Trabajo Fin de Máster

Predicción de fraude on-line en transacciones sobre canales digitales

Diego Calderón Barreda

Máster en Técnicas Estadísticas

Curso 2022-2023

Propuesta de Trabajo Fin de Máster

Título en galego: Predición de fraude en liña en transaccións sobre canais dixitais
Título en español: Predicción de fraude online en transacciones sobre canales digitales
English title: Online fraud prediction in transactions through digital channels
Modalidad: Modalidad B
Autor/a: Diego Calderón Barreda, Universidade de Vigo
Director/a: Rubén Fernández Casal, Universidade da Coruña; Manuel Oviedo de la Fuente, Universidade da Coruña
Tutor/a: Jorge García Romarís, ABANCA
Breve resumen del trabajo: El trabajo a realizar consistirá en desarrollar un modelo que prediga el fraude en las transacciones on-line. Este modelo afronta la dificultad de que las clases a predecir (fraude, no-fraude) son altamente desbalanceadas (volumen altísimo de transacciones con pocas transacciones fraudulentas), por lo que se aplicarán diferentes técnicas de re-muestreo, generación de datos ficticios (GANs), para entrenar modelos siguiendo distintas técnicas de Machine Learning.
Recomendaciones: Conocimientos avanzados en SQL y programación en Python.
Otras observaciones: Bolsa de ayuda de 800€ mensuales

Don/doña Rubén Fernández Casal, de la Universidade da Coruña, don/doña Manuel Oviedo de la Fuente, de la Universidade da Coruña y don/doña Jorge García Romarís de ABANCA, informan que el Trabajo Fin de Máster titulado

Predicción de fraude on-line en transacciones sobre canales digitales

fue realizado bajo su dirección por don/doña Diego Calderón Barreda para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, dan su conformidad para su presentación y defensa ante un tribunal.

En A Coruña, a 14 de julio de 2023.

FERNANDEZ CASAL RUBEN - 33317724Q
Firmado digitalmente por FERNANDEZ CASAL RUBEN - 33317724Q
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-33317724Q, givenName=RUBEN, sn=FERNANDEZ CASAL, cn=FERNANDEZ CASAL RUBEN - 33317724Q
Fecha: 2023.07.13 13:34:17 +02'00'

El/la director/a:
Don/doña Rubén Fernández Casal

Manuel Oviedo de la Fuente
Firmado digitalmente por Manuel Oviedo de la Fuente
Fecha: 2023.07.13 13:22:58 +02'00'

El/la director/a:
Don/doña Manuel Oviedo de la Fuente

El/la tutor/a:
Don/doña Jorge García Romarís

El/la autor/a:
Don/doña Diego Calderón Barreda

Declaración responsable. Para dar cumplimiento a la Ley 3/2022, de 24 de febrero, de convivencia universitaria, referente al plagio en el Trabajo Fin de Máster (Artículo 11, Disposición 2978 del BOE núm. 48 de 2022), **el/la autor/a declara** que el Trabajo Fin de Máster presentado es un documento original en el que se han tenido en cuenta las siguientes consideraciones relativas al uso de material de apoyo desarrollado por otros/as autores/as:

- Todas las fuentes usadas para la elaboración de este trabajo han sido citadas convenientemente (libros, artículos, apuntes de profesorado, páginas web, programas,...)
- Cualquier contenido copiado o traducido textualmente se ha puesto entre comillas, citando su procedencia.
- Se ha hecho constar explícitamente cuando un capítulo, sección, demostración,...sea una adaptación casi literal de alguna fuente existente.

Y, acepta que, si se demostrara lo contrario, se le apliquen las medidas disciplinarias que correspondan.

Índice general

Resumen

Resumen en español

El fraude es un grave problema que afecta a todas las entidades bancarias. Es de vital importancia minimizar las operaciones fraudulentas no sólo para evitar pérdidas de dinero sino también para prevenir que se debilite la relación del banco con los clientes. En este trabajo se utiliza un conjunto de datos de operaciones de la entidad ABANCA donde las operaciones fraudulentas componen tan sólo el 1.4 %, esto son 1 373 casos, del total y se trata por tanto de clases desbalanceadas. El objetivo es entrenar un modelo de clasificación binaria capaz de detectar el fraude para poder predecir operaciones futuras en tiempo real, por lo tanto interesa que esta se realice rápidamente.

Estudiamos el desempeño de tres modelos: regresión logística, CatBoost y redes neuronales. Para tratar el desbalanceo se usan cuatro métodos para cada modelo: RandomUnderSampler, RandomOverSampler, SMOTE y CTGAN. Teniendo en cuenta el caso en el que entrenamos cada modelo con la muestra original se tratan entonces 15 casos distintos. Estudiamos de ellos diferentes métricas usuales, incluyendo también una función de coste propia, para comparar su eficacia y determinar el más adecuado.

Concluimos que el modelo más adecuado es CatBoost remuestreando con RandomOverSampler. Con este modelo somos capaces de clasificar correctamente el 84 % de los casos fraudulentos y obtenemos el menor coste de todos los modelos estudiados.

English abstract

Fraud is a serious challenge that affects all banking institutions. Minimizing fraudulent operations is vital. Not only in order to prevent monetary losses but also to avoid the weakening of the relationship between the bank and the clients. A dataset corresponding to operations of the entity ABANCA is used in this work. Only 1.4 % of the provided data corresponds to fraudulent data, that is 1 373 operations, so the classes are unbalanced. The goal is to find a binary classification model that is able to detect fraudulent operations. We aim to implement this model so it makes real time predictions on operations, so it must be swift.

We study the performance of three models: logistic regression, CatBoost and neural networks. To tackle the unbalanced classes we use four resampling methods: RandomUnderSampler, RandomOverSampler, SMOTE y CTGAN. Taking into account the case without resampling, we have worked with a total of 15 different cases. We will apply different usual metrics, including our own cost function, in order to compare them and find the most convenient one.

We conclude that the best model for the problem is CatBoost resampling with RandomOverSampler. This model allows us to successfully classify 84 % of the fraudulent cases and it returns the lowest cost of all the studied models.

Capítulo 1

Introducción

1.1. El problema del fraude

El desarrollo de internet y las tecnologías ha conllevado un aumento de las operaciones realizadas mediante medios online. Cada vez se realiza un mayor número de operaciones a través de internet, llegando en 2021 a ser España el tercer país europeo en términos de facturación por ventas online con 68 400 millones de euros, más de tres cuartos de los usuarios de internet españoles realizan compras por internet regularmente.

Este aumento de las tecnologías y la familiarización de la gente joven con ellas ha supuesto que el dinero en efectivo vaya perdiendo peso poco a poco. Esta forma de pago se vio afectada por la pandemia del COVID-19, la cual fue una época de cambio para muchas personas. La gente prefería no realizar operaciones con dinero físico cuando era posible y esto provocó un aumento del volumen de las operaciones registradas por los bancos.

La pandemia no solo afectó al uso del dinero en efectivo sino que cambió los hábitos de compra de mucha gente. El confinamiento hizo que solamente fuese posible realizar compras online de una gran cantidad de productos y cuando se recuperó la normalidad de nuevo el comportamiento de la gente no volvió a su forma previa. Estos cambios de comportamiento provocan que los modelos entrenados previamente para predecir operaciones fraudulentas sean menos eficaces. El aumento del volumen de operaciones y el cambio en el comportamiento de los usuarios provocan la necesidad de seguir estudiando este problema y crear nuevos modelos para prevenir el fraude.

Una vez un tercero obtiene tus datos bancarios, este puede emplearlos para realizar transacciones online en tu nombre. Existen diferentes técnicas que se emplean para cometer conseguir esta información, las más habituales son:

- Fraude por suplantación de identidad o phishing: consiste en la suplantación de la identidad de un bancos, institución o entidad para engañar a una persona y que envíe su información bancaria.
- Fraude por robo de datos o skimming: consiste en el robo de datos al cliente al realizar él una operación normal.

Cuando se realiza una operación la entidad bancaria correspondiente recibe una serie de parámetros aportando gran cantidad de información sobre ella. Se reciben datos tanto del propio usuario, como del dispositivo desde el que se realiza o de la operación en sí. El trabajo de la entidad es analizar esos datos en tiempo real y dar una respuesta sobre si supones que es tu cliente, el dueño de la cuenta o tarjeta, quien realiza la operación o se trata de una operación fraudulenta.

Este trabajo trata el problema de predicción de fraude en ABANCA. La cantidad de datos disponible en la entidad es masiva puesto que se registra una media del orden de 1 000 000 operaciones diarias de las cuales solo una pequeña parte es fraudulenta. En concreto en torno a un 0.0012 % de las operaciones son fraudulentas, lo que equivale a unas 12 operaciones fraudulentas diarias luego claramente esto es un problema de datos desbalanceados.

El conjunto de datos es una pequeña muestra de los datos totales, ya que trabajar con bases de datos tan masivas genera problemas en términos de memoria y tiempos de computación. Se concretará más sobre la muestra empleada en el Capítulo 3.

1.2. Objetivos

El objetivo de este trabajo es construir un modelo que sea capaz de predecir el fraude a tiempo real en las operaciones sobre canales online en datos de la entidad ABANCA. Para esto se tienen como objetivos:

- Buscar un modelo de clasificación binaria que prediga el fraude.
- Preparar un programa que tome un dato real, lo procese para que concuerde con el input del modelo y realice la predicción.
- Obtener un tiempo de respuesta pequeño, menor a un segundo.

Al buscar una predicción en tiempo real entra en juego el tiempo de evaluación del modelo. Conviene buscar un modelo con un corto tiempo de respuesta ya que es de interés hacer esperar al cliente el menor tiempo posible para permitir realizar una operación.

Interesa tener en cuenta el número de falsos positivos que produce nuestro modelo ya que estos son casos de operaciones no fraudulentas que se etiquetan como fraudulentas, posiblemente deteniendo la operación. El no permitir realizar una operación a un cliente conlleva una pérdida de confianza del cliente sobre la entidad, lo cual es un resultado muy indeseable. Por otro lado, permitir un número alto de faltos negativos tampoco es favorable ya que esto implica que se dejan pasar transacciones fraudulentas y por tanto la entidad perderá dinero.

Capítulo 2

Metodología

En este capítulo se expone la base teórica de los procedimientos que se realizarán a lo largo del trabajo tanto para analizar los datos como de los modelos de clasificación.

2.1. Análisis de datos

El conjunto de datos proporcionado por la empresa es una lista de operaciones que contienen información de una serie de variables (apéndice A) sobre el comportamiento del cliente, el dispositivo desde la que se realiza y sobre la operación en sí. Este conjunto de datos masivo contiene una gran cantidad de información que debe ser filtrada y transformada para obtener modelos con mejor capacidad predictiva. Por lo tanto el primer paso es realizar una serie de modificaciones a dicho conjunto, para ello se seguirán los pasos expuestos a continuación.

2.1.1. Análisis exploratorio

Realizamos una comparativa gráfica de los valores que toma cada variable en los casos fraudulentos y no fraudulentos. Para ello se realiza un estudio por medio de boxplots y diagramas de barras incluyendo las densidades de probabilidad de ambos conjuntos. Se emplean estas gráficas para estudiar la distribución que sigue cada variable y se calculan los valores de sus medidas de tendencia central y dispersión.

2.1.2. Medidas de asociación

Conviene recordar la maldición de la dimensión que ocurre al tener un número elevado de variables. Una alta cantidad de variables afecta negativamente al tiempo de computación necesario para realizar tanto modificaciones al conjunto de datos como al tiempo de computación necesario para entrenar e incluso obtener respuestas de los diferentes modelos de clasificación.

El hecho de que varias variables tengan correlaciones altas entre ellas no necesariamente afecta a la precisión que presenta el modelo en la clasificación, pero como se ha comentado si afecta el tener una gran cantidad de variables. Por lo tanto interesa estudiar los casos en los que esto ocurre, eliminando variables altamente correlacionadas, antes de proceder con el estudio de los modelos.

Incluimos a continuación las medidas de asociación que se han usado para estudiar las correlaciones y la influencia del fraude en ellas para variables numéricas continuas y para variables numéricas discretas y categóricas.

Test de Kolmogorov-Smirnov de homogeneidad

Aplicaremos el test de homogeneidad de Kolmogorov-Smirnov para estudiar las diferencias entre las distribuciones de cada variable numérica en los casos fraudulentos y no fraudulentos. Este test está basado en la comparativa de la función de distribución empírica de los datos, que recordamos para una muestra ordenada $(x_1, ..., x_n)$ esta viene dada por:

$$F(x) = \begin{cases} 0 & \text{si } x \leq x_1 \\ \frac{i}{n} & \text{si } x_{i-1} < x \leq x_i \text{ con } i = 2, ..., n \\ 1 & \text{si } x > x_n \end{cases}$$

Para comparar la distribución de los datos en ambos casos, se utiliza el contraste de Kolmogorov-Smirnov que viene expresado por:

$$H_0 : \ f_f = f_n$$
$$H_1 : \ f_f \neq f_n$$

donde las funciones f son las densidades de probabilidad y los subíndices f y n indican los casos fraudulento y el no fraudulento para una variable respectivamente. El estadístico viene dado por:

$$D_{f,n} = \sup_x |F_f(x) - F_n(x)|$$

con $F_f(x)$ y $F_n(x)$ las distribuciones empíricas de una misma variable en los casos fraudulentos y no fraudulentos respectivamente. Así obtendremos una medida numérica de la discrepancia entre las distribuciones en ambos casos para cada variable numérica. Bajo H_0 se tiene igualdad de distribuciones y por tanto no influye el fraude en la variable. Por tanto consideraremos que para una variable no tiene influencia el fraude si el p-valor de la prueba tiene un valor superior a $\alpha = 0.05$.

Test χ^2 de homogeneidad

El test χ^2 es un test de hipótesis que puede emplearse para estudiar la independencia ente dos variables categóricas, utilizando la tabla de contingencia correspondiente a estas dos variables con los diferentes posibles valores.

Se mide nuevamente el contraste de igualdad de funciones de distribución f, expresado mediante:

$$H_0 : \ f_f = f_n$$
$$H_1 : \ f_f \neq f_n$$

con los subíndices f y n indican los casos fraudulento y el no fraudulento para una variable respectivamente.

Utilizando la notación empleada previamente, siendo A la matriz de contingencia de tamaño $r \times c$ denotamos n_{ij} al número de observaciones en la celda (i, j). Sabemos que el estadístico χ^2 es de la forma:

$$\chi^2 = \sum_i^r \sum_j^c \frac{(n_{ij} - e_{ij})^2}{e_{ij}}$$

donde $e_{ij} = \frac{n_{i.} n_{.j}}{n}$ es el número de casos esperado para la celda (i, j), con $n_{i.} = \sum_j n_{ij}$ y $n_{.j} = \sum_i n_{ij}$.

El p-valor se calcula gracias a este estadístico y a una distribución $\chi^2_{(r-1)(c-1)}$ podemos calcular el p-valor correspondiente.

Se estudiará nuevamente si para una variable categórica la distribución de los casos fraudulentos y los no fraudulentos coincide. Nuevamente consideraremos que para una variable no tiene influencia el fraude si el p-valor de la prueba tiene un valor superior a $\alpha = 0.05$.

Correlación de Pearson

Para medir la correlación lineal entre dos variables numéricas continuas se puede emplear el coeficiente de correlación de Pearson medida que fue introducida por Pearson (1896). Sean dos variables aleatorias continuas X e Y, este viene definido por:

$$\rho = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}$$

donde $\text{cov}(X, Y)$ es la covarianza entre las variables y σ_X, σ_Y son sus desviaciones estándar.

Para calcular el coeficiente de Pearson para dos variables con datos $\{(x_1, y_1), ..., (x_n, y_n)\}$ viene dado por:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}$$

con $\overline{x}$ y $\overline{y}$ las medias de x e y respectivamente.

El coeficiente de correlación de Pearson toma valores entre -1 y 1 donde un valor cercano a 1 indica alta correlación positiva, cercano a -1 indica alta correlación negativa y cercano a 0 indica variables incorreladas. Este estadístico mide únicamente las relaciones lineales, podrían emplearse otros coeficientes como el de Spearman que fue introducido por Spearman (1904) y mide relaciones monótonas entre variables, la cual es una condición menos estricta.

V de Cramér

La V de Cramér es una medida de asociación entre dos variables nominales basada en el estadístico de la χ^2. Esta medida puede emplearse entre variables discretas y fue propuesta por Cramér (1946).

Sea una muestra con n observaciones con A su matriz de contingencia de tamaño $r \times c$ denotamos π_{ij} a la probabilidad correspondiente a la celda (i, j). Recuperando el estadístico χ^2 introducido previamente se define la V de Cramér mediante:

$$V = \sqrt{\frac{\chi^2}{n(\text{mín}\,(c, r) - 1)}}$$

Esta medida devuelve un valor entre 0 y 1 que indican, respectivamente, nula y completa asociación. Si en la matriz de contingencia la suma de una fila o una columna es baja esto puede empeorar la calidad del estadístico, lo cual dificulta su interpretación. Por tanto se propone una corrección a la misma en Bergsma (2013) que viene dada por:

$$\tilde{V} = \sqrt{\frac{\tilde{\chi}^2}{n(\text{mín}\,(\tilde{c}, \tilde{r}) - 1)}}$$

donde $\tilde{\chi}^2$, $\tilde{r}$ y $\tilde{c}$ vienen dados por:

$$\tilde{\chi}^2 = \text{máx}\,(0, \chi^2 - \frac{n}{n-1}(r-1)(c-1))$$

$$\tilde{r} = r - \frac{1}{n-1}(r-1)^2$$

$$\tilde{c} = c - \frac{1}{n-1}(c-1)^2$$

El análisis de la medida corregida es completamente análogo al del caso original. Se obtiene un valor entre 0 y 1 indicando nula y completa asociación respectivamente.

Correlación de distancia

La correlación de distancia es una medida para medir la dependencia entre dos vectores. Se denota por $\mathcal{R}$ y fue introducida por Székely et al. (2007).

Para introducir la correlación de distancia es necesario introducir la distancia de covarianza, que también se define en el artículo original previamente mencionado. Esta distancia se aplica a un test de hipótesis de independencia entre dos vectores X e Y con formulación:

$$H_0 : \ \phi_{X,Y} = \phi_X \phi_Y$$
$$H_1 : \ \phi_{X,Y} \neq \phi_X \phi_Y$$

donde ϕ_X, ϕ_Y, $\phi_{X,Y}$ son respectivamente las funciones características de X, Y y la conjunta de ambos vectores.

Sean X e Y dos vectores con primeros momentos finitos, entonces la distancia de covarianza y la distancia de varianza se definen respectivamente como las raíces positivas de:

$$\mathcal{V}^2(X,Y) = ||\phi_{X,Y}(t,s) - \phi_X(t)\phi_Y(s)||^2$$

$$\mathcal{V}^2(X) = \mathcal{V}^2(X,X) = ||\phi_{X,X}(t,s) - \phi_X(t)\phi_X(s)||^2$$

Así se puede definir la correlación de distancia como la raíz positiva de:

$$\mathcal{R}^2(X,Y) = \begin{cases} \frac{\mathcal{V}^2(X,Y)}{\sqrt{\mathcal{V}^2(X)\mathcal{V}^2(Y)}}, & \text{si } \mathcal{V}^2(X)\mathcal{V}^2(Y) > 0 \\ 0, & \text{si } \mathcal{V}^2(X)\mathcal{V}^2(Y) = 0 \end{cases}$$

Para calcular este estadístico en una muestra problema es necesario introducir la siguiente notación:

$$a_{kl} = ||X_k - X_l||_p \qquad \overline{a}_{k\cdot} = \frac{1}{n}\sum_{l=1}^{n} a_{kl}, \qquad \overline{a}_{\cdot l} = \frac{1}{n}\sum_{k=1}^{n} a_{kl},$$

$$\overline{a}_{\cdot\cdot} = \frac{1}{n^2}\sum_{k,l=1}^{n} a_{kl} \qquad A_{kl} = a_{kl} - \overline{a}_{k\cdot} - \overline{a}_{\cdot l} + \overline{a}_{\cdot\cdot}$$

con $k, l = 1, ..., n$. De manera análoga se define $b_{kl} = ||Y_k - Y_l||_q$ y $B_{kl} = b_{kl} - \overline{b}_{k\cdot} - \overline{b}_{\cdot l} + \overline{b}_{\cdot\cdot}$. Podemos definir entonces los términos anteriores de la siguiente manera:

$$\hat{\mathcal{V}}_n^2(X,Y) = \frac{1}{n^2}\sum_{k,l=1}^{n} A_{kl}B_{kl}$$

$$\hat{\mathcal{V}}_n^2(X) = \hat{\mathcal{V}}_n^2(X,X) = \frac{1}{n^2}\sum_{k,l=1}^{n} A_{kl}^2$$

$$\hat{\mathcal{R}}_n^2(X,Y) = \begin{cases} \frac{\hat{\mathcal{V}}_n^2(X,Y)}{\sqrt{\hat{\mathcal{V}}_n^2(X)\hat{\mathcal{V}}_n^2(Y)}}, & \text{si } \hat{\mathcal{V}}_n^2(X)\hat{\mathcal{V}}_n^2(Y) > 0 \\ 0, & \text{si } \hat{\mathcal{V}}_n^2(X)\hat{\mathcal{V}}_n^2(Y) = 0 \end{cases}$$

Y por tanto la correlación de distancia estimada entre X e Y se define como la raíz positiva de $\hat{\mathcal{R}}_n^2(X,Y)$. Como se comentó previamente, la correlación de distancia generaliza la idea de correlación en el sentido de:

- $\mathcal{R}(X,Y)$ se define para X e Y de dimensiones arbitrarias.
- $\mathcal{R}(X,Y) = 0$ caracteriza la independencia de X e Y.

Entre sus propiedades está que $0 \leq \mathcal{R}(X,Y) \leq 1$. Usaremos este estadístico para medir la asociación entre una variable cualquiera y la variable indicadora de si se trata una operación fraudulenta. Nos interesa que estas variables sean lo más dependientes posibles, ya que así tendrán mayor poder de predicción. Por lo tanto a mayor correlación de distancia, mayor importancia para el problema.

2.1.3. Datos faltantes

Existen diferentes maneras de actuar cuando se trabaja con conjuntos de datos incompletos. Se puede obviar este problema, lo cual no siempre es posible, y trabajar con datos faltantes usando así toda la información disponible. Esto se denomina *pairwise deletion* y se eliminan los valores faltantes en las variables solamente cuando es necesario. En contraste se tiene *listwise deletion* a cuando se eliminan las observaciones con valores faltantes en alguna de las variables. También se pueden imputar los valores. Dependiendo de la cantidad de datos faltantes, su distribución o comportamiento y también de los modelos empleados para clasificar los datos conviene aplicar un procedimiento u otro.

Se clasifican los tipos de datos faltantes en función de su distribución, existen tres situaciones. Para introducirlas denotamos para esta sección las variables sin datos faltantes mediante X y las variables sin datos faltantes mediante Y. Entonces se denominan *Missing Completely At Random* (MCAR) cuando la probabilidad de que un dato sea faltante no depende de los datos faltantes ni de los observados. Esto se describe como que la probabilidad de que un dato sea faltante es totalmente aleatoria. Formalmente se puede escribir mediante:

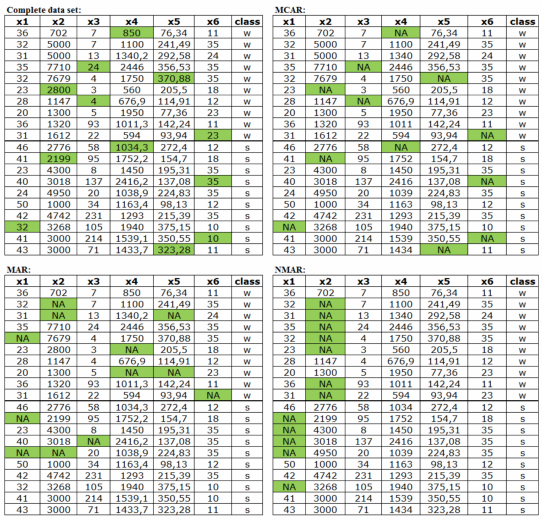
$$Pr(Y_{\text{faltante}}|X,Y) = Pr(Y_{\text{faltante}})$$

Se denomina *Missing At Random* (MAR) cuando la probabilidad de que la información faltante no depende de los datos faltantes pero puede que lo haga de los datos observados. Esto se expresa matemáticamente en Allison (2001) mediante:

$$Pr(Y_{\text{faltante}}|X,Y) = Pr(Y_{\text{faltante}}|X)$$

Existe otro tipo denominado *Missing Not At Random* (MNAR) donde los datos faltantes ocurren de manera no aleatoria, es decir la probabilidad de que un dato sea faltante depende de los datos faltantes. Este caso se da cuando no se dan ninguno de los otros dos casos.

Se puede observar un ejemplo de estas situaciones en la Figura 2.1, lo cual ayuda a entender sus definiciones.



Complete data set:

x1	x2	x3	x4	x5	x6	class
36	702	7	850	76,34	11	w
32	5000	7	1100	241,49	35	w
31	5000	13	1340,2	292,58	24	w
35	7710	24	2446	356,53	35	w
32	7679	4	1750	370,88	35	w
23	2800	3	560	205,5	18	w
28	1147	4	676,9	114,91	12	w
20	1300	5	1950	77,36	23	w
36	1320	93	1011,3	142,24	11	w
31	1612	22	594	93,94	23	w
46	2776	58	1034,3	272,4	12	s
41	2199	95	1752,2	154,7	18	s
23	4300	8	1450	195,31	35	s
40	3018	137	2416,2	137,08	35	s
24	4950	20	1038,9	224,83	35	s
50	1000	34	1163,4	98,13	12	s
42	4742	231	1293	215,39	35	s
32	3268	105	1940	375,15	10	s
41	3000	214	1539,1	350,55	10	s
43	3000	71	1433,7	323,28	11	s

MCAR:

x1	x2	x3	x4	x5	x6	class
36	702	7	NA	76,34	11	w
32	5000	7	1100	241,49	35	w
31	5000	13	1340	292,58	24	w
35	7710	NA	2446	356,53	35	w
32	7679	4	1750	NA	35	w
23	NA	3	560	205,5	18	w
28	1147	NA	676,9	114,91	12	w
20	1300	5	1950	77,36	23	w
36	1320	93	1011	142,24	11	w
31	1612	22	594	93,94	NA	w
46	2776	58	NA	272,4	12	s
41	NA	95	1752	154,7	18	s
23	4300	8	1450	195,31	35	s
40	3018	137	2416	137,08	NA	s
24	4950	20	1039	224,83	35	s
50	1000	34	1163	98,13	12	s
42	4742	231	1293	215,39	35	s
NA	3268	105	1940	375,15	10	s
41	3000	214	1539	350,55	NA	s
43	3000	71	1434	NA	11	s

MAR:

x1	x2	x3	x4	x5	x6	class
36	702	7	850	76,34	11	w
32	NA	7	1100	241,49	35	w
31	NA	13	1340,2	NA	24	w
35	7710	24	2446	356,53	35	w
NA	7679	4	1750	370,88	35	w
23	2800	3	NA	205,5	18	w
28	1147	4	676,9	114,91	12	w
20	1300	5	NA	NA	23	w
36	1320	93	1011,3	142,24	11	w
31	1612	22	594	93,94	NA	w
46	2776	58	1034,3	272,4	12	s
NA	2199	95	1752,2	154,7	18	s
23	4300	8	1450	195,31	35	s
40	3018	NA	2416,2	137,08	35	s
NA	NA	20	1038,9	224,83	35	s
50	1000	34	1163,4	98,13	12	s
42	4742	231	1293	215,39	35	s
32	3268	105	1940	375,15	10	s
41	3000	214	1539,1	350,55	10	s
43	3000	71	1433,7	323,28	11	s

NMAR:

x1	x2	x3	x4	x5	x6	class
36	702	7	850	76,34	11	w
32	NA	7	1100	241,49	35	w
31	NA	13	1340	292,58	24	w
35	NA	24	2446	356,53	35	w
32	NA	4	1750	370,88	35	w
23	NA	3	560	205,5	18	w
28	1147	4	676,9	114,91	12	w
20	1300	5	1950	77,36	23	w
36	NA	93	1011	142,24	11	w
31	NA	22	594	93,94	23	w
46	2776	58	1034	272,4	12	s
NA	2199	95	1752	154,7	18	s
NA	4300	8	1450	195,31	35	s
NA	3018	137	2416	137,08	35	s
NA	4950	20	1039	224,83	35	s
50	1000	34	1163	98,13	12	s
42	4742	231	1293	215,39	35	s
NA	3268	105	1940	375,15	10	s
41	3000	214	1539	350,55	10	s
43	3000	71	1434	323,28	11	s

Figura 2.1: Ejemplos de datos completos, con MCAR, MAR y MNAR. Imagen obtenida de Małgorzata (2013).

Imputación

La imputación de datos es una herramienta que cuenta con gran cantidad de métodos diferentes para llevarla a cabo. Dependiendo del tipo de datos problema y de como sea la distribución de los datos faltantes, conviene emplear un método u otro para realizar la imputación.

Imputar teniendo en cuenta otras variables mediante regresión es complicado, pues cada tipo de operación del conjunto de datos contiene datos faltantes de diferentes variables. Es decir para operaciones de login, sesión o transacción se tiene diferente comportamiento de algunas variables.

Una forma sencilla de imputar los datos es mediante la sustitución de los datos faltantes por una cantidad fija como puede ser la media o mediana en datos numéricos o la moda para datos categóricos. Sin embargo esto no es muy recomendable ya que puede acarrear diferentes problemas, como por ejemplo la introducción de sesgos o la modificación de la variabilidad o correlaciones. Sobre todo si existe alguna variable con muchos datos faltantes.

2.2. Técnicas de remuestreo

Tenemos un conjunto de datos problema que se encuentra dividido en dos categorías: operaciones fraudulentas y operaciones no fraudulentas. El número de casos en los que la operación es fraudulenta son mucho menor que si no lo es, el problema tratado es de datos desbalanceados. La eficacia de los modelos de clasificación puede verse afectada por este problema y por ello interesa tratarlo.

Los métodos de tratamiento del desbalanceo son muy variados pero su finalidad es la misma, conseguir que la cantidad de datos en cada clase sea igual. Algunas de las técnicas se fundamentan en el concepto de *undersampling*, el cual consiste en la reducción de la clase mayoritaria hasta igualar el número de datos a los de la clase minoritaria. La diferencia entre los distintos métodos radica en la manera en la que se realice esa reducción. Otras técnicas consisten en el proceso contrario, son técnicas de *oversampling* y estas consisten en el aumento de observaciones en la clase minoritaria hasta obtener una muestra balanceada. También existen técnicas que combinan ambos enfoques, brindando así un

abanico de opciones para abordar el desequilibrio de clases.

Se aplicarán cuatro técnicas de remuestreo para tratar el problema de datos desbalanceados. Estas técnicas son: RandomUnderSampler, RandomOverSampler, SMOTE y CTGAN. Además, se llevará a cabo una prueba de ajuste de los modelos utilizando datos sin remuestrear como medida de control.

2.2.1. Undersampling

La primera técnica aplicada será una del tipo *undersampling*. Como su nombre indica el método RandomUnderSampler de la librería imblearn de Python realiza un remuestreo aleatorio de los datos de la clase mayoritaria hasta obtener un número de muestras igual al de la clase minoritaria. Existe la opción de realizar este remuestreo con o sin reemplazamiento. Como tenemos muestras con tamaños muy diferentes se realizará sin reemplazamiento para evitar que se puedan tomar dos o más observaciones iguales de la clase mayoritaria, aunque esto sea poco probable, y así tener una mayor cantidad de información.

Se trata de un método rápido. Sin embargo, al escoger aleatoriamente las muestras puede ocurrir que alguna variable no tenga todos sus valores correctamente representados en la muestra ya que en el caso problema la cantidad de datos en ambas categorías es muy dispar, esto conllevaría una pérdida considerable de información. Se puede observar en la Figura 2.2 un ejemplo obtenido de la documentación del paquete imblearn para unos datos multiclase divididos en tres categorías, donde la clase minoritaria corresponde con los datos coloreados de morado. La figura de la izquierda corresponde con la clasificación con los datos originales y la de la derecha se trata de los datos resampleados con RandomUnderSampler de tal manera que las categorías coloreadas de amarillo y azul tengan la misma cantidad de datos que aquella coloreada de morada.

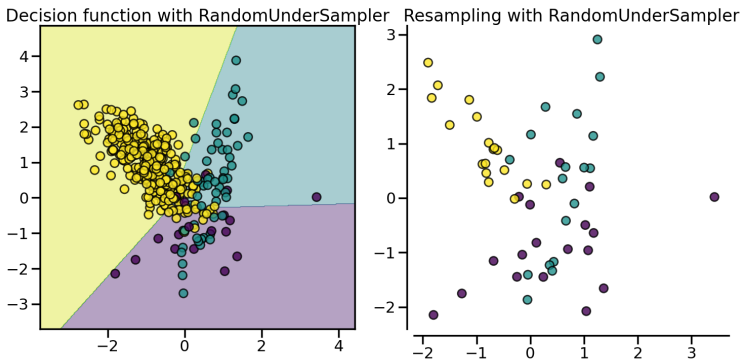


Figura 2.2: Gráfica de un conjunto de datos y su remuestreo usando RandomUnderSampler [1].

2.2.2. Oversampling

Una de las técnicas de *oversampling* aplicadas es el método RandomOverSampler de la librería imblearn de Python. Este realiza remuestreo aleatorio de la categoría minoritaria para conseguir balancear el conjunto. Con esto se tiene que los datos minoritarios aparecerán duplicados una cantidad determinada de veces, dependiendo del nivel de desbalanceo con el que se trabaje.

[1]Imagen obtenida de https://imbalanced-learn.org/stable/under_sampling.html (Consultado en abril de 2023)

Se puede observar en la Figura 2.3 un caso gráfico de clasificación multiclase con tres categorías. En la figura de la izquierda vemos la clasificación con los datos originales y en la figura de la derecha vemos la clasificación una vez aplicado el método. La clase minoritaria es aquella para la que los datos se han coloreado de morado y vemos que al aplicar el método cambia considerablemente la representación de la misma, consiguiendo así clasificar muestras de esta clase correctamente.

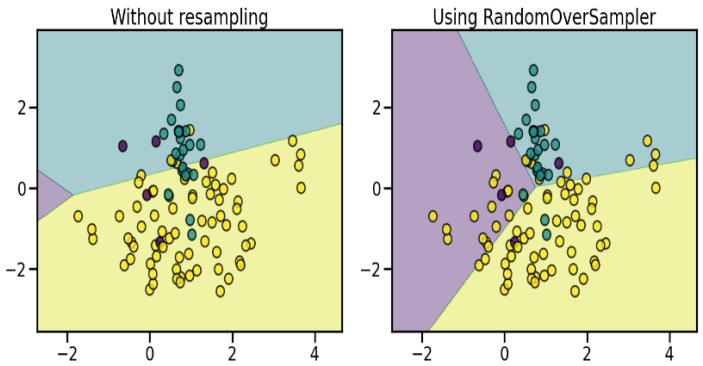


Figura 2.3: Gráfica de un conjunto de datos y su remuestreo usando RandomOverSampler [2].

2.2.3. Synthetic Minority Oversampling Technique

Otra técnica de *oversampling* es *Synthetic Minority Oversampling Technique* (SMOTE). Este algoritmo fue introducido por Chawla et al. (2002) y realiza un aumento de datos creando datos sintéticos basados en los originales. La forma en la que realiza esta tarea es introduciendo nuevos datos en los segmentos que unen cada operación con sus k vecinos más cercanos. El número k de vecinos escogido depende de la cantidad de muestras que sea necesario generar.

Este método tiene en cuenta el espacio de variables en vez de el conjunto de datos como tal. Por lo tanto, para la creación de un nuevo dato toma una de las operaciones, que es un vector en el espacio de variables, y calcula la diferencia entre este y su vecino más cercano. Esta diferencia se multiplica por un número aleatorio entre 0 y 1 para conseguir una nueva muestra entre ambas. El pseudocódigo del algoritmo se encuentra completo en Chawla et al. (2002). En la Figura 2.4 vemos una aplicación del remuestreo por SMOTE.

Este algoritmo tiene como ventaja que no crea réplicas de los datos ya observados. Esta técnica de remuestreo ha sido propuesta para problemas de fraude por sus autores. Han comprobado que este algoritmo es capaz de mejorar la exactitud del clasificador en dicho problema y por tanto interesa aplicarlo en nuestro estudio.

2.2.4. Conditional Tabular Generative Adversarial Networks

Los *Generative Adversarial Networks* (GANs) son un tipo de algoritmo de inteligencia artificial desarrollados en 2014 por Goodfellow et al. (2014) como un modelo de aprendizaje no supervisado, aunque su uso se ha expandido hacia otras ramas del aprendizaje automático. Se trata de unos juegos de suma nula donde dos redes neuronales compiten entre ellas siendo las ganancias de una las pérdidas de la otra. Explicado intuitivamente se tiene una red neuronal, el generador G, que genera datos con el objetivo de que cada vez sean más similares a los datos reales y otra red, el discriminador D, la

[2]Imagen obtenida de https://imbalanced-learn.org/stable/over_sampling.html (Consultado en abril de 2023)

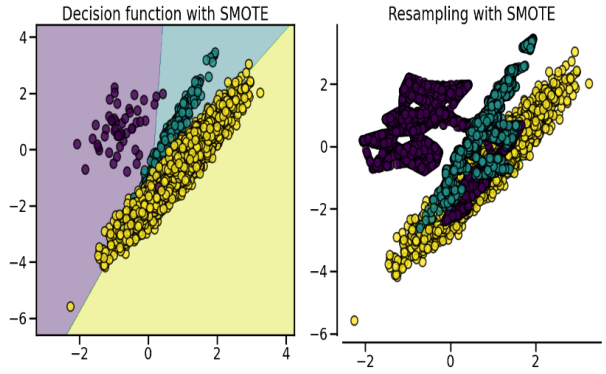


Figura 2.4: Gráfica de un conjunto de datos y su remuestreo usando SMOTE [2].

cual se entrena con datos reales con el fin de que distinga estos de los datos creados por el generador G.

Formalmente comenzamos definiendo p_g como la distribución del generador sobre los datos x. Para ello se definen primero las variables de ruido de entrada $p_z(z)$. Siendo G, el generador, una función diferenciable correspondiente a un perceptrón multicapa con parámetros θ_g, entonces $G(z;\theta_p)$ es una aplicación al espacio de datos. El perceptrón es una neurona artificial, este concepto y más sobre las redes neuronales se extenderán en la posterior Sección 2.3.3.
Definimos el discriminador D como otro perceptrón multicapa $D(x;\theta_d)$ que devuelve un escalar. $D(x)$ representa la probabilidad de que x provenga de los datos reales en vez de p_g. El objetivo, como ya se ha comentado, es entrenador D para maximizar la probabilidad de clasificar correctamente tanto los datos de la muestra como los provenientes de G. Al mismo tiempo se entrena G para minimizar $\log(1 - D(G(z)))$. Así la formulación del juego minimax entre D y G con función objetivo $V(G, D)$ es:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{datos}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$

La competición entre ambos modelos conduce a que estos mejoren hasta el punto de hacer indistinguibles los datos generados. Las GANs son capaces de generar nuevos datos muy similares a los datos originales. Un ejemplo conocido es el de creación de fotografías simulando seres humanos. Existe una página web[3] que muestra imágenes de personas ficticias creadas mediante el uso de GANs. Estudios realizados por Nightingale y Farid (2022) muestran que este tipo de imágenes llegan a ser tan reales que gente expuesta a ellas no es capaz de distinguirlas de imágenes de personas reales.

Se muestra un esquema del funcionamiento de esta técnica en la Figura 2.5. Usaremos GANs para generar datos ficticios de transacciones fraudulentas con el fin de balancear el conjunto de datos tratado.

Las GANs se desarrollaron en el contexto de generación de imágenes. En el problema tratado los datos vienen dispuestos en forma de tabla, por lo tanto el algoritmo original de las GANs no puede aplicarse. Este problema presenta una complejidad añadida ya que hay que tener en cuenta el tipo de dato que compone cada columna además de las relaciones entre ellas. Por ello Xu et al. (2019) desarrollan las *Conditional Tabular Generative Adversarial Networks* (CTGANs) que son algoritmos basado en las GANs para poder generar observaciones realistas de datos presentados en tablas. Estos procedimientos tienen en cuenta el tipo de variable tratada, si estas son continuas, discretas o categóricas.

[3] https://this-person-does-not-exist.com/
[4] Imagen obtenida de https://developer.ibm.com/articles/generative-adversarial-networks-explained/ (Consultado en mayo de 2023)

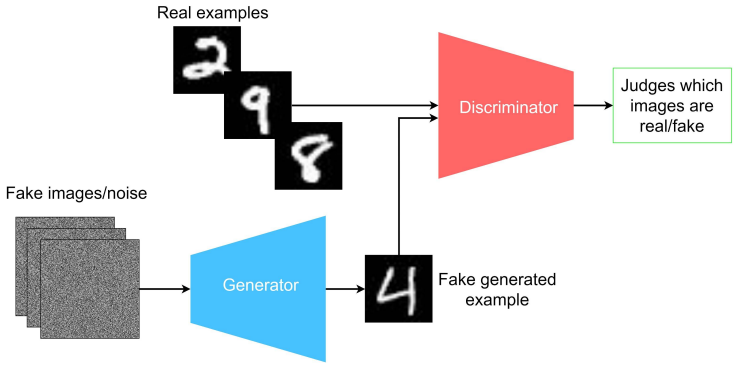


Figura 2.5: Esquema del funcionamiento de los GANs [4].

2.3. Modelos de clasificación

El problema a tratar es de clasificación binaria, queremos ser capaces de construir un modelo que ayude a decidir si una operación observada es fraude o no. Se estudiarán diferentes modelos y se determinará cual de todos desempeña mejor este trabajo.

2.3.1. Regresión logística

El modelo de regresión logística surge del deseo de modelar las probabilidades de que una observación pertenezca a una clase mediante funciones lineales. En nuestro problema tenemos dos clases, operación fraudulenta y no fraudulenta, por lo tanto será necesaria tan sólo una función lineal para realizar el modelo, lo cual lo convierte en un problema relativamente sencillo.

Nuestro modelo viene especificado mediante una función logit. Sea Y la respuesta del modelo, correspondiente a clase no fraudulenta $Y = 0$ y fraudulenta $Y = 1$, y X la observación sobre la que se quiere realizar la predicción. Siguiendo la formulación de Hastie et al. (2009) el problema se plantea mediante:

$$\log \frac{\Pr(Y = 0|X = x)}{\Pr(Y = 1|X = x)} = \beta_0 + \beta^\top x$$

Por lo tanto se tiene que

$$\Pr(Y = 0|X = x) = \frac{\exp(\beta_0 + \beta^\top x)}{1 + \exp(\beta_0 + \beta^\top x)}$$

$$\Pr(Y = 1|X = x) = \frac{1}{1 + \exp(\beta_0 + \beta^\top x)}$$

Este modelo es sencillo pero para emplearlo sobre variables predictoras categóricas conviene crear variables indicadoras o *dummy*. Para una variable con k categorías esto consiste en crear $k-1$ variables que corresponderán con $k-1$ de las categorías mientras que la categoría restante servirá de referencia. Vemos un problema claro con esto y es que si una variable posee gran cantidad de categorías se obtendría una gran cantidad de variables, lo que conlleva mayores tiempos de entrenamiento y predicción.

Para evitar problemas de colinealidad y sobreajuste se emplea regularización en la estimación de los parámetros. Para ello se consideran todas las variables pero se incluye una penalización que fuerza a que algunos coeficientes tomen valores muy cercanos a cero e incluso cero. Emplear regularización conlleva un aumento del sesgo pero una reducción en la varianza, disminuir la influencia de algunas

variables también ayuda en la interpretabilidad del modelo.
Existen diferentes tipos de regulación comunes como pueden ser las denominadas Ridge o Lasso. En este problema empleamos la llamada regularización L2 que es una penalización cuadrática de la magnitud de los coeficientes, por lo tanto todos los coeficientes son afectados de la misma manera y no se reduce considerablemente el efecto de ninguno.

2.3.2. Gradient boosting

La metodología *boosting* da lugar a un predictor robusto combinando modelos predictivos más débiles, es una metodología de aprendizaje lento Fernández et al. (2021). Los árboles de decisión poco profundos son buenos candidatos para usar como modelos pequeños ya que suelen ser ser malos predictores. Esta idea fue introducida por Kearns y Valiant (1994).

Gradient boosting refiere a métodos boosting que emplean algoritmos iterativos de descenso de gradientes desarrollados por Friedman (2001). La idea es encontrar un modelo aditivo en forma de ensamblaje de modelos más débiles.

El objetivo es encontrar una aproximación a la función F^* con la que realizar una predicción sobre un valor para el problema tratado, en nuestro caso de clasificación binaria. Partimos de una muestra de entrenamiento de la forma $\{x_i, y_i\}_{i=1}^n$ y se busca un función $\hat{F} : \mathbb{R}^n \to \mathbb{R}$ tal que minimice una función suave de pérdida $\mathcal{L}(y, F(x))$ sobre las distribuciones de los valores de la muestra. Siguiendo las ideas de Friedman (2001), esto es:

$$F^* = \arg\min_F \mathbb{E}_{x,y}[\mathcal{L}(y, F(x))]$$

Existen diversas funciones de pérdida, como la basada en errores cuadráticos o errores absolutos, y maneras de estimar $\hat{F}$, como usando familias paramétricas de funciones.

Construiremos la función $\hat{F}(x)$ como una suma ponderada, con pesos α, de T funciones h_t de una familia $\mathcal{H}$ denominadas predictores base. La expresión matemática es:

$$\hat{F}(x) = \sum_{t=1}^{T} \alpha_t h_t(x) + \text{cte}$$

Para un problema real el gradient boosting consiste en aproximar iterativamente una serie de funciones $F^t : \mathbb{R}^m \to \mathbb{R}$ con un algoritmo voraz partiendo de F_0.

$$F_0(x) = \arg\min_\alpha \sum_{i=1}^{n} \mathcal{L}(y_i, \alpha)$$

$$F_t(x) = F_{t-1}(x) + \left(\arg\min_{h_m \in \mathcal{H}} \left[\sum_{i=1}^{n} \mathcal{L}(y_i, F_{t-1}(x_i) + h_t(x_i)) \right] \right)$$

con $t = 1, ..., T$.

El problema surge de que no es computacionalmente viable elegir los valores óptimos de h_t en cada iteración para una función de pérdida arbitraria $\mathcal{L}$. Entonces se propone una simplificación del problema que es aplicar el mayor descenso para el gradiente. Se calcula el mínimo local de la función de pérdida iterando en F_{t-1}, desplazando una cantidad α suficientemente baja para considerar válida la aproximación lineal. Así se obtiene:

$$F_t(x) = F_{t-1}(x) - \alpha_t \sum_{i=1}^{n} \nabla F_{t-1} \mathcal{L}(y_i, F_{t-1}(x_i))$$

con

$$\alpha_t = \arg\min_{\alpha} \sum_{i=1}^{n} \mathcal{L}(y_1, F_{t-1}(x_i) - \alpha \nabla F_{t-1} \mathcal{L}(y_i, F_{t-1}(x_i)))$$

CatBoost

El algoritmo que utilizamos en este trabajo se denomina CatBoost y es una implementación de gradient boosting que usa arboles de decisión como predictores base. El algoritmo completo fue introducido por Prokhorenkova et al. (2019).

Se puede ver un árbol de decisión como particiones recursivas del espacio de variables. En el paso t tendríamos un árbol $h_t(x)$ con J_t hojas que dividen los datos en $R_{1m}, ..., R_{J_y t}$ regiones disjuntas. Usando la función indicadora, $\mathbb{1}_X$, podemos escribir las funciones h_t como:

$$h_t(x) = \sum_{j=1}^{J_t} b_{jt} \mathbb{1}_{R_{jt}}(x)$$

donde b_{jt} es el valor que se predice en la región R_{jt}. Así, tenemos:

$$F_t(x) = F_{t-1}(x) - \alpha_t h_t(x)$$

donde se puede tomar $\alpha_t = \alpha =$ cte o tomar un valor calculado a partir de una de las numerosas funciones de pérdida pérdida implementadas en el paquete, o incluso emplear una función de pérdida propia.

$$\alpha_t = \arg\min_{\alpha} \sum_{i=1}^{n} \mathcal{L}(y_i, F_{t-1}(x_i) + \alpha h_t(x_i))$$

Otros métodos de gradient boosting tienen en cuenta las respuestas de todas las observaciones de entrenamiento después de realizar varias iteraciones de descenso de gradiente. Esto provoca un desplazamiento en la predicción que se intenta corregir empleando un boosting ordenado como se expone en Prokhorenkova et al. (2019).

2.3.3. Redes neuronales

Las redes neuronales artificiales, en inglés *Artificial Neural Networks* (ANNs) o simplemente NNs, son una clase de modelos y algoritmos de aprendizaje profundo. El nombre deriva de las redes neuronales biológicas en las cuales los componentes principales son las neuronas que transmiten información en forma de señales eléctricas entre ellas. En lineas generales esto es lo que realizan las NNs, estas se componen de neuronas artificiales que transmiten información entre ellas en forma de números reales.

Neuronas

La neurona es el nivel más sencillo de una red neuronal. El funcionamiento básico de una neurona artificial es que recibe una información de entrada (*input*) en forma de $x_0, ..., x_p$, denominadas señales y proporciona una salida. La neurona tiene asociados unos pesos $w_0, ..., w_p$. Usualmente se toma $x_0 = 1$ y por tanto $w_0 = b_0$ sería el sesgo. La neurona toma esta información y realiza una combinación lineal del *input* y los pesos para posteriormente someterlos a una transformación mediante una función φ denominada la función de activación. Matemáticamente su estructura se describe como:

$$y = \varphi\left(\sum_{j=0}^{p} w_j x_j\right)$$

Las funciones de activación pueden tomar diferentes formas siendo las más comunes la ReLU (Rectified Linear Unit) y la sigmoid, sus expresiones son respectivamente:

$$\varphi_{ReLU}(x) = \text{máx}(0, x)$$

$$\varphi_{sigmoid}(x) = \frac{1}{1 + \exp(x)}$$

En la Figura 2.6 se presenta un esquema de la estructura de una neurona.

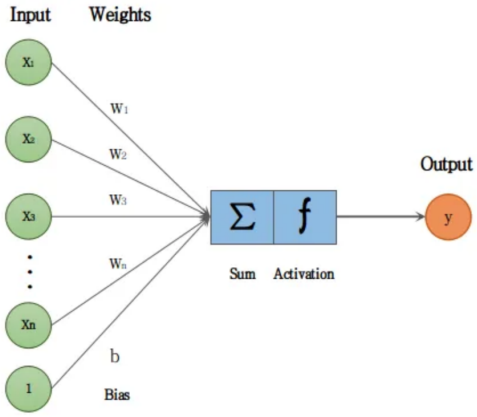


Figura 2.6: Esquema de la estructura de una neurona. Imagen obtenida de [5].

Capas

Las redes neuronales se componen de diferentes capas que a su vez no son más que conjuntos de neuronas, luego estas conforman la estructura intermedia de la red. El objetivo es aumentar el número de neuronas o capas para obtener mejores resultados a costa de complejizar el modelo.

Las capas se pueden clasificar en tres tipos dependiendo de su posición, y por tanto papel, en la red neuronal.

- Capa de entrada: consiste en las observaciones a partir de las cuales se quiere obtener una predicción.
- Capas ocultas: se trata de las capas intermedias. En ellas se realizan numerosas transformaciones las cuales no se observan, de ahí su nomenclatura.
- Capa de salida: se trata de la predicción realizada por el modelo. Es la información que recibe el usuario ya sea sobre la regresión o clasificación de la observación pasada al modelo.

Presentamos en la Figura 2.7 un esquema de la estructura de una red neuronal de una sola capa. Si el input tiene forma $x = (x_1, ..., x_p)$, la capa posee M neuronas y la salida es la asociada a un problema

[5] https://medium.com/@raycad.seedotech/convolutional-neural-network-cnn-8d1908c010ab (Consultado en mayo de 2023)

de clasificación de C clases entonces el funcionamiento se puede escribir matemáticamente mediante:

$$\begin{aligned} z_m &= \varphi(w_{0m} + w_m^\top x), & m = 1, ..., M \\ t_c &= \beta_{0c} + \beta_c^\top z, & c = 1, ..., C \\ F_c(x) &= g_c(t), & c = 1, ..., C \end{aligned}$$

con $z = (z_1, ..., z_M)$, $t = (t_1, ..., t_c)$ y g la función de salida, que puede coincidir con las de activación. En nuestro caso el problema es de clasificación binaria y por tanto $C = 2$. Esta estructura se puede generalizar para una red neuronal con un L capas.

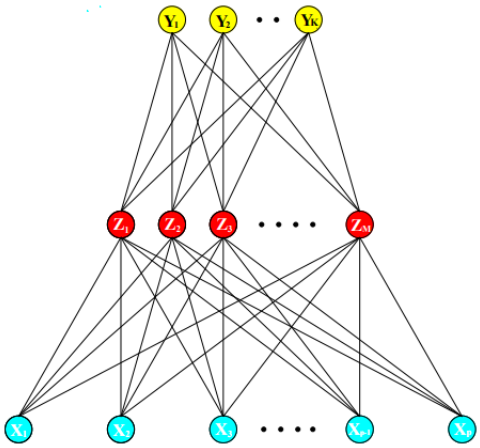


Figura 2.7: Esquema de la estructura de una red neuronal con una capa oculta. Imagen obtenida de Hastie et al. (2009).

La elección del número de capas es una tarea complicada, ya que una mayor cantidad de capas implica un mayor número de pesos y si este es muy grande entonces el modelo tiende a ajustarlos a valores cercanos a cero. Sin embargo, si hay pocas capas puede ocurrir que no tenga suficiente flexibilidad como para ajustar realmente el modelo Hastie et al. (2009).

Entrenamiento

Una red neuronal tiene como parámetros los pesos. Estos poseen valores desconocidos y es necesario ajustar el modelo a los datos de entrenamiento para obtener valores para estos. Los parámetros, con la notación anterior son:

$$\begin{aligned} \{w_{0m}, w_m; m = 1, ..., M\} & \quad M(p+1) \text{ pesos}, \\ \{\beta_{0c}, \beta_c; c = 1, ..., C\} & \quad C(M+1) \text{ pesos} \end{aligned}$$

Para determinar los pesos en un problema de clasificación, donde tenemos N datos, se utilizan la suma de errores cuadrados o la *cross-entropy*, que vienen dados respectivamente por:

$$R(\theta) = \sum_{c=1}^{C} \sum_{i=1}^{N} (y_{ic} - F_c(x_i))^2$$

$$R(\theta) = \sum_{i=1}^{N} \sum_{c=1}^{C} y_{ic} - \log(F_c(x_i))$$

Típicamente no se busca minimizar $R(\theta)$ ya que esto suele conllevar un sobreajuste del modelo. Por ello se realiza un enfoque basado en el descenso de gradiente denominado *backward propagation*. La idea es usar derivadas, que para este problema son fácilmente calculables gracias a la regla de la cadena, y realizar un descenso con un parámetro de tasa de aprendizaje para disminuir los errores. La formulación matemática para este problema se incluye en Hastie et al. (2009). Se realiza *forward propagation*, que consiste en calcular las predicciones del modelo en el paso, y luego se actualizan los pesos usando *backward propagation*, esto se repite un determinado número de veces (*epochs*). Para evitar el sobreajuste se mide también durante el desempeño del modelo con un conjunto de validación, si se observa que el error no disminuye de este conjunto entonces se para el entrenamiento del modelo.

Otras maneras de evitar el sobreajuste consisten en imponer una penalización a la función de error de la forma $R(\theta) + \lambda J(\theta)$. Un ejemplo de esto es el conocido como *weight decay* donde:

$$J(\theta) = \sum_{c,m} \beta_{cm}^2 + \sum_{m,l} w_{ml}^2$$

O la penalización denominada *weight elimination* donde:

$$J(\theta) = \sum_{c,m} \frac{\beta_{cm}^2}{1 + \beta_{cm}^2} + \sum_{m,l} \frac{w_{ml}^2}{1 + w_{ml}^2}$$

Una crítica es que estos modelos necesitan demasiado entrenamiento para aplicarse en casos reales debido a la gran cantidad de parámetros que precisan. Esto puede llevar a aumentos en la cantidad de memoria precisada o en un incremento en los tiempos de computación empleados. En nuestro problema no interesan tiempos grandes de computación, por lo tanto no se debe emplear un alto número de capas ocultas. Esta gran cantidad de parámetros dificulta también la interpretación del modelo, por lo que estos métodos se suelen denominar cajas negras como se Hastie et al. (2009).

2.4. Evaluación

Se muestran los procedimientos empleados para realizar el entrenamiento y posteriormente estudiar el desempeño de los modelos.

Se realizará una partición inicial de los datos en dos conjuntos: uno de validación y otro para entrenar y evaluar los modelos, en respectivas proporciones del 10 % y 90 %. El conjunto de validación se empleará para determinar los mejores hiperparámetros para cada modelo. El otro subconjunto se particionará a su vez en dos: entrenamiento y test, en proporciones 89 % y 11 %, que se emplearán para realizar el entrenamiento y evaluación de los modelos.

2.4.1. Repeticiones

Al realizar un entrenamiento y la posterior evaluación de un modelo las medidas de desempeño obtenidas dependerán de la partición realizada del conjunto de datos. Se pueden obtener resultados muy diferentes y esto nos puede llevar a aceptar como modelo más correcto uno que realmente no lo es.

Para evitar este problema de aleatoriedad se llevan a cabo 100 repeticiones de este procedimiento. Es decir se realizan particiones diferentes de la muestra de datos en entrenamiento y test para obtener así 100 métricas de evaluación diferentes. A partir de esto se calcularán los valores medios de dichas métricas y estos serán empleados para comparar los modelos.

2.4.2. Validación cruzada

La validación cruzada es una herramienta muy útil para evaluar el desempeño de un modelo, esta adopta diferentes formas dependiendo de como se particionen los conjuntos de datos. Una de las versiones más utilizadas es la validación cruzada de K grupos en la cual se divide la muestra en K subconjuntos. Uno de ellos se usará como datos de prueba y los $K-1$ restantes se emplean como datos de entrenamiento, este proceso se repite iterativamente hasta haber utilizado los K subconjuntos como muestras de test. Así se obtiene una medida más precisa al conseguir K entrenamientos del modelo como se expone en Refaeilzadeh et al. (2016), permitiendo así comparar los modelos de una manera más adecuada. La aplicación de este método se muestra gráficamente en la Figura 2.8.

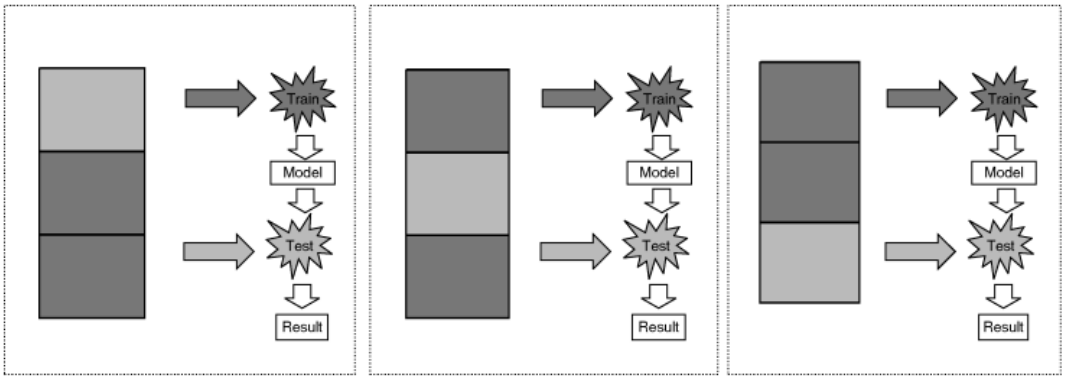


Figura 2.8: Procedimiento de validación cruzada de K grupos para el caso $K = 3$. Imagen obtenida de Refaeilzadeh et al. (2016).

En este trabajo se empleará la validación cruzada de K grupos en dos ocasiones distintas. La primera consistirá en aplicar dicha técnica, con $K = 5$, sobre un conjunto de datos de validación y entrenar los modelos con diferentes parámetros para así poder determinar cuales son más adecuados para ajustarlo. Así podemos determinar la combinación adecuada para analizar el resto de datos como se expone en Müller y Guido (2016).

A la hora de comprobar el desempeño de un modelo de clasificación hay que tener en cuenta que este puede generar diferentes resultados dependiendo de la partición entre entrenamiento y test utilizada. Para solventarlo aplicamos validación cruzada una segunda vez con el objetivo de obtener varios valores de las medidas de evaluación con diferentes particiones de los datos en entrenamiento y test y así evitar ese efecto de aleatoriedad mencionado. Esto ayudará a comparar más correctamente los diferentes modelos para estudiar cual produce mejores resultados. Para este caso se realizará validación cruzada con $K = 10$ iteraciones y este procedimiento se repetirá 10 veces con el objetivo de obtener 100 medidas diferentes. Al realizar este procedimiento se obtendrían realmente 100 modelos diferentes y puede surgir la duda que cual será el modelo final, para sortear esto toma como tal aquel modelo que obtiene mejores métricas reentrenado usando la totalidad de los datos salvo los de validación.

Como ya se ha comentado, un problema con el tratamiento de datos desbalanceados es que los clasificadores suelen ser más sensibles a la detección de la clase mayoritaria. Esto se intenta corregir empleando técnicas de remuestreo pero cabe recalcar que estas se han de llevar a cabo únicamente sobre el conjunto de entrenamiento para obtener resultados correctos. Si se realiza el remuestreo antes de particionar los datos cabe la posibilidad de que se tengan datos iguales o muy similares, dependiendo del tipo de remuestreo empleado, en las muestras de entrenamiento y test lo cual puede provocar que se produzca un sobreajuste en el entrenamiento del modelo.

2.4.3. Medidas de evaluación

Se comparará el desempeño de los diferentes modelos recurriendo a diferentes medidas todas ellas relacionadas con la capacidad del mismo a clasificar correctamente nuevas observaciones diferentes a las muestras de entrenamiento, las muestras de test. Recurrimos a la matriz de confusión, Tabla 2.1, del modelo para introducir las expresiones de las medidas.

		Predicción	
		No fraude	**Fraude**
Valor real	**No fraude**	Verdaderos negativos (VN)	Falsos positivos (FP)
	Fraude	Falsos negativos (FN)	Verdaderos positivos (VP)

Tabla 2.1: Estructura de la matriz de confusión.

Se presentan aquí diferentes medidas que se estudiarán. Dos medidas muy utilizadas son la precisión global o *accuracy* y el índice predictivo positivo o *precision*. En el problema planteado la clases están desbalanceadas, por lo que podría ocurrir que la *accuracy* no aportase demasiada información como se explica en Fernández et al. (2021). Por lo tanto se emplea la *balanced accuracy* (BA) en su lugar, para ello hay que introducir la sensibilidad y la especificidad, que son las tasas de verdaderos positivos (TPR) y la de verdaderos negativos (TNR). La tasa de verdaderos positivos también recibe el nombre de *recall*. Sus expresiones son:

$$accuracy = \frac{VP + VN}{VP + VN + FP + FN}$$

$$precision = \frac{VP}{VP + FP}$$

$$TPR = recall = \frac{VP}{VP + FN}$$

$$TNR = \frac{VN}{VN + FP}$$

$$BA = \frac{TPR + TNR}{2}$$

Existe otra medida, denominada F1-score, que es interesante pues combina *accuracy* y *recall* realizando su media armónica. Esta tiene forma:

$$F1 - score = \frac{2 \cdot precision \cdot recall}{precision + recall} = \frac{2VP}{2VP + FP + FN}$$

En nuestro problema no sólo interesa predecir correctamente la mayor cantidad de casos fraudulentos posibles, sino que también interesa no tener una gran cantidad de falsos positivos ya que no es conveniente para una entidad bloquear a sus clientes de realizar transacciones. Para elegir la medida adecuada para determinar el modelo más correcto es necesario tener en cuenta qué costes tiene el hecho

de que se clasifique erróneamente una observación. El importe medio de las operaciones fraudulentas es $valF = 298.7€$ y para las no fraudulentas es de $valN = 2\,700.4€$. El valor medio es mayor en las operaciones no fraudulentas puesto que muchas operaciones fraudulentas se realizan con valores bajos para intentar que estas pasen desapercibidas. Queremos emplear una función de coste de forma lineal, para ello es necesario asignar unos coeficientes a los diferentes valores de la matriz de confusión. Estos costes serán una primera aproximación que podrán modificarse en futuros trabajos dependiendo de los cambios en los comportamientos de los clientes y las necesidades de la entidad.

Hay que tener en cuenta que cuando se categoriza una observación como fraudulenta entonces es necesario realizar una llamada al cliente para así poder confirmar si esta fue realmente fraudulenta (VP) o si se trata de un falso positivo (FP). Este coste se estima a partir de la remuneración de un trabajador de un centro de atención y el tiempo que tarda en realizar la tarea en 2.7€. Además, hay que tener en cuenta que el hecho de bloquear una operación a un cliente conlleva dos pérdidas: la de una posible comisión, para la que tomamos un $0.2\,\%$, y una pérdida de confianza del cliente hacia la entidad cuyo valor es inmensurable e incluso puede llevarlo a abandonar los servicios. Este último valor se concreta en 2.7€ por paralelismo al precio de la llamada de comprobación. Teniendo en cuenta el importe medio de una operación no fraudulenta tenemos que el valor asociado a un falso positivo es de $costFP = valN * 0.2\,\% + 2.7 + 2.7 = 10.8€$.

El coste de un falso negativo se toma como el importe medio de las transacciones negativas. Si asumimos que el cliente realizará una llamada a la entidad cuando advierta la operación fraudulenta en su cuenta, hemos de sumar el precio de esta. De manera que el coste para un falso negativo se define mediante $costFN = valF + 2.7 = 301.4€$.

Para la función de coste no tendremos en cuenta los casos bien clasificados por el modelo, por lo tanto los verdaderos negativos tienen coste 0 y para los verdaderos positivos tenemos que el coste es simplemente el precio de la llamada $costVP = 2.7€$.

La métrica principal que utilizaremos para comparar los modelos será el coste, aunque también se estudiará el valor de las otras métricas de las expuestas en esta sección.

También nos interesa estudiar el tiempo de predicción de los modelos, ya que como se ha comentado la respuesta ha de ser menor a un segundo.

Una vez se haya seleccionado el modelo final se empleará los valores Shapley para estudiar el efecto final de las diferentes variables en él. El valor de Shapley es un concepto de solución en teoría de juegos cooperativos. Fue introducido por Shapley (1953) lo cual le llevó a recibir el premio Nobel en economía por ello. Calcular el valor teórico es complicado y por tanto en la práctica se suelen realizar estimaciones.

Al emplear árboles de decisión se puede calcular exactamente este valor. La predicción se puede explicar asumiendo que cada variable es un jugador de un juego donde la predicción es la indicación de fraude. El valor de Shapley es la media de las diferencias entre las contribuciones marginales al considerar predicciones entre las diferentes coaliciones posibles. Los tiempos de computación aumentan exponencialmente con el número de variables. Se expresa la formulación matemática en Molnar (2022). Se puede entender como que los valores Shapley aproximan el efecto correspondiente a eliminar una variable del conjunto de datos y por tanto muestran la importancia de cada variable en la predicción realizada. En nuestro problema se tiene que un mayor valor Shapley indica una mayor probabilidad de fraude.

Lundberg et al. (2020) emplearon los valores *Shapley Additive Explanation* (SHAP) para crear una serie de herramientas que permiten estudiar los efectos de las variables no solo en un contexto global sino mediante el estudio de predicciones individuales.

Capítulo 3

Tratamiento de datos

Los datos problema de operaciones proporcionados por ABANCA contienen 134 variables de las cuales dos son de vital importancia. La más importante de todo el conjunto de datos se denomina ETIQUETA e informa sobre si la operación correspondiente fue fraudulenta (valor 1) o no (valor 0), esta es por tanto nuestra variable respuesta. Otra variable importante es CANAL, mediante la cual se pueden particionar los datos entre operaciones realizadas mediante un dispositivo móvil (mobile) u online (online). Esta partición de los datos tiene mucha importancia ya que se observan algunas variables que sólo están presentes en uno de los casos. El objetivo de este trabajo es encontrar un modelo que sea capaz de clasificar una operación en fraudulenta o no fraudulenta en los casos donde el CANAL tome el valor online.

Para realizar la búsqueda del modelo partimos de una base de datos de gran tamaño, se trata de un conjunto operaciones realizadas entre el 01/06/2022 y el 17/02/2023 de las cuales tan sólo son 1 373 son fraudulentas. Este es por tanto un conjunto de datos desbalanceado ya que tan sólo un 1.4 %. Se someterán los datos a un análisis exploratorio inicial para poder descartar aquellas variables que no afecten al fraude o que tengan demasiados datos faltantes.

El plan de trabajo a seguir en este capítulo viene dado por:

- Análisis inicial: se realiza un estudio inicial para familiarizarnos con los datos y poder proponer nuevas variables y la modificación de otras ya existentes.
- Análisis exploratorio: se realiza un análisis exploratorio de los datos del problema. Se estudian las variables y su influencia sobre el fraude, aplicando medidas de asociación tanto entre dos variables diferentes como en una misma variable en sus casos fraudulentos y no fraudulentos.
- Problema con datos faltantes: se estudia la cantidad de datos faltantes presentes en el problema y se realiza la imputación de los mismos.

El objetivo de este capítulo es preparar los datos para poder aplicar sobre ellos las diferentes técnicas de remuestreo y posteriormente los modelos de clasificación.

3.1. Análisis inicial

El primer paso a la hora de trabajar con un conjunto de datos desconocido es estudiar su estructura. Interesa saber tanto las dimensiones del conjunto, el número de observaciones y de variables, como también es importante saber el tipo de datos que conforma cada variable y la distribución que poseen los valores que toma. Para realizar este estudio emplearemos la librería SweetViz de Python la cual permite realizar un breve informe para cada variable del conjunto problema comparando los

casos de operaciones fraudulentas y no fraudulentas. Este informe nos sirve de ayuda para proponer las distintas transformaciones que se exponen a continuación. En el apéndice A se incluye el listado de variables y una breve descripción de la información que contiene.

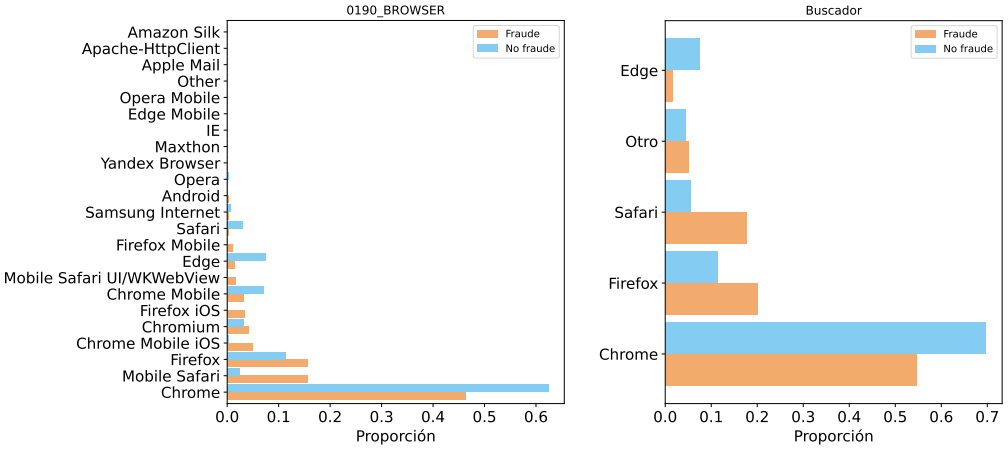
Es importante destacar que este desarrollo se lleva a cabo desde la perspectiva de implementar el modelo en producción. Por tanto todos los cambios realizados en el conjunto de datos original han de ser aplicados a las nuevas observaciones a la hora de hacer una nueva predicción, y esta modificación tomará un determinado tiempo en ejecutarse para realizar una predicción en tiempo real. Por lo tanto, interesa hacer el menor cambio de datos posible ya que, como se ha comentado en los objetivos, el modelo ha de realizar la predicción en el menor tiempo posible dentro de ciertos límites.

3.1.1. Creación de nuevas variables

En ocasiones casos interesa modificar las variables de distintas maneras e incluso crear variables nuevas antes de ajustar los diferentes modelos. Cabe destacar que se presentarán casos en los que una de estas nuevas variables no se pueda calcular debido a que los valores de las variables necesarias para calcularla sean nulos, se incluirá un valor nulo para la nueva variable en tal operación. Las nuevas variables creadas son:

Buscador

La variable 0190_BROWSER proporciona información sobre el navegador utilizado para realizar la operación. Esta variable posee 21 categorías de las cuales hay casos en los que la cantidad de datos es muy baja, como son *Amazon Silk* y *Maxthon* con 5 y 2 observaciones respectivamente. Se puede observar en la Figura 3.1 el gráfico de barras correspondiente a esta variable comparando las proporciones de casos fraudulentos y no fraudulentos. Para reducir el número de categorías se propone la variable Buscador que agrupa los valores de la variable en cinco categorías: Chrome, Firefox, Edge, Safari u Otro.



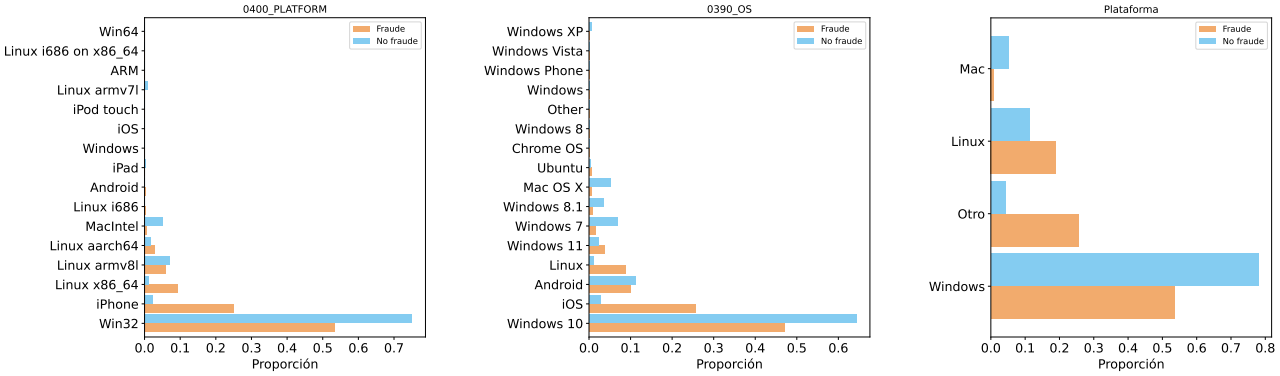
(a) (b)

Figura 3.1: Gráfico de barras de las variables 0190_BROWSER y Buscador comparando los casos fraudulentos y no los no fraudulentos.

La nueva variable Buscador estar relacionada con el fraude ya que se observa una gran discrepancia entre el porcentaje de casos fraudulentos y no fraudulentos para algunas de las categorías.

Plataforma

Las variables 0400_PLATFORM y 0390_OS son categóricas, con 21 y 16 casos distintos respectivamente, y presentan información similar correspondiente al sistema operativo del dispositivo utilizado. Estos valores se agrupan en una nueva variable, Plataforma, en cuatro categorías: Windows, Mac, Linux y Otro. La variable 0390_OS tiene menor cantidad de datos faltantes, un 1 % frente a un 6 % de 0400_PLATFORM. El modo de actuar para una operación concreta es obtener el valor de 0390_OS y si este es un valor faltante, obtenerlo de 0400_PLATFORM si está disponible. La Figura 3.2 nos permite observar que existe discrepancia entre la distribución de los datos fraudulentos y la de los no fraudulentos.



(a) (b) (c)

Figura 3.2: Gráfico de barras de las variables 0400_PLATFORM, 0390_OS y Plataforma comparando los casos fraudulentos y no los no fraudulentos.

Region

Las variables 0700_CONTINENTCODE, 0710_COUNTRY y 0780_REGION ofrecen información sobre el continente, el país y la región desde la que se realiza la operación. Estas contienen una gran cantidad de categorías distintas (5, 85 y 231 respectivamente) por lo que se propone agruparlas. Para reducir el números sin perder demasiada información, se agrupan los valores en comunidades autónomas si la operación se realizó desde España o por continentes si se realizó desde fuera. Esta separación se justifica ya que la mayor parte (un 96 %) de las transacciones se realizan desde España. Se crea así la variable Region cuya distribución se presenta en la Figura 3.3. Se incluye en esta figura también las distribuciones de las variables originales, aunque para los casos de 0710_COUNTRY y 0780_REGION se muestran únicamente los casos con mayor número de observaciones.

Teleoperadora

La variable 0740_ISP es una variable categórica con información sobre el proovedor de internet. Esta posee 840 casos distintos y algunos casos tienen poca información. Además, existen instancias en los que una misma teleoperadora aparece en dos categorías debido a una diferente nomenclatura como son el caso por ejemplo de *Orange España* y *Orange Spain Network*. Se crea la variable Teleoperadora para agrupar en los siguientes casos: Telefonica, R Cable, Vodafone, Orange, Yoigo y Otro. Así conseguimos categorías con un número de casos más homogéneo. La Figura 3.4 muestra el diagrama de barras para estas dos variables, apareciendo en el caso de 0740_ISP únicamente los 15 valores más frecuentes. Observamos que existe una gran discrepancia entre los diferentes casos, por lo tanto esta

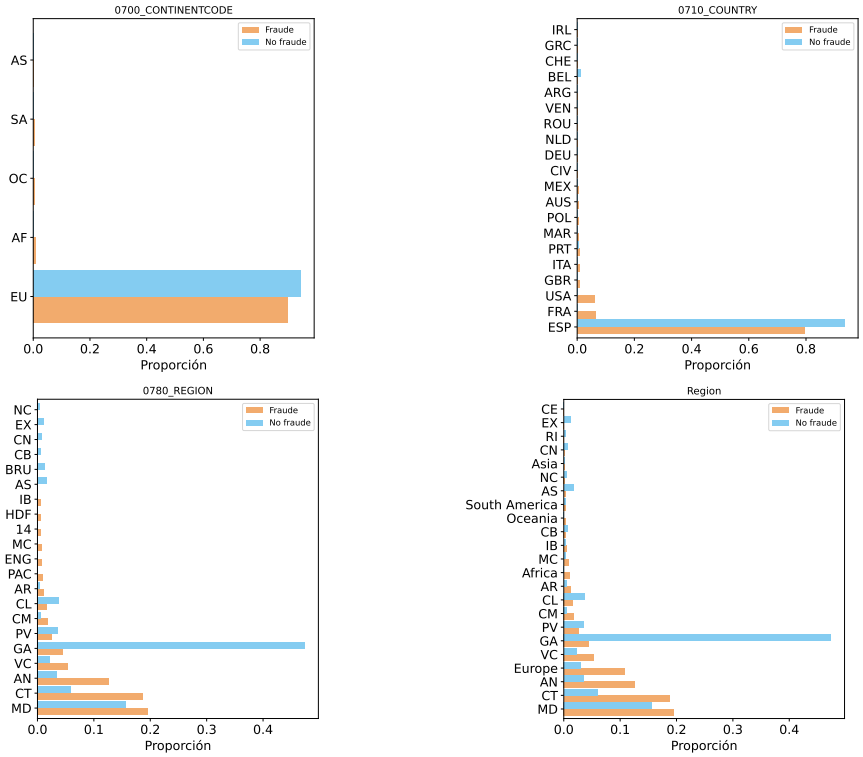


Figura 3.3: Gráfico de barras de las variables 0700_CONTINENTCODE, 0710_COUNTRY y 0780_REGION y Region comparando los casos fraudulentos y no los no fraudulentos.

variable resulta de interés.

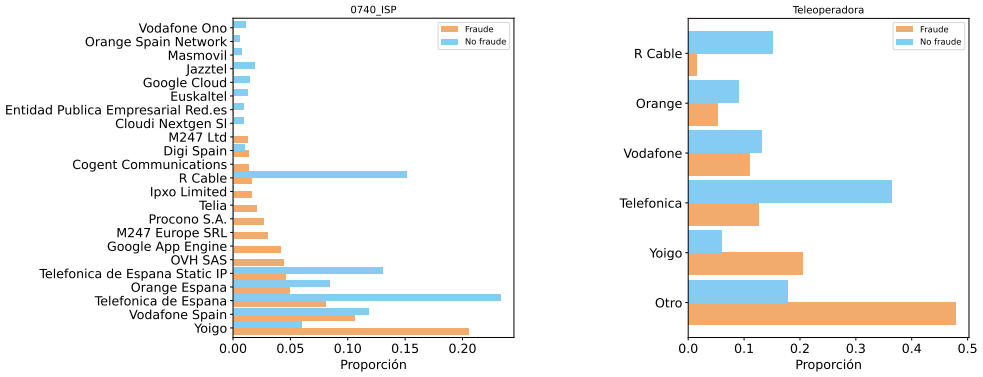


Figura 3.4: Gráfico de barras de las variables 0740_ISP y Teleoperadora comparando los casos fraudulentos y no los no fraudulentos.

DistanciaNavegador

La variable NAVEGADOR incluye información sobre el navegador usado en la transacción mientras que la variable NAVEGADOR_LIST incluye un listado sobre los navegadores habituales del usuario. Estas variables tienen 6532 y 30408 categorías únicas. Una de las observaciones para NAVEGADOR

es de la forma:

Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36

Por lo tanto creamos la variable DistanciaNavegador que presenta valores continuos entre 0 y 100 dependiendo si el navegador utilizado se parece a alguno de los navegadores usuales o no. Si el navegador está en la lista entonces toma el valor 100 y si no está toma valores más bajos hasta un mínimo de 0 dependiendo de cuanto de diferente sea. Se presenta en la Figura 3.5 las funciones de densidad de valores para los casos fraudulentos y no fraudulentos además de su boxplot. Se observa una clara discrepancia en la distribución de los valores fraudulentos y los no fraudulentos.

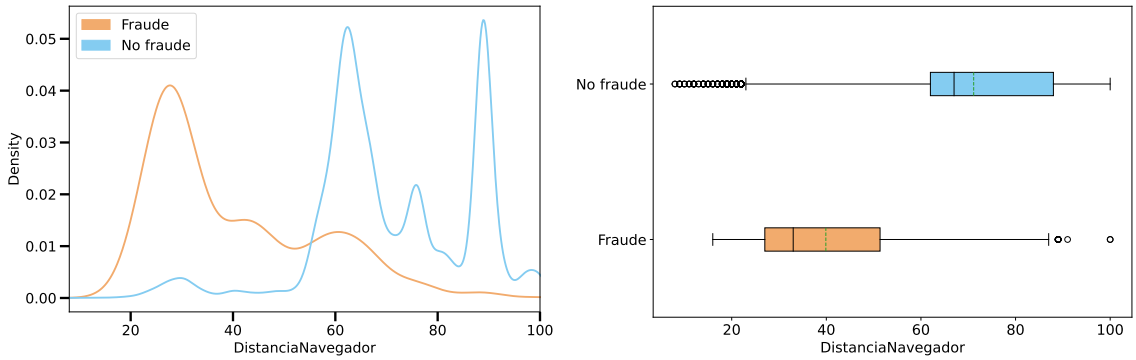


Figura 3.5: Densidades de valores (izda) y boxplots (dcha) para la variable DistanciaNavegador comparando los casos fraudulentos y no los no fraudulentos.

Pais_Usual

Retomando la variable 0710_COUNTRY previamente introducida y usando la variable COUNTRY_LIST que da información de los países usuales desde los que un cliente realiza operaciones se crea la variable indicadora Pais_Usual da información sobre si el país desde el que se realiza la operación está dentro los países usuales del cliente (valor 1) o no (valor 0). Vemos en el diagrama de barras de la Figura 3.6 que sí esta realcionada con el fraude.

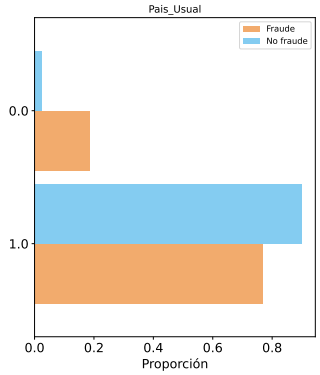


Figura 3.6: Gráfico de barras de la variable Pais_Usual comparando los casos fraudulentos y no los no fraudulentos.

3.1.2. Transformaciones logarítmicas

Al realizar el análisis inicial se ha comprobado que existe un número de variables que toman valores en un rango muy alto, desde cero hasta valores superiores a un millón. Se realiza una transformación logarítmica de estas variables para reducir la asimetría.

Las variables que poseen una distribución con gran asimetría positiva y que por tanto serán sometidas a la transformación logarítmica son:

- TXN_W1_TXN_AMT_MEAN
- TXN_M1_TXN_AMT_MEAN
- TXM_3M_TXN_AMT_MEAN
- TXN_M1_TXN_AMT_STD
- TXM_3M_TXN_AMT_STD
- TXN_W1_TXN_MAX_AMT
- TXN_M1_TXN_MAX_AMT
- TXM_3M_TXN_MAX_AMT
- DIGITALIDAD_RELATIVA
- DIGITALIDAD_RELATIVA_EDAD
- SAME_IP_ACROSS_CANAL_W1_CNT
- SAME_IP_ACROSS_CANAL_M1_CNT
- IMPORTE
- ACCESSES_BMOVIL
- ACCESSES_BMOVIL_W1
- ACCESSES_BMOVIL_M1
- ACCESSES_BMOVIL_3M
- ACCESSES_BE
- ACCESSES_BE_W1
- ACCESSES_BE_M1
- ACCESSES_BE_3M
- COUNTRIES_SWITCH_30_MIN_W1_CNT
- COUNTRIES_SWITCH_30_MIN_M1_CNT

Una de las variables es IMPORTE y corresponde con la cantidad de dinero enviada en la operación. Las variables DIGITALIDAD_RELATIVA y DIGITALIDAD_RELATIVA_EDAD son coeficientes que indican el grado de digitalidad del usuario y dicho grado pero teniendo en cuenta la edad del mismo.

Las variables cuyo nombre tiene nomenclatura TXN_X contienen información sobre las medias, desviaciones típicas y máximos de los importes de transacciones en diferentes periodos de tiempos. Se incluye una descripción detallada en el apéndice A.

Aquellas variables con nombre de la forma ACCESSES_X contienen información sobre el número de accesos a banca móvil o banca electrónica totales y en distintos periodos de tiempo. De nuevo se encuentran descritas detalladamente en el apéndice A.

Se tiene que log_COUNTRIES_SWITCH_30_MIN_W1_CNT y log_COUNTRIES_SWITCH_30_MIN_M1_CNT indican el número de veces que ha habido un cambio de país entre operaciones distintas en un tiempo menor a 30 minutos para el usuario en la última semana y mes respectivamente.

Por otro lado, log_SAME_IP_ACROSS_CANAL_W1_CNT, y log_SAME_IP_ACROSS_CANAL_M1_CNT proporcionan el número de IPs utilizadas por el usuario en la última semana y el último mes respectivamente.

A pesar de haber un gran número de variables, en concreto 23 variables, a las que se le aplica transformación logarítmica para corregir este problema hemos visto que la información que estas proporcionan está relacionada entre ellas mediante grupos.

Estas nuevas variables recibirán como nombre su nombre inicial precedido de log_, por ejemplo IMPORTE pasará a ser log_IMPORTE.

A continuación mostramos en la Figura 3.7 un ejemplo de la diferencia de comportamiento de la variable TXM_3M_TXN_AMT_MEAN, que contiene información sobre la media de importes de las transacciones del último mes, antes y después de aplicar transformación logarítmica usando boxplots. Observamos que se corrige la asimetría de la distribución de la variable.

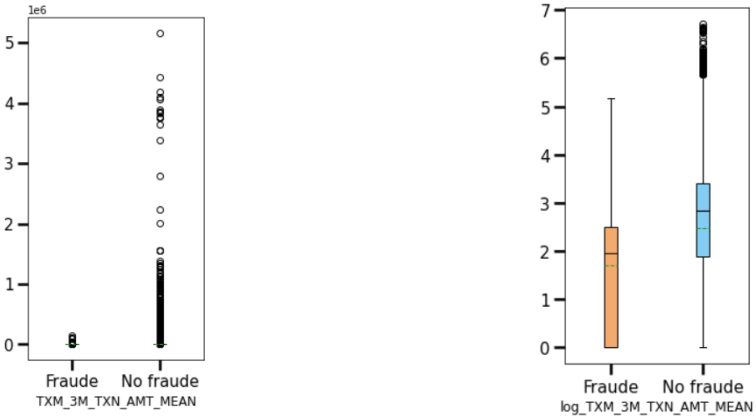


Figura 3.7: Boxplots de las variables TXM_3M_TXN_AMT_MEAN y log_TXM_3M_TXN_AMT_MEAN comparando los casos fraudulentos y no los no fraudulentos. La linea verde horizontal indica la media y la negra la mediana.

Se observa esta mejora en la distribución de los datos en todos los casos aplicados.

3.1.3. Eliminación de variables

Los datos problema proceden de una base de datos con operaciones realizadas mediante canales mobile u online por lo que existen variables correspondientes a características únicas de cada tipo de canal. Como el objetivo es conseguir un modelo para operaciones de canal online, debemos eliminar las variables que no se correspondan con ese canal. Estas variables incluyen 0450_BATTERYCHARGING, 0530_ISCALLINPROGRESS, 0550_LINECARRIER, 0630_SMSSTEALERS y 1490_batteryPercentage entre otras.

Ocurre también que hay algunas variables que no son más que la normalización de otras variables tal y como se indica en las descripciones de las variables del apéndice A. Por lo tanto estas no van a aportar nueva información al modelo y conviene eliminarlas para evitar efectos de colinealidad. Estas variables son:

- ACCESSES_BMOVIL_W1_ZS
- ACCESSES_BMOVIL_M1_ZS
- ACCESSES_BMOVIL_3M_ZS
- ACCESSES_BMOVIL_ZS
- ACCESSES_BE_W1_ZS
- ACCESSES_BE_M1_ZS

- ACCESSES_BE_3M_ZS

- ACCESSES_BE_ZS

3.2. Análisis exploratorio

Estudiamos diferentes medidas para estudiar el comportamiento de los datos que compone cada variable. Nos interesa también saber que distribución tienen los mismos en los casos fraudulentos y no fraudulentos. Así podremos determinar cuales de las variables varían más en ambos casos y por tanto sería de esperar que resultaran de más utilidad a la hora de predecir el fraude.

3.2.1. Medidas de asociación

Como se comentó en la Sección 2.4.3 se usarán diferentes medidas para medir las asociaciones entre variables. La correlación de Pearson y la V de Cramér se emplear para medir la asociación entre variables continuas y discretas respectivamente. Para estudiar la influencia del fraude en las variables usaremos tests de homogeneidad de Kolmogorov-Smirnov para variables continuas y un test χ^2 para casos discretos. Finalmente, estudiamos la correlación de distancia para comprobar la asociación entre cada variable y la variable ETIQUETA, viendo así la influencia del fraude en ellas.

Correlaciones de Pearson

Estudiamos ahora las correlaciones de Pearson entre las variables numéricas continuas. Incluimos en la Figura 3.8 el mapa de calor de las correlaciones. En el apéndice B se presenta la tabla con las correlaciones entre pares de variables en los casos en los que esta toma valores mayores a 0.9 o menores a -0.9.

Observamos que existe una gran correlación entre algunas variables, sobre todo aquellas que tratan los mismos comportamientos del usuario pero diferentes determinados intervalos de tiempo, como es de esperar. Se encuentran entre estas por ejemplo los ratios del número de accesos a la banca electrónica móvil: totales, en los últimos tres meses, en el último mes y en la última semana. Similarmente, se encuentran altamente correlacionadas las variables correspondientes al logaritmo del número de accesos a la banca móvil o la banca electrónica.

Mostramos en la Figura 3.9 el diagrama de dispersión de las variables log_TXN_W1_TXM_AMT_MEAN y log_TXN_W1_TXM_MAX_AMT y así comprobamos también gráficamente que estas variables están relacionadas.

Con el objetivo de evitar problemas de colinealidad eliminamos las variables altamente correlacionadas y de cada par escogemos mantener aquellas que poseen mayor correlación de distancia con ETIQUETA. Así disminuimos también el número de variables de problema. Las variables de las que prescindimos son:

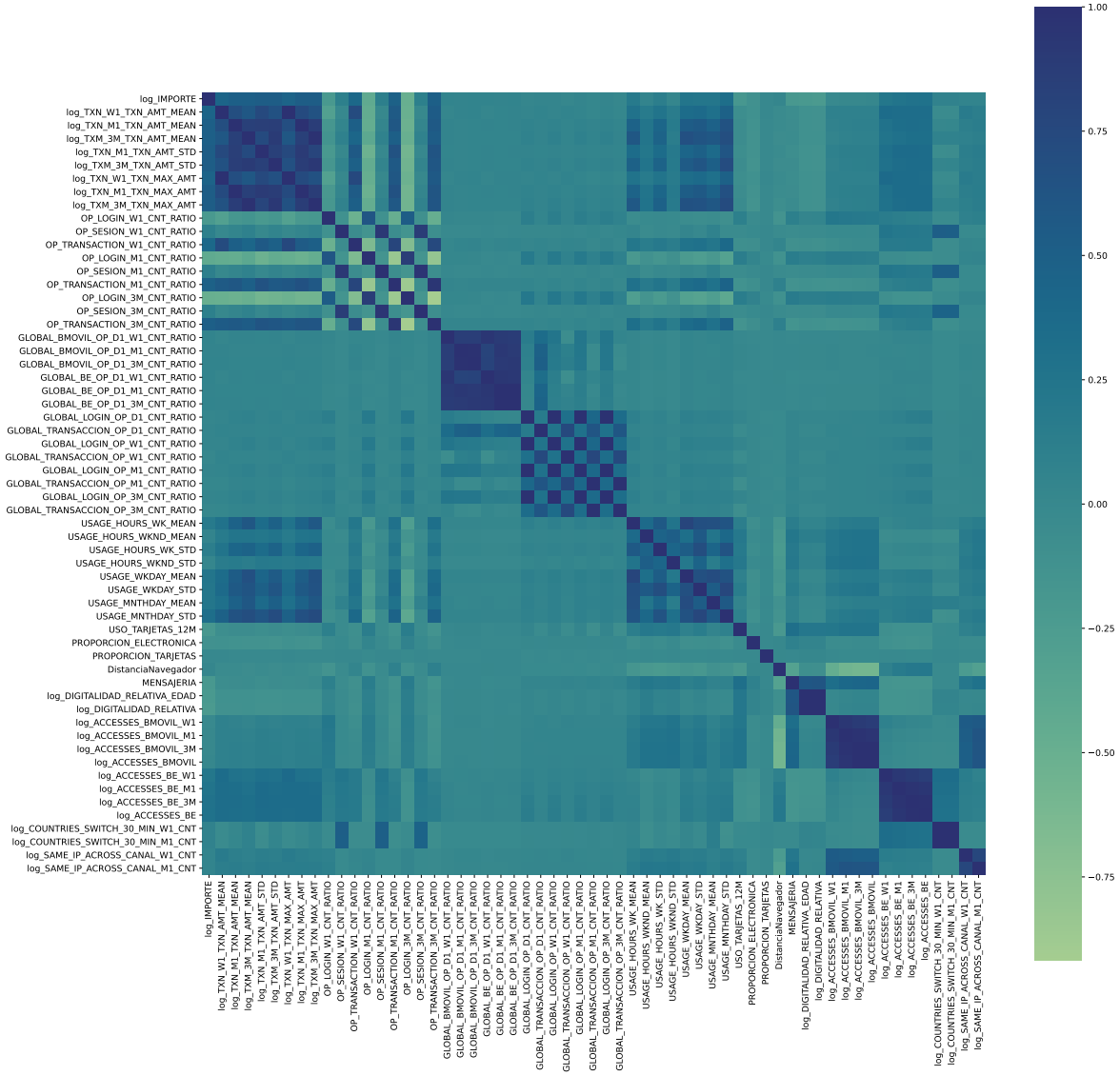


Figura 3.8: Mapa de calor de las correlaciones de Pearson entre variables continuas.

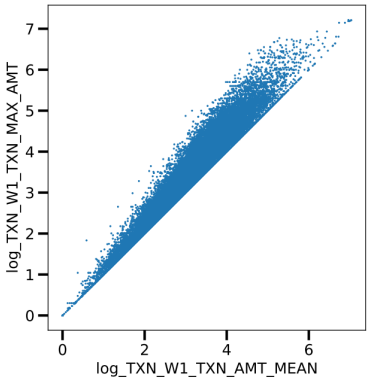


Figura 3.9: Diagrama de dispersión de log_TXN_W1_TXM_AMT_MEAN y log_TXN_W1_TXM_MAX_AMT.

- GLOBAL_BE_OP_D1_M1_CNT_RATIO
- GLOBAL_BE_OP_D1_W1_CNT_RATIO
- GLOBAL_BE_OP_D1_3M_CNT_RATIO
- GLOBAL_BMOVIL_OP_D1_M1_CNT_RATIO
- GLOBAL_LOGIN_OP_D1_CNT_RATIO
- GLOBAL_LOGIN_OP_M1_CNT_RATIO
- GLOBAL_LOGIN_OP_W1_CNT_RATIO
- GLOBAL_TRANSACCION_OP_M1_CNT_RATIO
- OP_LOGIN_3M_CNT_RATIO
- OP_TRANSACTION_M1_CNT_RATIO
- log_ACCESSES_BE_3M
- log_ACCESSES_BE_M1
- log_ACCESSES_BMOVIL
- log_ACCESSES_BMOVIL_M1
- log_ACCESSES_BMOVIL_W1
- log_DIGITALIDAD_RELATIVA_EDAD
- log_TXM_3M_TXN_MAX_AMT
- log_TXN_M1_TXN_AMT_MEAN
- log_TXN_W1_TXN_AMT_MEAN

V de Cramér

Utilizamos la V de Cramér para comprobar la asociación presente entre pares de variables discretas. Se muestra en la Figura 3.10 el mapa de calor de esta medida entre las diferentes variables. En la Tabla 3.1 vemos los casos con valores más altos de V de Cramér para cada par de variables, se muestra también la correlación de distancia de cada variable con ETIQUETA con el fin de comparar la influencia de cada una de ellas en el fraude. Eliminamos por tanto aquellas con menor correlación de distancia: MENSAJERIA, 0440_SCREENWIDTH Y 0420_SCREENHEIGHT.

Test de Kolmogorov-Smirnov

Aplicamos el test de Kolmogorov-Smirnov para medir si la distribución de los datos fraudulentos es la misma que la de los no fraudulentos en los casos de variables numéricas continuas. Aplicamos también la correlación de distancias entre cada variable y la variable ETIQUETA y la representación en gráficos de barras con las densidades de distribución de ambos casos para apoyar los resultados obtenidos con el test. En la Tabla 3.2 presentamos los resultados en aquellos casos para los que se ha obtenido un p-valor mayor 0.05.

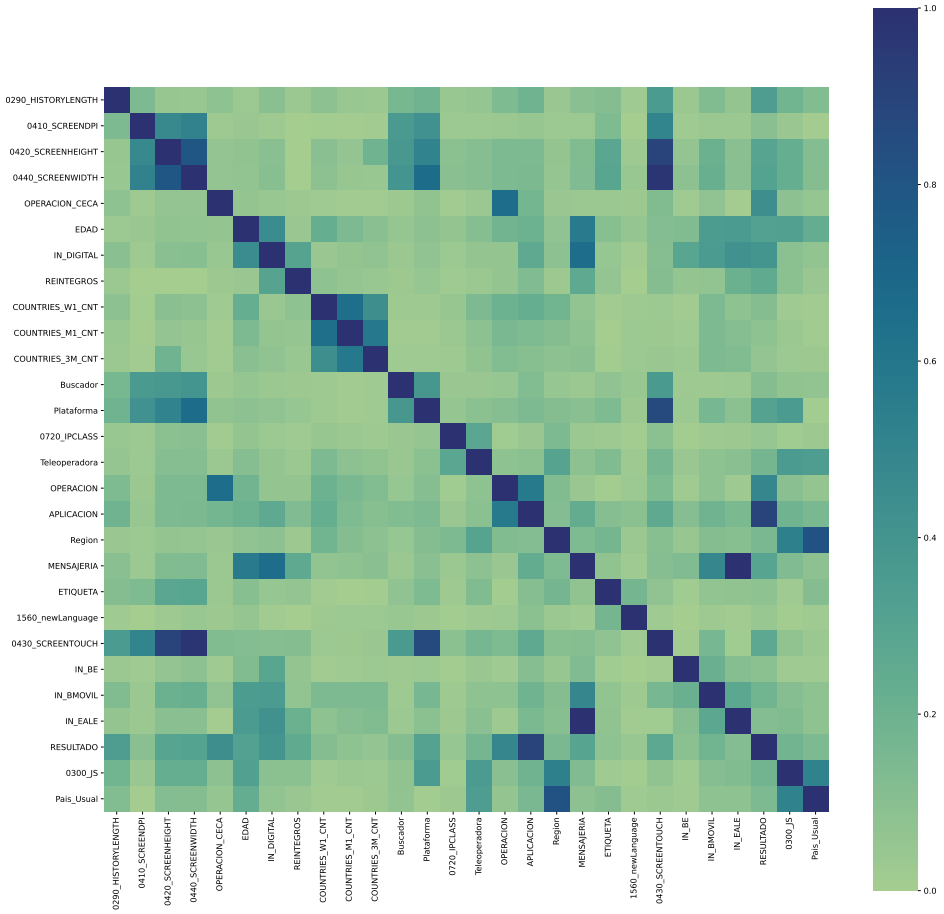


Figura 3.10: Mapa de calor de la V de Cramér entre variables discretas.

Variable 1	dcor 1	Variable 2	dcor 2	V de Cramér
IN_EALE	0.0740	MENSAJERIA	0.0697	0.9932
0430_SCREENTOUCH	0.0760	0440_SCREENWIDTH	0.0693	0.9748
0420_SCREENHEIGTH	0.0373	0430_SCREENTOUCH	0.0760	0.8950

Tabla 3.1: Valores de la V de Cramér para cada pares de variables en los casos donde esta toma valores más elevados. Se incluyen las correlaciones de la distancia entre las variables 1 y 2 y la respuesta, variable ETIQUETA, denominadas respectivamente dcor1 y dcor2.

Variable	p-valor	Correlación de distancia
OP_SESION_W1_CNT_RATIO	0.994	0.0046
OP_SESION_M1_CNT_RATIO	0.465	0.0073
PROPORCION_ELECTRONICA	0.720	0.0108
PROPORCION_TARJETAS	1	0.0022
log_COUNTRIES_SWITCH_30_MIN_W1_CNT	1	0.0096
log_COUNTRIES_SWITCH_30_MIN_M1_CNT	1	0.0086

Tabla 3.2: Resultados obtenidos al usar tests de Kolmogorov-Smirnov a las diferentes variables numéricas. Se muestran los p-valores del test y la correlación de la distancia con ETIQUETA obtenidas para algunas variables.

Observamos que hay seis variables las cuales devuelven un alto p-valor al aplicar los tests de Kolmogorov-Smirnov. Estos p-valores son tan altos que para cualquier nivel de significación usual (α =0.01,0.05,0.1) se obtiene que las distribuciones de los datos fraudulentos y no fraudulentos para estas variables no son significativamente distintos. Por comparativa, en las variables con mayor valor de correlación de distancia con la variable ETIQUETA, que se mostrarán más adelante en la Tabla 3.3, se obtienen valores de 0.0697 para la decimoquinta variable más importante y estos valores aumentan hasta 0.2178 en el mejor de los casos. Por lo tanto aceptamos que la correlación de distancias con la variable ETIQUETA es muy baja en estos casos.

También se observa esta idea gráficamente mediante los gráficos de barras con las funciones de densidad. Por todas estas razones podemos concluir que no hay efecto significativo del fraude en estas variables y por tanto consideramos que no serán de utilidad para predecirlo.

Test χ^2

Aplicamos el test χ^2 para comprobar si la distribución de los datos fraudulentos es la misma que la de los no fraudulentos en los casos de variables discretas. Aplicamos también la correlación de distancias entre cada variable y la variable ETIQUETA y la representación en gráficos de barras, con las densidades de distribución en casos de variables numéricas, para corroborar los resultados obtenidos con el test.

Solamente la variable COUNTRIES_M1_CNT presenta un p-valor mayor 0.05, concretamente el valor obtenido es de 0.8310. Su valor de correlación de distancia con ETIQUETA es de 0.0022 el cual es un valor muy bajo comparado con otras variables. Observando el gráfico de barras comparando los casos fraudulentos y no fraudulentos comprobamos que aparentemente esta variable no está relacionada con el fraude y por lo tanto decidimos prescindir de ella.

Importancia de las variables

Gracias a la correlación de la distancia con la variable ETIQUETA obtenemos una medida de la importancia que tiene cada variable en el fraude. Esta es una medida univariante, al considerar modelos multivariantes esto podría cambiar por ello en la Sección 5.2.2 se medirá la importancia real de las variables en el modelo final. Se muestran en la Tabla 3.3 todas las doce variables con mayor valor de correlación de distancia.

Variable	Correlación de distancia
log_ACCESSES_BE	0.2178
DistanciaNavegador	0.2082
1560_newLanguage	0.1682
log_ACCESSES_BE_W1	0.1527
0410_SCREENDPI	0.1269
Pais_Usual	0.1175
APLICACION	0.1068
log_ACCESSES_BMOVIL_3M	0.0977
Region	0.0875
Resultado	0.0821
GLOBAL_TRANSACCION_OP_3M_CNT_RATIO	0.0769
0430_SCREENTOUCH	0.0760

Tabla 3.3: Variables y su correlación de la distancia con la variable ETIQUETA para los casos de mayor de esta.

Procedemos a analizar las variables con mayor valor de correlación de distancia con ETIQUETA para mostrar el comportamiento de las mismas. En la Tabla 3.4 mostramos un resumen con distintas medidas descriptivas de estas variables y se representan en la Figura 3.11 los boxplots de los mismos.

Variable	Caso	Media	Std	25 %	50 %	75 %
log_ACCESSES_BE	Fraude	0.683	0.789	0	0.301	1.230
log_ACCESSES_BE	No fraude	2.057	0.700	1.663	2.124	2.474
DistanciaNavegador	Fraude	39.000	16.199	27.000	33.000	51.360
DistanciaNavegador	No fraude	71.149	15.300	62.000	67.000	88.000
log_ACCESSES_BE_W1	Fraude	0.180	0.372	0	0	0.301
log_ACCESSES_BE_W1	No fraude	0.821	0.572	0.477	0.845	1.146

Tabla 3.4: Medidas descriptivas de log_ACCESSES_BE, DistanciaNavegador y log_ACCESSES_BE_W1. Se incluye la media, desviación típica y los cuantiles (0.25, 0.50, 0.75) en los casos de fraude y no fraude.

Observamos que la distribución de los datos es muy diferente al comparar ambos casos lo cual da una idea de la importancia que realmente van a tener estas variables a la hora de predecir el fraude. Vemos que estas variables tienen un mayor número de posibles datos atípicos en los casos no fraudulentos. Es de esperar que sea más común que usuarios con un menor número de accesos a la banca electrónica sean más susceptibles a sufrir fraude, lo cual explica que las operaciones asociadas a usuarios con mayor número de accesos, en log_ACCESSES_BE y log_ACCESSES_BE_W1, sean de carácter no fraudulento.

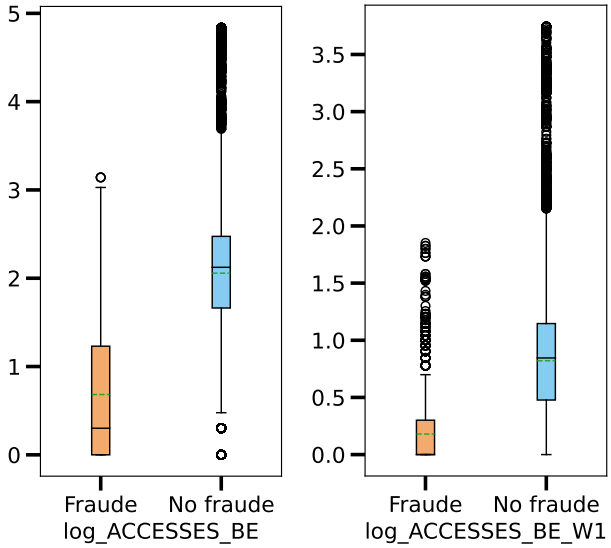


Figura 3.11: Boxplot de las variables numéricas log_ACCESSES_BE y log_ACCESSES_BE_W1. La linea verde horizontal indica la media y la negra la mediana.

Recordamos que en la Figura 3.5 se incluye el boxplot para la variable DistanciaNavegador. Se espera que el navegador usado por un usuario sea casi siempre el mismo. Sin embargo, existen situaciones en los que un usuario puede acceder desde un navegador diferente por diversos motivos, por lo que en el caso de la variable DistanciaNavegador ocurre que son posibles datos atípicos aquellos no fraudulentos con valores bajos para esta variable que se corresponderían con estas situaciones.

Se muestran los diagramas de barras de las variables categóricas 1560_newLanguage y 0410_SCREENDPI en la Figura 3.12. Observamos que existe una mayor proporción de casos fraudulentos con respecto a los no fraudulentos si la variable 1560_newLanguage toma valor 1, que se corresponde a que el idioma del dispositivo es nuevo para ese usuario, tal y como se esperaría. También observamos que para 0410_SCREENDPI es más frecuente que se den casos fraudulentos si los DPI de la pantalla son 32 que cualquier otro de los valores. Al contrario ocurre si los DPI son 24, es menos frecuente que se de fraude.

Finalmente, la lista de variables explicativas queda reducida a tan sólo 58 casos.

3.2.2. Problema con datos faltantes

El conjunto de datos con el que trabajamos contiene datos faltantes en algunas de las variables. La cantidad de datos faltantes depende de la variable, en la Figura 3.13 podemos observar una gráfica con el porcentaje de datos faltantes en aquellas variables que sufren este problema.

Estos datos faltantes se pueden deber a diversos factores. Observamos agrupaciones en el porcentaje de datos faltantes en algunas variables que presentan información de carácter similar, como IN_BE, IN_EALE e IN_BMOVIL, que indican que dichos datos faltantes están relacionados. Además se observa que las observaciones que tienen valor nulo en uno de estos parámetros lo tienen en los otros dos. Esto nos hace suponer que estos datos presentan valores faltantes con una distribución aleatoria o MAR.

Se observa que los porcentajes de datos faltantes en los conjuntos de datos fraudulentos y no fraudulentos el es el mismo en las distintas variables. Por lo tanto el hecho de que una operación sea

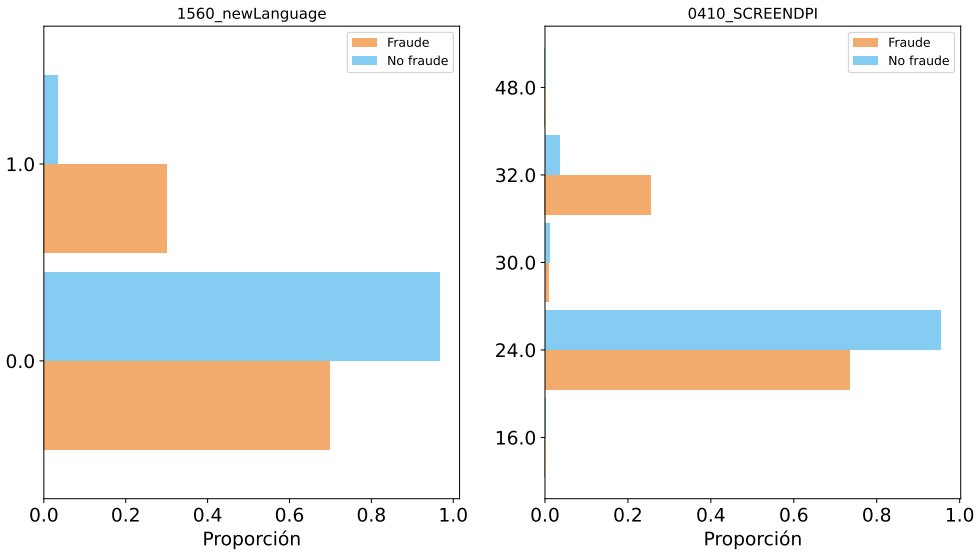


Figura 3.12: Gráfico de barras de las variables numéricas 1560_newLanguage y 0410_SCREENDPI.

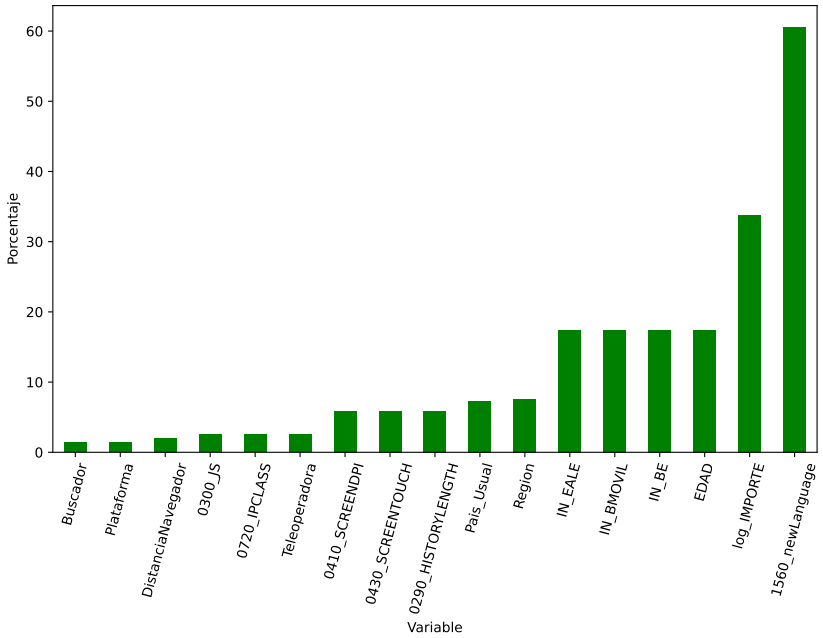


Figura 3.13: Porcentaje de datos faltantes en las variables incompletas.

fraudulenta no tiene aparentemente influencia en si alguna variable va a tener datos faltantes. Esta medida es univariante, puede darse el caso de que esto afecte de manera multivariante.

Algunos de los modelos de clasificación, o su implementación, que se emplearán para tratar el problema no se pueden aplicar para conjuntos con datos faltantes. Recordamos las alternativas que se pueden tomar para afrontar el problema:

- Eliminar las variables con datos faltantes, lo cual no suele ser recomendable.
- Eliminar las observaciones con datos faltantes: *listwise* o *casewise deletion*.
- Imputar los datos.

Como se ha comprobado en la Figura 3.13 hay un gran número de variables con valores faltantes. Algunas de ellas son aparentemente muy influyentes en el fraude y por lo tanto no conviene eliminarlas.

Algunas de las variables poseen una gran cantidad de datos faltantes y si se eliminasen las observaciones faltantes en estos casos se reduciría la muestra considerablemente. Esto conllevaría a una reducción de los casos fraudulentos, lo cual supondría una importante pérdida de información ya que el número de casos de esta clase es bajo como ya se ha comentado anteriormente.

Se opta por tanto por imputar los datos y se utiliza para esta labor la librería MissForest de Python. Nuestra decisión de imputar también viene apoyada por un estudio externo a este trabajo que ha sido realizado en la entidad mostrando que los mejores resultados se obtienen imputando. El algoritmo presentado por dicha librería se basa en bosques aleatorios y su pseudocódigo se presenta en Stekhoven y Bühlmann (2011). Se emplea este método ya que se ha observado que obtiene buenos resultados comparándolo con otros métodos, como imputación por K vecinos más cercanos, como vemos en Penone et al. (2014) y además se expone en Małgorzata (2013) que este ofrece resultados favorables incluso para alto porcentaje de datos faltantes en los casos de datos faltantes MAR.

Capítulo 4

Resultados

En este capítulo se muestran los resultados obtenidos al entrenar y realizar predicciones de los distintos modelos. Se sigue un esquema que comienza con la introducción de los parámetros de cada modelo y la elección realizada. A continuación, se incluyen los resultados de las repeticiones introducidas en la Sección 2.4.1. Finalmente se hace una comparativa entre los modelos para determinar cual resuelve mejor el problema.

4.1. Regresión logística

En esta sección se muestran los resultados obtenidos usando un modelo de regresión logística implementados en el paquete sklearn de Python empleando regularización L2, este método tiene disponibles diferentes tipos de regularización pero este es el más recomendado. En este estudio se emplea la totalidad de las variables para el método de regresión logística.

4.1.1. Resultados de las repeticiones

En las tablas 4.1-4.5 se muestran los valores medios de las diferentes métricas incluidas en el Capítulo 2. Se representan los costes correspondientes en las Figuras 4.1-4.5.

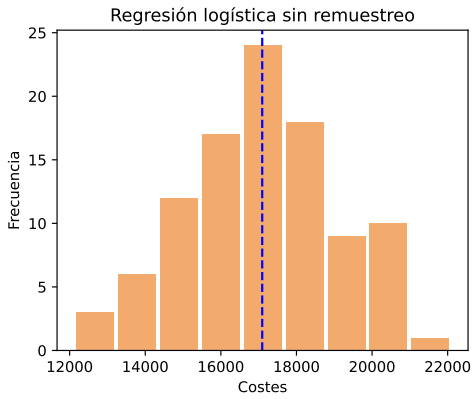


Figura 4.1: Histograma de los valores de costes obtenidos para el modelo de regresión logística sin remuestreo. Se incluye el valor medio mediante la linea azul discontinua.

Medida	**Media**	**Desviación típica**
F1-score	0.65	0.03
Precision	0.80	0.04
Recall	0.55	0.04
Accuracy	0.99	0.01
Balanced accuracy	0.78	0.02
Verdaderos negativos	8 613.98	10.07
Falsos positivos	17.12	3.77
Falsos negativos	55.49	6.54
Verdaderos positivos	68.41	7.23
Coste	17 093.52	1 969.53

Tabla 4.1: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de regresión logística sin remuestreo.

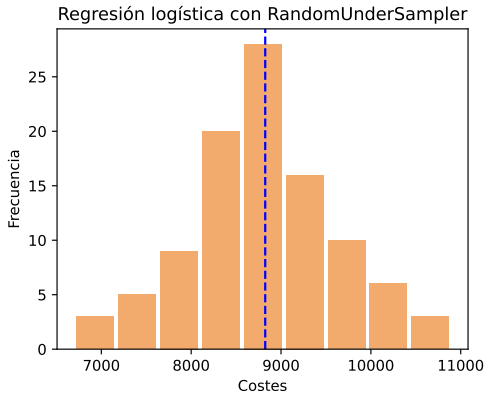


Figura 4.2: Histograma de los valores de costes obtenidos para el modelo de regresión logística con RandomUnderSampler. Se incluye el valor medio mediante la linea azul discontinua.

Medida	Media	Desviación típica
F1-score	0.29	0.02
Precision	0.17	0.01
Recall	0.94	0.02
Accuracy	0.93	0.01
Balanced accuracy	0.94	0.01
Verdaderos negativos	8 065.27	36.08
Falsos positivos	565.84	35.59
Falsos negativos	7.96	2.88
Verdaderos positivos	115.93	9.88
Coste	8 823.63	825.59

Tabla 4.2: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de regresión logística con RandomUnderSampler.

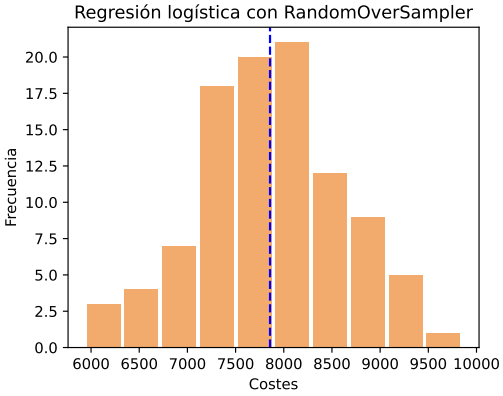


Figura 4.3: Histograma de los valores de costes obtenidos para el modelo de regresión logística con RandomOverSampler. Se incluye el valor medio mediante la linea azul discontinua.

Medida	Media	Desviación típica
F1-score	0.32	0.02
Precision	0.19	0.01
Recall	0.94	0.02
Accuracy	0.94	0.01
Balanced accuracy	0.94	0.01
Verdaderos negativos	8 145.63	22.88
Falsos positivos	485.48	20.48
Falsos negativos	7.63	2.42
Verdaderos positivos	116.26	9.90
Coste	7 857.04	741.23

Tabla 4.3: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de regresión logística con RandomOverSampler.

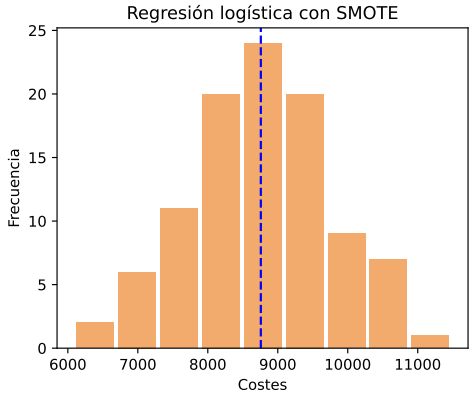


Figura 4.4: Histograma de los valores de costes obtenidos para el modelo de regresión logística con SMOTE. Se incluye el valor medio mediante la linea azul discontinua.

Medida	**Media**	**Desviación típica**
F1-score	0.35	0.02
Precision	0.22	0.02
Recall	0.89	0.03
Accuracy	0.95	0.01
Balanced accuracy	0.92	0.01
Verdaderos negativos	8 243.96	20.56
Falsos positivos	387.15	17.68
Falsos negativos	14.20	3.41
Verdaderos positivos	109.69	9.49
Coste	8 757.42	1 019.41

Tabla 4.4: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de regresión logística con SMOTE.

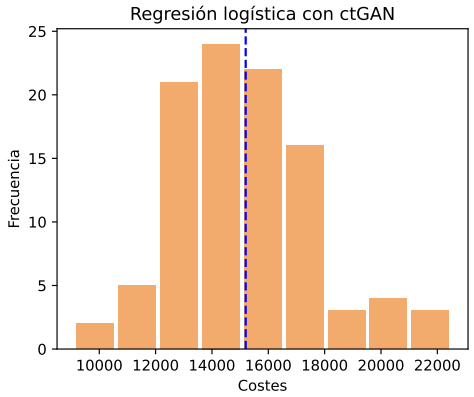


Figura 4.5: Histograma de los valores de costes obtenidos para el modelo de regresión logística con ctGAN. Se incluye el valor medio mediante la linea azul discontinua.

Medida	**Media**	**Desviación típica**
F1-score	0.53	0.06
Precision	0.47	0.08
Recall	0.63	0.06
Accuracy	0.98	0.01
Balanced accuracy	0.81	0.03
Verdaderos negativos	8 539.91	33.30
Falsos positivos	91.20	34.01
Falsos negativos	46.46	8.63
Verdaderos positivos	77.44	9.22
Coste	15 196.47	2 449.54

Tabla 4.5: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de regresión logística con ctGAN.

4.2. CatBoost

4.2.1. Selección de parámetros

Se empleará el modelo CatBoostClassifier la librería catboost de Python. Los parámetros usados, y los valores estudiados, en la búsqueda de rejilla son:

- ***iterations***: Número máximo de árboles que se construyen para resolver el problema. Valor empleado: [1000]
- ***learning_rate***: paso empleado para el descenso por gradiente. Valores empleados: [0.1, 0.2, 0.3, 0.4, 0.5]
- ***max_depth***: profundidad máxima del árbol resultante. Valores empleados: [4, 6, 8, 10]
- ***colsample_bytree***: porcentaje de variables que se toman para realizar el árbol. Valores empleados: [0.5, 0.7, 1]
- ***scale_pos_weight***: multiplicador para los pesos que se le dan a la clase positiva. Valores empleados: [0.1, 1, 10]

Por lo que se estudian 180 combinaciones de parámetros. El mejor modelo se obtiene gracias a la curva ROC empleando los datos de validación y es el que toma valores $learning\_rate = 0.5$, $max\_depth = 6$, $colsample\_bytree = 1$ y $scale\_pos\_weight = 1$. Por lo tanto obtendremos un árbol de máxima profundidad 6 que tendrá en cuenta todas las variables, no aplicará ningún peso extra a las clases positivas y realizará 1000 iteraciones.

4.2.2. Resultados de las repeticiones

Se presentan los resultados de las repeticiones al aplicar las diferentes técnicas de remuestreo para el caso de CatBoost. En las tablas 4.6-4.10 se muestran los valores medios de las diferentes métricas. Se representan los costes correspondientes en las Figuras 4.6-4.10.

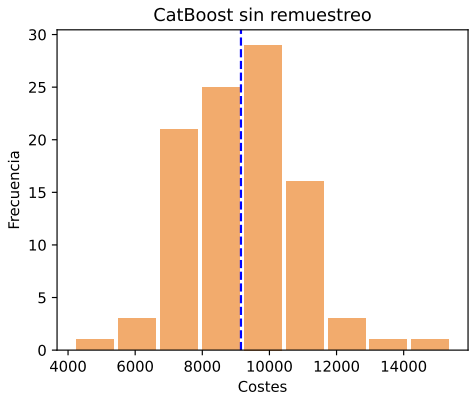


Figura 4.6: Histograma de los valores de costes obtenidos para CatBoost sin remuestreo. Se incluye el valor medio mediante la linea azul discontinua.

Medida	Media	Desviación típica
F1-score	0.83	0.03
Precision	0.91	0.03
Recall	0.76	0.04
Accuracy	0.99	0.01
Balanced accuracy	0.88	0.02
Verdaderos negativos	8 622.03	10.58
Falsos positivos	9.07	3.11
Falsos negativos	29.19	5.63
Verdaderos positivos	94.71	8.78
Coste	9 151.13	1 699.28

Tabla 4.6: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de CatBoost sin remuestreo.

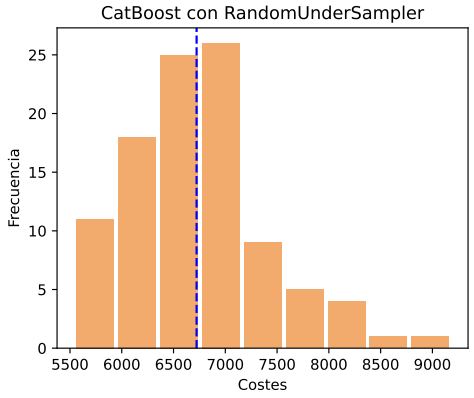


Figura 4.7: Histograma de los valores de costes obtenidos para CatBoost con RandomUnderSampler. Se incluye el valor medio mediante la linea azul discontinua.

Medida	Media	Desviación típica
F1-score	0.34	0.03
Precision	0.20	0.02
Recall	0.96	0.02
Accuracy	0.95	0.01
Balanced accuracy	0.95	0.01
Verdaderos negativos	8 164.12	39.65
Falsos positivos	466.99	38.87
Falsos negativos	4.50	1.98
Verdaderos positivos	119.39	9.60
Coste	6 722.45	675.50

Tabla 4.7: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de CatBoost con RandomUnderSampler.

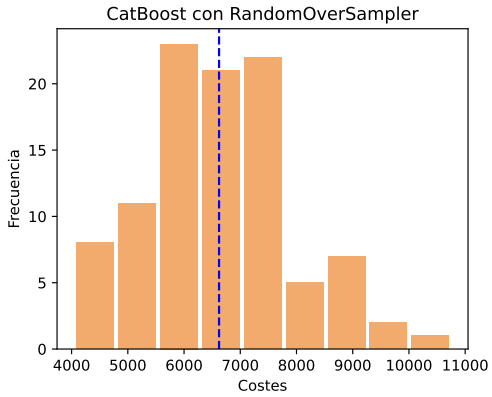


Figura 4.8: Histograma de los valores de costes obtenidos para CatBoost con RandomOverSampler. Se incluye el valor medio mediante la linea azul discontinua.

Medida	Media	Desviación típica
F1-score	0.78	0.03
Precision	0.72	0.04
Recall	0.84	0.03
Accuracy	0.99	0.01
Balanced accuracy	0.92	0.02
Verdaderos negativos	8 590.12	12.36
Falsos positivos	40.98	6.99
Falsos negativos	19.57	4.24
Verdaderos positivos	104.33	8.97
Coste	6 622.40	1 283.90

Tabla 4.8: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de CatBoost con RandomOverSampler.

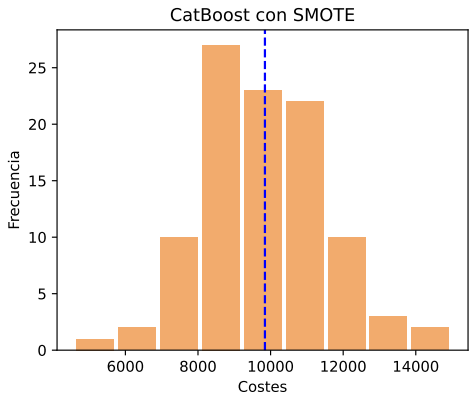


Figura 4.9: Histograma de los valores de costes obtenidos para CatBoost con SMOTE. Se incluye el valor medio mediante la linea azul discontinua.

Medida	**Media**	**Desviación típica**
F1-score	0.76	0.03
Precision	0.76	0.04
Recall	0.75	0.04
Accuracy	0.99	0.01
Balanced accuracy	0.87	0.02
Verdaderos negativos	8 601.88	11.06
Falsos positivos	29.22	5.26
Falsos negativos	30.78	5.96
Verdaderos positivos	93.12	9.01
Coste	9 843.67	1 796.10

Tabla 4.9: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de CatBoost con SMOTE.

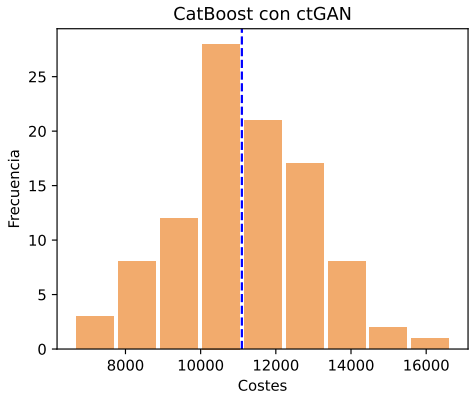


Figura 4.10: Histograma de los valores de costes obtenidos para CatBoost con ctGAN. Se incluye el valor medio mediante la linea azul discontinua.

Medida	**Media**	**Desviación típica**
F1-score	0.79	0.03
Precision	0.88	0.03
Recall	0.71	0.04
Accuracy	0.99	0.01
Balanced accuracy	0.86	0.02
Verdaderos negativos	8 618.96	10.24
Falsos positivos	12.14	3.25
Falsos negativos	35.59	5.92
Verdaderos positivos	88.31	8.78
Coste	11 095.88	1 781.71

Tabla 4.10: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el caso de CatBoost con ctGAN.

4.3. Redes neuronales

4.3.1. Selección de parámetros

Se empleará el modelo KerasClassifier del paquete scikeras de Python. Los parámetros usados en la búsqueda de rejilla y sus valores son:

- ***batch_size***: número de muestras que se toman para realizar el descenso de gradiente. Valores empleados: [100, 200]
- ***drop***: el dropout técnica que consiste en sustuir el input de cada neurona por un valor nulo, se realiza aleatoriamente con probabilidad *drop*. El resto de inputs se escalaran de tal manera que el valor de la suma total de los inputs se mantenga igual. Valores empleados: [0.2, 0.25, 0.3]
- ***n_layers***: número de capas de la red. Valores empleados: [3, 4, 5]
- ***epochs***: número de veces que se actualizan los pesos, es decir el número de veces que se realiza la *backward* y *forward propagation*. Valores empleados: [100]
- ***n_neuronas***: número de neuronas por capa. Valores empleados: [58]

El mejor modelo ha sido obtenido con los datos de validación empleando la curva ROC y es el que toma como valores $batch\_size = 200$, $drop = 0.25$, $n\_layers = 3$, $epochs = 100$ y $n\_neuronas = 58$. Se elige el modelo que mejores resultados obtenga pero considerando los tiempos de ejecución. A mayor número de capas y neuronas, mayor número de parámetros y por tanto será necesario cargar un mayor número de parámetros a la hora de realizar una predicción. Así se escoge una red de 3 capas de 58 neuronas, que es el número de variables explicativas que obtuvimos después del tratamiento de datos.

4.3.2. Resultados de las repeticiones

Se presentan los resultados de las repeticiones al aplicar las diferentes técnicas de remuestreo para una red neuronal. En las tablas 4.11-4.15 se muestran los valores medios de las diferentes métricas incluidas en el capítulo 1. Se representan los costes correspondientes en las Figuras 4.11-4.15.

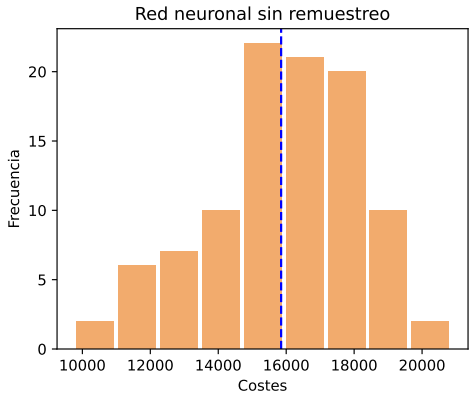


Figura 4.11: Histograma de los valores de costes obtenidos para el modelo de redes neuronales sin remuestreo. Se incluye el valor medio mediante la linea azul discontinua.

Medida	Media	Desviación típica
F1-score	0.68	0.04
Precision	0.82	0.05
Recall	0.59	0.05
Accuracy	0.99	0.01
Balanced accuracy	0.79	0.02
Verdaderos negativos	8 614.91	9.77
Falsos positivos	16.20	4.57
Falsos negativos	51.36	7.54
Verdaderos positivos	72.54	8.05
Coste	15 850.01	2 258.66

Tabla 4.11: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el modelo de redes neuronales sin remuestreo.

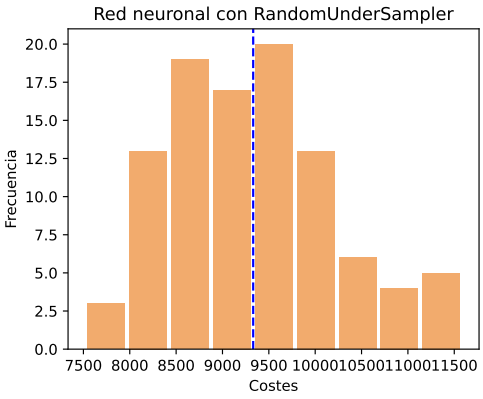


Figura 4.12: Histograma de los valores de costes obtenidos para el modelo de redes neuronales con RandomUnderSampler. Se incluye el valor medio mediante la linea azul discontinua.

Medida	**Media**	**Desviación típica**
F1-score	0.27	0.02
Precision	0.16	0.02
Recall	0.94	0.02
Accuracy	0.93	0.01
Balanced accuracy	0.93	0.01
Verdaderos negativos	8 017.20	60.91
Falsos positivos	613.91	60.09
Falsos negativos	7.92	2.91
Verdaderos positivos	115.97	9.74
Coste	9 330.85	895.56

Tabla 4.12: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el modelo de redes neuronales con RandomUnderSampler.

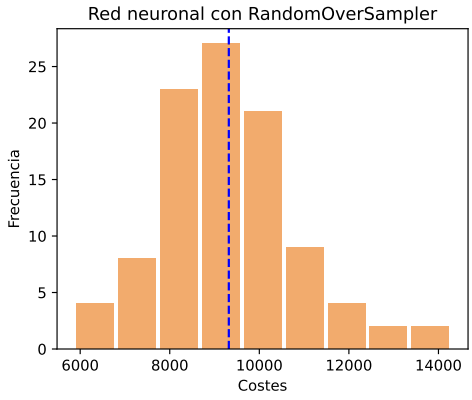


Figura 4.13: Histograma de los valores de costes obtenidos para el modelo de redes neuronales con RandomOverSampler. Se incluye el valor medio mediante la linea azul discontinua.

Medida	**Media**	**Desviación típica**
F1-score	0.67	0.03
Precision	0.58	0.04
Recall	0.78	0.04
Accuracy	0.99	0.01
Balanced accuracy	0.89	0.02
Verdaderos negativos	8 562.29	15.12
Falsos positivos	68.81	10.98
Falsos negativos	27.59	5.26
Verdaderos positivos	96.31	9.34
Coste	9 318.44	1 553.68

Tabla 4.13: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el modelo de redes neuronales con RandomOverSampler.

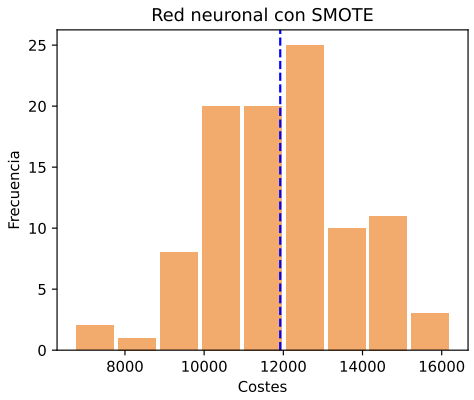


Figura 4.14: Histograma de los valores de costes obtenidos para el modelo de redes neuronales con SMOTE. Se incluye el valor medio mediante la linea azul discontinua.

Medida	Media	Desviación típica
F1-score	0.61	0.03
Precision	0.54	0.04
Recall	0.71	0.04
Accuracy	0.99	0.01
Balanced accuracy	0.85	0.02
Verdaderos negativos	8 555.80	13.71
Falsos positivos	75.30	10.79
Falsos negativos	36.07	6.04
Verdaderos positivos	87.83	8.84
Coste	11 921.39	1 776.96

Tabla 4.14: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el modelo de redes neuronales con SMOTE.

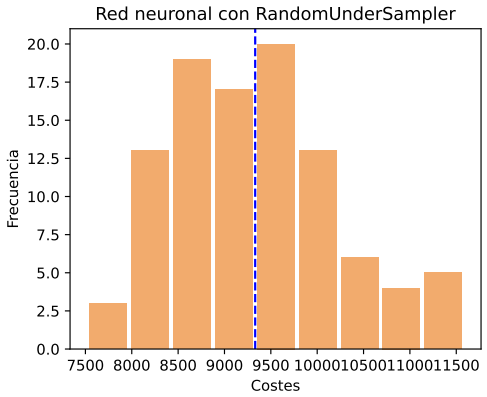


Figura 4.15: Histograma de los valores de costes obtenidos para el modelo de redes neuronales con ctGAN. Se incluye el valor medio mediante la linea azul discontinua.

Medida	Media	Desviación típica
F1-score	0.64	0.04
Precision	0.78	0.05
Recall	0.55	0.05
Accuracy	0.99	0.01
Balanced accuracy	0.77	0.03
Verdaderos negativos	8 611.76	11.19
Falsos positivos	19.34	6.09
Falsos negativos	56.01	7.82
Verdaderos positivos	67.89	8.29
Coste	17 272.82	2 334.85

Tabla 4.15: Valores medios de las distintas medidas obtenidos de las repeticiones realizadas para el modelo de redes neuronales con ctGAN.

4.4. Comparativa de los modelos

Tiempos de ejecución

Se ha calculado el tiempo que tarda cada modelo en realizar una predicción a partir de una nueva observación. Para esto se han de aplicar las modificaciones, la invocación al modelo y efectuar la predicción. Para obtener una medida más fiable se ha calculado la media de tiempos al realizar este procedimiento sobre más de 9000 observaciones. Así se han obtenido los tiempos expuestos en la Tabla 4.16.

Modelo	Tiempo (ms)	Desviación típica (ms)
Regresión logística	3.50	0.18
CatBoost	5.32	0.14
Red neuronal	79.33	17.29

Tabla 4.16: Tiempos, con sus desviaciones típicas, de llamada a los modelos y predicción de una nueva observación.

El modelo más rápido para proporcionar una predicción es regresión logística. El modelo CatBoost, aun siendo casi el doble de lento que regresión logística, también ofrece tiempos de ejecución muy cortos. No ocurre esto con las redes neuronales puesto que la gran cantidad de parámetros ralentiza la predicción y por lo tanto este modelo puede no ser el más adecuado pues ofrece un mayor riesgo de tardar demasiado en proporcionar una respuesta, aunque los tiempos siguen siendo bastante inferiores a un minuto.

Comparaciones gráficas

Analizamos ahora los resultados de las Figuras 4.16-4.18 en las que se muestran los boxplots de los costes, F1-scores, número de falsos negativos y falsos positivos respectivamente. La diferencia que hay entre los valores máximos y mínimos obtenidos para cada medida en cada modelo nos reafirma la necesidad de efectuar diferentes repeticiones para llevar a cabo más correctamente la comparación de los modelos.

Comparando las técnicas de remuestreo, las más complejas, i.e. SMOTE y CTGAN, no proporcionan en media mejores resultados que las más simples o que incluso el caso sin remuestreo para el problema tratado. Esto se debe a que estos generan una mayor cantidad de falsos negativos los cuales están muy penalizados en nuestra función de costes. También generan una menor cantidad de falsos positivos, pero el peso de estos en la función de costes es menor y por lo tanto no compensa la cantidad de falsos positivos. El hecho de que haya un pequeño número de datos de la clase minoritaria en comparación con el número de variables puede influir en la generación de los nuevos datos, llevando a estos métodos de remuestreo a generar peores resultados de los esperados.

Analicemos los resultados para los modelos CatBoost. En la Figura 4.16 vemos que el modelo con menor coste, tanto en valor medio como mediano, es CatBoost remuestreando con RandomOverSampler. Se tiene que CatBoost con RandomUnderSampler también posee un coste similar a este caso. Por lo general ocurre que el modelo CatBoost, con cualquiera de las técnicas de remuestreo, obtiene costes relativamente bajos. Observando los F1-scores vemos que se obtiene el valor más alto, de todos los modelos vistos, para el caso sin remuestreo. También se obtiene valores altos de F1-score, aunque ligeramente inferiores a dicho caso, para el resto de caso de CatBoost salvo con RandomUnderSampler. Para este caso se tiene un gran número de falsos positivos y esto es desfavorable como ya se ha

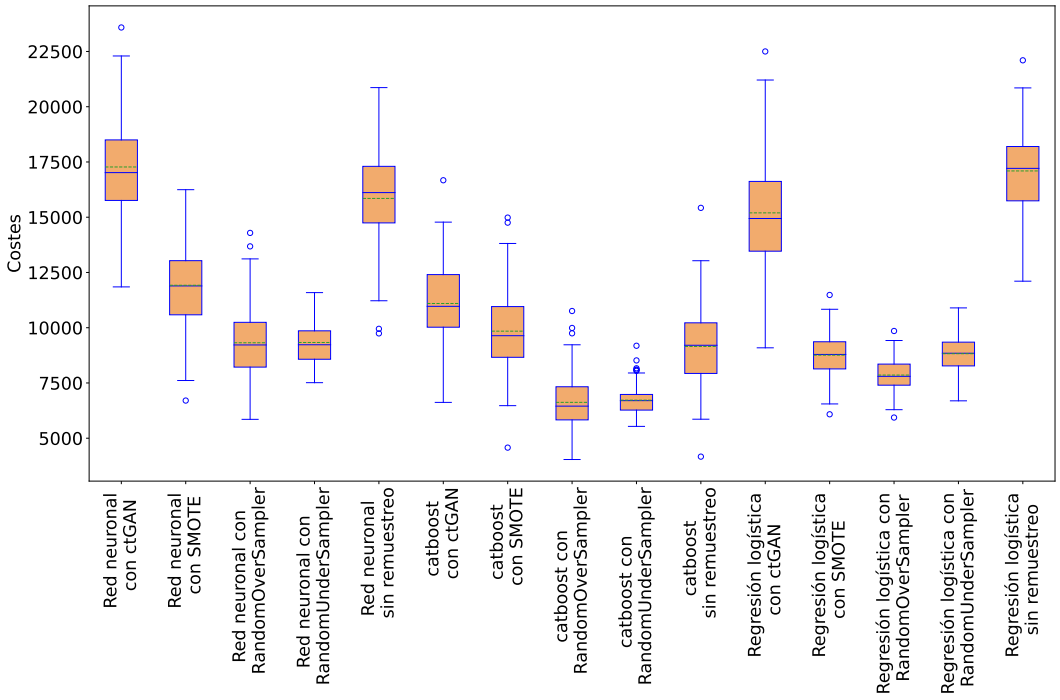


Figura 4.16: Boxplots de los costes obtenidos para las repeticiones de los modelos con las diferentes técnicas de remuestreo.

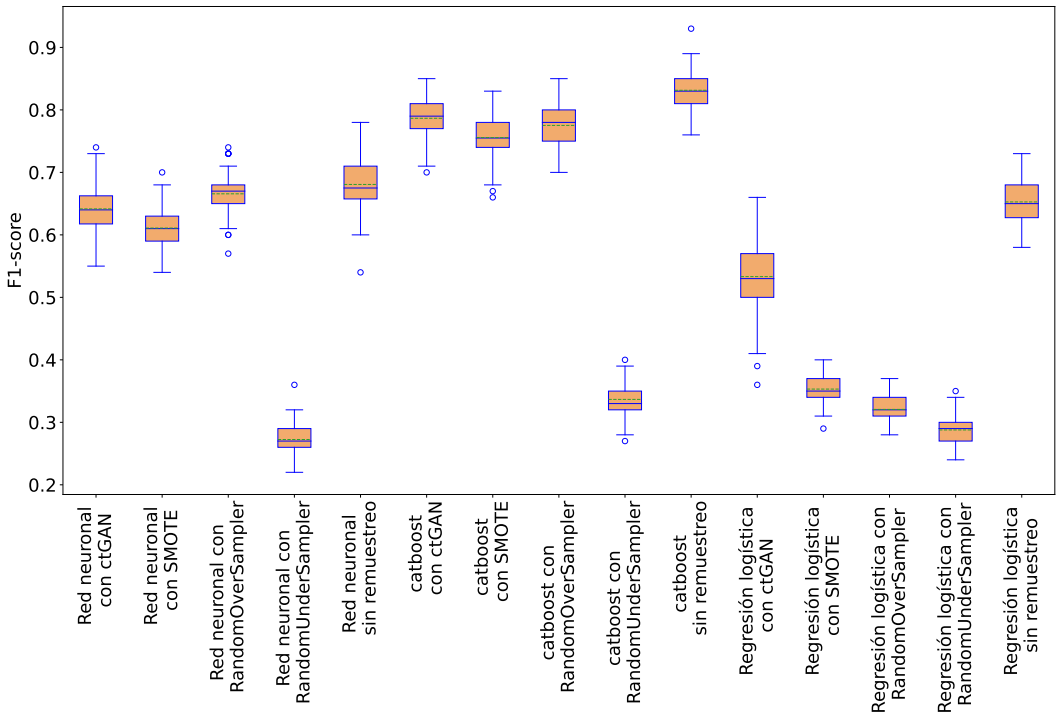


Figura 4.17: Boxplots de los F1-scores obtenidos para las repeticiones de los modelos con las diferentes técnicas de remuestreo.

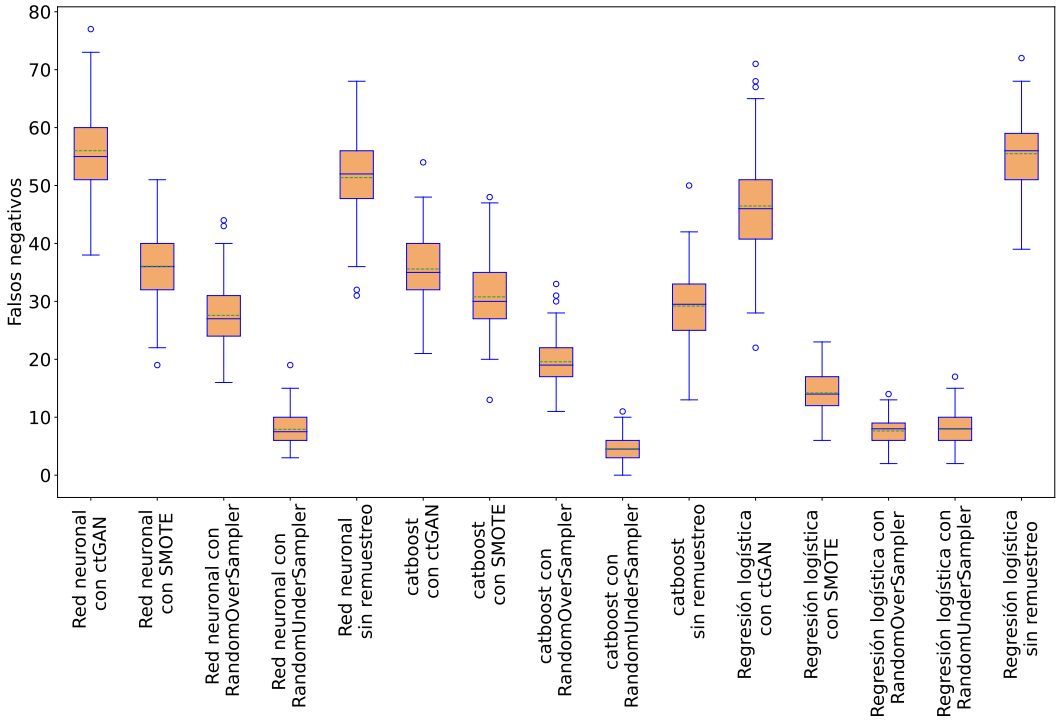


Figura 4.18: Boxplots de los falsos negativos obtenidos en las repeticiones para todos los modelos con las diferentes técnicas de remuestreo.

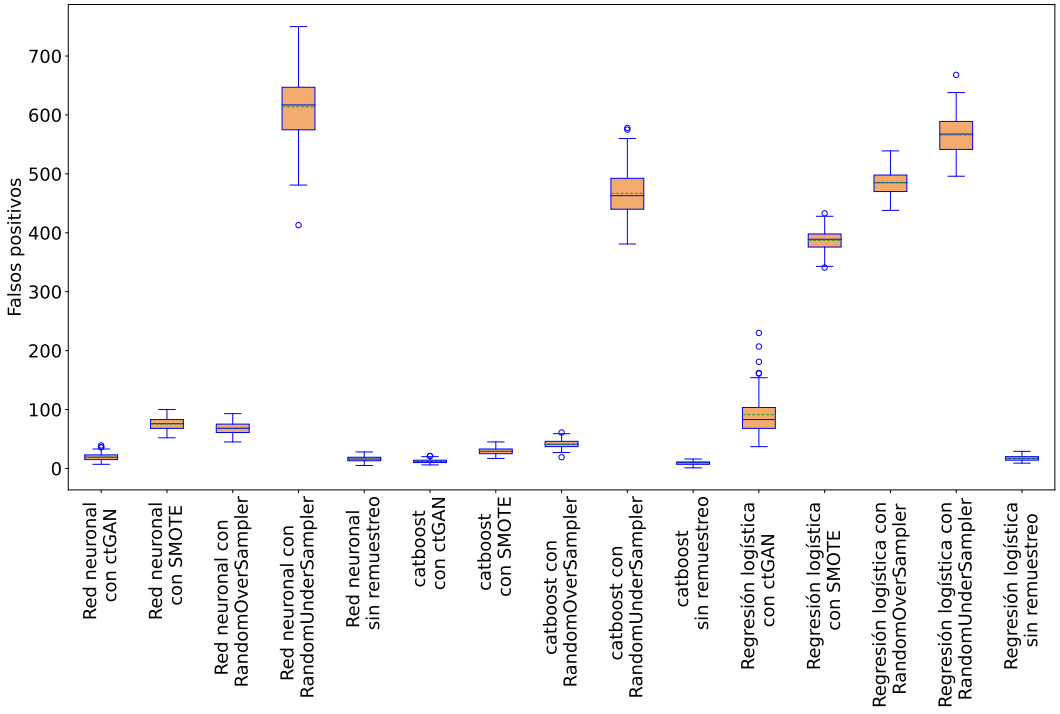


Figura 4.19: Boxplots de los falsos positivos obtenidos en las repeticiones para todos los modelos con las diferentes técnicas de remuestreo.

comentado previamente debido a que ellos conllevan un daño inmensurable en la relación del cliente con la entidad, por ello este modelo sería menos adecuado. El caso sin remuestreo ofrece una menor cantidad de falsos positivos a coste de un aumento de los falsos negativos.

Estudiemos ahora los resultados de los modelos de regresión logística. Los casos remuestreados con SMOTE, RandomOverSampler y RandomUnderSampler poseen bajos costes en media. En estos casos tenemos valores de F1-score muy bajos debido a que estos modelos generan un gran número de falsos positivos lo cual sabemos que es desfavorable. El modelo de regresión logística que mejor medida F1-score produce es el caso sin remuestreo pero este produce una gran cantidad de falsos negativos, esto es que no detecta muchas operaciones fraudulentas lo cual tampoco es deseado.

Las redes neuronales con remuestreo mediante RandomOverSampler y RandomUnderSampler también poseen valores medios de coste bajos, comparados con los otras técnicas de remuestreo. Para este primer caso tenemos una cantidad baja de falsos negativos y una cantidad considerable de falsos positivos, luego el modelo desempeña relativamente bien. Para el segundo caso tenemos una cantidad muy alta de falsos positivos y baja de falsos negativos, por lo que el modelo sobreestima la cantidad de casos fraudulentos. Se obtienen valores de F1-score de entre 0.6 y 0.7 para todos los casos excepto para RandomUnderSampler. Los resultados obtenidos para los modelos de redes neuronales no mejoran los de los modelos CatBoost. Sumando a eso que, como observamos en la sección anterior, los tiempos de ejecución para las redes neuronales son más altos no consideramos estos métodos como los más aptos para su puesta en producción.

Parece que el modelo más adecuado será uno de los CatBoost, concretamente con RandomOverSampler, RandomUnderSampler o sin remuestreo, dependiendo de los objetivos y necesidades.

Capítulo 5

Conclusiones

Presentamos en este capítulo las conclusiones del estudio realizado. Comenzamos con los resultados obtenidos sobre el desempeño de los diferentes modelos y técnicas de remuestreo. Acabamos presentando un análisis del modelo que se decide como el más apropiado para el problema planteado.

5.1. Desempeño de las técnicas de remuestreo y los modelos empleados

En cuanto a las técnicas de remuestreo las conclusiones sacadas dependen del objetivo buscado. Si se busca detectar una mayor cantidad de casos positivos sin importar la cantidad de falsos positivos entonces el método más adecuado según nuestro estudio es RandomUnderSampler. Los modelos de RandomOverSampler, CTGAN y SMOTE proporcionan por lo general un mejor balance entre falsos positivos y falsos negativos. Aunque dentro de estos CTGAN genera mayor cantidad de falsos negativos, seguido de SMOTE y por último RandomOverSampler. Utilizar los modelos sin emplear remuestreo genera sorprendentemente buenos resultados incluso con clases tan desbalanceadas como las tratadas. En este caso la cantidad de falsos positivos es comparable a las tres últimas técnicas mencionadas y la cantidad de falsos negativos es de las menores observadas, por lo tanto proporcionan los mejores resultados de F1-scores.

Con respecto a los modelos, ya se comentó en el capítulo anterior que el que realiza mejores predicciones para este problema es CatBoost, seguido por redes neuronales y por último regresión logística. Aunque cabe comentar que todo esto depende de los criterios empleados. Si a la entidad le interesa aumentar la cantidad de casos fraudulentos detectados nos decantaríamos por un modelo y si el interés sería también tener en cuenta la cantidad de falsos positivos, como es nuestro caso, entonces interesaría emplear otro modelo.

Si existen muchos falsos negativos ocurre que se deja pasar muchas operaciones fraudulentas y por el contrario si hay mucho falso positivo se esta sobre estimando la cantidad de operaciones fraudulentas. Dependiendo del coste que se asocie a cada una de estas situaciones entonces interesa permitir uno u otro caso.

5.2. Mejor modelo para el problema

Como ya se ha comentado, existe una gran complejidad a la hora de realizar un balance entre permitir una mayor cantidad de falsos positivos o de falsos negativos. En este problema se asociaron determinados costes a los falsos positivos y negativos los cuales conllevaron a aceptar como mejor

modelo CatBoost remuestreando con RandomOverSampler. En esta sección analizaremos los resultados asociados a este modelo, determinando la importancia de las variables obtenidas.

5.2.1. Medidas del rendimiento

De acuerdo con lo recogido en el capítulo 1, teniendo en cuenta las medias de las repeticiones realizadas entonces la matriz de confusión con estos valores está expuesta en la Tabla 5.1. Recordamos que en la Tabla 4.8 se muestran los valores medios de las medidas calculadas para las distintas repeticiones de este modelo. Atendiendo a la *precision*, esta indica la proporción de valores que son verdaderamente positivos entre los que el modelo clasifica como positivos y se ha obtenido un valor de 0.72. Esto implica que el 28 % de los casos que predecimos como fraudulentos realmente no lo son. El *recall* indica la proporción de positivos que se clasifican como tal, obtenemos un valor de 0.84 que es un valor alto. Este valor nos informa que somos incapaces de detectar el 16 % de los casos fraudulentos. Como ya se comentó en el capítulo anterior, es necesario que haya cierto balance entre el número de falsos positivos y falsos negativos que ofrece. Este modelo es el que mejor balance proporciona para los pesos asociados a cada uno de estos casos, obteniendo un coste de 6622.40€ para él.

		Predicción	
		0	**1**
Valor	**0**	8 590.12	40.98
real	**1**	19.57	104.33

Tabla 5.1: Matriz de confusión con los valores medios obtenidos en las diferentes muestras de test de las repeticiones del Capítulo 4.

5.2.2. Importancia de las variables en el modelo final

Se muestran las diferentes gráficas realizadas a partir de los valores SHAP en las Figuras 5.1-5.3. Recordamos que los valores SHAP son una medida que da cuenta de la importancia de las variables en la respuesta dada por el modelo.

En la Figura 5.1 se muestra la importancia de las variables medida como la media de los valores absolutos de los valores SHAP para cada variable. Efectuaremos una comparativa de estos valores con los obtenidos en el análisis del capítulo 2 con la correlación de distancia con la variable ETIQUETA, en la Tabla 3.3. Vemos que la variable más influyente es DistanciaNavegador y esta fue la segunda en términos de correlación de distancia. La tercera más influyente es log_ACCESSES_BE la cual tenía mayor valor de correlación de distancia. Estas variables se esperaba que fuesen muy influyentes, por lo tanto no sorprende su posicionamiento en la lista. Sin embargo, la segunda y cuarta más importantes son APLICACION y Region y estas se situaron más abajo en valor de correlación de distancia, la séptima y novena respectivamente, aparentemente contienen información exclusiva que no es compatida con otras variables.

Otras variables que aparecen en las posiciones más altas en ambas listas son Pais_Usual, log_ACCESSES_BMOVIL_3M, 1560_newLanguage y GLOBAL_TRANSACCION_OP_3M_CNT_RATIO. De las 10 de variables más influyentes, Figura 5.1, tan sólo GLOBAL_LOGIN_OP_3M_CNT_RATIO y Teleoperadora no tienen valores altos de correlación de distancia con ETIQUETA. Aunque se observa que los valores de correlación de distancia con ETIQUETA para estas dos variables no se encuentran muy alejados a los valores más bajos de correlación de distancia de los presentados en la Tabla 3.3.

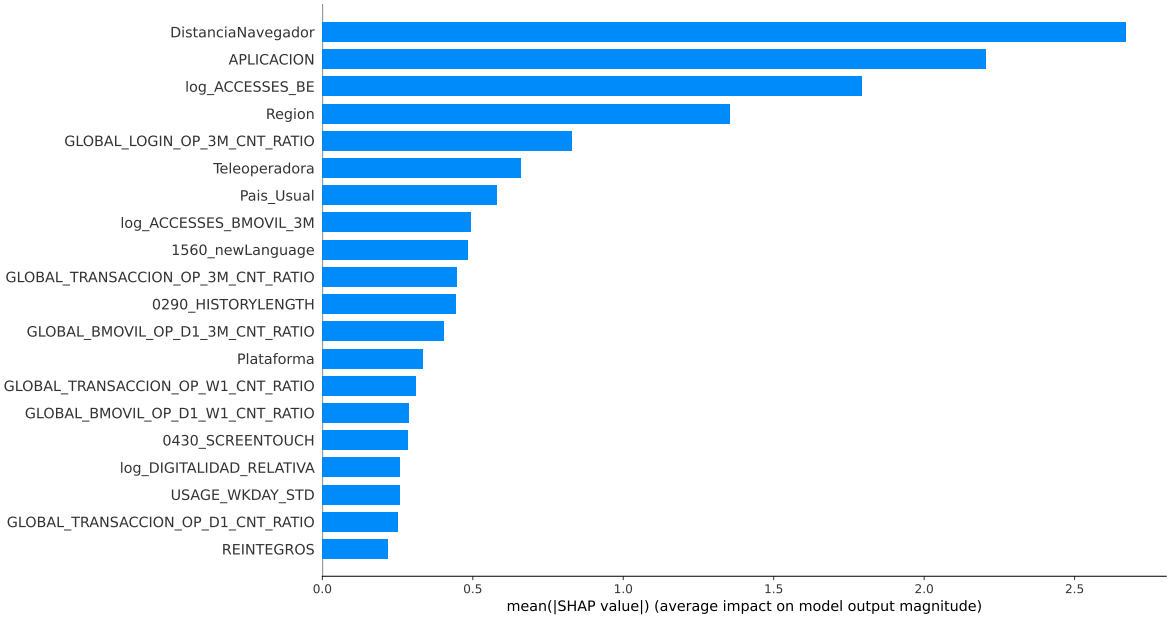


Figura 5.1: Gráfico de barras la media de valores absolutos de los valores SHAP.

La Figura 5.2 muestra los valores SHAP de las observaciones para las variables más influyentes y se colorean estas mediante el valor de la variable mediante la leyenda presente. Esto nos permite observar la diferencia de influencia en la respuesta con el valor que toma cada variable. Valores altos de los valores SHAP indican una mayor probabilidad de que una observación sea fraudulenta. Se incluyen las variables categóricas mediante un código numérico establecido en el apéndice A en cada una de las variables para facilitar su representación.

Se puede observar en la Figura 5.3 el valor SHAP para las dos variables más importantes de cada observación en función del valor de la variable y distinguiendo entre casos fraudulentos y no fraudulentos. Estas gráficas nos permiten estudiar la importancia que tienen los diferentes valores obtenidos para cada variable para determinar una observación como fraudulenta o no. Esto puede servir de ayuda en estudios posteriores para dar mayor peso a ciertas variables o incluso a ciertos valores de ellas que se sabe son muy indicativos de fraude.

Para la variable DistanciaNavegador, Figura 5.3 (izda), valores altos implican un menor valor SHAP como era de esperar por como se construye la variable. Valores bajos de esta variable proporcionan valores SHAP muy bajos y por lo tanto son muy influyentes para posiblemente clasificar dichas observaciones como no fraudulentas.

En cuanto a la variable APLICACION se observa que el valor 0 es el único que produce valores de SHAP positivos. Las operaciones fraudulentas producen los mayores valores de SHAP como se observa en la Figura 5.3 (dcha). Por lo tanto vemos que los valores distintos a 0 tienden a una clasificación de las observaciones como no fraudulentas incluso en casos realmente positivos.

Para el resto de variables vemos existe una concentración de valores con valor SHAP cercano a 0, con colas hacia ambos lados en diferentes formas. Son estos valores de las colas los que más influirán en las predicciones del modelo. También se observa que los valores fraudulentos por lo general posee un mayor valor de SHAP que los no fraudulentos, lo cual es una muestra más de la utilidad del modelo.

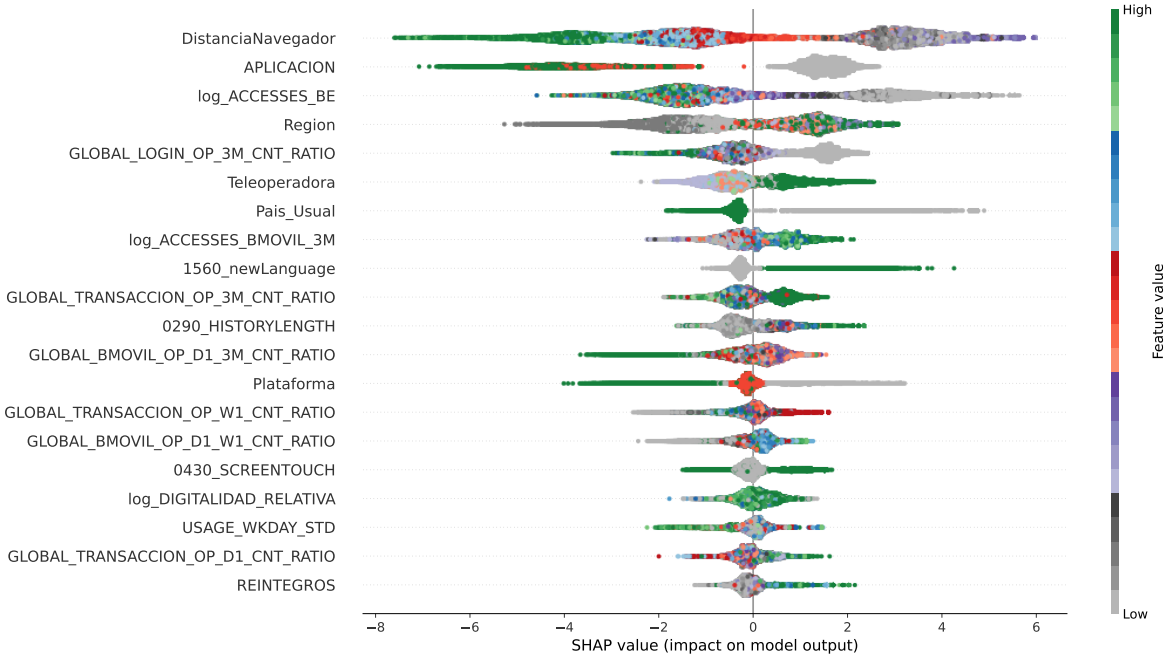


Figura 5.2: Gráfico del valor SHAP de las observaciones para las variables más importantes.

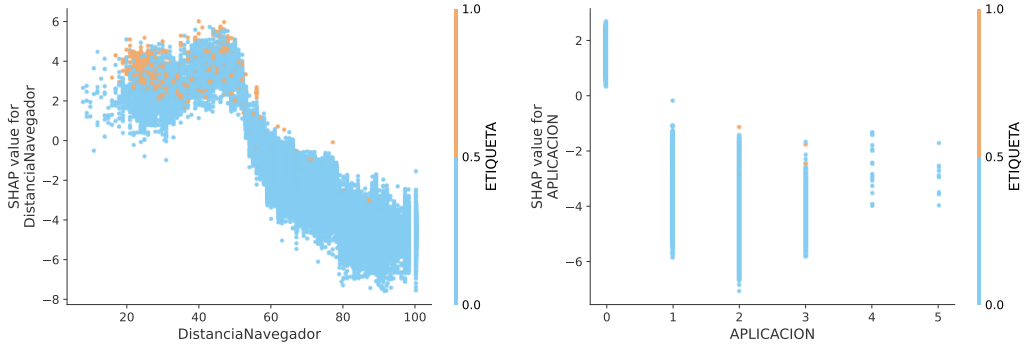


Figura 5.3: Gráficos de los valores SHAP frente a los valores tomados por las variables para DistanciaNavegador (izda) y APLICACION (dcha).

Cabe destacar que las variables que fueron creadas para el problema parecen ser de utilidad, pues cuatro de ellas se encuentran entre las diez variables más influyentes para el modelo. Además, existe un equilibrio entre la importancia de las variables que van asociadas al comportamiento de clientes y las asociadas propiamente al dispositivo desde las que se realiza.

Al elegir un modelo basado en árboles de decisión no es necesario imputar los datos antes de realizar la predicción, por lo tanto se acorta el tiempo de ejecución lo cual es un resultado muy positivo.

5.3. Comentarios finales y futuras líneas de investigación

Como resumen final concluimos que se han cumplido todos los objetivos planteados. El mejor modelo para clasificar operaciones fraudulentas en los datos tratados corresponde con un CatBoost remuestreando los datos con RandomOverSampler.
Se ha obtenido un modelo capaz de predecir el 84 % de los casos fraudulentos en media. Aunque el 28 % de los casos que se clasifican como fraudulentos realmente no lo son. El tiempo de predicción es corto, correspondiendo con lo buscado por la entidad para la que se realiza el trabajo.

Los comportamientos de los usuarios cambian constantemente y es posible que con el paso del tiempo la influencia del fraude con las diferentes variables cambie. Es plausible pensar que se puede obtener una mayor cantidad de información, o variables, por las entidades. Por estos motivos es necesario volver a medir el desempeño de los modelos al enfrentarse a estas nuevos datos e incluso reentrenar los modelos con la nueva información. Los datos correspondientes a operaciones bancarias fraudulentas siempre será un número mucho menor que el de las no fraudulentas, por lo tanto el problema de tratar clases desbalanceadas también se presentará. Por todo esto será necesario revisitar este tipo de problemas con regularidad y asegurarse de que todo funciona correctamente, probando nuevas funciones de coste con pesos actualizados e incluso nuevos modelos, como por ejemplo regresión logística con GAMs, o técnicas novedosas para intentar mejorar la eficacia en la predicción. Se pueden estudiar también métodos bagging para combinar, por ejemplo, 100 modelos y calcular el promedio para modelo final lo cual ayudaría a reducir la variabilidad.

Apéndice A

Variables

En este apéndice se muestra la lista de las variables tratadas y una breve descripción de cada una.

- **FECHA:** fecha y hora en la que se realizó la operación.
- **ETIQUETA:** indicador si la operación es fraudulenta o no.
- **ENTIDAD:** entidad de la operación.
- **OPERACION_CECA:** valor de la operación para la Confederación Española Cajas de Ahorros (CECA).
- **APLICACION:** la aplicación del banco utilizada para realizar la operación. Valores: WELE200:0, BEPRJ001:2, NBEPRJ001:1, BEPRJ0UO:3, PISP_BEP:4, PISP_BEE:5.
- **APLICACION_ORIGEN:**
- **USERPUID:** ID del usuario.
- **RESULTADO:** indica si la operación se realiza o no.
- **CANAL:** medio utilizado para realizar la operación.
- **NAVEGADOR:** nombre técnico del navegador utilizado.
- **OPERACION:** tipo de operación realizada. Valores: LOGIN:0, TRANSACCION:1, SESION:2.
- **ID_PAIS_DESTINO:** ID asociada al país de destino.
- **ID_BANCO_DESTINO:** ID asociada al banco de destino.
- **IMPORTE:** cantidad de dinero usada en una transacción.
- **DIVISA:** divisa en la que se realiza la transacción.
- **STATUS:** indica si la operación se puede realizar o no.
- **TXN_W1_TXN_AMT_MEAN:** media de los importes de las transacciones de la última semana.
- **TXN_M1_TXN_AMT_MEAN:** media de los importes de las transacciones del último mes.
- **TXM_3M_TXN_AMT_MEAN:** media de los importes de las transacciones de los últimos tres meses.

- **TXN_M1_TXN_AMT_STD:** desviación típica de los importes de las transacciones del último mes.
- **TXM_3M_TXN_AMT_STD:** desviación típica de los importes de las transacciones de los últimos tres meses.
- **TXN_W1_TXN_MAX_AMT:** máximo de los importes de las transacciones de la última semana.
- **TXN_M1_TXN_MAX_AMT:** máximo de los importes de las transacciones del último mes.
- **TXM_3M_TXN_MAX_AMT:** máximo de los importes de las transacciones de los últimos tres meses.
- **ACCESSES_BMOVIL_W1:** accesos en la última semana a la banca móvil.
- **ACCESSES_BMOVIL_W1_ZS:** accesos en la última semana a la banca móvil normalizado.
- **ACCESSES_BMOVIL_M1:** accesos en el último mes a la banca móvil.
- **ACCESSES_BMOVIL_M1_ZS:** accesos en el último mes a la banca móvil normalizado.
- **ACCESSES_BMOVIL_3M:** accesos en los últimos tres meses a la banca móvil.
- **ACCESSES_BMOVIL_3M_ZS:** accesos en los últimos tres meses a la banca móvil normalizado.
- **ACCESSES_BMOVIL:** accesos a la banca móvil.
- **ACCESSES_BMOVIL_ZS:** accesos a la banca móvil normalizado.
- **ACCESSES_BE_W1:** accesos en la última semana a la banca electrónica.
- **ACCESSES_BE_W1_ZS:** accesos en la última semana a la banca electrónica normalizado.
- **ACCESSES_BE_M1:** accesos en el último mes a la banca electrónica.
- **ACCESSES_BE_M1_ZS:** accesos en el último mes a la banca electrónica normalizado.
- **ACCESSES_BE_3M:** accesos en los últimos tres meses a la banca electrónica.
- **ACCESSES_BE_3M_ZS:** accesos en los últimos tres meses a la banca electrónica normalizado.
- **ACCESSES_BE:** accesos a la banca electrónica.
- **ACCESSES_BE_ZS:** accesos a la banca electrónica normalizado.
- **OP_LOGIN_W1_CNT_RATIO:** proporción de logins en la última semana.
- **OP_SESION_W1_CNT_RATIO:** proporción de sesiones en la última semana.
- **OP_TRANSACTION_W1_CNT_RATIO:** proporción de transacciones en la última semana.
- **OP_LOGIN_M1_CNT_RATIO:** proporción de logins en el ultimo mes.
- **OP_SESION_M1_CNT_RATIO:** proporción de sesiones en el ultimo mes.
- **OP_TRANSACTION_M1_CNT_RATIO:** proporción de transacciones en el ultimo mes.
- **OP_LOGIN_3M_CNT_RATIO:** proporción de logins en los tres últimos meses.
- **OP_SESION_3M_CNT_RATIO:** proporción de sesiones en los tres últimos meses.

- **OP_TRANSACTION_3M_CNT_RATIO:** proporción de transacciones en los tres últimos meses.
- **GLOBAL_BMOVIL_OP_D1_W1_CNT_RATIO:** proporción de operaciones del usuario en banca móvil en el último día por última semana.
- **GLOBAL_BMOVIL_OP_D1_M1_CNT_RATIO:** proporción de operaciones del usuario en banca móvil en el último día por último mes.
- **GLOBAL_BMOVIL_OP_D1_3M_CNT_RATIO:** proporción de operaciones del usuario en banca móvil en el último día por últimos tres meses.
- **GLOBAL_BE_OP_D1_W1_CNT_RATIO:** proporción de operaciones del usuario en banca electrónica en el último día por última semana.
- **GLOBAL_BE_OP_D1_M1_CNT_RATIO:** proporción de operaciones del usuario en banca electrónica en el último día por último mes.
- **GLOBAL_BE_OP_D1_3M_CNT_RATIO:** proporción de operaciones del usuario en banca electrónica en el último día por últimos tres meses.
- **GLOBAL_LOGIN_OP_D1_CNT_RATIO:** proporción de logins realizadas por el usuario en el último día.
- **GLOBAL_TRANSACCION_OP_D1_CNT_RATIO:** proporción de transacciones realizadas por el usuario en el último día.
- **GLOBAL_LOGIN_OP_W1_CNT_RATIO:** proporción de logins realizadas por el usuario en la última semana.
- **GLOBAL_TRANSACCION_OP_W1_CNT_RATIO:** proporción de transacciones realizadas por el usuario en la última semana.
- **GLOBAL_LOGIN_OP_M1_CNT_RATIO:** proporción de logins realizadas por el usuario en el último mes.
- **GLOBAL_TRANSACCION_OP_M1_CNT_RATIO:** proporción de transacciones realizadas por el usuario en el último mes.
- **GLOBAL_LOGIN_OP_3M_CNT_RATIO:** proporción de logins realizadas por el usuario en los últimos tres meses.
- **GLOBAL_TRANSACCION_OP_3M_CNT_RATIO:** proporción de transacciones realizadas por el usuario en los últimos tres meses.
- **USAGE_HOURS_WK_MEAN:** media de horas de uso por semana.
- **USAGE_HOURS_WKND_MEAN:** media de horas de uso por fin de semana.
- **USAGE_HOURS_WK_STD:** desviación típica de horas de uso por semana.
- **USAGE_HOURS_WKND_STD:** desviación típica de horas de uso por fin de semana.
- **USAGE_WKDAY_MEAN:** media de horas de uso por semana laborable.
- **USAGE_WKDAY_STD:** desviación típica de horas de uso por semana laborable.
- **USAGE_MNTHDAY_MEAN:** media de las horas de uso por día en el último mes.
- **USAGE_MNTHDAY_STD:** desviación típica de las horas de uso por día en el último mes.

- **COUNTRIES_W1_CNT:** cantidad de países a los que se ha realizado una operación en la última semana.
- **COUNTRIES_M1_CNT:** cantidad de países a los que se ha realizado una operación en el último mes.
- **COUNTRIES_3M_CNT:** cantidad de países a los que se ha realizado una operación en los tres últimos meses.
- **COUNTRIES_SWITCH_30_MIN_W1_CNT:** cambios de país en 30 minutos en la última semana.
- **COUNTRIES_SWITCH_30_MIN_M1_CNT:** cambios de país en 30 minutos en el último mes.
- **SAME_IP_ACROSS_CANAL_W1_CNT:** número de IPs usadas por el usuario en la última semana.
- **SAME_IP_ACROSS_CANAL_M1_CNT:** número de IPs usadas por el usuario en el último mes.
- **COUNTRY_LIST:** lista de paises a los que se ha realizado alguna transacción.
- **NAVEGADOR_LIST:** lista de navegadores usados por el usuario.
- **IN_BE:** indicador de presencia en banca electrónica.
- **IN_BMOVIL:** indicador de presencia en banca móvil.
- **IN_EALE:** indicador de activación de alertas móviles.
- **USO_TARJETAS_12M:** proporción de uso de la tarjeta.
- **PROPORCION_ELECTRONICA:** proporción electrónica.
- **PROPORCION_TARJETAS:** proporción tarjetas.
- **MENSAJERIA:** indica que servicios de mensajes tiene activado el usuario.
- **DIGITALIDAD_RELATIVA:** coeficiente de digitalidad.
- **EDAD:** edad del usuario.
- **DIGITALIDAD_RELATIVA_EDAD:** coeficiente de digitalidad teniendo en cuenta su edad.
- **IN_DIGITAL:** indicador de la digitalidad del usuario.
- **REINTEGROS:** cantidad de ingresos del usuario.
- **0190_BROWSER:** nombre comercial del navegador usado.
- **0200_BROWSERVERSION:** versión del buscador.
- **0230_CLIENTTIMEZONE:** hora del cliente.
- **0290_HISTORYLENGTH:** números de URLs en el historial.
- **0300_JS:** indicador de activación del JavaScript.
- **0340_USERAGENT:** ID de agente del usuario.

- **0390_OS:** sistema operativo del dispositivo.
- **0400_PLATFORM:** plataforma utilizada para la operación.
- **0410_SCREENDPI:** DPI de la pantalla del dispositivo.
- **0420_SCREENHEIGHT:** altura de la pantalla.
- **0430_SCREENTOUCH:** indicador de si la pantalla es táctil.
- **0440_SCREENWIDTH:** anchura de la pantalla.
- **0450_BATTERYCHARGING:** indicador si el dispositivo está siendo cargado.
- **0460_CALCSTATE:** indicador de si la calculación de elementos de riesgos ha finalizado.
- **0470_CPUTYPE:** modelo de CPU del dispositivo.
- **0480_DATACARRIER:** indicador de si el dispositivo está conectado a una red móvil.
- **0490_DEVICELANGUAGE:** idioma del dispositivo.
- **0500_DEVICEROOTED:** indicador de si el dispositivo ha sido *rooteado*.
- **0510_DEVICETYPE:** nombre técnico de dispositivo.
- **0520_EMULATOR:** indicador de si se está usando un emulador en el dispositivo.
- **0530_ISCALLINPROGRESS:** indicador de si se está realizando una llamada mientras se realiza la operación.
- **0550_LINECARRIER:** nombre de la teleoperadora.
- **0570_MRSTAPPCOUNT:** indica el número de herramientas de control remoto que hay instaladas en el dispositivo.
- **0590_ROOTHIDERS:** indica si hay *roothiders*.
- **0630_SMSSTEALERS:** indica si hay ladrones de SMS en el dispositivo.
- **0640_TIMEZONE:** huso horario del dispositivo.
- **0670_WIFISECURITYSCORE:** valor entre 0 y 1000 que indica si la conexión wifi utilizada es (valores altos) o no segura (valores bajos).
- **0690_CITY:** ciudad desde la que se realiza la operación.
- **0700_CONTINENTCODE:** código del continente desde el que se realiza la operación.
- **0710_COUNTRY:** país desde el que se realiza la operación.
- **0720_IPCLASS:** clase de IP. Valores: A:0, B:2, C:1.
- **0730_IPTIMEZONE:** huso horario de la IP de conexión a internet.
- **0740_ISP:** agente que proporciona conexión a internet.
- **0750_LATITUDENETWORK:** latitud de la conexión.
- **0760_LONGITUDENETWORK:** longitud de la conexión.
- **0780_REGION:** región desde donde se realiza la operación.

- **1490_batteryPercentage:** porcentaje de batería del dispositivo.
- **1500_cameraMegaPixelSize:** megapíxeles de la cámara del dispositivo.
- **1510_developerSettings:** indica si están activadas los ajustes de desarrollador.
- **1520_numberOfInstalledApplications:** número de aplicaciones instaladas.
- **1530_unlockType:** tipo de desbloqueo del dispositivo.
- **1560_newLanguage:** indicador de si el idioma es nuevo.
- **Buscador:** buscador de internet utilizado para realizar la operación. Valores: Chrome:0, Firefox:1, Otro:2, Edge:3, Safari:4.
- **Plataforma:** plataforma utilizada por el usuario. Valores: Linux:0, Windows:1, Otro:2, Safari:3
- **Region:** región desde la que se realiza la operación (comunidad autónoma si es desde España o continente si es desde fuera de España). Valores: PV:0, MD:1, GA:2, VC:3, AN:4, Europe:5, CL:6, AS:7, CM:8, Asia:9, Africa:10, CT:11, CB:12, EX:13, Oceania:14, CN:15, IB:16, South America:17, AR:18, NC:19, MC:20, RI:21, CE:22.
- **Teleoperadora:** teleoperadora que ofrece el acceso a internet. Valores: Yoigo:0, Telefonica:1, Vodafone:2, R Cable:3, Orange:4, Otro:5
- **DistanciaNavegador:** distancia del navegador usado a los navegadores usuales del usuario.
- **Pais_Usual:** indicador de si el país desde donde se realiza la operación se encuentra en la lista de países usuales del usuario.

Apéndice B

Correlaciones de Pearson entre las variables

Se incluyen la Tabla B.1 que incluye los pares de variables continuas en los que las correlaciones de Pearson entre ellas toman valores superiores a 0.9 o inferiores a -0.9.

Variable 1	Variable 2	Correlación
GLOBAL_BE_OP_D1_3M_CNT_RATIO	GLOBAL_BE_OP_D1_M1_CNT_RATIO	0.985
GLOBAL_BE_OP_D1_3M_CNT_RATIO	GLOBAL_BE_OP_D1_W1_CNT_RATIO	0.931
GLOBAL_BE_OP_D1_3M_CNT_RATIO	GLOBAL_BMOVIL_OP_D1_3M_CNT_RATIO	0.917
GLOBAL_BE_OP_D1_M1_CNT_RATIO	GLOBAL_BE_OP_D1_W1_CNT_RATIO	0.934
GLOBAL_BE_OP_D1_M1_CNT_RATIO	GLOBAL_BMOVIL_OP_D1_3M_CNT_RATIO	0.910
GLOBAL_BE_OP_D1_M1_CNT_RATIO	GLOBAL_BMOVIL_OP_D1_M1_CNT_RATIO	0.921
GLOBAL_BE_OP_D1_W1_CNT_RATIO	GLOBAL_BMOVIL_OP_D1_W1_CNT_RATIO	0.941
GLOBAL_BMOVIL_OP_D1_3M_CNT_RATIO	GLOBAL_BMOVIL_OP_D1_M1_CNT_RATIO	0.978
GLOBAL_LOGIN_OP_3M_CNT_RATIO	GLOBAL_LOGIN_OP_D1_CNT_RATIO	0.985
GLOBAL_LOGIN_OP_3M_CNT_RATIO	GLOBAL_LOGIN_OP_M1_CNT_RATIO	0.999
GLOBAL_LOGIN_OP_3M_CNT_RATIO	GLOBAL_LOGIN_OP_W1_CNT_RATIO	0.992
GLOBAL_LOGIN_OP_D1_CNT_RATIO	GLOBAL_LOGIN_OP_M1_CNT_RATIO	0.985
GLOBAL_LOGIN_OP_D1_CNT_RATIO	GLOBAL_LOGIN_OP_W1_CNT_RATIO	0.989
GLOBAL_LOGIN_OP_M1_CNT_RATIO	GLOBAL_LOGIN_OP_W1_CNT_RATIO	0.993
GLOBAL_TRANSACCION_OP_3M_CNT_RATIO	GLOBAL_TRANSACCION_OP_M1_CNT_RATIO	0.979

continúa

Variable 1	Variable 2	Correlación
OP_LOGIN_3M_CNT_RATIO	OP_TRANSACTION_3M_CNT_RATIO	-0.917
OP_TRANSACTION_3M_CNT_RATIO	OP_TRANSACTION_M1_CNT_RATIO	0.941
log_ACCESSES_BE	log_ACCESSES_BE_3M	0.985
log_ACCESSES_BE	log_ACCESSES_BE_M1	0.944
log_ACCESSES_BE_3M	log_ACCESSES_BE_M1	0.968
log_ACCESSES_BE_M1	log_ACCESSES_BE_W1	0.928
log_ACCESSES_BMOVIL	log_ACCESSES_BMOVIL_3M	0.988
log_ACCESSES_BMOVIL	log_ACCESSES_BMOVIL_M1	0.953
log_ACCESSES_BMOVIL_3M	log_ACCESSES_BMOVIL_M1	0.974
log_ACCESSES_BMOVIL_M1	log_ACCESSES_BMOVIL_W1	0.937
log_DIGITALIDAD_RELATIVA	log_DIGITALIDAD_RELATIVA_EDAD	0.994
log_TXM_3M_TXN_AMT_MEAN	log_TXM_3M_TXN_MAX_AMT	0.981
log_TXM_3M_TXN_AMT_STD	log_TXM_3M_TXN_MAX_AMT	0.929
log_TXN_M1_TXN_AMT_MEAN	log_TXN_M1_TXN_MAX_AMT	0.988
log_TXN_W1_TXN_AMT_MEAN	log_TXN_W1_TXN_MAX_AMT	0.995

Tabla B.1: Pares de variable en los que la correlación de Pearson toma valores más altos en valor absoluto.

Bibliografía

Allison, P. (2001). *Missing Data*. SAGE Publications.

Bergsma, W. (2013). A bias-correction for Cramér's V and Tschuprow's T. *Journal of the Korean Statistical Society*, *42*(3), 323-328.

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, W. (2002). SMOTE: Synthetic Minority Oversampling Technique. *J. Artif. Intell. Res. (JAIR)*, *16*, 321-357.

Cramér, H. (1946). En *Mathematical Methods of Statistics*. Princeton University Press.

Fernández, R., Costa, J., & Oviedo, M. (2021). *Aprendizaje estadístico*. https://rubenfcasal.github.io/aprendizaje_estadistico

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, *29*(5), 1189-1232.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, *27*.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning*. Springer.

Kearns, M., & Valiant, L. (1994). Cryptographic Limitations on Learning Boolean Formulae and Finite Automata. *Journal of the Association for Computing Machinery*, *41*(1), 67-95.

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., & Lee, S.-I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, *2*(1).

Małgorzata, M. (2013). Some remarks on the data imputation using "MissForest"method. *Acta Univertitatis Lodziensis. Folia oeconomica*, *285*, 169-179.

Molnar, C. (2022). *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable* (2.ª ed.). https://christophm.github.io/interpretable-ml-book

Müller, A., & Guido, S. (2016). *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media.

Nightingale, S. J., & Farid, H. (2022). AI-synthesized faces are indistinguishable from realfaces and more trustworthy. *Proceedings of the National Academy of Sciences*, *119*(8).

Pearson, K. (1896). Mathematical Contributions to the Theory of Evolution. III. Regression, Heredity, and Panmixia. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, *187*, 253-318.

Penone, C., Davidson, A. D., Shoemaker, K. T., Di Marco, M., Rondinini, C., Brooks, T. M., Young, B. E., Graham, C. H., & Costa, G. C. (2014). Imputation of missing data in life-history trait datasets: which approach performs the best? *Methods in Ecology and Evolution*, *5*(9), 961-970.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2019). CatBoost: unbiased boosting with categorical features.

Refaeilzadeh, P., Tang, L., & Liu, H. (2016). Cross-Validation. En *Encyclopedia of Database Systems* (pp. 1-7). Springer New York.

Shapley, L. S. (1953). Contributions to the Theory of Games II. Princeton University Press.

Spearman, C. (1904). The Proof and Measurement of Association between Two Things. *The American Journal of Psychology*, *15*(1), 72-101.

Stekhoven, D. J., & Bühlmann, P. (2011). MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, *28*(1), 112-118.

Székely, G. J., Rizzo, M. L., & Bakirov, N. K. (2007). Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, *35*(6), 2769-2794.

Xu, L., Skoularidou, M., Cuesta-Infante, A., & Veeramachaneni, K. (2019). Modeling Tabular Data Using Conditional GAN. En *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc.