas técnicas para el tratamiento de la no respuesta y métodos de imputación. También explicaremos algunas librerías de Python que implementan métodos de imputación de valores faltantes.

En el tercer capítulo describiremos el conjunto de datos, analizaremos algunas variables y se realizará el preprocesado de los datos incluyendo el tratamiento de los valores faltantes.

En el cuarto se describirá el método de trabajo, las métricas utilizadas, los algoritmos de clasificación usados (Regresión Logística, XGBoost y CatBoost), presentaremos los resultados y los compararemos.

En el quinto y último capítulo se incluyen las conclusiones y posibles trabajos a futuro para seguir avanzando en este problema.

Capítulo 2

Tratamiento de datos faltantes

Este capítulo tratará la problemática de tener datos faltantes en un problema de ciencia de datos, la teoría de datos y distintas técnicas para tratar los datos faltantes.

2.1 Notación

Parte de la notación está basada en la presente en el libro Multivariate Imputation by Chained Equations in R de Van Buuren [2].

En primer lugar llamaremos a $X_j (j = 1, ..., k)$ una de las k variables del dataset, teniendo $\mathbf{X} = (X_1, ..., X_k)$ como el conjunto de las variables. El dataset tendrá un total de n observaciones, de las cuales m tendrán valores faltantes. Consideramos $Y_j (j = 1, ..., p)$ una de las $p \leq k$ variables con valores faltantes, por lo que $\mathbf{Y} = (Y_1, ..., Y_p)$. La parte de observada de Y_J será Y_j^{obs} mientras que Y_j^{mis} será la parte que contiene valores faltantes. La variable Y_j imputada se llamará $\hat{Y}_j$. Por último $Z_j (j = 1, ..., l)$ será una de las l variables sin valores faltantes ($m + l = k$), mientras que $\mathbf{Z} = (Z_1, ..., Z_l)$ será el conjunto de las variables sin datos faltantes.

2.2 Introdución a la problemática de datos faltantes

El análisis y explotación de datos son cada vez más comunes y necesarios en cualquier ámbito y empresa. Para ello es necesario tener unos datos completos y de calidad, pero eso no siempre es posible. Por ejemplo, en una encuesta las personas que cobran sueldos más altos podrían ser más propensos a no decir su salario. En preguntas sobre preferencias musicales, algunas personas podrían no ser capaces de dar una respuesta. También podría darse que se pierdan observaciones por el camino.

Los datos faltantes pueden aparecer en cualquier estudio estadístico de cualquier ámbito como estudios médicos, análisis de mercados o prevención de fraude. Hacer un tratamiento

de los datos faltantes es necesario pues muchos algoritmos de machine learning como por ejemplo xgboost [3] no permiten datos faltantes.

Un tratamiento incorrecto de los datos faltantes puede acarrear un estudio o modelo predictivo sesgado o erróneo. Por ejemplo, en el caso de una encuesta de diez preguntas siendo una de ellas acerca del salario. Las personas que tengan salarios más elevados podrían ser más dadas a no responder la cuestión salarial pero sí las otras preguntas. Si eliminamos las observaciones con algún valor faltante, tendríamos unos datos sesgados pues las personas con sueldos más altos no estarían incluidas.

2.3 Teoría de datos faltantes

Es muy habitual encontrarnos con conjuntos de datos incompletos o que tienen datos perdidos. En estos casos es deseable saber el por qué la ausencia de estos datos, para tomar la mejor decisión.

Rubin (1976) [4] propuso una clasificación en tres tipos de modelos de datos faltantes: MCAR, MAR y MNAR.

2.3.1 MCAR (missing completely at random)

MCAR es el más sencillo de los tres. Un conjunto de datos es MCAR si la probabilidad de que falte un dato es igual para todos los individuos y no depende de las medidas de otras variables, es decir, no existe ninguna relación entre que un dato sea faltante u observado.

Considerando un ejemplo con una única variable con valores faltantes llamada Y_1 y Z siendo las variables restantes con todos los valores observados tenemos:

$$\mathcal{P}(Y_1 \text{ sea dato faltante}|Z, Y_1) = \mathcal{P}(Y_1 \text{ sea dato faltante}) \text{ [5]}$$

MCAR es la situación ideal ya que los datos faltantes serían tan solo un subconjunto aleatorio de los datos completos, aunque podría haber casos en los que perdamos todo el tamaño muestral.

Ejemplo 2.1 *Partiendo de una tabla de una base de datos, si ocurre un problema informático y se pierden algunos valores de algunas observaciones de forma aleatoria tendríamos pérdida MCAR.*

Para comprobar si un conjunto de datos tiene una pérdida MCAR podemos emplear el test de Little [6]. Este test se basa en la hipótesis nula de que los datos faltantes están distribuidos de forma aleatoria en relación con las variables observadas; por lo que no habría ningún factor oculto que influyera en la probabilidad de la presencia de un valor faltante. Este test nos devuelve un p-valor; con un nivel de significación del 5% aceptaremos que los datos tienen pérdida MCAR cuando el p-valor sea superior a 0.05.

2.3.2 MAR (missing at random)

Un conjunto de datos tiene pérdida MAR cuando la probabilidad de que una varibale tenga datos faltantes es independiente de los valores de la misma variable, pero dependiente de los valores de otras variables presentes en el conjunto de datos.

Utilizando la notación anterior:

$$\mathcal{P}(Y_1 \text{ sea dato faltante}|Z, Y_1) = \mathcal{P}(Y_1 \text{ sea dato faltante}|Z)$$

Ejemplo 2.2 *Partiendo de un estudio en el que tenemos las variables puesto de trabajo y sueldo, estaríamos con una pérdida MAR si la gente con determinados puestos de trabajo fuera más reacia a contestar a la pregunta del sueldo.*

2.3.3 MNAR (missing not at random)

Tenemos el caso de MNAR cuando la probabilidad de que una observación sea dato faltante es dependiente del valor de la propia variable. Su efecto no se puede ignorar ya que el valor faltante está relacionado con la razón por la que falta el dato.

Empleando la notación anterior:

$$\mathcal{P}(Y_1 \text{ sea dato faltante}|Z, Y_1) \neq \mathcal{P}(Y_1 \text{ sea dato faltante}|Z)$$

Ejemplo 2.3 *En un estudio acerca de malos hábitos alimenticios, si algunas personas con esos malos hábitos fueran menos propensas a contestar las preguntas.*

2.4 Métodos básicos para tratamiento de datos faltantes

En esta sección trataremos las técnicas básicas para tratamiento de datos faltantes. Estas técnicas consisten en quedarse con un subconjunto del conjunto de datos.

2.4.1 Análisis con datos completos (listwise)

Es un método muy habitual y sencillo de utilizar. Consiste en eliminar todas las observaciones que contengan algún valor faltante en alguna variable del conjunto de datos, es decir, para realizar el análisis estadístico solo se usarían las observaciones que disponen de todos los valores.

Ejemplo 2.4 *Partamos de un dataset X con $k = 3$ variables y $n = 100$ observaciones. Una variable tendrá valores faltantes $(p = 1)$ en $m = 10$ observaciones. Si realizamos un análisis con datos completos estaríamos eliminando los datos de las m observaciones con valores faltantes y nuestro dataset pasaría de $n = 100$ observaciones a $n - m = 90$ observaciones.*

Si la pérdida del conjunto de datos es MCAR, los resultados del análisis serán insesgados pues se trataría de una muestra aleatoria de los datos. La desventaja es que no es habitual la presencia de una pérdida MCAR y el análisis sería insesgado en el resto de los casos. Otra desventaja es que se puede perder mucha información, sobre todo si se tienen muchas variables con valores faltantes. Siguiendo con el ejemplo anterior, si tuviéramos un 10% de datos faltantes en cada una de las 3 variables podríamos llegar a eliminar hasta el 30% de las observaciones.

2.4.2 Análisis con datos disponibles

El análisis con datos disponibles se basa en emplear todos los datos disponibles en cada parte del análisis. Es decir, si estamos analizando primero un subconjunto de tres variables y luego otro subconjunto de cuatro variables; en el primer subconjunto usaremos todas las observaciones sin valores faltantes en esas tres variables y en el segundo análisis emplearemos todas las observaciones sin valores faltantes en esas cuatro variables.

La mayor desventaja de este enfoque es que podríamos estar usando diferentes tamaños muestrales para cada análisis. Esto puede presentar problemas, por ejemplo, en la selección de variables significativas para un modelo de regresión.

Es un buen método bajo el supuesto de MCAR. Comparando con el método anterior (análisis con datos completos) tenemos la ventaja de estar usando al menos la misma información y, en muchos casos más información para cada parte del análisis.

2.5 Métodos de imputación

Otra forma de tratar los valores faltantes es imputar el valor faltante por un valor. Se usa la información presente en los valores observados para establecer un valor en aquellos valores no observados. Es importante saber elegir bien el método de imputación ya que cada uno tiene sus ventajas e inconvenientes.

A la hora de hacer la imputación es importante mantener la consistencia de los datos. También es necesario mantener las distribuciones de las variables, así como sus correlaciones para evitar una distorsión de los datos.

2.5.1 Imputación por un valor central de posición

Incluimos aquellos métodos en los que imputamos todos los valores faltantes de una misma variable por un mismo valor. Este valor se puede conseguir con la media, mediana o moda entre otros métodos.

Imputación por la media

Se trata de uno de los métodos de imputación más usados debido a su facilidad de implementación e interpretabilidad. Sustituye los valores faltantes de cada variable por la media muestral de la propia variable.

Este método funciona bien bajo el supuesto de datos MCAR, pero no lo hace cuando la pérdida depende de otra variable. Esto es porque si se sustituyen los datos faltantes por la media, estaríamos reduciendo la varianza de cada variable y, por ende, también se modificarían las matrices de covarianza y de correlaciones.

Otra desventaja de este procedimiento es que tan solo es aplicable a variables cuantitativas y no a a variables cualitativas.

Ejemplo 2.5 *Sea Y_1 una variable con valores faltantes y codificando los los valores faltantes como na; partimos de $Y_1 = [1, 3, 4, na, na, 3, na, 1, 3, 13]$. Para imputar, calculamos la media de los valores presentes siendo esta $\bar{Y_1} = 4$ quedando la variable $\hat{Y_1} = [1, 3, 4, 4, 4, 3, na, 1, 3, 13]$ tras realizar la imputación.*

Imputación por la mediana

Se trata de un procedimiento similar al de la media, con la diferencia de que se sustituyen los valores de cada variable por la mediana de la propia variable. Su ventaja respecto a la imputación por la mediana es que es más robusta a la aparición de datos atípicos.

Al igual que la imputación por la media funciona bien bajo el supuesto MCAR, pero si no, estaríamos reduciendo mucho la varianza de cada variable y modificando las matrices de covarianza y correlaciones. También es aplicable sólo a variables cuantitativas.

Ejemplo 2.6 *Partamos del ejemplo anterior, siendo $Y_1 = [1, 3, 4, na, na, 3, na, 1, 3, 13]$. La mediana de esta variable es 3 e $\hat{Y_1} = [1, 3, 4, 3, 3, 3, 3, 1, 3, 13]$ sería la variable imputada.*

Podemos observar en los ejemplos 2.5 y 2.6 la diferencia con el método de la media ya que el valor imputado es 3 en vez de 4. Se observa la robustez del método de la mediana a los valores atípicos (en este caso la última observación con un valor de 13) a diferencia del método de la media.

Imputación por la moda

Procedimiento muy similar a los dos descritos anteriormente, con la diferencia de que se imputa con el valor más frecuente o moda.

Su ventaja respecto a los métodos anteriores es que se puede usar para imputar las variables cualitativas o categóricas. Por el contrario, no es recomendable usarlo para imputar

variables cuantitativas pues la media o mediana son una mejor representación de este tipo de variables.

Ejemplo 2.7 *Partiendo del mismo ejemplo anterior, se tiene* $Y_1 = [1, 3, 4, na, na, 3, na, 1, 3, 13]$. *El valor más frecuente es* 3 *e* $\hat{Y}_1 = [1, 3, 4, 3, 3, 3, 3, 1, 3, 13]$ *sería la forma de la variable imputada con la moda.*

2.5.2 Imputación mediante un modelo de regresión o estimación de media condicional

Este método consiste en realizar un modelo de regresión para imputar los valores faltantes con los valores predichos por el modelo de regresión.

Si se usan modelos paramétricos de regresión se pueden llegar a producir estimaciones sesgadas. Por ejemplo, si se usa un modelo lineal los valores imputados caerán en una línea recta o en el hiperplano, dependiendo de la cantidad de dimensiones. Otro problema es que la correlación entre los datos imputados es igual a 1, por lo que las correlaciones estarían sobreestimadas.

Sea Y_1 una variable con valores faltantes y X_1 y X_2 las otras dos variables del conjunto de datos, la ecuación para imputar Y_1 mediante regresión lineal es:

$$Y_1 = \beta_0 + \beta_1 X_1 + \beta_2 X_2$$

Una solución puede ser añadir un resíduo aleatorio a cada valor imputado (regresión estocástica). Así conseguimos mantener las correlaciones entre las variables y se podría reducir el sesgo de la imputación por regresión. El residuo aleatorio se suele generar a partir de una distribución normal. Otras opciones son usar modelos de regresión no paramétricos, que cada vez son más usados, o el remuestreo de los residuos.

Sea Y_1 una variable con valores faltantes, X_1 y X_2 las otras dos variables del conjunto de datos y $\mathcal{E}$ el residuo aleatorio que sigue una normal , la ecuación para imputar Y_1 mediante regresión estocástica es:

$$Y_1 = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \mathcal{E}$$

Los modelos de regresión lineal o de regresión estocástica tan solo sirven para variables numéricas. Si deseamos imputar una variable categórica podríamos usar la regresión logística. Se trata de un método paramétrico al igual que los dos métodos anteriores. Asume relación lineal entre las variables independientes y la variable categórica a imputar a partir de una función de enlace. Para cada variable categórica a imputar se ajusta un modelo con las variables restantes, a continuación, se predice sobre las observaciones con valores faltantes en la variable a imputar y se les asigna la categoría con mayor probabilidad devuelta por el modelo.

2.5.3 Imputación cold deck

Este método consiste en imputar un valor faltante mediante valores externos al conjunto de datos con el que se está trabajando. Un caso de uso es en el ámbito de las encuestas, se pueden usar encuestas anteriores que tuvieran preguntas similares para imputar valores faltantes en encuestas actuales. En otros ámbitos se podrían usar datos históricos para realizar la imputación de los valores faltantes.

2.5.4 Imputación hot deck

La imputación hot deck es un tipo de imputación no parámetrica [7] muy usada en encuestas debido a que ayuda a reducir el sesgo provocado por la no respuesta. Este procedimiento hace uso del valor de otras observaciones con características similares para imputar los valores faltantes. Tenemos varios tipos de imputación hot deck.

Hot deck aleatorio

Se imputa haciendo uso de valores aleatorios de la muestra. Usando este método tenemos la ventaja de que sirve tanto para variables categóricas como numéricas, también que se preserva la distribución de cada variable. La desventaja es que podría incurrir en sesgos en las estimaciones de las correlaciones de las variables.

Ejemplo 2.8 *Partamos de un conjunto de datos* $Y_1 = [1, 4, 8, na, 6]$ *; aleatoriamente se elegiría un valor para imputar el único valor faltante. Supongamos que sale elegido el segundo valor, nuestro conjunto de datos imputado quedaría con la forma* $\hat{Y}_1 = [1, 4, 8, 4, 6]$

Hot deck estratificado

En este método las observaciones se dividen en grupos teniendo en cuenta alguna característica común de las observaciones. Dentro de cada grupo se elige un valor aleatorio al igual que en el método anterior.

Ejemplo 2.9 *En un conjunto de datos con las variables sexo y edad, tenemos la primera variable completa y la segunda con valores faltantes; podríamos dividir los datos en hombres y mujeres para realizar el procedimiento descrito en el método anterior.*

Hot deck basado en vecinos

En este método se escogen las observaciones más cercanas mediante una medida de distancia (Euclídea, Mahalanobis, ...). Además de la selección del tipo de distancia dependiendo del tipo de problema, otro parámetro de interés es el número de vecinos. Mediante los valores

de estos vecinos se obtiene el valor imputado. Si queremos imputar variables numéricas podemos emplear la media o mediana de los valores de los k vecinos elegidos; mientras que si queremos imputar variables categóricas se puede emplear la moda. Otra opción para imputar variables categóricas sería emplear tan solo el vecino más cercano. Si usamos más vecinos tenemos la ventaja de que estamos usando más información y estaríamos incluyendo menos sesgo.

Hot deck secuencial

Su uso está pensado para datos temporales ya que consiste en imputar la observación faltante con su observación anterior.

Ejemplo 2.10 *Si la variable* $Y_1 = [1, 5, na, 8, 2, na, 6]$ *es una serie temporal y realizamos esta imputación los valores anterior a los valores faltantes serán* 5 *y* 2 *respectivamente, quedando la variable imputada como* $\hat{Y}_1 = [1, 5, 5, 8, 2, 2, 6]$.

El principal problema de este método es la posible presencia de múltiples valores faltantes consecutivos. Por ejemplo, en un supuesto de toma del valor de la temperatura cada hora, si tenemos ocho valores faltantes consecutivos estaríamos imputando el último valor faltante con una observación que es ocho horas anterior. Una solución a este problema podría ser usar el ajuste de una serie de tiempo.

2.5.5 Métodos basados en árboles

Lo más habitual con los métodos basados en árboles es usarlos como algoritmos de clasificación y regresión, pero también pueden ser usados para la imputación de datos faltantes. Se tratan de métodos no paramétricos que permiten tanto la imputación de variables numéricas como variables categóricas.

Estos métodos emplean la estructura del árbol para realizar estimaciones de los valores faltantes. Explicaremos el principal algoritmo que es MissForest, propuesto por por Stkhoven y Bühlmann [8]. Como se dice en el artículo citado, algunos datasets pueden tener interacciones complejas entre las variables y estructuras de relaciones no lineales que no pueden ser capturadas con métodos paramétricos. Algunos de los parámetros son la profundidad máxima de los árboles o el número de iteraciones del algoritmo.

El algoritmo es el siguiente:

1. Se crean varios conjuntos de datos completos a partir del dataset original. A los valores faltantes de dichos conjuntos se les asignan unos valores iniciales. Esta asignación inicial variará en diferentes ejecuciones del algoritmo ya que se están empleando métodos boostrap.

2. Para cada conjunto de datos se construye un modelo de regresión o clasificación basado en árboles de decisión y lo usamos para predecir los valores faltantes
3. Se realiza la imputación de los valores faltantes con las predicciones del paso anterior.
4. Se repiten los pasos 2 y 3 hasta obtener una convergencia.

Una de sus principales ventajas es la imputación simultánea de diferentes variables, además de tener la capacidad de imputar variables categóricas. También destaca su flexibilidad y escalabilidad ya que podemos usar este algoritmo en conjuntos de datos pequeños y grandes obteniendo una buena eficiencia computacional.

2.5.6 Imputación múltiple

La imputación múltiple [9] se caracteriza por devolver más de un valor para cada valor faltante. Cada uno de los valores faltantes se imputan m veces, obteniendo m conjuntos de datos completos. Estos múltiples valores se combinan para obtener los valores imputados. Para combinar estos valores se puede usar la media o mediana en el caso de variables numéricas, mientras que para variables categóricas podemos emplear la moda. También se podría escoger uno de los valores de forma aleatoria.

Al generar varias imputaciones por cada valor faltante y combinarlas, estamos realizando estimaciones más precisas y menos sesgadas de los valores faltantes. Otra ventaja es su posible aplicación tanto en variables numéricas como variables categóricas. Sus desventajas podrían ser el coste computacional de realizar varias imputaciones para cada valor faltante y el hecho de elegir un criterio para combinar estos valores.

2.5.7 Otros métodos de imputación

Existen una gran cantidad de métodos de los cuales no hemos hablado en secciones anteriores. Un método habitual de imputación es la estimación por máxima verosimilitud [10] que trata de estimar los parámetros del modelo y los valores faltantes de forma simultánea, maximizando la verosimilitud de la muestra. Por verosimilitud entendemos como la medida de probabilidad de que los datos observados hayan sido generados por el modelo propuesto.

Otros métodos de imputación existentes son la imputación por modelo de Markov oculto (HMM) o la imputación basada en el remuestreo con boostrap.

2.6 Librerías de Python para imputación

En el lenguaje de programación Python existen una serie de librerías con diferentes funciones para poder imputar valores faltantes. Algunas de las librerías principales se explican a continuación.

2.6.1 Scikit-learn

Scikit-learn o sklearn [11] se trata de una de las librerías más utilizadas tanto para algoritmos de clasificación o regresión como para imputación de valores faltantes. Esta librería cuenta con la función *SimpleImputer* que permite imputar mediante la media, mediana o moda según lo indique usuario. También se puede imputar usando el método del vecino más cercano mediante el *KNNImputer* que por defecto tendrá en cuenta los cinco vecinos más cercanos.

Otra funcionalidad de esta librería es el *IterativeImpute*. Para cada variable con valores faltantes realiza un modelo predictivo con las variables restantes y así poder estimar valores de los valores faltantes de la variable elegida; realiza este proceso de forma iterativa sobre cada columna hasta alcanzar convergencia o el número máximo de iteraciones. Esta función permite escoger entre modelos como *RandomForestRegressor*, *BayesianRidge* o *ExtraTreesRegressor* para estimar los valores faltantes.

2.6.2 Autoimpute

Autoimpute [12] es un paquete centrado en la imputación. Incluye distintos métodos univariantes ya mencionados como la imputación con la media, mediana o moda. También contiene métodos multivariantes como regresión lineal, estocástica o logística.

Su principal funcionalidad es la función *Mice* que permite la imputación múltiple original; pero también permite la elección de diferentes estrategias de imputación para cada variable mediante el parámetro *strategy*. Esta función también permite elegir que variables predictoras se van a utilizar para cada variable a imputar mediante el parámetro *predictors*.

Si estamos con series de tiempo, este paquete también permite realizar interpolaciones lineal, cuadrática o cúbica entre otras.

2.6.3 Impyte

Impyte [13] es otro paquete pensado para imputar valores faltantes en Python. Con el parámetro *estimator* de la función *impute* podemos elegir entre distintos estimadores para realizar la imputación. Estos estimadores incluyen Random Forest, vecino más cercano o KNN, árboles de decisión, Gradient Boosting o Bayes entre otros.

2.6.4 Missingpy

Missingpy [14] se trata de una librería diseñada para la imputación de valores faltantes en Python. Actualmente recoje dos métodos de imputación: el vecino más cercano (al igual que la librería anterior) e imputación con missForest que emplearemos en el siguiente capítulo.

Para la imputación con missForest emplea su función *MissForest*, cuyos parámetros principales son:

- Número máximo de iteraciones que realizará el algoritmo.
- Profundidad máxima de los árboles
- Criterios de parada para variables categóricas y para variables numéricas.
- Número de estimadores o árboles.

Si deseamos imputar variables categóricas es necesario indicar cuales son estas variables mediante el parámetro *cat_vars*.

Capítulo 3

Análisis y limpieza de los datos

En este capítulo se describe el conjunto de datos del que se dispone y al que se le realizan las diferentes transformaciones como eliminación, creación o modificación de variables.

3.1 Conjunto de datos

El conjunto de datos usado en este problema se trata de una submuestra sobre el total de operaciones de tipo transacción realizadas entre el *01/01/2022* y el *30/03/2023*. Se parte de un conjunto de datos con $p = 134$ variables y $n = 973000$ observaciones, pero tras el procesado y limpieza del conjunto de datos el número de variables no será el mismo.

3.1.1 Descripción de variables iniciales

La variable principal y objetivo es FRAUDE que tomará el valor **Y** si se trata de una transacción fraudulenta y su valor será **N** si no lo es. En la clase de transacciones fraudulentas contamos con 1031 (0.105%) observaciones frente a las 971969 (99.895%) operaciones no fraudulentas, por lo que tenemos un problema con datos muy desbalanceados.

Otra variable con mucha importancia es CANAL, que puede tomar dos valores: **online** cuando se trata de una transacción realizada con la banca electrónica y **mobile** cuando se trata de una transacción realizada con la banca móvil. La importancia de esta variable reside en que hay diecinueve variables como por ejemplo 0510_DEVICETYPE, 1530_unlockType o 1520_numberOfInstalledApplication que solo están presentes cuando el canal es mobile, teniendo todo nulos cuando el canal es online. Por la contra, también hay cinco variables como 0290_HISTORYLENGTH o 0410_SCREENDPI que sólo están presentes cuando el canal es online, teniendo todo valores nulos cuando el canal es mobile. Vistas estas diferencias entre los dos canales, se separarán los datos en dos conjuntos en función del canal y se realizarán dos modelos distintos para no mezclar información que puede ser clave a la hora de detectar operaciones fraudulentas.

En el conjunto de datos online se cuenta con $n = 312891$ (un 32.15% del total inicial) observaciones de las cuales 371 son fraudulentas y 312520 son no fraudulentas. En el conjunto de datos mobile tenemos $n = 660109$ observaciones (67.85%) de las cuales 660 son fraudulentas y 659449 son no fraudulentas.

Las variables explicativas se dividen en tres grandes grupos:

- Variables propias de la transacción como la fecha, aplicación, canal, url de la operación, id del banco destino, importe, etc.
- Variables del dispositivo con el que se realizó la transacción como el navegador, plataforma, sistema operativo, altura y ancho de la pantalla, supuestas latitud y longitud desde donde se hizo la transacción, lenguaje del dispositivo, porcentaje de batería, etc.
- Variables acerca del comportamiento del cliente como el número de accesos a la banca electrónica en la última semana, mes y tres meses; medias y desviaciones típicas de las transacciones hechas en la última semana, mes y tres meses; digitalidad relativa y edad del cliente; número de países visitados en el último mes; lista de países y navegadores habituales del cliente; etc.

En el apéndice A se puede ver una explicación más precisa de cada variable.

3.2 Análisis descriptivo

De cara a tener más información acerca de los conjuntos de datos se realiza un análisis de algunas variables. Realizaremos este análisis mostrando gráficas acerca de las variables.

3.2.1 Importe

En la Gráfica 3.1 (página 16) se puede observar que el importe medio es superior para operaciones fraudulentas y también para el canal online. Se ve que las transacciones con un importe superior se corresponden con operaciones no fraudulentas probablemente debido a las medidas de seguridad que existen para este tipo de operaciones.

La mediana de las transacciones de banca electrónica es de 750€ frente a los 70€ de la banca móvil. La dispersión en las operaciones fraudulentas es inferior ya que tienen una desviación típica de 4146€ frente a la desviación típica de las operaciones no fraudulentas que es 106247€. Viendo esto y el gráfico se puede concluir que la dispersión de las operaciones fraudulentas es inferior.

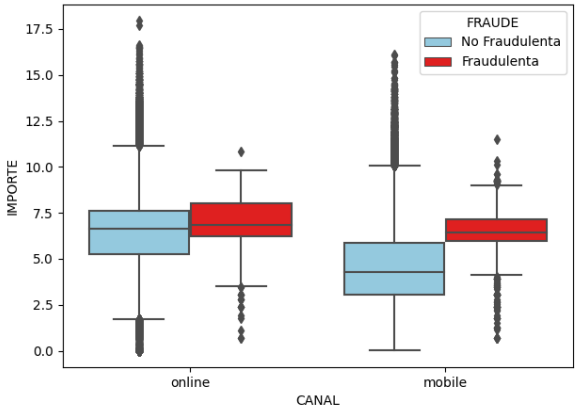

Figura 3.1: Box plot del importe de las transacciones en escala logarítmica en función del canal y de si la operación es fraudulenta o no.

3.2.2 Edad

En la Figura 3.2 (página 16) se observa que las edades de las operaciones fruadulentas son muy similares a las de operaciones no fraudulentas. La edad es inferior para operaciones realizadas con dispositivos móviles. La edad media de los clientes que realizan transacciones con banca electrónica es de 41.5 años, mientras que la media de edad en banca móvil es de 36.8 años.

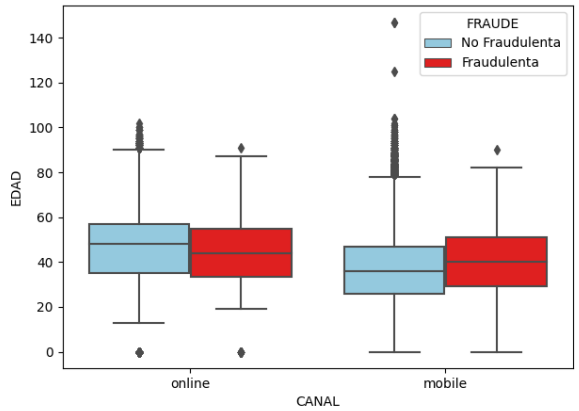

Figura 3.2: Box plot de la edad de las personas que realizaron las transacciones en función del canal y de si la operación es fraudulenta o no.

3.2.3 Digitalidad relativa

En la Figura 3.3 (página 17) tenemos la variable DIGITALIDAD_RELATIVA.

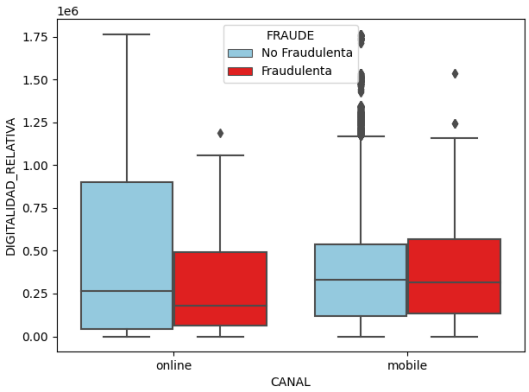

Figura 3.3: Box plot de la digitalidad relativa de las personas que realizaron las transacciones en función del canal y de si la operación es fraudulenta.

Se puede percibir que el valor de esta variable es ligeramente superior para la banca móvil. Para el conjunto de datos de banca electrónica se observa que el valor de esta variable es inferior cuando se trata de una operación fraudulenta; también se ve una mayor dispersión en las observaciones no fraudulentas. En cuanto a la banca móvil, los valores son más similares, pero podemos ver que la mayoría de los valores atípicos se corresponden con observaciones no fraudulentas.

3.2.4 Cociente digitalidad relativa y edad

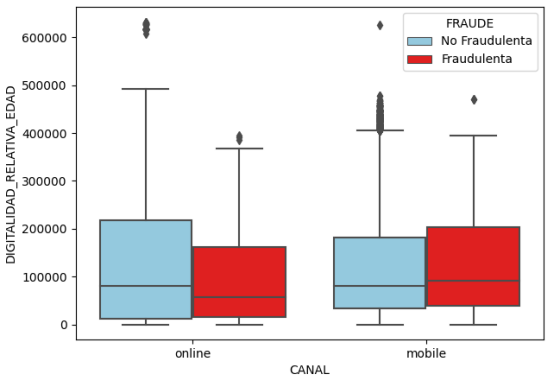

Figura 3.4: Box plot del cociente entre la digitalidad relativa y la edad en función del canal y de si la operación es fraudulenta.

En la Figura 3.4 (página 17) tenemos la variable DIGITALIDAD_RELATIVA_EDAD. En las observaciones de banca electrónica, esta variable tiene valores ligeramente superiores cuando la operación es no fraudulenta; en cambio, en las observaciones de banca móvil, esta variable toma valores ligeramente superiores cuando es fraudulenta.

La gran mayoría de valores atípicos de esta variable se corresponden con operaciones que no fueron fraude.

3.2.5 Accesos a banca electrónica

Esta variable, solo será analizada para los datos de banca electrónica pues nos indica el número de accesos a la banca electrónica del cliente y está más relacionada con este conjunto de datos.

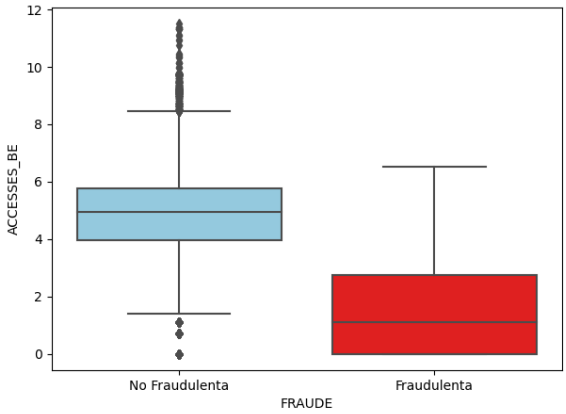

Figura 3.5: Box plot del número de accesos (en escala logarítmica) a banca electrónica del cliente en función de si la transacción es fraudulenta

.

En la Figura 3.5 (página 18) se puede observar que el número de accesos es claramente muy inferior para las operaciones no fraudulentas y se puede suponer que esta variable va a tener bastante importancia en los modelos de banca electrónica. La mediana es de 2 accesos para las operaciones fraudulentas y de 140 accesos para las operaciones no fraudulentas, lo cual es una diferencia muy significativa.

En cuanto a la dispersión, también es muy inferior en las operaciones fraudulentas con una desviación típica de 108.6 accesos frente a una desviación típica de 754.18 en las observaciones no fraudulentas.

3.2.6 Accesos a banca móvil

En contraposición con el apartado anterior, esta variable solo se analizará para las transacciones de banca móvil.

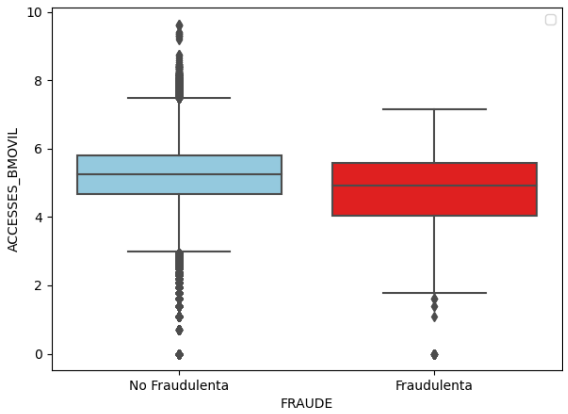

Figura 3.6: Boxplot de la cantidad de accesos a banca móvil en escala logarítmica en función de si la operación es fraudulenta o no.

En la Figura 3.6 (página 19) se puede ver que hay menos diferencias entre las operaciones fruadulentas y no fraudulentas. La mediana es de 135 accesos para las operaciones fraudulentas y 188 accesos para las no fraudulentas, lo cual es una diferencia considerable, aunque sea inferior a la de la variable anterior. En cuanto a la dispersión, la desviación típica es mayor para las operaciones fraudulentas con un valor de 260.7 accesos frente a un valor de 197.4 para las no fraudulentas. Otra observación es que los valores atípicos más grandes se corresponden con observaciones no fraudulentas.

3.2.7 Porcentaje de batería

Esta variable solo está presente en el canal mobile, por lo que en el Gráfico 3.7 (página 20) solo se diferencia por la etiqueta. Podemos percibir que el porcentaje de batería es inferior en las operaciones fraudulentas respecto a las no fraudulentas. La mediana para las transacciones no fraudulentas es de 62% de batería frente al 48% de las operaciones fraudulentas.

3.2.8 Número de aplicaciones

Esta variable también está solo presente en el canal mobile, en la Figura 3.8 (página 20) se tiene el número de aplicaciones del móvil en escala logarítmica en función de la etiqueta. Se puede observar claramente que está variable también tiene un valor inferior cuando las operaciones son fraudulentas. En cambio, los valores superiores se corresponden con operaciones

no fraudulentas.

La mediana de las operaciones fraudulentas es de 36 contra la de las operaciones no fraudulentas con un valor de 64 aplicaciones. Las desviaciones típicas de Ambas clases son 35.6 y 35.5 aplicaciones respectivamente. En cuanto a los valores atípicos podemos observar que la gran mayoría son de operaciones no fraudulentas, mientras que los valores de operaciones fraudulentas están más agrupados.

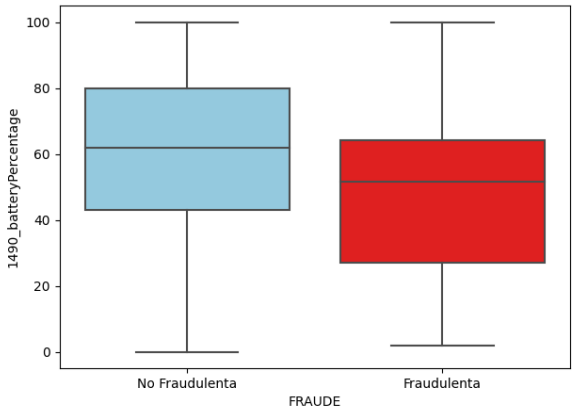

Figura 3.7: Box plot de la batería del dispositivo móvil en función de si la operación es fraudulenta o no.

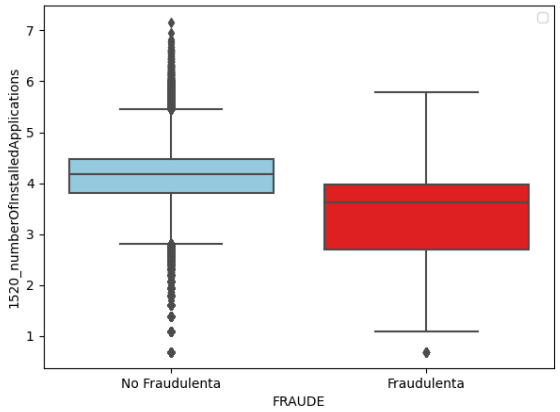

Figura 3.8: Boxplot del número de aplicaciones en función de si la operación es frauduelenta o no.

3.3 Limpieza de los datos

Realizar una limpieza de los datos es necesaria para tener unos datos de calidad, para ello eliminaremos variables con información repetida o que se considere poco importante, modificaremos algunas variables para facilitar su uso y entendimiento por parte de los algoritmos de machine learning y crearemos variables nuevas a partir de variables ya existentes.

3.3.1 Eliminación variables

En el conjunto de datos inicial contamos con información repetida o muy similar, otras variables tienen información que no es relevante de cara a realizar predicciones, otras variables tienen el mismo valor para todos los datos y otras variables no estarán en los conjuntos de datos finales, pero serán usadas para crear nuevas variables.

Eliminamos las siguientes variables:

- CANAL: puesto que separamos los datos en función del canal, esta variable tomará el mismo valor en cada conjunto y será eliminada.
- FECHA: no se considera relevante puesto que el objetivo es predecir fraude en fechas futuras. También se eliminan dos variables sobre fechas como FECHA_PLANIF_OPER y FECHA_DIA.
- RESULTADO: todas las transacciones tienen el mismo valor en este campo, por lo que se elimina esta variable.
- OPERACION: todos los datos usados son transacciones por lo que tienen el mismo valor en esta variable.
- ID_BANCO_DESTINO: demasiados valores distinos y un enorme porcentaje de nulos. Además, es difícil establecer un criterio para agrupar los datos de esta variable.
- 0200_BROWSERVERSION: relacionada con 0190_BROWSER, nos quedaremos con esta última.
- 0340_USERAGENT: aporta información acerca del navegador, cpu, sistema operativo y dispositivo; pero esta información ya la tenemos por separado en otras variables.
- 0690_CITY y 0710_COUNTRY: aportan información acerca de la supuesta localización de la persona que realiza la operación, nos quedaremos con la variable 0780_REGION para tener esta información además de 0750_LATITUDENETWORK y 0760_LONGITUDENETWORK.

- COUNTRY_LIST: esta variable será usada para la creación de varias variables que se explicarán en la siguiente subsección, pero no estará en los conjuntos de datos finales.
- NAVEGADOR_LIST: al igual que la anterior se usará para la creación de variables, pero no estará en los conjuntos finales.
- ACCESSES_BMOVIL_W1_ZS: tiene información repetida ya que se trata de la variable normalizada de ACCESSES_BMOVIL_W1. Pasa lo mismo con ACCESSES_BMOVIL_M1_ZS, ACCESSES_BMOVIL_M3_ZS, ACCESSES_BMOVIL_ZS, ACCESSES_BE_W1_ZS, ACCESSES_BE_M1_ZS, ACCESSES_BE_M3_ZS y ACCESSES_BE_ZS que también serán eliminadas.

También eliminaremos en el conjunto de datos online aquellas variables que presenten todo valores nulos al estar solo presentes en el canal mobile. Tenemos diecinueve variables que lo cumplen como 0520_EMULATOR.

Realizamos el mismo proceso con el canal mobile, hay 5 variables que presenten todo valores nulos al estar solo en canal online como la variable 0430_SCREENTOUCH.

3.3.2 Creación de variables

A partir de variables difíciles de usar como la lista de navegadores habituales del cliente o de la lista de países habituales, crearemos distintas variables que pueden ser más útiles y aportar información al modelo final.

Variable SCORE_BROWSER

La variable NAVEGADOR nos indica el navegador con el que se realizó la operación, mientras que la variable NAVEGADOR_LIST contiene la lista de navegadores habituales del cliente para realizar operaciones. En cuanto a la primera variable tiene un total de 15704 valores distintos, mientras que la segunda tiene un total de 125981 valores posibles, por lo que para aprovechar la información de estas dos variables es necesario aplicarles algún tipo de transformación. Con las funciones *extractOne* y *token_set_ratio* de la librería de python fuzzywuzzy [15] obtendremos un valor numérico entre 0 y 100 acerca de la similitud entre el navegador de la lista que más se parece al navegador utilizado. El objetivo es comprobar si el navegador usado en la operación tiene relación con sus navegadores habituales.

Ejemplo 3.1 *Sean X = NAVEGADOR e Y = NAVEGADOR_LIST las variables iniciales y Z = SCORE_BROWSER la nueva variable.*

Tenemos una observación con los valores X_1 y Y_1 siendo respectivamente ABANCA/1044548 CFNetwork/ 1333.0.4 Darwin/21.5.0 y [ABANCA/1044548 CFNetwork/1331.0.7 Darwin 21.4.0 ABANCA/999239 CFNetwork/1331.0.7 Darwin 21.4.0].

Con la función descrita obtendremos una puntuación de 96 de similitud entre el navegador usado y el navegador que más se parece de la lista. Por lo que $Z_1 = 96$.

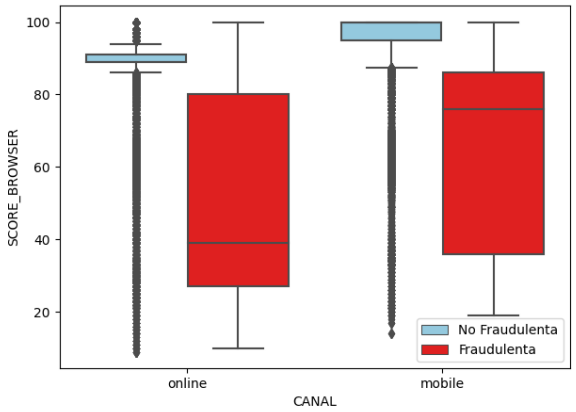

Figura 3.9: Boxplot del scoreBrowser en función de si la operación es fraudulenta o no.

En la Figura 3.9 (página 23), podemos ver que los valores de esta variable son muy distintos según la operación es fraudulenta o no. La mediana de las operaciones fraudulentas es de 46, mientras que de las no fraudulentas es 98, lo cual es una diferencia de más del doble. La dispersión también es mayor para las operaciones fraudulentas con una desviación típica para estas de 26.8 y 9.1 para las no fraudulentas. Con estos datos y viendo el gráfico, podemos concluir que esta variable podría tener un gran poder discriminativo y tener bastante importancia en nuestros modelos.

Variable SAME_COUNTRY

La variable ID_PAIS_DESTINO nos indica el país al que se realizó la transacción, mientras que la variable COUNTRY_LIST indica la lista de países habituales a los que el cliente realiza transacciones. Con la función *countries.get* de pycountry [16] pasamos el id del país a su nombre en inglés y podremos comprobar si este país está en la lista.

Ejemplo 3.2 *Sea X = ID_PAIS_DESTINO, Y = COUNTRY_LIST y Z = SAME_COUNTRY siendo esta última la nueva variable creada.*

Partimos de la observación (X_1, Y_1) *con la forma* $X_1 = Es$ *y* $Y_1 =$ *[Spain, Portugal]. Obtenemos que el país destino está en la lista de países y por lo tanto* $Z_1 = 1$.

Variable SAME_COUNTRY2

La variable 0710_COUNTRY indica el supuesto país desde el que se realizó la operación, por lo que usaremos esta variable y ID_PAIS_DESTINO para comprobar si el país destino y

origen de la transacción coinciden. Usamos las mismas funciones que en el apartado anterior para comprobar si estos países coinciden.

Ejemplo 3.3 *Sean las variables iniciales X = ID_PAIS_DESTINO e Y = 0710_COUNTRY, Z = SAME_COUNTRY2 será la nueva variable.*

Si partimos de una observación cuyos valores sean $X_1 =$ Es y $Y_1 =$ ESP, obtenemos que son el mismo país y $Z_1 = 1$.

Variable SAME_COUNTRY3

Partiendo de las variables 0710_COUNTRY y COUNTRY_LIST. Usamos las mismas funciones y el mismo procedimiento que en los dos apartados anteriores para comprobar si el país origen de la transacción se encuentra en la lista de países habituales de destino del cliente.

Ejemplo 3.4 *Sean X = 0710_COUNTRY e Y = COUNTRY_LIST las variblaes iniciales y Z = SAME_COUNTRY3 la nueva variable.*

Con una observación cuyos valores sean $X_1 =$ ESP y $Y_1 =$ [Spain, Portugal], obtenemos que el supuesto país desde el que se hizo la transacción se encuentra en la lista de países habituales y tendremos que $Z_1 = 1$.

3.3.3 Selección de variables

En este apartado se reducirá el tamaño del conjunto de datos eliminando variables que aparentemente no sean relevantes. El objetivo es aumentar la velocidad y mejorar el rendimiento de los algoritmos al mismo tiempo que se ahorran recursos al tener unos conjuntos de datos más pequeños. Para ello emplearemos test de hipótesis y distinguiremos entre variables categóricas, variables mixtas y variables continuas y discretas.

Una vez terminada la selección de variables:

- En el conjunto de datos online pasamos de 96 a 84 variables.
- En el conjunto de datos mobile pasamos de 111 a 87 variables.

Variables categóricas

Se considera una variable categórica a aquella variable que indica la pertenencia a una clase o categoría. Usaremos el contraste χ^2 para medir las discrepancias entre las distribuciones de los datos de las operaciones fraudulentas y las no fraudulentas. Esto será realizado mediante la función *chi2_contingency* de scipy [17]. Los resultados posibles son:

- Ambas distribuciones son similares, por lo que consideramos que la variable no tiene información relevante y no usaremos dicha variable.

- Las distribuciones son significativamente diferentes, por lo que consideramos que la variable es útil y la mantendremos en el conjunto de datos.

En el conjunto de datos online se eliminan tres variables categóricas como NAVEGADOR, mientras que en el de online se eliminan trece variables como 0630_SMSSTEALERS o 0700_CONTINENTCODE.

Variables continuas y discretas

Consideramos variables continuas aquellas que pueden tomar cualquier valor en un rango, mientras que serán variables discretas aquellas que tienen una cantidad contable de observaciones en un rango. Usaremos el contraste de Kolmogorov-Smirnov que compara la función de distribución empírica de los datos de operaciones fraudulentas contra las no fraudulentas. Emplearemos la función *kstest* de scipy [17]. Los posibles resultados son:

- Ambas distribuciones son similares, por lo que consideramos que la variable no tiene información relevante y no usaremos dicha variable.
- Las distribuciones son significativamente distintas, por lo que consideramos que la variable es útil y la mantendremos.

En el conjunto de datos online se elimina solo la variable COUNTRIES_W1_CNT y en el mobile se eliminan tres variables como ACCESES_BE_M1.

Variables mixtas

Llamaremos variables mixtas a aquellas variables continuas o discretas en las que se repita en un 20% o más de las observaciones el mismo valor k. Dada una variable X en la que se repita el mismo valor k en un 20% o más de las observaciones, creamos una variable Z que tendrá la forma:

$$Z_i = \begin{cases} 1 & \text{si xi toma el valor k,} \\ 0 & \text{si xi no toma el valor k} \end{cases}$$

En el primer paso de este proceso realizaremos el contraste de χ^2 considerando la variable Z obteniendo los siguientes resultados:

- Ambas distribuciones son similares por lo que iremos al siguiente paso para dedicir sin conservamos la variable.
- Las distribuciones son significativamente distintas, por lo que consideramos que la variable tiene importancia y la mantendremos en dataset.

Sobre las variables que no superen el anterior test, aplicamos el segundo paso. Se elimina el valor k de la variable X para realizarle el test de Kolmogorov-Smirnov igual que en la sección 3.3.3. Los resultados pueden ser:

- Las dos distribuciones son similares, consideramos que la variable no tiene información relevante y la eliminamos.
- Son significativamente distintas las distribuciones, por lo que se considera que la variable es de utilidad y seguirá en el conjunto de datos.

3.4 Valores faltantes

En esta sección analizaremos los valores faltantes en los conjuntos de datos de mobile y online por separado, también buscaremos relaciones entre estos valores y explicaremos las distintas técnicas que usamos para hacer el tratamiento de estos valores faltantes.

3.4.1 Análisis valores faltantes

En primer lugar, realizamos el test de Little [6] sobre el conjunto de datos online para comprobar si la pérdida de los datos es MCAR o no. Obtenemos un p-valor menor que 0.001 y con un nivel de significación del 5% rechazamos que la pérdida sea MCAR, por lo que podría ser MAR o MNAR.

Si estudiamos los valores faltantes más en profundidad, observamos grupos de variables que tienen exactamente los mismos valores faltantes en las mismas observaciones. Esto se debe a que el origen de estas variables es el mismo; por lo que cuando no tenemos una, tampoco tendremos el resto. Los grupos de variables son:

- Las variables 0400_PLATFORM, 0410_SCREENDPI, 0420_SCREENHEIGHT, 0430_SCREENTOUCH y 0440_SCREENWIDTH (un 8.01% de las observaciones).
- Las variables 0720_IPCLASS y 0740_ISP (un 7.93% de las observaciones).
- Las variables 0230_CLIENTTIMEZONE y 0290_HISTORYLENGTH (un 8.08% de las observaciones).
- Las variables IN_BE, IN_BMOVIL y IN_EALE (19.35% de las observaciones).
- Las variables 0750_LATITUDENETWORK y 0760_LONGITUDENETWORK (solo en el 0.17% de las observaciones).

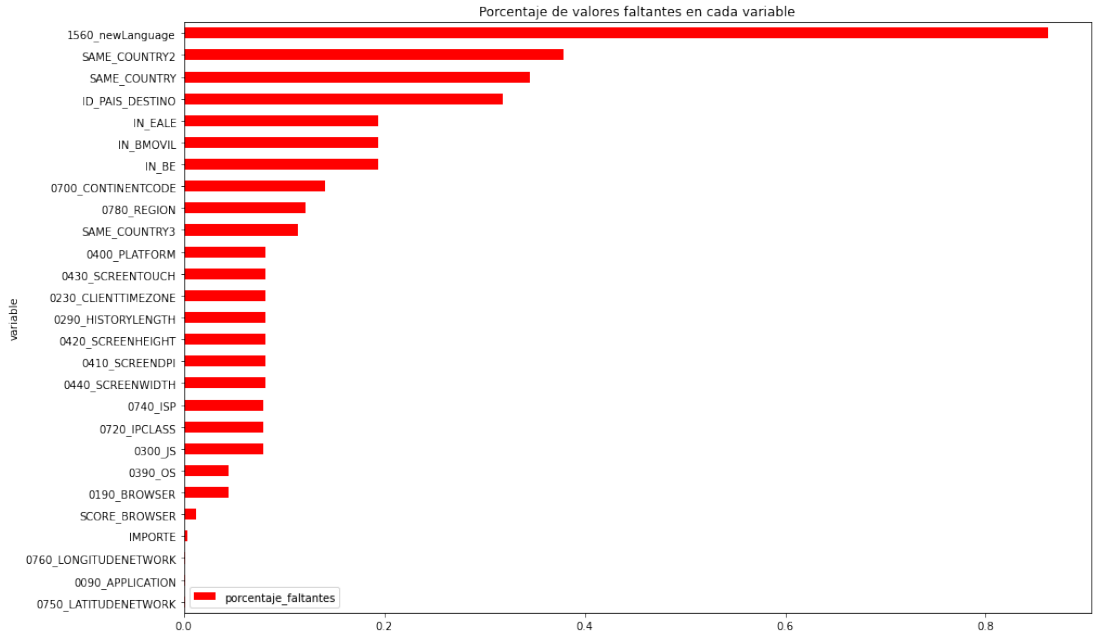

Figura 3.10: Representación del porcentaje de valores faltantes en cada variable del conjunto de datos de banca electrónica

Observando el Gráfico 3.10 (página 27) podemos ver que la variable con mayor porcentaje de valores faltantes es 1560_newLanguage superando el 80% de las observaciones nulas. También que tenemos 27 variables (32.54% de las variables del conjunto de datos online) con al menos un valor nulo; por lo que será muy importante realizar un correcto tratamiento de estas variables.

Si realizamos el test de little en el conjunto de datos mobile, obtenemos un p-valor también menor que 0.001, por lo que también rechazamos que la pérdida del conjunto de datos mobile sea MCAR con un nivel de significación del 5%. Realizando el mismo análisis de los valores faltantes, también podemos encontrar grupos de variables que tienen valores faltantes en exactamente las mismas observaciones.

- Las variables IMPORTE y DIVISA (7.53% de las observaciones). Si investigamos más a fondo encontramos que las observaciones con estas dos variables nulas son aquellas en las que la variable APLICACION_ORIGEN es TjVirtualAltaTarjeta. Estas dos variables son nulas en este caso porque cuando se da de alta una tarjeta virtual no se guarda el importe en las tablas con las que estamos trabajando.
- Las variables 0190_BROWSER y 0390_OS (un 0.3% de las observaciones).
- Las variables 0400_PLATFORM, 0420_SCREENHEIGHT y 0440_SCREENWIDTH (el 2.7% de las observaciones de mobile).

- Las variables 0470_CPUTYPE, 0490_DEVICELANGUAGE, 0510_DEVICETYPE y 0640_TIMEZONE (tan solo en el 0.97%).
- Las variables 0500_DEVICE, 0590_ROOTHIDERS, 0750_LATITUDENETWORK y 0760_LONGITUDENETWORK (solo un 0.24% de observaciones).
- Las variables 0720_IPCLASS y 0740_ISP (0.32% de observaciones).
- Las variables 1490_batteryPercentage y 1530_unlockType (en un 9.6%).

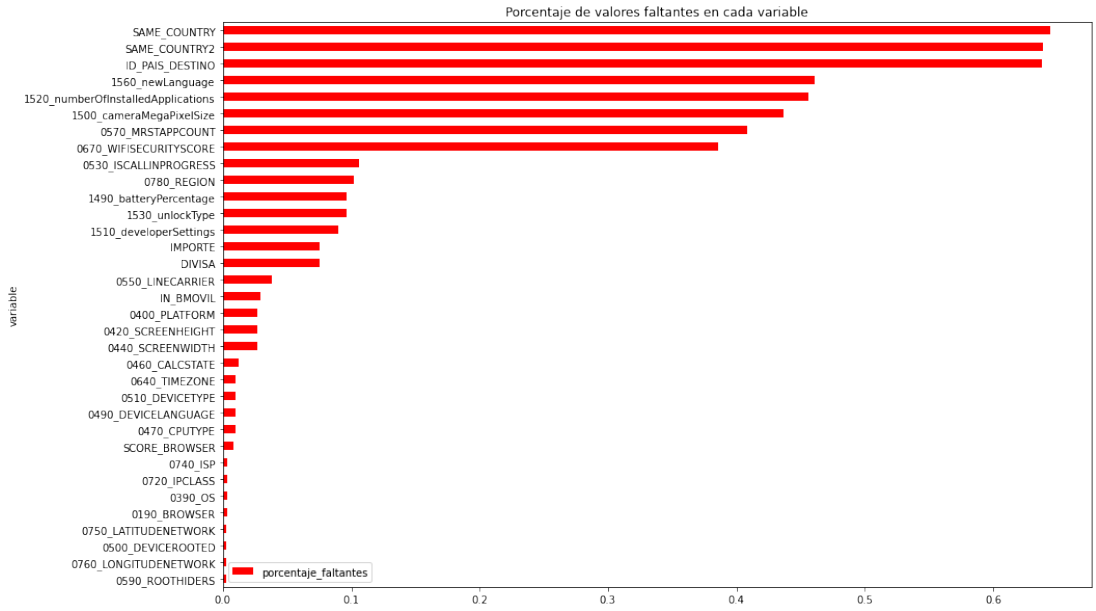

Figura 3.11: Representación del porcentaje de valores faltantes en cada variable del conjunto de datos de banca móvil

A parte de los grupos ya mencionados, también tienen gran relación las variables ID_PAIS_DESTINO, SAME_COUNTRY y SAME_COUNTRY2. Esto se debe a que se, como se vió en la sección 3.3.2, se usa la primera variable para obtener las dos restantes, por lo que cuando la primera sea nula, las otras dos también lo serán. Podemos observar que las tres variables superan el 60% de valores faltantes en sus observaciones.

En el caso del conjunto de datos mobile, tenemos hasta 34 variables (un 39.08% del total de variables) que no tienen todas las observaciones, por lo que también será clave un buen tratamiento de estas variables.

3.4.2 Tratamiento valores faltantes

Como hemos visto, tenemos una gran cantidad de variables con valores nulos, por lo que tiene mucha importancia tratarlas de una manera correcta. En este apartado se explican los

métodos que se usarán para tratar estas variables. Se probarán estos métodos en los distintos tipos de modelos que se usan en los capítulos posteriores.

En primer lugar, se eliminan aquellas variables con un porcentaje de valores faltantes superior al 50%. Se realiza esto, porque hay varias variables en cada conjunto que superan este umbral y no vale la pena imputarlas. Lo ideal sería que en la recolección de datos se pudiera reducir la cantidad de valores faltantes en estas variables para poder tener más información a la hora de detectar patrones de las operaciones fraudulentas.

Todos los métodos que usemos han sido explicados en el capítulo anterior. Se usarán los siguientes métodos o combinación de métodos:

1. Listwise, nos quedamos con aquellas observaciones que no tengan ningún valor faltante. En estos conjuntos de datos el problema es que tan solo un pequeño porcentaje de las observaciones tiene los datos completos. En el caso del conjunto de datos online pasamos de un dataset de 312891 observaciones a 27748, un 8.87% de las observaciones iniciales; también se reducen el número de transacciones fraudulentas pasando de 371 originalmente a 61. En los datos del canal mobile partimos de 660109 observaciones por las $42212(6.39\%$ del total) que quedan después de aplicar este método; el número de operaciones fraudulentas se reduce aún más que en online pasando de 660 transacciones fraudulentas a tan solo 12. Con esta reducción de las operaciones en ambos datasets será muy complicado conseguir buenos resultados con este método.

2. Imputación por la media para las variables numéricas e imputación con la moda para las variables categóricas. Mediante la función *SimpleImputer* de sklearn [18]. La ventaja de este método respecto al anterior es la no pérdida de ninguna observación ni de ninguna variable, además destaca la rapidez y simpleza con la que se pueden imputar los datos.

3. Imputación por la mediana para las variables numéricas e imputación con la moda para las variables categóricas. También con la función *SimpleImputer* de sklearn. Características similares al método anterior, pero es más robusto a valores atípicos. Por ejemplo, en la variable IMPORTE en el conjunto de datos online cuando se imputa con la media, se imputa el valor 7924.4€, mientras que si se imputa la mediana se tienen 748.6€ un valor 10 veces inferior que probablemente se ajuste mejor con la realidad.

4. Imputación mediante KNN, se usa un solo vecino para poder imputar tanto variables numéricas y categóricas. La medida de distancia usada es la distancia euclídea. Se emplea la función *KNNImputer*,también perteneciente a la librería sklearn.

5. Imputación con regresión lineal para las variables numéricas y regresión logística para las variables categóricas. Para cada variable numérica con valores faltantes se construyó una regresión lineal con las variables restante mediante la función LinearRegression de

sklearn [11] y así poder predecir los valores faltantes e imputarlo. Con las variables categóricas se siguió un proceso similar con la función LogisticRegression también de sklearn.

6. Imputación con regresión estocástica para variables numéricas y regresión logística para categóricas. Para conseguir mantener las correlaciones de las variables se añade un residuo aleatorio a la regresión lineal. La metodología ha sido muy similar al método anterior con la diferencia de añadir dicho residuo.

7. Imputación múltiple tanto para variables numéricas como para variables categóricas. La imputación de todas las variables se hace de forma simultánea. Se usa la función *MiceImputer* de la librería Autoimpute [12]. Para combinar las distintas imputaciones que realiza se usa la mediana para variables numéricas y la moda para variables categóricas.

8. Imputación con MissForest para todas las variables ya que sirve tanto para las variables categóricas como para las variables numéricas. Se emplea la función *missForest* de missingpy [14]. Con esta función es importante indicar aquellas variables que son categóricas mediante el parámetro *cat_vars* para que las trate como tal, ya que si no las tratará como variables numéricas. La ventaja respecto al resto de métodos utilizados es que para cada valor faltante se le imputa un valor distinto por lo que es mucho más probable mantener la distribución de cada variable. El problema es el tiempo necesario para ejecutar este método, que es muy superior al resto de métodos empleados.

3.4.3 Comparación métodos de imputación

Vistos los métodos usados para imputar los valores faltantes, se compara la distribución del logaritmo de la variable 1520_numberOfInstalledApplication en cada método para ver los distintos efectos de los métodos de tratamiento de datos faltantes. Esta variable tiene alrededor del 45% de valores faltantes como se ve en el Gráfico 3.11 (página 28). Se analizan las distintas funciones de densidad del logaritmo de esta variable.

En primer lugar, en la Figura 3.12 (página 31) tenemos 2 gráficas: la función de densidad de los valores originales de esta variable y los valores tras aplicar listwise. Se puede observar que aplicando listwise se obtiene una distribución similar salvo en los valores más bajos de esta variable, ya que apenas tienen presencia tras aplicar este método.

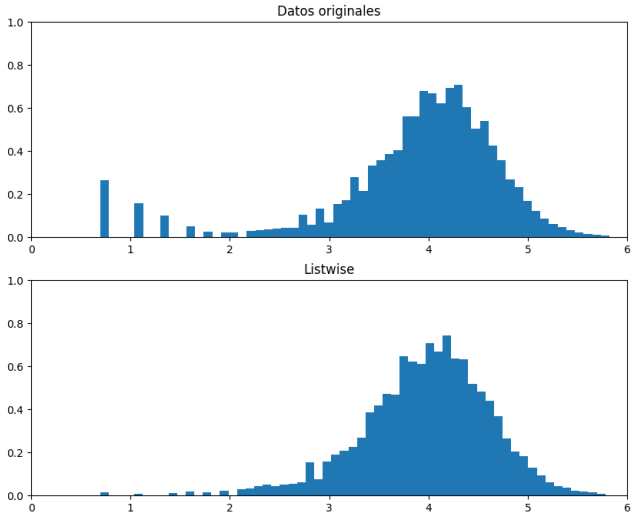

Figura 3.12: Funciones de densidad del logaritmo de la variable 1520_numberOfInstalledApplication con los valores originales y aplicando listwise

En la Figura 3.13 (página 32) se usó la imputación con la media y mediana respectivamente. En estas dos distribuciones se observa cómo se transforma la distribución de la variable con un pico muy alto en cada una de sus correspondiente media y mediana. Esto se debe a que esta variable tiene un 45% de valores faltantes, por lo que no son las mejores formas de imputación para esta variable.

Los siguientes métodos de imputación fueron vecino más cercano, regresión lineal y regresión estocástica y podemos ver sus distribuciones en la figura 3.14 (página 33). La imputación con vecino más cercano es muy similar a la distribución original, siendo la que más parecido tiene en los valores más bajos. En cuanto a la regresión lineal la función de densidad resultante agrupa mucho los valores centrales y no tiene demasiado parecido con la original. Por último, la regresión estocástica si es capaz de dar una representación aparentemente más cercana a la realidad, aunque en los valores bajos no tiene mucho parecido.

Por último, se realiza la imputación con imputación múltiple y missForest en la Figura 3.15 (página 34). La distribución de los datos imputados con imputación múltiple sigue una distribución similar a los datos originales, con la diferencia de que es más escalonada que la original. También hay diferencias en los valores más pequeños. En cuanto a missForest se observan bastantes parecidos con la distribución original salvo en los valores más pequeños, ya que en la distribución original había más valores en ese rango.

Viendo estas figuras, podemos concluir que los métodos de media, mediana y regresión lineal transforman mucho la distribución de esta variable. El método de imputación múltiple es

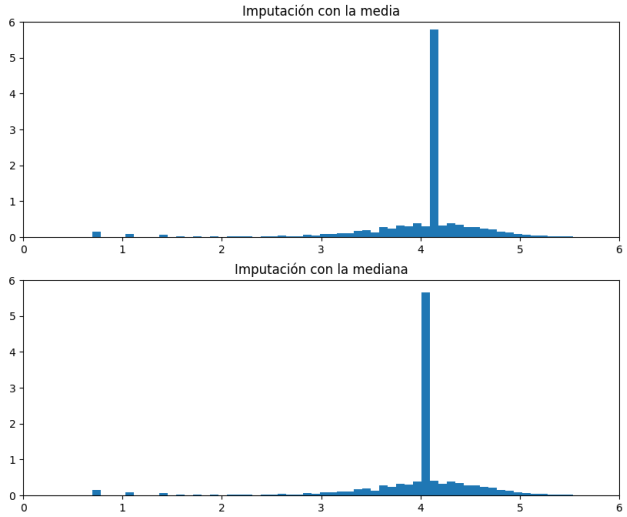

Figura 3.13: Funciones de densidad del logaritmo de la variable 1520_numberOfInstalledApplication imputando con la media y mediana.

intermedio y mantiene ciertas similitudes. Los métodos que mejor mantienen la distribución original de la variable son missForest, vecino más cercano y regresión estocástica. En el caso de análisis de listwise también se mantiene la distribución, pero se han eliminado más del 90% de las observaciones y es una pérdida de información poco razonable.

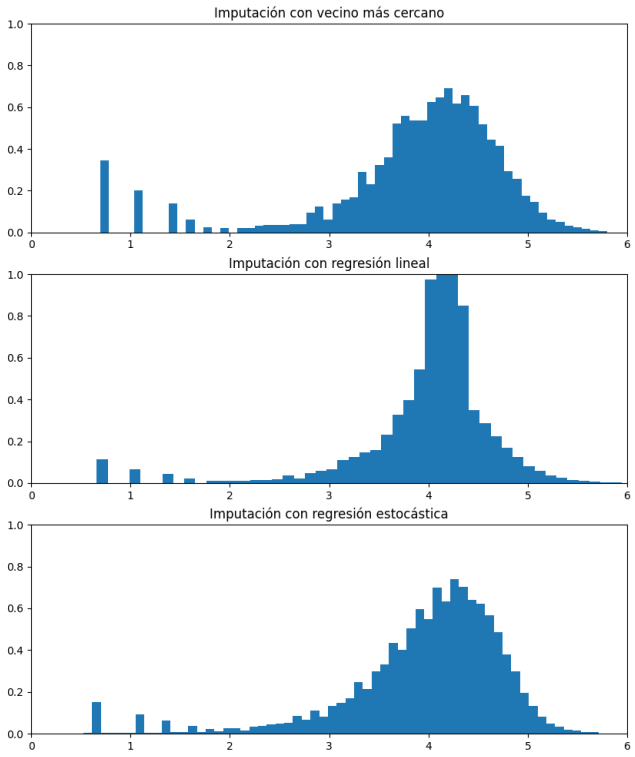

Figura 3.14: Funciones de densidad del logaritmo de la variable 1520_numberOfInstalledApplication imputando con vecino más cercano, regresión lineal y regresión estocástica

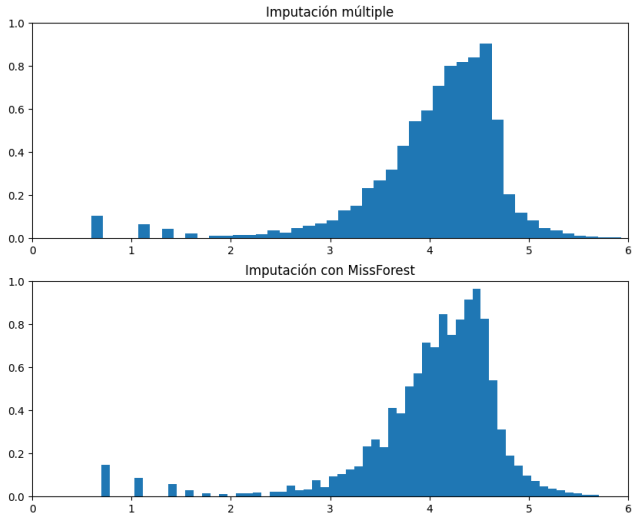

Figura 3.15: Funciones de densidad del logaritmo de la variable 1520_numberOfInstalledApplication imputando con imputación múltiple y missForest

Capítulo 4

Modelos entrenados y resultados

En este capítulo se describirán los modelos de machine learning empleados, así como los pasos realizados para su entrenamiento y la metodología empleada. Emplearemos tres modelos distintos: Regresión Logística, XGBoost y CatBoost . Para cada modelo y cada conjunto de datos (online y mobile) se busca la mejor combinación de parámetros (hiperparámetros propios del modelo y método para tratar los datos faltantes) posible. Para ello se realiza una búsqueda óptima de los hiperparámetros del modelo sobre cada método de imputación. Para comparar los resultados de los modelos usaremos los criterios de selección descritos en la subsección 4.3.

4.1 Método de trabajo

Para cada modelo buscaremos encontrar la mejor combinación de hiperparámetros propios del modelo, método de imputación y umbral para determinar que una operación es fraudulenta. Por umbral entendemos el punto de corte a partir del cual consideramos que una operación es fraudulenta. Los modelos nos devolverán un valor entre 0 y 1, siendo fraudulentas las operaciones que superen un umbral determinado.

Ejemplo 4.1 *Por ejemplo, si tenemos el umbral en* 0.5 *y obtenemos una predicción con un valor de* 0.55*, se considera esa operación como fraudulenta al superar el umbral.*

Se realiza una búsqueda de parámetros con cada algoritmo y método de tratar los valores faltantes. Teniendo en cuenta que emplearemos tres algoritmos distintos y usaremos ocho métodos distintos de tratamiento de valores faltantes, se realizarán un total de 24 búsquedas de parámetros (para cada conjunto de datos) para decidir qué modelo es mejor, que técnica de tratamiento de valores faltantes es superior y la mejor combinación de ambas tanto para el conjunto de datos online como para el conjunto de mobile.

Se divide cada uno de los dos conjuntos de datos en entrenamiento (80% de las observaciones) y test (20%). Se usará el conjunto de entrenamiento para entrenar el modelo y el

conjunto de test para ver sus métricas y decidir qué modelos son superiores. Para cada modelo estableceremos inicialmente una serie de hiperparámetros que queremos optimizar. Para ello, estableceremos una serie de valores para cada hiperparámetro y se probarán todas las combinaciones posibles. Con cada combinación se entrenará un modelo con el conjunto de entrenamiento y se obtendrán las métricas sobre el conjunto de test en base a distintos umbrales o puntos de corte. Una vez tenemos las métricas se puede establecer que combinación es mejor.

4.2 Modelos

Los algoritmos de machine learning que emplearemos son regresión logística, XGBoost y CatBoost.

4.2.1 Regresión Logística con penalización cuadrática

Se usa este algoritmo mediante la función *LogisticRegression* de sklearn [11]. La regresión logística se trata de un algoritmo de aprendizaje supervisado para clasificación ya que predice el resultado de una variable categórica. Este algoritmo mide la relación entre las diferentes características del conjunto de datos y la probabilidad de pertenecer a una clase, usa una función logística para estimar las probabilidades. Los parámetros que se pretenden optimizar son:

- Max_iter: número máximo de iteraciones para ajustar los coeficientes para minimizar la función de coste. En la regresión logística tradicional, se realiza un enfoque analítico para calcular estos coeficientes, mientras que en esta función de sklearn se realiza de forma iterativa.
- Class_weight: debido al gran desbalanceo entre las clases, le asignaremos un peso a cada clase para darle más o menos importancia.

4.2.2 XGBoost

XGBoost [3] es un algoritmo de aprendizaje supervisado que sirve tanto para clasificación como para regresión y su abreviatura significa extreme gradient boosting. Como su abreviatura indica está basado en el algoritmo Gradient Boosting. XGBoost está basado en arboles de decisión y es paralelizable y escalable.

Para emplear este algoritmo, se usa la función *XGBClassifier* de la librería XGBoost. Los parámetros que se buscan optimizar son:

- Max_depth: profundidad máxima de cada árbol.

- Learning_rate: tasa de aprendizaje del algoritmo.
- Scale_pos_weight: parámetro usado para controlar el equilibrio entre las clases positivas y negativas. Al tener un problema desbalanceado, si aumentamos el valor de este parámetro, le estamos dando más importancia a los ejemplos positivos.

4.2.3 CatBoost

CatBoost [19] es otro algoritmo de aprendizaje supervisado que también sirve para clasificación y regresión. Al igual que el método anterior está basado en Gradient Boosting y árboles de decisión. Su principal diferencia es el manejo de las variables categóricas, con XGBoost era necesario codificar estas variables como números, pero con CatBoost este paso no es necesario. Otra diferencia con XGBoost es que CatBoost emplea árboles simétricos.

Se usa la función *CatBoostClassifier* de la librería CatBoost para emplear este algoritmo. Los parámetros que se buscan optimizar son:

- Depth: profundidad máxima de cada árbol.
- Scale_pos_weight: parámetro usado para controlar el equilibrio entre las clases positivas y negativas. Al tener un problema desbalanceado, si aumentamos el valor de este parámetro, le estamos dando más importancia a los ejemplos positivos.
- Learning_rate:tasa de aprendizaje del modelo.

4.3 Criterios selección

Para medir el rendimiento de los modelos se tendrán en cuenta dos aspectos: sus métricas y su tiempo de respuesta. En base a estos datos podremos encontrar las configuraciones óptimas de parámetros y decidir que un modelo es mejor que otro

4.3.1 Métricas

Partiendo de la base de que nos encontramos con un problema de clasificación muy desbalanceado no es posible usar cualquier métrica. Por ejemplo, la accuracy no sería un buen indicativo ya que si el modelo predijera todas las operaciones como no fraudulentas tendríamos un accuracy superior al 99% y sería superior a la accuracy de cualquier otro modelo. Por ello usaremos métricas que no tengan en cuenta los verdaderos negativos. Las métricas que usaremos serán precision, recall y f1-score.

Precision

Porcentaje de ejemplos positivos correctamente predichos. En el contexto del problema, de las operaciones fraudulentas que devuelve el modelo, que porcentaje son realmente fraudulentas. Su rango de valores está en entre 0 y 1.

$$\text{Precision} = \frac{\text{Verdaderos Positivos}}{\text{Verdaderos Positivos} + \text{Falsos Positivos}}$$

Recall

Porcentaje de ejemplos positivos identificados por el modelo de entre todos los positivos reales, también llamado índice predictivo positivo. En el contexto de este problema de clasificación, de todas las operaciones fraudulentas que tenemos, que porcentaje devuelve el modelo. Su rango de valores está en entre 0 y 1.

$$\text{Recall} = \frac{\text{Verdaderos Positivos}}{\text{Verdaderos Positivos} + \text{Falsos Negativos}}$$

F1-Score

Trata de representar la precision y el recall en una sola métrica realizando una media armónica de ambas. Su rango de valores está en entre 0 y 1. Existen variaciones para dar más peso a una de las métricas sobre la otra.

$$\text{F1-score} = 2 \cdot \frac{\text{Precisión} \cdot \text{Recall}}{\text{Precisión} + \text{Recall}}$$

4.3.2 Tiempo de respuesta

Para tener un buen modelo de predicción de fraude es importante también tener un tiempo de respuesta lo más pequeño posible. Es por ello que mediremos el tiempo de respuesta medio para cada modelo de machine learning empleado y se tendrá en cuenta para las conclusiones finales.

4.4 Resultados de los modelos

Para la presentación de resultados explicaremos la mejor combinación de hiperparámetros y umbral para cada modelo de machine learning con cada técnica para tratamiento de valores faltantes y cada conjunto de datos. Se mostrará una tabla para cada uno de los dos conjuntos de datos. En esta tabla tendremos el algoritmo usado, método de imputación y las métricas ya descritas en la subsección 4.3.1 evaluadas sobre el conjunto de datos de test.

Con el objetivo de establecer un criterio, se establecerá que un modelo es superior a otro cuando tenga un mayor f1-score.

4.4.1 Conjunto de datos de banca electrónica

En primer lugar, se muestra el tiempo de predicción para cada uno de estos tres algoritmos con este conjunto de datos.

Cuadro 4.1: Tabla con los tiempos de predicción de los algoritmos para el conjunto de banca electrónica.

Algoritmo	Tiempo medio predicción	Desviación típica
Reg. Logística	2.42 ms	0.027 ms
XGBoost	3.32 ms	0.189 ms
CatBoost	4.93 ms	0.259 ms

En segundo lugar, en cada algoritmo enumeramos sus parámetros óptimos para cada método de tratamiento de datos faltantes. En la Tabla 4.2 tendremos las métricas de todos los modelos enumerados.

Regresión Logística con penalización cuadrática

A continuación, se enumera cada método de imputación con sus parámetros óptimos para el algoritmo de regresión logística.

1. Listwise: 2500 iteraciones, un peso de 50 a las observaciones positivas y 0.4 de umbral.
2. Media para variables numéricas y moda para categóricas: 2500 iteraciones, peso de 10 a las observaciones positivas y 0.4 de umbral.
3. Mediana para numéricas y moda para categóricas: 2500 iteraciones, peso de 10 a las observaciones positivas y 0.4 de umbral.
4. Vecino más cercano: 1000 iteraciones, peso de 10 a las observaciones positivas y 0.5 de umbral.
5. Regresión lineal para variables numéricas y regresión logística para categóricas: 2500 iteraciones, peso de 10 a las observaciones positivas y 0.3 de umbral.
6. Regresión estocástica para numéricas y regresión logística para categóricas: 1000 iteraciones, peso de 10 a las observaciones positivas y 0.4 de umbral.

Cuadro 4.2: Tabla con las métricas del mejor modelo para cada método de imputación en el conjunto de datos de banca electrónica.

Algoritmo	**Método imp.**	**Recall**	**Precision**	**F1-Score**
Reg. Logística	Listwise	42.86%	1.25%	4.14%
Reg. Logística	Media	32.2%	15.2%	20.65%
Reg. Logística	Mediana	32.2%	15.97%	21.35%
Reg. Logística	KNN	20.34%	12.12%	15.19%
Reg. Logística	Reg. lineal	35.59%	10.07%	16.54%
Reg. Logística	Reg. estocástica	23.73%	13.08%	16.87%
Reg. Logística	Imp. múltiple	22.03%	13.27%	16.56%
Reg. Logística	MissForest	20.34%	13.79%	16.44%
XGBoost	Listwise	42.86%	60%	50%
XGBoost	Media	71.19%	82.35%	77.78%
XGBoost	Mediana	72.88%	84.31%	78.18%
XGBoost	KNN	76.27%	84.91%	80.36%
XGBoost	Reg. lineal	84.75%	76.92%	80.65%
XGBoost	Reg. estocástica	76.27%	86.54%	81.08%
XGBoost	Imp. múltiple	76.27%	78.95%	77.59%
XGBoost	MissForest	76.27%	88.54%	81.94%
CatBoost	Listwise	28.57%	66.67%	40%
CatBoost	Media	72.88%	87.76%	79.63%
CatBoost	Mediana	72.88%	84.31%	78.18%
CatBoost	KNN	74.58%	86.27%	80%
CatBoost	Reg. lineal	72.88%	84.31%	78.18%
CatBoost	Reg. estocástica	74.58%	84.71%	79.32%
CatBoost	Imp. múltiple	66.1%	92.86%	77.23%
CatBoost	MissForest	74.58%	86.27%	80%

7. Imputación múltiple: 2500 iteraciones, peso de 10 a las observaciones positivas y 0.4 de umbral.

8. MissForest: 2500 iteraciones, peso de 10 a las observaciones positivas y 0.4 de umbral.

XGBoost

A continuación, se enumera cada método de imputación con sus parámetros óptimos.

1. Listwise: profundidad máxima 8, 0.3 de learning rate, 100 de scale_pos_weight y 0.4 de umbral.

2. Media para variables numéricas y moda para categóricas: profundidad máxima 10, 0.2 de learning rate, 10 de scale_pos_weight y 0.5 de umbral.

3. Mediana para numéricas y moda para categóricas: profundidad máxima 8, 0.2 de learning rate, 10 de scale_pos_weight y 0.3 de umbral.

4. Vecino más cercano: profundidad máxima 8, 0.2 de learning rate, 10 de scale_pos_weight y 0.3 de umbral.

5. Regresión lineal para variables numéricas y regresión logística para categóricas: profundidad máxima 10, 0.2 de learning rate, 100 de scale_pos_weight y 0.3 de umbral.

6. Regresión estocástica para numéricas y regresión logística para categóricas: profundidad máxima 6, 0.2 de learning rate, 100 de scale_pos_weight y 0.6 de umbral.

7. Imputación múltiple: profundidad máxima 8, 0.3 de learning rate, 10 de scale_pos_weight y 0.15 de umbral.

8. MissForest: profundidad máxima 8, 0.3 de learning rate, 10 de scale_pos_weight y 0.3 de umbral.

CatBoost

A continuación, se enumera cada método de imputación con sus parámetros óptimos.

1. Listwise: profundidad de 6, 0.1 de learning rate, 1 de scale_pos_weight y umbral de 0.8.

2. Media para numéricas y moda para categóricas: profundidad de 8, 0.1 de learning rate, 10 de scale_pos_weight y umbral de 0.3.

3. Mediana para numéricas y moda para categóricas: profundidad de 6, 0.2 de learning rate, 1 de scale_pos_weight y umbral de 0.3.

4. Vecino más cercano: profundidad de 6, 0.1 de learning rate, 10 de scale_pos_weight y umbral de 0.5.

5. Imputación con regresión lineal para variables numéricas y regresión logística para categóricas: profundidad de 8, 0.1 de learning rate, 10 de scale_pos_weight y umbral de 0.3.

6. Imputación con regresión estocástica para numéricas y regresión logística para categóricas: 6 de profundidad, 0.1 de learning rate, 40 de scale_pos_weight y umbral de 0.6.

7. Imputación múltiple: 8 de profunidad, 0.2 de learning rate, 1 de scale_pos_weight y 0.3 de umbral.

8. MissForest: 6 de profundidad, 0.1 de learning rate, 40 de scale_pos_weight y umbral de 0.5.

4.4.2 Conjunto de datos de banca móvil

Al igual que en el conjunto de datos de banca electrónica, en primer lugar se muestra el tiempo medio de predicción de cada uno de los tres algoritmos para el conjunto de datos de banca móvil. Se muestra en la Tabla 4.3.

Cuadro 4.3: Tabla con los tiempos de predicción de los algoritmos para el conjunto de datos de banca móvil.

Algoritmo	Tiempo medio predicción	Desviación típica
Reg. Logística	2.53 ms	0.031 ms
XGBoost	3.56 ms	0.202 ms
CatBoost	5.09 ms	0.265 ms

A continuación, se explica para cada algoritmo usado, sus parámetros óptimos en cada método de imputación. En la Tabla 4.4 tendremos sus correspondientes métricas.

Regresión Logística con penalización cuadrática

A continuación, se enumeras cada método de imputación con sus parámetros óptimos para el algoritmo de regresión logística.

1. Listwise: 2500 iteraciones, un peso de 50 a las observaciones positivas y 0.4 de umbral.

Cuadro 4.4: Tabla con las métricas del mejor modelo para cada método de imputación con los datos de banca móvil.

Algoritmo	**Método imp.**	**Recall**	**Precision**	**F1-Score**
Reg. Logística	Listwise	0%	0%	0%
Reg. Logística	Media	0.78%	1.2%	1.2%
Reg. Logística	Mediana	20.34%	12.00%	15.09%
Reg. Logística	KNN	20.34%	12.12%	15.19%
Reg. Logística	Reg. lineal	27.12%	11.27%	15.92%
Reg. Logística	Reg. estocástica	32.2%	14.5%	20%
Reg. Logística	Imp. múltiple	25.42%	10.42%	14.78%
Reg. Logística	MissForest	37.29%	12%	18.16%
XGBoost	Listwise	0%	0%	0%
XGBoost	Media	69.77%	91.84%	79.3%
XGBoost	Mediana	68.99%	93.68%	79.46%
XGBoost	KNN	70.54%	88.35%	78.45%
XGBoost	Reg. lineal	68.22%	91.67%	78.22%
XGBoost	Reg. estocástica	69.77%	91.84%	79.30%
XGBoost	Imp. múltiple	68.22%	92.63%	78.57%
XGBoost	MissForest	70.54%	90.1%	79.13%
CatBoost	Listwise	0%	0%	0%
CatBoost	Media	67.44%	93.55%	78.38%
CatBoost	Mediana	66.67%	91.49%	77.13%
CatBoost	KNN	67.44%	90.62%	77.33%
CatBoost	Reg. lineal	68.22%	94.62%	79.28%
CatBoost	Reg. estocástica	67.44%	92.55%	78.03%
CatBoost	Imp. múltiple	66.67%	92.47%	77.48%
CatBoost	MissForest	69.77%	92.78%	79.65%

2. Media para variables numéricas y moda para categóricas: 2500 iteraciones, peso de 100 a las observaciones positivas y 0.3 de umbral.
3. Mediana para numéricas y moda para categóricas: 1000 iteraciones, peso de 50 a las observaciones positivas y 0.8 de umbral.
4. Vecino más cercano: 1000 iteraciones, peso de 10 a las observaciones positivas y 0.5 de umbral.
5. Regresión lineal para variables numéricas y regresión logística para categóricas: 1000 iteraciones, peso de 50 a las observaciones positivas y 0.8 de umbral.
6. Regresión estocástica para numéricas y regresión logística para categóricas: 2500 iteraciones, peso de 100 a las observaciones positivas y 0.8 de umbral.
7. Imputación múltiple: 1000 iteraciones, peso de 50 a las observaciones positivas y 0.8 de umbral.
8. MissForest: 2500 iteraciones, peso de 10 a las observaciones positivas y 0.4 de umbral.

XGBoost

Se enumera cada método de imputación con sus parámetros óptimos para el conjunto de datos mobile.

1. Listwise: profundidad máxima 8, 0.2 de learning rate, 1000 de scale_pos_weight y 0.5 de umbral.
2. Media para variables numéricas y moda para categóricas: profundidad máxima 10, 0.2 de learning rate, 10 de scale_pos_weight y 0.4 de umbral.
3. Mediana para numéricas y moda para categóricas: profundidad máxima 6, 0.2 de learning rate, 10 de scale_pos_weight y 0.4 de umbral.
4. Vecino más cercano: profundidad máxima 6, 0.2 de learning rate, 100 de scale_pos_weight y 0.6 de umbral.
5. Regresión lineal para numéricas y regresión logística para categóricas: profundidad máxima 6, 0.3 de learning rate, 100 de scale_pos_weight y 0.8 de umbral.
6. Regresión estocástica para numéricas y regresión logística para categóricas: profundidad máxima 10, 0.3 de learning rate, 10 de scale_pos_weight y 0.4 de umbral.
7. Imputación múltiple: profundidad máxima de 6, 0.2 de learning rate, 10 de scale_pos_weight y 0.3 de umbral

8. MissForest: profundidad máxima 8, 100 estimadores, 100 de scale_pos_weight y 0.8 de umbral.

CatBoost

En este apartado se enumera cada método de imputación con sus parámetros óptimos con el algoritmo CatBoost para el conjunto de datos mobile

1. Listwise: profundidad 8, 0.2 de learning rate, 1 de scale_pos_weight y umbral de 0.3.
2. Media para numéricas y moda para categóricas: profundidad 10, 0.1 de learning rate, 40 de scale_pos_weight y umbral de 0.6.
3. Mediana para numéricas y moda para categóricas: profundidad 6, 0.1 de learning rate, 1 de scale_pos_weight y umbral de 0.3.
4. Vecino más cercano: 6 de profundidad, 0.1 de learning rate, 1 de scale_pos_weights y 0.3 de umbral.
5. Regresión lineal para variables numéricas y regresión logística para categóricas: profundidad 6, 0.2 de learning rate, 10 de scale_pos_weight y umbral de 0.5.
6. Regresión estocástica para variables numéricas y regresión logística para categóricas: profundidad 8, 0.1 de learning rate, 40 de scale_pos_weight y umbral de 0.5.
7. Imputación múltiple: profundidad de 10, 0.2 de learning rate, 1 de scale_pos_weight y 0.3 de umbral.
8. MissForest: profundidad 8, 0.2 de learning rate, 40 de scale_pos_weight y umbral de 0.6.

4.5 Comparación de los modelos

En esta sección se comparan los resultados para cada conjunto de datos por separado. Se compara el tiempo de respuesta y las métricas de los distintos modelos con sus correspondientes parámetros y métodos de imputación.

4.5.1 Comparación conjunto de datos banca electrónica

Se comienza analizando los resultados del conjunto de datos de banca electrónica, del que tenemos sus resultados en las Tablas 4.1 y 4.2. Se puede observar que el algoritmo más rápido a la hora de predecir es el de regresión logística con un tiempo medio de 2.42 ms frente a los 3.32 ms de XGBoost y 4.93 de CatBoost. Las diferencias de tiempo son considerables, ya que

la regresión logística tarda de media, menos de la mitad del tiempo que tarda el algoritmo de CatBoost.

En cuanto a las métricas, con un vistazo general a la tabla de resultados, se puede observar que los modelos en los que se usa regresión logística como algoritmo de clasificación obtienen los peores resultados en todas las métricas, con bastante diferencia respecto a los que usan XGBoost y CatBoost. Se puede observar que estos dos últimos obtienen resultados similares. El modelo que consigue un mayor recall es el de XGBoost usando regresión lineal como método de imputación, el que obtiene una mayor precisión y f1-score es el de XGBoost con missForest como método de imputación.

Comparando los resultados de los métodos de tratamiento de valores faltantes, podemos observar que el método de listwise es el que peor resultados obtiene en los tres algoritmos y con bastante diferencia respecto al resto de métodos, probablemente se deba a la gran cantidad de datos faltantes que tenemos y la consiguiente eliminación de muchas observaciones. En dos de los tres algoritmos la imputación con la mediana obtiene mejores resultados que imputando con la media, mientras que la imputación con regresión estocástica se obtienen mejores resultados en los tres algoritmos que la imputación regresión lineal. En cuanto al método del vecino más cercano (KNN) se obtienen mejores resultados que imputando con un valor central y resultados similares que con regresión lineal, regresión estocástica o imputación múltiple.

El modelo con mejores métricas es el de XGBoost usando missForest para imputar los valores faltantes. Con este modelo obtenemos unas métricas de 76.27% de recall, 88.54% de precision y 81.94% de f1-score.

4.5.2 Comparación conjunto de datos banca móvil

Los resultados del conjunto de datos de banca móvil están presentes en las Tablas 4.3 y 4.4. El algoritmo con un menor tiempo medio de predicción vuelve a ser la regresión logística con una media de 2.53 ms, seguido del XGBoost con 3.56 ms y de CatBoost con 5.09 ms.

Los modelos mediante regresión logística tienen resultados muy inferiores a los otros dos algoritmos si miramos cualquiera de las tres métricas. XGBoost y CatBoost obtienen resultados muy similares al igual que en el conjunto de datos anterior. El modelo que mejor recall obtiene es XGBoost con el vecino más cercano como método de imputación con un recall de 70.54%; en cuanto a la precision el mejor modelo es XGBoost con CatBoost con regresión lineal ya que obtiene una precision del 94.62% y el CatBoost con missForest es el que mejor f1-score tiene con un 79.65%.

Podemos observar que, en los tres algoritmos, todas las métricas dan un 0% empleando listwise debido a la cantidad muy pequeña de datos que quedan al usar este método. Imputando con la mediana se obtienen mejores métricas en dos de los tres algoritmos, siendo las diferencias muy considerables en la regresión logística. Con la regresión estocástica se consi-

guen mejores resultados que la regresión lineal también en dos (regresión logística y XGBoost) de los tres algoritmos. En el caso de imputación con el vecino más cercano las métricas son estándar y no consigue sobresalir. En cambio, empleando missForest como método de imputación obtenemos el mejor modelo usando Catboost con unas métricas de 69.77% de recall, 92.78% de precision y 79.65% de f1-score.

4.6 Análisis mejores modelos

Una vez vistos los mejores modelos para cada conjunto de datos, se procede a ver que variables tuvieron una mayor importancia. Comenzamos con el modelo de banca electrónica. El mejor modelo para este conjunto de datos fue un XGBoost con el algoritmo missForest como método de imputación. En la Figura 4.1 podemos observar las 20 variables más importantes. Se consigue la importancia de las variables con la función *feature_importances_* del modelo XGBoost.

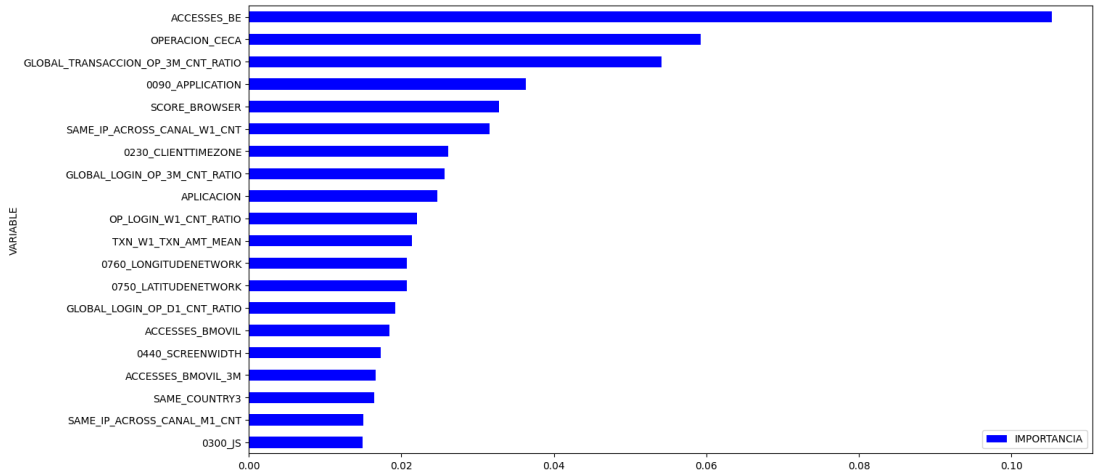

Figura 4.1: Representación de las 20 variables más importantes del modelo de banca electrónica

Se puede observar que la variable con mayor importancia es ACCESSES_BE que indica la cantidad de accesos a la banca electrónica del cliente, esta Variable tiene casi el doble de importancia que la segunda variable más importante. Las siguientes variables con importancia son OPERACION_CECA, GLOBAL que indica el valor de la operación para CECA, también tiene importancia el ratio de transacciones en los últimos tres meses o la aplicación que se usó. Otra variable con bastante importancia es SCORE_BROWSER que fue una variable creada anteriormente que indica la similitud entre el navegador usado para la operación y los navegadores habituales del cliente. Otras variables con menor importancia son la zona horaria desde la que se realiza la operación, latitud y longitud, accesos a banca móvil del cliente o

el ratio de logins del cliente.

Si se analizan las variables con menor importancia encontramos cinco variables que tuvieron una importancia de prácticamente 0, por lo que habríamos obtenido prácticamente los mismos resultados sin ellas. Estas variables son IN_EALE (si el cliente tiene alertas móviles), 0430_SCREENTOUCH (si el dispositivo es táctil), IN_BMOVIL (si el usuario tiene banca móvil), 0700_CONTINENTCODE y IN_BE (si el cliente tiene banca electrónica).

A continuación, se procede a analizar la importancia de las variables del mejor modelo de banca móvil. Este modelo es un CatBoost con missForest como algoritmo de imputación. En la Figura 4.2 tenemos el gráfico con las 20 variables con mayor importancia.

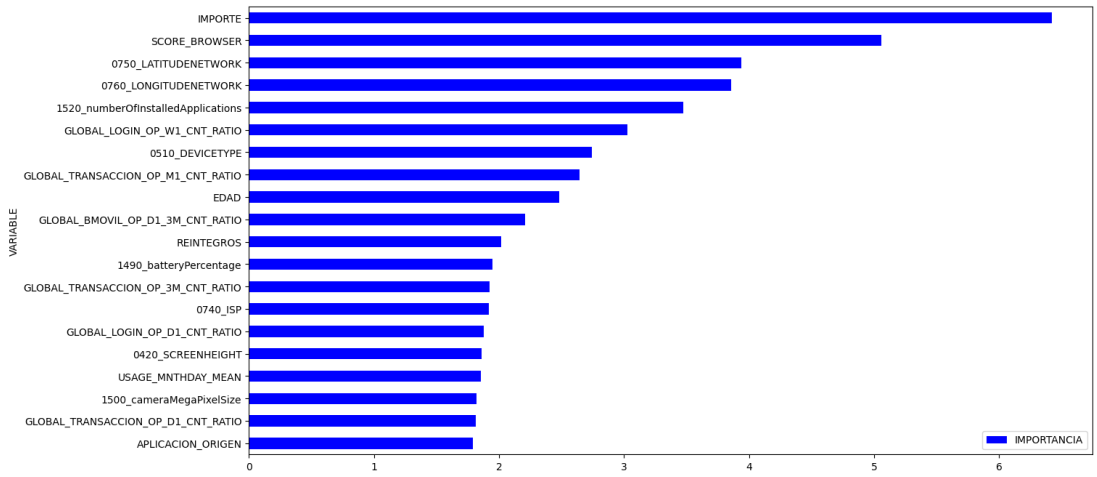

Figura 4.2: Representación delas 20 variables más importantes del modelo de banca móvil

En este caso la variable con mayor importancia es la que indica el importe de la transacción, seguida de la variable SCORE_BROWSER que fue creada en capítulos anteriores y también tenía importancia en banca electrónica. La latitud y longitud también tienen más importancia que en el modelo banca electrónica. La quinta variable más importante nos indica la cantidad de aplicaciones instaladas en el dispositivo. Otras variables con importancia son el tipo de dispositivo, ratio de transacción en el último mes y tres meses, la edad del cliente, porcentaje de batería del dispositivo, el proveedor de INTERNET del dispositivo (0740_ISP) o la aplicación de origen.

En las variables con menor importancia, hay seis variables con cero importancia. Estas variables son 0500_DEVICEROOTED (si el dispositivo está rooteado), 0670_WIFISECURITYSCORE, 0400_PLATFORM(plataforma del dispostivo), SAME_COUNTRY2 (creada anteriormente), DIVISA y IN_BMOVIL.

Capítulo 5

Conclusiones

Conclusiones En este capítulo se incluirán las conclusiones y el trabajo que se podría realizar a futuro.

5.1 Conclusiones

Una vez vistos los distintos modelos entrenados y sus métricas, se observa que:

- En el conjunto de datos de banca electrónica los mejores resultados en la muestra de test se obtienen con el algoritmo XGBoost usando el algoritmo missForest como método de imputación consiguiendo las siguientes métricas: 76.27% de recall, 88.54% de precision y 81.94% de f1-score. En el caso de banca móvil, los mejores resultados en la muestra de test los obtiene el CatBoost con missForest como método de imputación consiguiendo las siguientes métricas: 69.77% de recall, 92.78% de precision y 79.65% de f1-score.
- El algoritmo que peores resultados obtiene en ambos conjuntos de datos es el de regresión logística. Por su contra, XGBoost y CatBoost obtienen los mejores resultados, siendo estos similares. Era esperado que estos dos algoritmos tuvieran mejores resultados ya que son dos algoritmos más flexibles. Además, son dos algoritmos muy usados en la actualidad.
- El algoritmo que menos tiempo tarda en predecir es el de regresión logística, seguido de XGBoost y CatBoost. Cabe destacar que los tiempos de predicción de los tres algoritmos son bastante bajos.
- La técnica de imputación que mejores resultados obtiene es el algoritmo de missForest; pero las diferencias de las métricas no son demasiados grandes. Si se quisiera una técnica de imputación más simple, la imputación con la mediana o con el vecino más cercano también obtienen buenos resultados.

- Si se descartan los modelos en los que se usa missForest como técnica de imputación, el mejor modelo para banca electrónica sería un XGBoost con imputación mediante regresión estocástica en variables numéricas y regresión logística en variables categóricas. En el caso de banca móvil, el mejor modelo sería un XGBoost imputando con la mediana para variables numéricas y moda para categóricas.
- La variable con mayor importancia en el mejor modelo de banca electrónica es el número de accesos del cliente a propia banca electrónica. En el caso de la banca móvil es el importe de la transacción. En ambos modelos la variable creada SCORE_BROWSER está entre las 5 variables con mayor importancia.

5.2 Trabajo futuro

A continuación, se presentan algunas áreas en las que se podría trabajar a futuro de cara a intentar conseguir mejores resultados.

- Este trabajo se ha centrado en tres algoritmos de clasificación, pero hay otros muchos algoritmos que podrían ser usados como redes neuronales, random forest, etc.
- Hemos empleado un total de ocho técnicas de imputación, pero podrían usarse otras técnicas como el algoritmo de máxima verosimilitud [10]. También se podrían combinar las técnicas usadas para variables numéricas con las usadas para variables categóricas, por ejemplo, podría usarse la imputación con la mediana para variables numéricas y la regresión logística para categóricas.
- Debido a la gran cantidad de datos faltantes en determinadas variables, sería interesante saber la causa de la pérdida de esta información y también la mejora en la recolección de los datos para poder reducir al máximo el número de valores faltantes.
- Se podrían intentar la creación de otras variables de interés que pudieran tener poder discriminatorio.
- Otro trabajo a futuro podría ser incidir más en la selección de variables para intentar reducir el número de variables eliminando aquellas variables que no tengan importancia o aquellas que tengan una importancia muy pequeña.
- Otro punto interesante sería el estudio en profundidad el efecto de las variables predictoras y sus interacciones entre sí.
- El último punto a realizar sería la puesta en producción de los dos modelos para poder detectar las transacciones fraudulentas tanto en banca electrónica como en banca móvil

Apéndices

Apéndice A

Descripción de las variables

En este capítulo adicional, se hará una descripción breve de cada variable presente en el problema.

- FECHA: fecha y hora en que se realizó la operación.
- APLICACION: código de la aplicación (banca electrónica, banca móvil, etc) que realiza la operación.
- OPERACION_CECA: valor de la operación para CECA.
- RESULTADO: resultado de la operación.
- CANAL: medio que realizó la transacción (online: banca electrónica /mobile: banca móvil).
- OPERACION: indica el tipo de operación (transacción).
- NAVEGADOR: navegador con que se hace la transacción.
- ID_PAIS_DESTINO: código del país destino de la transacción.
- ID_BANCO_DESTINO: código del banco destino de la transacción.
- IMPORTE: importe de la transacción.
- DIVISA: divisa en que se realiza la transacción.
- FECHA_PLANIF_OPER: fecha y hora en que se planifica la transacción.
- APLICACION_ORIGEN: aplicación origen de la transacción (BizumEnviarDinero, Transferencias, PagoMoviles, TjVirtualAltaTarjeta, etc)
- FRAUDE: indica si la transacción es fraudulenta (Y) o no (N).

- 0090_APPLICATION: id de la aplicación (1037151 / 52327).
- 0190_BROWSER: nombre del buscador (Android, Firefox, Safari, etc).
- 0200_BROWSERVERSION: versión del buscador.
- 0230_CLIENTTIMEZONE: offset en minutos de supuesta ubicación geográfica.
- 0290_HISTORYLENGTH: número de urls en el historial del navegador.
- 0300_JS: indica si el dispositivo tenía activado JavaScript (1/0).
- 0340_USERAGENT: agente del usuario del dispositivo.
- 0390_OS: sistema operativo del dispositivo (Windows 10, iOs - 15.6.1, etc).
- 0400_PLATFORM: plataforma del dispositivo (android, iOs, Win32, etc).
- 0410_SCREENDPI: DPI de la pantalla del dispositivo.
- 0420_SCREENHEIGHT: altura de la pantalla del dispositivo.
- 0430_SCREENTOUCH: indica si el dispositivo es táctil (1/0).
- 0440_SCREENWIDTH: ancho de la pantalla del dispositivo.
- 0450_BATTERYCHARGING: indica si el dispositivo está cargando (true) o no (false).
- 0460_CALCSTATE: indica si cálculo de elementos de riesgo se ha completado en el momento que el SDK móvil envía solicitud de notificación (phase1_incomplete/ phase1_complete/ calc_complete).
- 0470_CPUTYPE: modelo de CPU del dispositivo (arm64-v8a, arm64, armeabi-v7a, arm).
- 0480_DATACARRIER: si el dispositivo está conectado a la red de datos móviles (true/-false).
- 0490_DEVICELANGUAGE: código del lenguaje del dispositivo.
- 0500_DEVICEROOTED: si el dispositivo está rooteado o jailbroken (true) o no (false).
- 0510_DEVICETYPE: modelo del dispositivo (iPhone12,1, Xiaomi Redmi Note 9S, etc).
- 0520_EMULATOR: si la aplicación se ejecuta en un emulador (true/false).
- 0530_ISCALLINPROGRESS: estado de llamadas del dispositivo (yes/no/ringing).
- 0550_LINECARRIER: nombre del portador de la línea (Movistar, Vodafone, etc).

- 0570_MRSTAPPCOUNT: número de herramientas de soporte móvil remoto instaladas en el dispositivo.
- 0590_ROOTHIDERS: indica si hay root hiders en el dispositivo (true/false).
- 0630_SMSSTEALERS: indica si hay ladrones de SMS en el dispositivo (true/ false).
- 0640_TIMEZONE: offset del tiempo en minutos de la supuesta ubicación geográfica.
- 0670_WIFISECURITYSCORE: si se está usando conexión wifi segura (0) o no segura (1000).
- 0690_CITY: supuesta ciudad en la que se encuentra el dispositivo.
- 0700_CONTINENTCODE: supuesto continente en que se encuentra el dispositivo.
- 0710_COUNTRY: supuesto país en el que se encuentra el dispositivo.
- 0720_IPCLASS: tipo de dirección ip (A, B, C).
- 0730_IPTIMEZONE: offset del tiempo universal coordinado en minutos de la ubicación geográfica.
- 0740_ISP: supuesto proveedor de servicios de internet del dispositivo (Jazztel, Yoigo).
- 0750_LATITUDENETWORK: supuestas coordenadas de la latitud del dispositivo.
- 0760_LONGITUDENETWORK: supuestas coordenadas de longitud del dispositivo.
- 0780_REGION: ubicación de la dirección ip.
- 1490_batteryPercentage: valor del porcentaje de batería del dispositivo.
- 1500_cameraMegaPixelSize: tamaño en megapíxeles de la cámara del dispositivo.
- 1510_developerSettings: si opciones de desarrollador están habilitadas (true/false).
- 1520_numberOfInstalledApplications: número de aplicaciones instaladas.
- 1530_unlockType: método que usa el usuario para desbloquear el dispositivo.
- 1560_newLanguage: indica si el idioma del dispositivo es nuevo en la cuenta.
- FECHA_DIA: fecha en que se realizó la operación.
- ACCESSES_BMOVIL_W1: cantidad de accesos a banca móvil en la semana previa.
- ACCESSES_BMOVIL_M1: cantidad de accesos a banca móvil en el mes previo.

- ACCESSES_BMOVIL_3M: cantidad de acceso a banca móvil en los 3 meses previos.
- ACCESSES_BMOVIL: cantidad de accesos a la banca móvil.
- ACCESSES_BE_W1: cantidad de accesos a banca electrónica en la semana previa.
- ACCESSES_BE_M1: cantidad de accesos a banca electrónica en el mes previo.
- ACCESSES_BE_3M: cantidad de acceso a banca electrónica en los 3 meses previos.
- ACCESSES_BE: cantidad de accesos a la banca electrónica.
- ACCESSES_BMOVIL_W1_ZS: cantidad normalizada de accesos a banca móvil en la semana previa.
- ACCESSES_BMOVIL_M1_ZS: cantidad normalizada de accesos a banca móvil en el mes previo.
- ACCESSES_BMOVIL_3M_ZS: cantidad normalizada de accesos a banca móvil en los 3 meses previos.
- ACCESSES_BMOVIL_ZS: cantidad normalizada de accesos a banca electrónica.
- ACCESSES_BE_W1_ZS: cantidad normalizada de accesos a banca electrónica en la semana previa.
- ACCESSES_BE_M1_ZS: cantidad normalizada de accesos a banca electrónica en el mes previo.
- ACCESSES_BE_3M_ZS: cantidad normalizada de accesos a banca electrónica en los 3 meses previos.
- ACCESSES_BE_ZS : cantidad normalizada de accesos a banca electrónica.
- TXN_W1_TXN_AMT_MEAN: importe medio de las transacciones de la semana previa.
- TXN_M1_TXN_AMT_MEAN: importe medio de las transacciones del último mes.
- TXM_3M_TXN_AMT_MEAN: importe medio de las transacciones de los 3 meses anteriores.
- TXN_M1_TXN_AMT_STD: desviación típica de las transacciones del último mes.
- TXM_3M_TXN_AMT_STD: desviación típica de las transacciones de los 3 meses anteriores.

- TXN_W1_TXN_MAX_AMT: importe máximo de la última semana.
- TXN_M1_TXN_MAX_AMT: importe máximo del último mes.
- TXM_3M_TXN_MAX_AMT: importe máximo de los últimos 3 meses.
- OP_LOGIN_W1_CNT_RATIO: ratio de las operaciones login en la semana previa.
- OP_SESION_W1_CNT_RATIO: ratio de las operaciones sesión en la semana previa.
- OP_TRANSACTION_W1_CNT_RATIO: ratio de las operaciones transacción en la semana previa.
- OP_LOGIN_M1_CNT_RATIO: ratio de las operaciones login en el mes previo.
- OP_SESION_M1_CNT_RATIO: ratio de las operaciones sesión en el mes previo.
- OP_TRANSACTION_M1_CNT_RATIO: ratio de las operaciones transacción en el mes previo.
- OP_LOGIN_3M_CNT_RATIO: ratio de las operaciones login en los 3 meses previos.
- OP_SESION_3M_CNT_RATIO: ratio de las operaciones sesión en los 3 meses previos.
- OP_TRANSACTION_3M_CNT_RATIO: ratio de las operaciones transacción en los 3 meses previos.
- GLOBAL_BMOVIL_OP_D1_W1_CNT_RATIO: ratio (último día/ última semana) de transacciones realizadas en la banca móvil.
- GLOBAL_BMOVIL_OP_D1_M1_CNT_RATIO: ratio (último día/ último mes) de transacciones realizadas en la banca móvil.
- GLOBAL_BMOVIL_OP_D1_3M_CNT_RATIO: ratio (último día/ últimos 3 meses) de transacciones realizadas en la banca móvil.
- GLOBAL_BE_OP_D1_W1_CNT_RATIO: ratio (último día/ última semana) de transacciones realizadas en la banca electrónica.
- GLOBAL_BE_OP_D1_M1_CNT_RATIO: ratio (último día/ último mes) de transacciones realizadas en la banca electrónica.
- GLOBAL_BE_OP_D1_3M_CNT_RATIO: ratio (último día/ últimos 3 meses) de transacciones realizadas en la banca electrónica.
- GLOBAL_LOGIN_OP_D1_CNT_RATIO: ratio de logins del cliente en el último día.

- GLOBAL_TRANSACCION_OP_D1_CNT_RATIO: ratio de transacciones hechas en el último día.
- GLOBAL_LOGIN_OP_W1_CNT_RATIO: ratio de logins hechos en la última semana.
- GLOBAL_TRANSACCION_OP_W1_CNT_RATIO: ratio de transacciones hechas en la semana anterior.
- GLOBAL_LOGIN_OP_M1_CNT_RATIO: ratio de logins hechos en el mes anterior.
- GLOBAL_TRANSACCION_OP_M1_CNT_RATIO: ratio de transacciones hechas en el mes anterior.
- GLOBAL_LOGIN_OP_3M_CNT_RATIO: ratio de logins hechos durante los últimos 3 meses.
- GLOBAL_TRANSACCION_OP_3M_CNT_RATIO: ratio de transacciones hechas en los últimos 3 meses.
- USAGE_HOURS_WK_MEAN: media de horas de uso semanales.
- USAGE_HOURS_WKND_MEAN: media de horas de uso en fin de semana.
- USAGE_HOURS_WK_STD: desviación típica de uso semanal.
- USAGE_HOURS_WKND_STD: desviación típica de uso en fines de semana.
- USAGE_WKDAY_MEAN: uso medio en horas en días de semana.
- USAGE_WKDAY_STD: desviación típica de uso en días de semana.
- USAGE_MNTHDAY_MEAN: media de uso en horas cada mes.
- USAGE_MNTHDAY_STD: desviación típica de uso mensual.
- IN_BE: se indica si el usuario tiene banca electrónica (S/N).
- IN_BMOVIL: se indica si el usuario tiene banca móvil (S/N).
- IN_EALE: se indica si el usuario tiene tiene alertas móviles (S/N).
- USO_TARJETAS_12M: indicador entre 0 y 1 de cuanto ha usado la tarjeta el cliente.
- PROPORCION_ELECTRONICA: indicador entre 0 y 1 que indica la proporción de transacciones electrónicas del cliente.

- PROPORCION_TARJETAS: indicador entre 0 y 1 que indica la proporción de transacciones con tarjetas del cliente.
- MENSAJERIA: indica si tiene alertas móviles y/o de mensajes.
- DIGITALIDAD_RELATIVA: coeficiente que indica el nivel de digitalidad relativa del cliente.
- EDAD: edad del cliente.
- DIGITALIDAD_RELATIVA_EDAD: cociente entre digitalidad relativa y edad.
- IN_DIGITAL: indica el tipo de digitalidad del cliente.
- REINTEGROS: cantidad de reintegros realizados.
- COUNTRY_LIST: países habituales del cliente.
- NAVEGADOR_LIST: navegadores habituales desde los que el cliente realiza operaciones
- COUNTRIES_W1_CNT: cantidad de países en los que ha estado el cliente en los últimos 7 días.
- COUNTRIES_M1_CNT: cantidad de países en los que ha estado en el último mes.
- COUNTRIES_3M_CNT: cantidad de países en los que ha estado en los últimos 3 meses.
- COUNTRIES_SWITCH_30_MIN_W1_CNT: cantidad de países desde los que el cliente realizó operaciones de riesgo en la última semana.
- COUNTRIES_SWITCH_30_MIN_M1_CNT: cantidad de países desde los que el cliente realizó operaciones de riesgo en el último mes.
- SAME_IP_ACROSS_CANAL_W1_CNT: cantidad de direcciones IP iguales desde las cuales se conectó el usuario en los últimos 7 días.
- SAME_IP_ACROSS_CANAL_M1_CNT : cantidad de direcciones IP iguales desde las cuales se conectó el usuario en el último mes..

Bibliografía

[1] Banco de España, "Estudio sobre hábitos de uso del efectivo," https://www.bde.es/f/webbde/INF/MenuVertical/BilletesYMonedas/Estudios_e_informes/Estudio_sobre_habitos_efectivo/InformeEjecutivo_Estudio_habitos_uso_efectivo_VF_CE.pdf, 2022, informe ejecutivo.

[2] S. Van Buuren, *mice: Multivariate Imputation by Chained Equations in R.* Journal of Statistical Software, 2011.

[3] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 785–794. [En línea]. Disponible en: http://doi.acm.org/10.1145/2939672.2939785

[4] D. B. RUBIN, *Inference and missing data*, 2nd ed. Oxford University Press, 1976.

[5] P. D. Allison, *Missing data.* Sage publications, 2001.

[6] R. J. Little, "A test of missing completely at random for multivariate data with missing values," *Journal of the American Statistical Association*, vol. 83, no. 404, pp. 1198–1202, 1988.

[7] J. Carpenter and M. G. Kenward, *Multiple Imputation and Its Application.* Wiley, 2013.

[8] D. J. Stekhoven and P. Bühlmann, "Missforest – non-parametric missing value imputation for mixed-type data," *Bioinformatics*, vol. 28, no. 1, pp. 112–118, 2012.

[9] D. B. Rubin, *Multiple Imputation for Nonresponse in Surveys.* New York: John Wiley & Sons, 1987.

[10] ——, "Inference and missing data," *Biometrika*, vol. 63, no. 3, pp. 581–592, 1976.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[12] M. Johnson, "Autoimpute: Python package for multivariate imputation," 2021. [En línea]. Disponible en: https://pypi.org/project/autoimpute/

[13] "Impyte: A python package for data imputation," https://github.com/andirs/impyte, 2023, version 0.1.0.

[14] D. N. Reshef, "Missingpy: Missing data imputations in Python," https://github.com/epsilon-machine/missingpy, 2018.

[15] S. Inc, *fuzzywuzzy: Fuzzy String Matching in Python*, 2023. [En línea]. Disponible en: https://github.com/seatgeek/thefuzz

[16] F. Circusio, *pycountry: Fuzzy String Matching in Python*, 2022. [En línea]. Disponible en: https://github.com/flyingcircusio/pycountry

[17] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," https://scikit-learn.org/stable/, 2011. [En línea]. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html

[19] L. Prokhorenkova, G. Gusev, A. Vorobev, A. Dorogush, and A. Gulin, "Catboost: gradient boosting with categorical features support," https://catboost.ai, 2017.