# Predicción de compra en el entorno ecommerce:

Comparación de modelos, variables y procedimientos en conjuntos de datos desbalanceados

Victor Ayala Sánchez

**Analítica Empresarial**

**Tutor/a de TF**
Santiago Rojo Muñoz

**Profesor/a responsable de la asignatura**
Albert Solé Ribalta

Fecha Entrega:
15/01/2023

Universitat Oberta
de Catalunya

# Tabla de contenidos

# Ficha del Trabajo Final

| | |
|---|---|
| **Título del trabajo:** | **Predicción de compra en el entorno ecommerce: comparación de modelos, variables y procedimientos en conjuntos de datos desbalanceados** |
| **Nombre del autor/a:** | Víctor Ayala Sánchez |
| **Nombre del Tutor/a de TF:** | Santiago Rojo Muñoz |
| **Nombre del/de la PRA:** | Albert Solé Ribalta |
| **Fecha de entrega:** | 12/2022 |
| **Titulación o programa:** | Analítica Empresarial |
| **Área del Trabajo Final:** | Área de TF |
| **Idioma del trabajo:** | Castellano |
| **Palabras clave** | Modelos de clasificación, Conjuntos de datos desbalanceados, Random Forest |

**Resumen del Trabajo**

El trabajo consistió en probar 5 algoritmos de clasificación: regresión logística, árbol de decisión, random forest, AdaBoost y XGBoost. La comparación fue sobre un conjunto de datos de clases desbalanceadas de compradores y no compradores de un portal web de venta de información financiera. Cada algoritmo fue entrenado con 4 conjuntos de datos, el original y 3 reducidos con diferentes técnicas: PCA, UMAP y TSNE. Además cada conjunto de datos fue balanceado con diferentes técnicas: ROS, SMOTE y ADASYN con diferentes proporciones entre las clases.

Posteriormente se eligieron los modelos de mayor performance y se probaron contra los conjuntos de datos sin balancear, como métrica principal de evaluación se tomó el F1-Score.

El modelo de mayor performance fue XGBoost con el conjunto de datos reducido por UMAP y balanceado al 15% con ROS.

Se notaron ligeras diferencias en el rendimiento de los algoritmos si fueron entrenados con conjuntos reducidos con PCA, UMAP y TSNE. Se descubrió que el procedimiento menos adecuado para balancear es igualar las clases, los resultados más adecuados se obtuvieron balanceando al 15%.

**Abstract**

The work consisted of testing 5 classification algorithms: logistic regression, decision tree, random forest, AdaBoost and XGBoost. The comparison was based on an unbalanced data set of buyers and non-buyers of a web portal that sells financial information. Each algorithm was trained with 4 data sets, the original and 3 reduced with different techniques: PCA, UMAP and TSNE. Furthermore, each data set was balanced with different techniques: ROS, SMOTE, and ADASYN with different proportions between classes.

Subsequently, the models with the highest performance were chosen and tested against the unbalanced data sets, using the F1-Score as the main evaluation metric.

The model with the highest performance was XGBoost with the data set reduced by UMAP and balanced to 15% with ROS.

Slight differences in the performance of the algorithms were observed if they were trained with reduced sets with PCA, UMAP and TSNE. It was discovered that the least suitable procedure to balance is to equalize the classes, the most suitable results were obtained by balancing at 15%.

# 1 - Introducción

## 1.1 Contexto y justificación del Trabajo

Géron (2019) define el aprendizaje automático como la ciencia y el arte de programar a los ordenadores para que estos puedan aprender de los datos.

Una segunda y también acertada definición es propuesta por Igual et al.( 2017), el aprendizaje automático es la disciplina encargada de la codificación de programas que ajustan automáticamente su rendimiento de acuerdo con su exposición a la información en los datos. Dicho ajuste, es referido como aprendizaje y es logrado gracias a modelos que son susceptibles de presentar cambios en sus parámetros. El aprendizaje automático se puede considerar un subcampo de la inteligencia artificial (IA).

El aprendizaje automático puede ser clasificado de varias formas, una de las más comunes sería en aprendizaje supervisado y no supervisado (Igual et al., 2017; Géron, 2019). El presente trabajo y la literatura consultada se relacionan principalmente con el aprendizaje supervisado, de forma específica con una clase de este tipo llamada clasificación.

Los problemas de clasificación dentro del dominio de ciencia de datos implican la correcta predicción de una etiqueta: Si o No. Es decir se busca la predicción de una variable binaria (Igual et al., 2017).

Los problemas de clasificación utilizan algoritmos de clasificación para su modelaje estadístico. A continuación se describen algunos algoritmos comúnmente utilizados en el tipo de problema mencionado:

**Naive Bayes**, este modelo basado en el teorema de Bayes, busca estimar una probabilidad de encontrar un valor de la variable dependiente dado una serie de valores de las variables predictoras que han aparecido (Bruce et al., 2020).

Para cada registro en los datos el algoritmo aplicará los siguientes paso (Bruce et al., 2020):

1. Buscar todos los registros con el mismo perfil de predictor (es decir, donde los valores de las variables independientes son los mismos).
2. Determinar a qué clases pertenecen esos registros y qué clase prevalece más (es decir, es más probable).
3. Asignar esa clase al nuevo registro. El enfoque anterior equivale a encontrar todos los registros en la muestra que son exactamente como el nuevo registro

**Regresión Logística**, es equiparable a la regresión lineal múltiple, excepto que el resultado es binario. El modelo emplea varias transformaciones para convertir el problema en lineal. Debido a su velocidad computacional se trata de un método popular (Bruce et al., 2020).

Otra definición del método es presentada por Igual et al. (2017), como un tipo de modelo de clasificación probabilístico. Representa un modelo binario para predecir una respuesta binaria a partir de uno o varios predictores.

**SVM** las máquinas de vectores de soporte clasifican los datos encontrando el hiperplano que maximiza el margen entre las clases en los datos de entrenamiento (separa de mejor forma). En un ejemplo bidimensional es decir con dos clases, se puede pensar en un hiperplano como la línea recta más ancha (es decir, la línea con márgenes) que separa las dos clases (Albon, 2018).

**Árboles de Decisión** son algoritmos versátiles de aprendizaje automático que pueden desempeñar tareas de clasificación y regresión, e incluso tareas de salida múltiple. Son algoritmos muy potentes, capaces de ajustar conjuntos de datos complejos y con variables heterogéneas (con diferentes escalas de medidas). Los árboles de decisión también son los componentes fundamentales de Random Forests, uno de los modelos de mayor uso en la actualidad (Géron, 2019).

**Random Forest** se trata de un modelo combinado, este caso se compone por un conjunto de árboles de decisión (Géron, 2019). Las técnicas de modelos combinados suelen tener buenas propiedades para combatir el sobreajuste al conjunto de datos. En el caso del

algoritmo de Random Forest, la agregación de clasificadores utilizando una técnica de votación reduce la varianza del clasificador final. Lo mencionado aumenta la robustez del clasificador y generalmente logra un muy buen desempeño en la clasificación (Igual et al., 2017).

La utilización de modelos y algoritmos de clasificación necesariamente implica la evaluación de los mismos, dado que el objetivo es la correcta predicción y no sólo generar modelos nuevos (Albon, 2018). A continuación se explican brevemente algunas de las métricas utilizadas en la evaluación de la performance de los modelos de clasificación:

La medida o métrica básica para evaluar un algoritmo de clasificación es el **Accuracy** en inglés (Igual et al., 2017). A continuación una fórmula describe el cálculo de la métrica:

$$Accuracy = Número\ de\ predicciones\ Correctas\ /\ N$$

Otra métrica utilizada para la evaluación de modelos es la **Precisión**; definida por Albon (2018) como la proporción de cada observación predicha como positiva que es en realidad positiva (verdaderos positivos), formalmente, la precisión es:

$$Precision = Verdaderos\ Positivos\ /\ Verdaderos\ Positivos + Falsos\ Positivos$$

El **Recall** es la proporción de cada observación positiva que es verdaderamente positiva. Dicha métrica mide la capacidad del modelo para identificar una observación de la clase positiva (Albon, 2018). Se define formalmente así:

$$Recall = Verdaderos\ Positivos\ /\ Verdaderos\ Positivos + Falsos\ Negativos$$

El **F1-Score** es definido como una métrica que representa un balance entre la precisión y el recall, este se calcula con la media armónica entre las métricas mencionadas (Albon, 2018):

$$F1\ Score = 2 \times (Precision \times Recall\ /\ Precision + Recall)$$

Por último, es importante nombrar otra métrica de evaluación, especial dentro de los problemas de clasificación donde se tienen clases desbalanceadas. El **Balanced Accuracy** calcula la precisión equilibrada, lo que evita las estimaciones de rendimiento infladas en

conjuntos de datos desbalanceados; cuando se tiene una clase que representa más del 90% de los registros, los algoritmos de clasificación tendrán valores inflados del accuracy.

La métrica de **Balanced Accuracy** se define como el macro promedio de las puntuaciones del recall por clase o, de manera equivalente, la precisión bruta donde cada muestra se pondera de acuerdo con la prevalencia inversa de su verdadera clase. Por lo tanto, para conjuntos de datos equilibrados, la puntuación es igual a la precisión.

En el caso binario, la precisión equilibrada es igual a la media aritmética de la sensibilidad (tasa de verdaderos positivos) y la especificidad (tasa de verdaderos negativos), o el área bajo la curva ROC con predicciones binarias en lugar de puntajes:

$$Balanced\ Accuracy\ = 1/2\ \times\ ((VP/\ VP\ +\ FN)\ +\ (VN/VN\ +\ FP))$$

VP: Verdaderos Positivos

VN: Verdaderos Negativos

FN: Falsos Negativos

VN: Verdaderos Negativos

Hasta este punto se han abordado de forma breve tanto algunos modelos comúnmente utilizados dentro de los problemas de clasificación como su forma de evaluarlo. Sin embargo es importante resaltar la existencia de problemas de clasificación en dónde las clases o etiquetas se encuentran desbalanceadas. Es decir existe una clase dominante que representa al menos el 95% de la muestra frente a una minoritaria. Teniendo esto en consideración se mencionan algunas técnicas utilizadas para solventar dicho problema al momento de entrenar a los modelos:

- Submuestreo o Undersampling, implica tomar una muestra de los registros de la clase dominante para así equiparar ambas.
- Sobremuestreo o Oversampling, implica realizar un muestreo aleatorio con reemplazo sobre la clase inferior. Este procedimiento duplica datos de la clase

inferior para sobre representarla dentro del conjunto de datos y equiparar así ambas clases.

- Generación de datos, implica crear datos sintéticos similares (pero no iguales) de los registros de la clase minoritaria. Esta técnica se ha vuelto popular desde la aparición del algoritmo SMOTE (Synthetic Minority Oversampling Technique), destinado justamente a crear registros similares de la clase minoritaria y así balancear el conjunto de datos.

Teniendo en cuenta todo el contexto comentado con anterioridad, se pretende utilizar una combinación de los algoritmos, técnicas de reducción y técnicas de muestreo para poder clasificar de forma correcta los visitantes / usuarios de una página web que contiene una parte de comercio electrónico. El portal de comercio electrónico mencionado tiene como objetivo principal la venta de informes financieros.

La justificación de este tipo de investigaciones se relaciona al poder lograr una ventaja competitiva con respecto a otras empresas y sitios web que ofrecen servicios similares. Lo comentado es una situación común dentro del ecosistema del comercio electrónico, dónde triunfan aquellas personas con mayor comprensión de los usuarios y clientes.

## 1.2 Objetivos del Trabajo

**Objetivo principal**

Construir un modelo mediante la comparación de múltiples algoritmos, que permite la correcta clasificación entre usuarios que son clientes y los que no

**Hipótesis**

Los usuarios compradores y no compradores pueden ser clasificados a partir de su comportamiento en la página web y una serie de variables asociadas.

**Objetivos Parciales**

- Determinar el procedimiento más adecuado para el tratamiento de conjunto de datos desbalanceados
- Comparar los diferentes modelos y elegir el más adecuado para el conjunto de datos.
- Determinar el procedimiento de reducción de dimensiones más adecuado para la performance de los modelos de clasificación.
- Determinar variables relevantes para la clasificación de los usuarios. Es decir, determinar las variables relevantes asociadas a los usuarios que efectivamente realizan una compra de servicios de información financiera.

## 1.3 Impacto en sostenibilidad, ético-social y de diversidad

Dentro del contexto del comercio electrónico y la ciencia de datos aplicada se tocan diversos temas éticos que deben ser tomados en consideración. Es importante que toda investigación realizada se mantenga dentro del marco regulatorio de la Unión Europea con respecto a la privacidad y la protección de datos en todas las fases.

Actualmente los algoritmos y sistemas inteligentes no solamente son capaces de inferir características de las personas tales como su género o nivel educativo, sino en cambio también son capaces de inferir la orientación sexual, sus opiniones religiosas o sus opiniones políticas. Por lo tanto se debe asegurar que el producto o servicio de una investigación no excluya o discrimine de alguna forma a las personas (Porcelli, 2020).

Además de lo comentado, debe también asegurarse la transparencia y trazabilidad de los sistemas en los que está involucrado el aprendizaje automático y la inteligencia artificial; si el usuario de un servicio se ve perjudicado de alguna forma por la decisión tomada por un algoritmo, este se encuentra en su derecho de conocer el funcionamiento del mismo (Porcelli, 2020).

Por último, la actividad de la ciencia de datos aplicada al comercio electrónico y la presente investigación deben asegurar de alguna forma el bienestar social y medioambiental. Es decir se debe tomar en cuenta el impacto medioambiental (si lo hubiese) de la actividad de entrenar modelos y utilizarlos dentro del contexto empresarial (Porcelli, 2020).

Teniendo en cuenta el contexto ético descrito, se comenta que con respecto a la presente investigación y al tratarse de data de navegación web (totalmente anonimizada), se cumplen con criterios de protección de datos personales y trazabilidad de usuarios. El objetivo de la investigación no es determinar posiciones políticas, valores u orientación sexual de los usuarios, teniendo en cuenta que no se tienen datos suficientes como para plantearlo dentro de la investigación.

## 1.4 Breve sumario de productos obtenidos

El principal producto de la investigación es una serie de modelos estadísticos capaces de determinar con algún grado de efectividad si alguien es un potencial comprador o no.

# 2 - Estado del arte - Investigaciones previas

## 2.1 Técnicas de reducción de dimensionalidad

En este apartado se describen algunas técnicas de reducción de dimensiones utilizadas en la actualidad.

**PCA**

Géron, (2019) comenta que el análisis de componentes principales es en la actualidad una de las técnicas más populares y comúnmente utilizadas en la reducción de dimensiones. Consiste en primer lugar en identificar un hiperplano cercano a los datos para posteriormente proyectarlos en dicho hiperplano.

Otra definición es planteada por Müller & Guido,( 2016) como un método que rota el conjunto de datos de una manera tal que las características o variables rotadas no están correlacionadas estadísticamente. Dicha rotación es a menudo seguida por una selección de un subconjunto de las nuevas características, de acuerdo con la importancia que tienen para explicar la varianza.

**TSNE**

Müller y Guido, (2016) comentan que la idea principal detrás del algoritmo T-SNE es encontrar una representación bidimensional de los datos que sea capaz de conservar las distancias entre puntos lo mejor posible.

**UMAP**

McInnes et al., (2018) escribieron la primera publicación donde comentaban y definían UMAP (Uniform Manifold Aproximation and Projection) como una técnica nueva de aprendizaje múltiple para la reducción de dimensiones. UMAP se basa en en la geometría riemanniana. Se trata de un algoritmo escalable práctico que es aplicable a datos del mundo real. El algoritmo UMAP es competitivo con t-SNE en calidad de visualización y podría decirse que conserva más de la estructura global con rendimiento de tiempo de ejecución superior.

## 2.2 Técnicas de Oversampling

En este apartado es importante destacar que existen técnicas de oversampling y undersampling, las primeras consisten en la creación de muestras sintéticas para equilibrar las clases dentro de un problema de clasificación. La segunda por el contrario implica la eliminación aleatoria de muestras de la clase mayoritaria para equilibrar las clases; debido a que la segunda implica eliminación de información potencialmente valiosa para el proceso de entrenamiento de los algoritmos, se decide trabajar solo con Oversampling en la presente investigación.

A continuación se describen algunas técnicas utilizadas dentro del oversampling:

**ROS - Random Oversampling**

Se trata de una de las técnicas más básicas y consiste principalmente en generar nuevas muestras en las clases que están subrepresentadas. La estrategia más básica consiste en generar nuevas muestras mediante muestreo aleatorio con reemplazo de las muestras disponibles actualmente (Lemaítre et al., 2017). Es decir se toman de forma aleatoria muestras de la clase minoritaria y estas se duplican en la proporción deseada.

**SMOTE - Synthetic Minority Over-sampling Technique**

Se trató en su momento de una técnica nueva planteada en un paper por (Chawla et al., 2002). Los autores de SMOTE proponen sobremuestrear la clase minoritaria creando registros o ejemplos "sintéticos" en lugar de sobremuestrear con reemplazo (como en el caso de ROS).

Dicho enfoque está inspirado en una técnica que resultó exitosa en el reconocimiento de caracteres escritos a mano (Ha & Bunke, 1997 citado en Chawla et al., 2002). Quienes crearon datos de entrenamiento adicionales al realizar ciertas operaciones en datos reales.

La clase minoritaria se sobre muestrea tomando cada muestra de clase minoritaria e introduciendo ejemplos sintéticos a lo largo de los segmentos de línea que unen cualquiera o todos los vecinos más cercanos de la clase minoritaria. Dependiendo de la cantidad de

sobremuestreo requerida, los vecinos de los vecinos más cercanos se eligen de forma aleatoria.

**ADASYN - Adaptive Synthetic**

Fue presentado por He et al.,( 2008) como una nueva técnica de muestreo. Dicha técnica de muestreo plantea la creación de nuevas muestras sintéticas a partir de las distribuciones de las variables de la clase minoritaria y tomando ciertos registros como más difíciles de detectar para los algoritmos de clasificación y por lo tanto tomando este tipo de casos con mayor peso en el momento de creación de muestras.

## 2.3 Investigaciones previas en problemas de clasificación

Es variada la documentación, la literatura y los trabajos relacionados al análisis de datos y la aplicación de la ciencia de datos al campo del comercio electrónico y la predicción de compra. A continuación se realiza una breve revisión de la literatura actual y del estado del arte en el contexto mencionado.

Bigon et al.( 2019) , realizaron un estudio de predicción de compra relacionado a un sitio web de moda basándose principalmente en la conducta y los clicks de los usuarios.Como parte del cumplimiento de los aspectos éticos, los investigadores en este caso anonimizaron los datos para hacer imposible su cruce y la identificación de usuarios.

El objetivo del estudio fue en primer lugar generar un conjunto de datos estándar para compartir y que la comunidad de personas dedicadas a la ciencia de datos y el análisis del comportamiento en el sector de la moda puedan tener una norma con respecto a la identificación de potenciales compradores. En segundo lugar se compararon diferentes algoritmos de clasificación para determinar los compradores de los no compradores (Bigon et al., 2019).

La principal métrica utilizada para evaluar los algoritmos fue el accuracy, teniendo mayor performance las redes neuronales. Los autores también identifican como uno de los retos principales el lidiar con el desbalance de la clase no compradora y el sesgo que puede producir en el momento de generar predicciones (Bigon et al., 2019).

Similar a la línea de investigación antes mencionada Niu et al.(2017) trabajaron con datos del sitio web (tracking del comportamiento de los usuarios) de una cadena de retail en Estados Unidos. El propósito de la investigación fue probar algunos algoritmos de aprendizaje automático y evaluar su eficacia en la clasificación de sesiones y usuarios compradores.

Los investigadores reportan que la información originalmente venía en formato json y fue procesada por medio de algunos scripts desarrollados en Python, posteriormente para la construcción de los modelos utilizaron R. Dentro del estudio se definieron las siguientes variables de interés para la construcción de los modelos (Niu et al., 2017):

- Longitud de la query, definida como el número de palabras de una consulta que realiza el usuario dentro del buscador del sitio web de la cadena de retail.
- La posición de la consulta, definida como la posición de la consulta durante la sesión, es decir si fue la primera, segunda o tercera búsqueda del usuario dentro del sitio web
- Número total de consultas, el número total de búsquedas o consultas que realizó el usuario.
- Posición del clic, definida como el orden del click dentro de la lista de interacciones
- Posición del clic promedio, posición promedio dentro de la lista de interacciones
- Número de resultados en la búsqueda, número de productos en la consulta
- Número de clicks, propuesta como el número total de clicks la consulta
- Número de clicks por sesión, número total de clicks dentro de una sesión
- La entropía de los clicks en la consulta, la entropía fue definida como el nivel de ambigüedad dentro de la búsqueda realizada por el usuario.
- Tiempo en pantalla, tiempo total que ha pasado el usuario en la página de un producto.

- Tipo de usuario, si el usuario se encontraba o no registrado
- Dispositivo , nombre del dispositivo desde el cual ha ocurrido la interacción con el sitio web

Debido a la experiencia con el desarrollo de modelos reportada por los autores, los modelos basados en árboles de decisión tienden a tener una performance más elevada debido a la capacidad que tienen este tipo de algoritmos de predecir patrones complejos dentro de los datos. A su vez, ellos sugieren que esto se debe a la insensibilidad que tienen ante la interacción de los factores que son introducidos. En el estudio descrito, se utilizó el random forest como algoritmo principal para clasificar dos grupos: compradores y no compradores. Además de lo mencionado se construyó también un modelo de regresión logística para utilizarlo como línea base dentro del estudio.

Se procesaron un total de millón y medio de transacciones, del cual solo un 4% representan casos reales de conversión (compra efectiva). Los investigadores reportan un total de 8 búsquedas en promedio por usuario dentro de una sesión, alrededor de 3 minutos en promedio en la visualización de cada item o producto y 1.92 clicks en promedio por búsqueda. Debido al evidente problema de clases desbalanceadas dentro del conjunto de datos (96% no compradores, 4% compradores), los investigadores decidieron realizar un submuestreo de la clase dominante (los no compradores).

Los autores del estudio concluyen que el modelo combinado de random forest tiene un accuracy alto de 0.70, por lo tanto el modelo propuesto es eficiente en la tarea de clasificación de compradores, sin embargo no son capaces de determinar cuales predictores son más importantes en la conducta mencionada (Niu et al., 2017).

Otro estudio realizado por Jia et al., (2017), representa también un ejemplo de búsqueda de modelos adecuados para problemas de clasificación entre usuarios que compran y no compran dentro del contexto de comercio electrónico. En este caso, los investigadores tuvieron un enfoque distinto, utilizando algoritmos de clasificación Bayesianos.

Tomaron un conjunto de datos públicos de la tienda retailer Ali (en China), liberado en una competencia de 2015. El análisis de los autores estuvo centrado en los clicks,

descubriendo un patrón de hasta 5 veces más en el número de clicks entre aquellos usuarios que terminaban efectuando una compra.

Una de las hipótesis propuestas dentro de la investigación comentada fue que los usuarios que muestran interés en comprar un determinado ítem o producto por lo general pasan más tiempo en pantalla en ese ítem y se ve un incremento de los click asociados. Otra variable de interés para los investigadores es el tiempo que transcurre desde la última visita antes de la compra (Jia et al., 2017).

El algoritmo de clasificación principalmente utilizado dentro del estudio fue el Naive Bayes y además lo combinaron con una validación cruzada en 10 particiones del conjunto de datos. Los autores también reportaron un problema común de este tipo de datos, el desbalance entre las clases. Sin embargo no reportan alguna medida, metodología o procedimiento ante tal problema o los efectos del mismo sobre las predicciones que se generan (Jia et al., 2017).

Como métricas utilizadas para la evaluación del modelo, tomaron la precisión y el recall, obteniendo en ambas métricas promedios mayores a 0.80. Se tomaron como conclusiones del estudio que una aproximación bayesiana puede ser una vía efectiva para encarar los problemas de clasificación con el tipo de datos mencionado (Jia et al., 2017).

Otra aproximación diferente fue tomada por los autores Suchacka et al.,( 2015), quienes en lugar de analizar a los usuarios, tomaron una menor granularidad e intentaron trabajar y clasificar las sesiones. En este caso los investigadores definen una sesión como una serie de peticiones HTTP que realiza un usuario o potencial comprador a un servidor web.

Las sesiones analizadas en este estudio pertenecen al sitio web de una tienda de libros en línea, el total 39.000 registros fue separado en los conjuntos de entrenamiento y test y la variable objetivo a predecir fue si la sesión había terminado en una compra efectiva o no. Se utilizó el algoritmo de vecinos cercanos (K-NN, en inglés), junto con la distancia euclídea (Suchacka et al., 2015).

Se evaluó el modelo con las métricas de accuracy y la sensibilidad (verdaderos positivos / (verdaderos positivos + falsos negativos)). El resultado fue un accuracy elevado de

99%, sin embargo no se reportan problemas de desbalance o posible sesgos en los datos debido a este.

Como ejemplo de estudios que utilizan otra familia de modelos, tenemos la investigación realizada por Dou, (2020) quien utilizó el modelo Cat Boost (familia de descenso de gradientes). El autor recomienda este tipo de aproximaciones, con este tipo de modelos, cuando se tiene un problema de clasificación con una gran cantidad de variables predictoras, heterogéneas en su medición y conjunto de datos complejos.

El conjunto de datos final del estudio estuvo compuesto por 12300 registros que describen los patrones de navegación, clicks y detalle de los productos consultados de una plataforma de comercio electrónico. La variable a predecir fue la compra o no por parte de los usuarios y como variables predictoras se tomaron en cuenta: el número de páginas visitadas, tiempo total en cada página, el producto asociado a la página vista, la tasa de rebote, la tasa de salida, si se trataba de un día feriado o no, el tipo de navegador, el tipo de usuario (visitante recurrente o no) y si la fecha de navegación fue durante el fin de semana o no (Dou, 2020).

El autor reportó un conjunto de datos desbalanceado y por lo tanto tomó como criterios de evaluación del modelo el área bajo la curva (la curva ROC) y el F1 score, teniendo esta última un valor elevado de 0.83 luego de probar el modelo con diferentes hiper parámetros (Dou, 2020, #).

Se concluye dentro de la investigación que el modelo Cat Boost es adecuado para tratar con conjuntos de datos desbalanceados y además efectivo para la clasificación de compradores (Dou, 2020).

Siguiendo con la revisión de estudios, se tiene la investigación de Bingbing et al., (2021) quienes realizaron un estudio comparativo con la finalidad de predecir la conducta de compra.

Entre los algoritmos utilizados se tuvieron en cuenta: redes neuronales recurrentes (LSTM), máquina de soporte vectorial y vecinos cercanos (Knn). El objetivo principal de la investigación fue probar un modelo ensamblado: en primer lugar se utilizó la red neuronal recurrente para modelar el comportamiento de clicks y búsquedas a lo largo del tiempo,

además de las variables género, edad y nivel educativo, la red fue utilizada para extraer los features o variables de relevancia que funcionaron como input a la máquina de soporte vectoriales, con este último algoritmo se realizaba la predicción de compra o no compra (Bingbing et al., 2021).

Se analizaron los datos de 105.320 usuarios (de la web de comercio electrónico de un centro comercial) y el conjunto de datos fue dividido en 80% entrenamiento y 20% test. Los investigadores reportaron problemas de desbalance, sin embargo no mencionan una solución formal o método para lidiar con el problema. Como indicadores principales para evaluar los modelos puestos a prueba los autores del estudio utilizaron 3 métricas principales: la precisión, el recall y el F1 score (Bingbing et al., 2021).

Los resultados obtenidos muestran una performance mayor en las 3 métricas utilizadas en el modelo compuesto por redes neuronales recurrentes y la máquina de soporte vectorial, con respecto al k vecinos y la máquina de soporte vectorial sola (Bingbing et al., 2021).

Otro estudio propuesto como un benchmark donde se compararon los siguientes algoritmos: XGBoost, SVM, Random Forest y CatBoost fue realizado por Cao et al., (2021). Se utilizaron datos de un sitio de comercio electrónico relacionado a la venta de cursos y formación en línea. Entre las variables analizadas estaban: información básica del usuario, información de visitas anteriores al sitio, información de si hubo o no compra e información de logeo.

Los investigadores reportan que los algoritmos utilizados dentro del estudio tienen un mayor rendimiento dentro del contexto de problemas de clasificación con conjuntos de datos desbalanceados. Como indicadores los autores del estudio también han utilizado el F1 score (Cao et al., 2021).

Los autores de la investigación reportaron haber realizado una limpieza de datos antes de aplicar las técnicas mencionadas: borraron registros duplicados e incompletos. Posteriormente también borraron algunas variables que se consideraban innecesarias en el estudio. Los resultados demuestran un pobre rendimiento en relación al F1-Socre de la máquina de soporte vectorial y el random forest. Tanto el modelo XG Boost como CatBoost tuvieron un F1-Score, moderado alto de 0.76 y 0.77. Todos los algoritmos reportan un accuracy

elevado por encima de 0.95, lo que da cuenta del problema de sobre entrenamiento en un problema de clases desbalanceadas (Cao et al., 2021).

Cao et al.,( 2021) concluyen que el modelo con la performance más alta es el CatBoost, con un área bajo la curva de 0.84, lo cual indica una alta sensibilidad del modelo para realizar predicciones. Además resumen otros hallazgos relevantes en los siguientes puntos:

- Un número elevado de visitas recurrente se relaciona con la conducta de no compra.
- La promoción continua y el regalo de cupones de descuento se relaciona a la compra.
- Una optimización de la experiencia del usuario se relaciona con la conducta de compra.

Otro ejemplo de investigación donde se han comparado múltiples modelos fue realizada por Wu & Li, (2022) en este caso el objetivo era comparar modelos combinados compuestos por redes neuronales y árboles de decisión con boosting. Los investigadores como métricas para comparar la performance de los diferentes modelos utilizaron el accuracy, el recall, el F1-Score. y el área bajo la curva (ROC).

La hipótesis principal de los autores del estudio fue que el modelo combinado que incluye redes neuronales y árboles de decisión tendría una performance mayor con respecto a los demás algoritmos de aprendizaje automático utilizados (Wu & Li, 2022).

Los datos utilizados en el estudio corresponden a una fuente de comercio electrónico con 2.300.000 millones de registros de conducta de usuarios, incluída la de compra. Los investigadores reportan que el conjunto de datos no se encuentra desbalanceado entre las clases de compra y no compra y por lo tanto no son necesarias medidas para dicho problema. El conjunto fue separado en entrenamiento (75%) y test (25%) y usó la validación cruzada con criterio de 5 (Wu & Li, 2022).

Se compararon un rango amplio de modelos de aprendizaje automático entre los que se encuentran: regresión logística, vecinos cercanos, árboles de decisión con descenso de gradiente, random forest, Light GBM, NG Boost, CatBoot, Multi layer perceptron, árboles de decisión con gradient boost combinado con regresión logística y árboles de decisión con gradient boost con redes neuronales; siendo este último modelo el que obtuvo mayores resultados en accuracy, recall, F1 Score y en la curva ROC.

Los autores del estudio concluyen que el modelo basado en redes neuronales y árboles de decisión tiene mayor eficiencia computacional y logra realizar predicciones más adecuadas a la realidad en comparación con otros modelos base incluidos en el estudio (Wu & Li, 2022).

Otra aproximación en la que se ha comparado múltiples modelos fue realizada por los autores Lee et al. (2021), se tenían 2 objetivos principales que se describen a continuación:

- Encontrar el modelo de aprendizaje automático que más se adecua para la predicción de datos de compra de comercio electrónico.
- Encontrar el método de muestreo más adecuado para mejorar las predicciones, teniendo en cuenta que el conjunto de datos se encuentra desbalanceado.

El conjunto de datos utilizado fue recolectado por medio de Gooogle analytics y fue recolectado durante los años 2016, 2017 y 2018. Cabe destacar que el conjunto de datos es público y se puede encontrar en repositorios como Kaggle; los datos corresponden a una tienda de comercio electrónico de google, desde la cual se pueden comprar todo tipo de mercancía relacionada a la marca (Lee et al., 2021).

El tamaño del conjunto de datos se encuentra en el orden del 1.5 millones de usuarios y solo el 2% representan transacciones efectivas; por lo que los autores han reportado el problema de desbalance dentro del contexto de la aplicación de diferentes algoritmos (Lee et al., 2021).

Se utilizaron una gran cantidad de métricas para la comparación de los diferentes modelos,

sin embargo los autores principalmente reportan 3:

- El área bajo la curva o curva ROC,
- La sensibilidad calculada como la proporción de Verdaderos Positivos/ Verdaderos Negativos + Falsos Negativos
- La especificidad, calculada como la proporción de Verdaderos Negativos / Verdaderos Negativos + Falsos Negativos.

El estudió comparó los siguientes algoritmos: árboles de decisión, redes neuronales, vecinos cercanos, regresión logística, máquina de soporte vectorial, random forest, GBM y XG Boost. De todos los modelos utilizados dentro del estudio se realizó la optimización de hiper parámetros y se calcularon las métricas de evaluación promediando los resultados de la validación cruzada aplicada con 5 particiones del conjunto de datos (Lee et al., 2021).

Los resultados indican que el modelo XG Boost muestra el rendimiento más alto en el área bajo la curva (0.86). Con respecto a los métodos de muestreos lo autores han utilizado principalmente tres aproximaciones diferentes:

- La primera, utilizar los datos completos sin modificaciones de muestreo.
- La segunda, utilizando undersampling. Es decir, igualando el número de registros de la clase mayoritaria al número de registros de la clase minoritaria.
- La tercera, Oversampling, igualando el número de registros de la clase minoritaria al número de registros de la clase mayoritaria. Lo mencionado, por medio de la generación de muestras sintéticas de compradores efectivos.

Los resultados indican que el SMOTE (oversampling con muestras sintéticas), como aproximación al problema del desbalance produce una mayor performance de las métricas comentadas (Lee et al., 2021).

Por último, otro estudio llevado a cabo por Chaudhuri et al., (2021) tuvo como objetivo comparar algoritmos de aprendizaje automático y algoritmos de aprendizaje profundo para evaluar su rendimiento en el problema de clasificación de compradores y no compradores.

El conjunto de datos pertenece a la web de comercio electrónico de una tienda de retail con operaciones en Alemania y Bélgica. Consta de 430000 registros que corresponden a sesiones, en las que el 67% corresponde a transacciones efectivas y el resto no (Chaudhuri et al., 2021).

Las técnicas tradicionales de aprendizaje automático utilizada en la investigación fueron: árboles de decisión, random forest y la máquina de soporte vectorial (SVM); por otra parte, como técnica principal de aprendizaje profundo, utilizaron redes neuronales profundas (Chaudhuri et al., 2021). Como paso previo realizaron una limpieza general del conjunto de

datos, con reemplazo de valores nulos o perdidos y tratamiento de valores extremos. Los autores también realizaron una validación cruzada con 5 particiones del conjunto de datos original (Chaudhuri et al., 2021).

Para poder establecer comparaciones entre modelos, los investigadores hicieron uso de hasta 8 métricas diferentes: Accuracy, Recall, F-1 Score, PPV (Positive Predictive Value o Valores Positivos Predichos), NPV(Negative Predictive Value o Valores negativos predichos), FPR (False Positive Rate o Ratio de falsos positivos),curvas ROC y el coeficiente de correlación de Matthews

El mayor rendimiento en todas las métricas lo obtuvo la red neuronal profunda, con un F1 score de 0.92. Los autores del estudio concluyeron que la aproximación al problema de clasificación por vía de aprendizaje profundo puede ser la más adecuada para lograr resultados sobresalientes (Chaudhuri et al., 2021).

Luego de una breve revisión de la literatura referida al tema de estudio se pueden establecer las siguientes conclusiones:

- Es importante establecer una o más métricas de comparación y evaluación de modelos.
- Los modelos combinados parecen mostrar mayor robustez ante el tipo de problema planteado.
- Las redes neuronales pueden representar una alternativa válida para tratar el problema.
- No se establecen demasiadas comparaciones sobre cómo proceder ante los conjuntos de datos desbalanceados, propios del tipo de fenómeno que se estudia.

# 3 - Materiales y Métodos

El desarrollo del proyecto fue realizado enteramente en el lenguaje de programación python, en específico python 3.8.8.

Se utilizaron además librerías propias de python para la ciencia de datos como: pandas, para la manipulación de datos, numpy para cálculos científicos y manipulación de vectores, seaborn / matplotlib para la visualización de datos. Además se ha utilizado también Scikit learn para la explotación de los datos, importación de modelos y procedimientos de evaluación de performance e imbalanced-learn para la implementación de los procedimientos de sobremuestreo.

Por último se utilizaron otras dos librerías: XGB y UMAP, la primera es la librería para algoritmos ensamblados y la otra es un algoritmo para reducción de dimensiones.

Para consultar al detalle las librerías utilizadas en el proyecto, se tiene un fichero de **requeriments.txt** en el repositorio del código fuente (ver apartado 7.)

# 4 - Descripción del conjunto de datos

El conjunto de datos se encuentra compuesto principalmente por 4 tablas. Los datos se corresponden a información histórica obtenida de las sesiones / visitas, compras y consumos promocionales de los usuarios dentro de un portal de comercio electrónico cuyo principal objetivo es la venta de información financiera sobre diferentes empresas de Colombia: Informes Comerciales y módulos de información detallada sobre datos financieros, Prensa, Administradores, Incidencias, etc, Informes Sectoriales,Base de datos a medida, Productos de Marketing (mercadeo), Información de accionistas, Información de proveedores y clientes, etc (Rojo Muñoz, 2022).

Los potenciales compradores siguen el siguiente flujo: visitan la página de comercio electrónico, realizan el proceso de registro y comienzan a consumir promociones, posteriormente pagan por algún tipo de información concreta o informes mensajes y pasan a ser clientes.

Se presenta a continuación una breve descripción de las tablas:

**Usuarios**

Se trata de la tabla de los registros de los usuarios:

| Campo | Ejemplo | Descripción |
|---|---|---|
| IDUSUARIO | 8107310 | Id Único de Usuario |
| TIPOUSUARIO | PF | PJ-Persona Jurídica, PF= Persona Física/Persona Natural, PX=Puede ser PJ pero no es seguro |
| FECHA_REGISTRO | 22/10/2019 | Fecha de Registro del Usuario |
| CANAL_REGISTRO | 3 | Canal de Registro del Usuario, 1 se relaciona con SEM, 4 con SEO. 2, 3, 7 son directorios populares. Resto son directorios especializados |
| IND_CLIENTE | 1 | Indicador de Cliente. Variable Target dicotómica, 1=Compra, es cliente. |
| FEC_CLIENTE | 22/10/2019 | Fecha en la que el Usuario se convierte a Cliente |
| TIPOEMAIL | gmail.com | Dominio email del usuario |
| BONDAD_EMAIL | 0 | Bondad/Ponderación del email, resultado de campañas de emailing:<br>20= Verde (OK)<br>9 = Naranja (Ha dado un error temporal pero seguimos enviando)<br>1 = SPAM<br>0 = Rojo (Invalido)<br>-10 = Dominio invalido (invalido)<br>-20 = No emai |
| USU_TIPO | PJ | Tipología Usuario, PJ- Persona Jurídica, PN-Persona Natural, PX-Indefinido |

| | | |
|---|---|---|
| USU_TAMANIO | GR | Tamaño de la Compañía del Usuario si TIPOUSUARIO=PJ<br>GR - Grande<br>MD - Mediana<br>PQ - Pequeña<br>MC - Micro<br>SD - Sin Definir |
| USU_CIIU | A0100 | Código de Actividad CIIU si TIPOUSUARIO=PJ |
| USU_ESTADO | CANCELACION | Estado/Situación de la Compañía, si TIPOUSUARIO=PJ |
| USU_DEPARTAMENTO | BOGOTÁ | Departamento/Provincia d |

**Consumos**

Esta tabla lleva el registro de los diferentes consumos gratuitos o promocionales que el usuario ha utilizado:

| Campo | Ejemplo | Descripción |
|---|---|---|
| IDCONSUMO | 40057306 | Identificador Único del Consumo |
| IDUSUARIO | 6868835 | Identificador del Usuario |
| IDPRODUCTO | 144920 | Identificador de Producto, del producto consumido |
| DESCPRODUCTO | Perfil Promocional | Descripción del Producto, del producto consumido |

| | | |
|---|---|---|
| FECHACONSUMO | 1/1/2018 | Fecha del Consumo |
| EMPCONSUL_ID | 986515 | Id Único de la empresa asociada al producto consumido |
| EMPCONSUL_CIIU | H4921 | Código de Actividad CIIU de la empresa asociada al producto consumido |
| EMPCONSUL_PROV | ATLANTICO | Departamento de la empresa asociada al producto consumido |
| EMPCONSUL_EST | ACTIVA | Estado de la empresa asociada al producto consumido |

**Sesiones**

Esta tabla lleva contiene la información de las visitas y sesiones de los usuarios a la web:

| Campo | Ejemplo | Descripción |
|---|---|---|
| IDUSUARIO | 7339406 | Identificador del Usuario |
| FECHA_SESION | 31/8/2018 | Fecha Sesión |
| SESIONS | 12 | Número de Sesiones abiertas por el usuario en esa fecha |

**Ventas**

Contiene toda la información relativa a las ventas realizadas, dicha tabla no formará parte del análisis:

| Campo | Ejemplo | Descripción |
|---|---|---|
| IDVENTA | 40218367 | Id de Registro de Venta |
| IDUSUARIO | 6889066 | Identificador del Usuario |
| FECHAVENTA | 16/1/2018 | Fecha Primera Venta |
| TIPOVENTA | Suscripción | Tipo de la primera Venta: Venta Puntual Informe, Venta Puntual Listado, Suscripción, Bono |
| IMPORTE | 252000 | Importe de la primera Venta |
| NUMVENTAS | 2 | Numero de compras realizadas incluida la primera |

| | | |
|---|---|---|
| IMPORTES | 277000 | Suma Importe de todas las ventas |
| VP Informe | 1 | 1 si se le ha vendido un Informe en alguna ocasión. De lo contrato es Nulo |
| BONO | | 1 si se le ha vendido un Bono en alguna ocasión. De lo contrato es Nulo |
| SUSCRIPCION | 1 | 1 si se le ha vendido una Suscripción en alguna ocasión. De lo contrato es Nulo |
| VP Listado | | 1 si se le ha vendido un Listado en alguna ocasión. De lo contrato es Nulo |

# 5 - Resultados

## 5.1 Análisis Exploratorio de datos

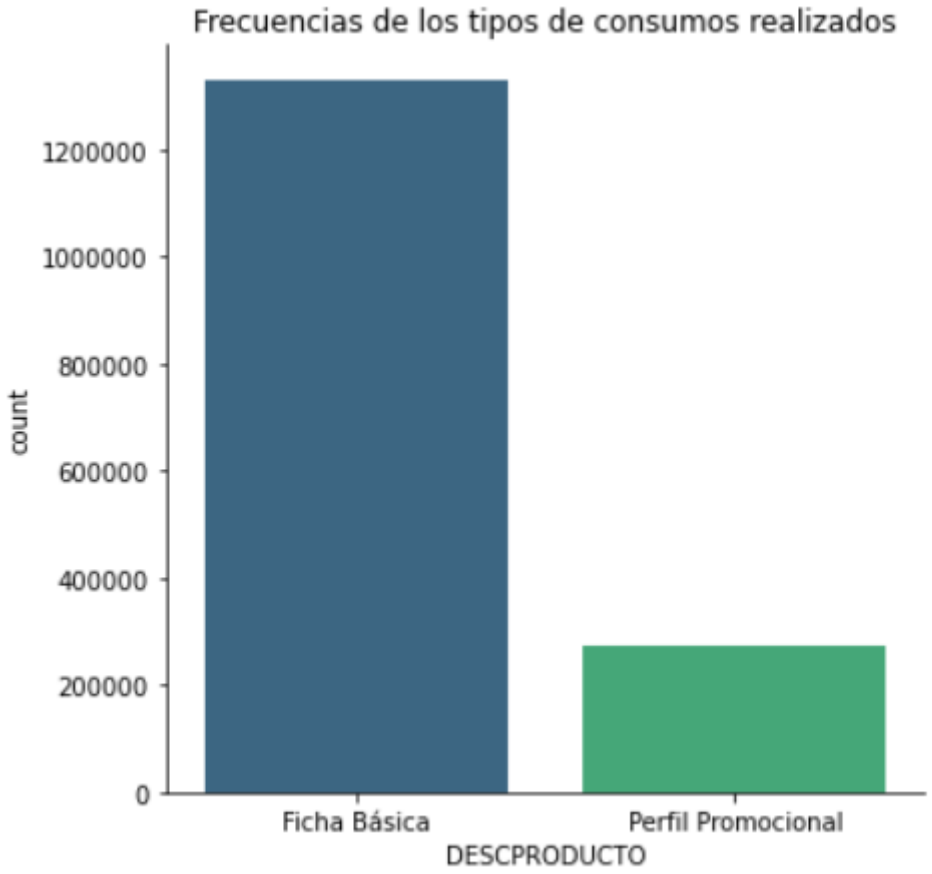
Los usuarios / visitantes de la página de comercio electrónico tienen en promedio unos 11 consumos promocionales. A continuación se presentan algunos estadísticos descriptivos:

| COUNT_DESCPRODUCTO | |
|---|---|
| count | 142.709 |
| mean | 11,2 |
| std | 671,2 |
| min | 1,0 |
| 25% | 1,0 |
| 50% | 2,0 |
| 75% | 6,0 |
| max | 134.694 |

La mediana de la distribución es 2 y se observa que existen valores extremos siendo el máximo de consumos promocionales 134.694

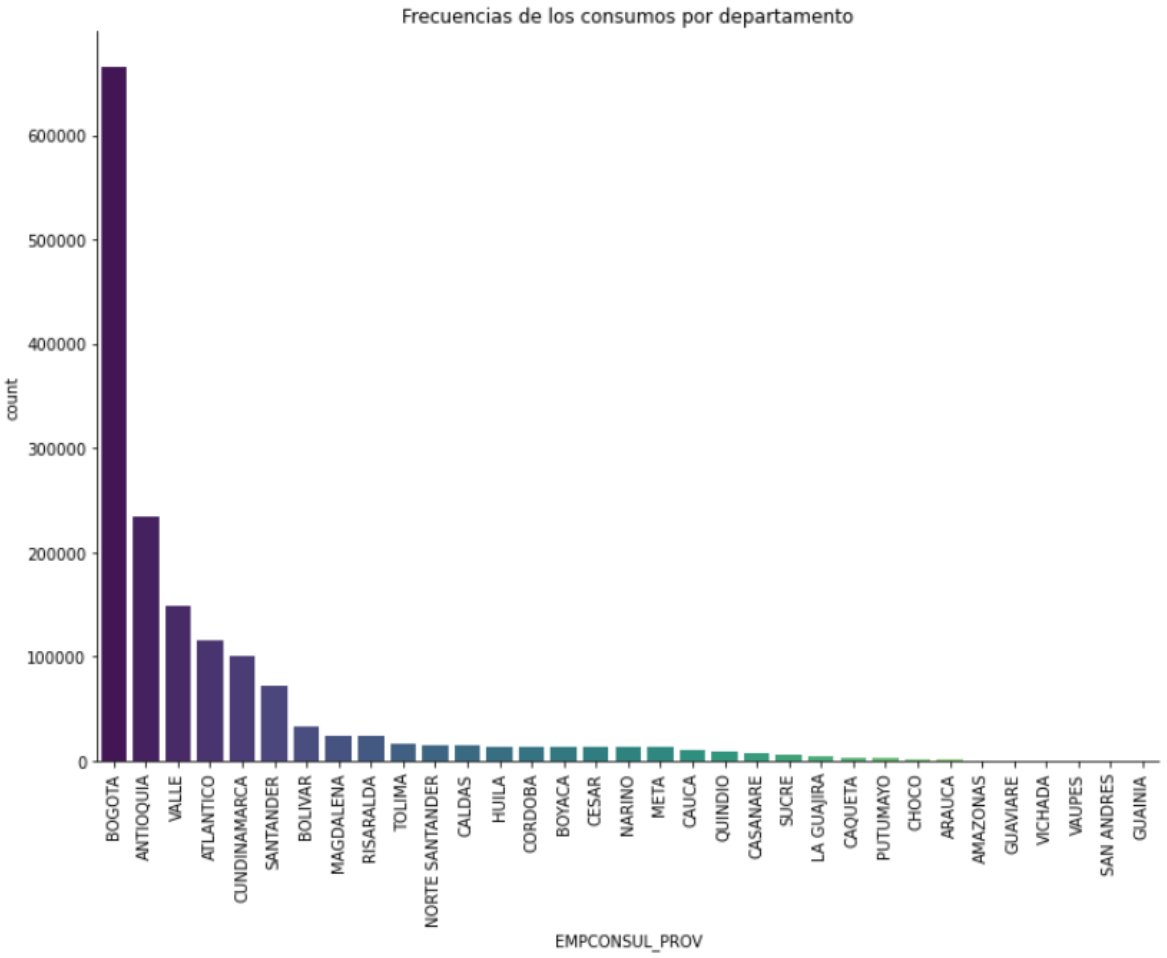
Con respecto a los tipos de consumos promocionales, la ficha básica representa el 83% de los casos son del tipo perfil promocional:

| DESCPRODUCTO | Count | % |
|---|---|---|
| Ficha Básica | 1.328.336 | 83.048 |
| Perfil Promocional | 271.148 | 16.952 |




Con respecto a la localización de las empresas asociadas a los consumos promocionales, se tienen que en el 41% de los casos se corresponde a sociedades en Bogotá:

| EMPCONSUL_PROV | Count | % |
|---|---|---|
| BOGOTA | 666.231 | 41.907 |
| ANTIOQUIA | 234.739 | 14.766 |
| VALLE | 148.956 | 9.370 |
| ATLANTICO | 116.172 | 7.307 |
| CUNDINAMARCA | 100.556 | 6.325 |
| SANTANDER | 71.704 | 4.510 |
| BOLIVAR | 33.379 | 2.100 |
| MAGDALENA | 24.003 | 1.510 |
| RISARALDA | 23.874 | 1.502 |
| TOLIMA | 15.760 | 0.991 |




Por último se destaca que el 82% de los consumos promocionales se encuentran asociados a empresas que se encuentran activas.

Con respecto a los datos de sesiones, tenemos que en promedio un usuario ha tenido unas 7 sesiones, el mismo valor que el percentil 75. Se observan valores extremos en la distribución de sesiones:

| | SESIONES |
|---|---|
| **count** | 182054.0 |
| **mean** | 7,04 |
| **std** | 9,90 |
| **min** | 1,00 |
| **25%** | 4,00 |
| **50%** | 5,00 |
| **75%** | 7,00 |
| **max** | 580,00 |

Analizando los datos de las sesiones agrupadas por año, se observa que el 97% de los datos corresponden a los año 2018 y 2019

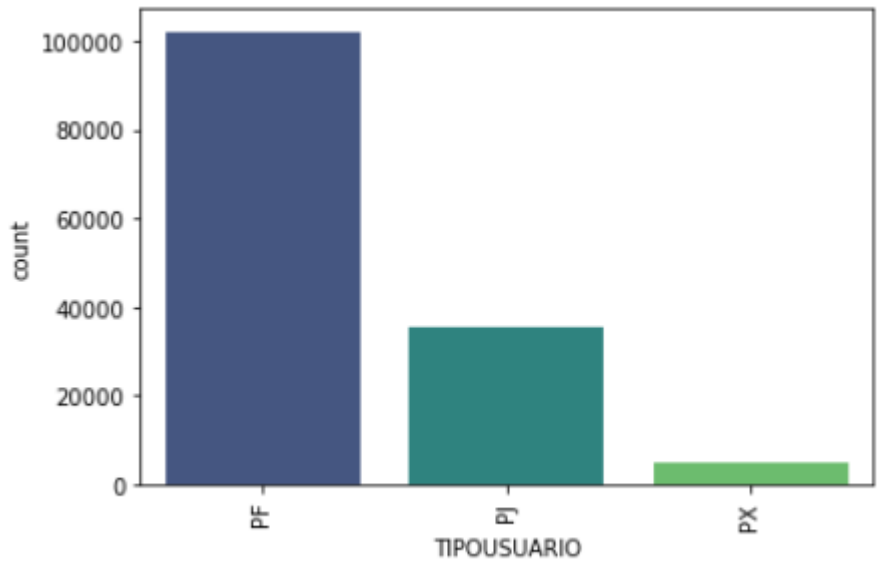
| FECHA_SESION | SESIONES | Pocentaje% |
|---|---|---|
| 2018-12-31 | 606930 | 47.3036 |
| 2019-12-31 | 650727 | 50.7171 |
| 2020-12-31 | 18682 | 1.4560 |
| 2021-12-31 | 6490 | 0.5058 |

En relación a la tabla de usuarios, se tienen un total de 142736 registros únicos. De estos registros un 1.83% corresponde a usuarios que terminaron realizando una compra y por lo tanto hicieron conversión a clientes (2615 registros en total).

| IND_CLIENTE | Count | Porcentaje |
|---|---|---|
| 0 | 140121 | 98.167946 |
| 1 | 2615 | 1.832054 |

Del total de usuarios comentado con anterioridad, el 72% corresponde a personas físicas o naturales:

| TIPOUSUARIO | count | % |
|---|---|---|
| PF | 102160 | 71.573 |
| PJ | 35644 | 24.972 |
| PX | 4932 | 3.455 |



Los usuarios tienen como principal canal de registro el directorio popular, seguido del SEM:

| CANAL_REGISTRO | count | % |
|---|---|---|
| DirectorioPopular | 75780 | 54.434 |
| SEM | 29196 | 20.972 |
| DirectorioEspecializado | 20693 | 14.864 |
| SEO | 13546 | 9.730 |

Con respecto a la bondad o ponderación realizada a los mails de registro de los usuarios, casi el 70% de estos se encuentran en condiciones de ser valorados como válidos:

| BONDAD_EMAIL | count | % |
|---|---|---|
| OK | 99024 | 69.376 |
| Invalido | 26499 | 18.565 |
| DominioInvalido | 9458 | 6.626 |
| NoMail | 3893 | 2.727 |
| Error_Temporal | 1996 | 1.398 |
| SPAM | 1866 | 1.307 |

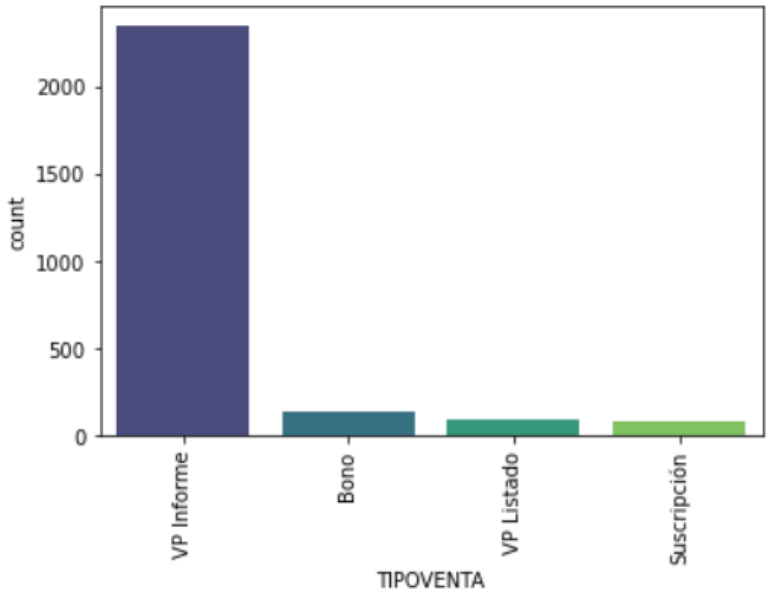
De aquellos usuarios registrados como persona jurídicas o empresas dentro de la tabla de datos de usuarios, el 70% corresponde a empresas de tamaño micro. Además el 80% de los usuarios registrados como empresas se encuentran activas actualmente y al menos el 40% se encuentran localizadas en Bogotá.

| USU_TAMANIO | count | % |
|---|---|---|
| MC | 23096 | 69.339 |
| PQ | 4387 | 13.171 |
| MD | 2612 | 7.842 |
| GR | 2524 | 7.578 |
| SD | 690 | 2.072 |

Top 5 localizaciones de los usuarios registrados:

| USU_DEPARTAMENTO | count | % |
|---|---|---|
| BOGOTA | 14168 | 39.414 |
| ANTIOQUIA | 4366 | 12.146 |
| VALLE | 2898 | 8.062 |
| CUNDINAMARCA | 1941 | 5.400 |
| ATLANTICO | 1665 | 4.632 |

Por último con respecto a la tabla de ventas, en el 88% de los casos, representan Informes específicos:




## 5.2 Preprocesamiento y extracción de features

En esta sección se comentará el proceso para la construcción de una única tabla para la fase de entrenamiento de los diferentes modelos de clasificación para determinar compradores de no compradores.

### 5.2.1 Tabla de consumos

Teniendo en cuenta dicha tabla, recordando que se trata de información de diferentes promociones aprovechadas por los usuarios; se realizaron agregaciones para determinar el número total de consumos realizados por cada id de cliente; lo mencionado con anterioridad nos permite construir 3 variables nuevas: el número total de consumos por usuarios, el número total de consumos de ficha promocional por usuario y el número total de consumos de perfil promocional por usuario. A continuación se presenta un muestra de los primeros registros luego de la agregación de consumos totales:

| IDUSUARIO | NumeroConsumos |
|---|---|
| 6868841.0 | 7 |
| 6868844.0 | 1 |
| 6868850.0 | 1 |
| 6868859.0 | 4 |
| 6868870.0 | 5 |

Estos campos se cruzaron con la tabla de usuarios realizando un left join por el campo ID cliente. De esta forma a la tabla original de usuarios se le han sumado tres variables más, todas de naturaleza cuantitativa y asociadas al conteo de consumos realizados por cada usuario.

### 5.2.2 Tabla de sesiones

Con relación a la tabla de información que lleva el registro de las visitas de los usuarios al portal de comercio electrónico. En una primera instancia se calcularon dos variables cuantitativas: se contaron los días diferentes que los usuarios visitaron la página (totalDays) y se sumaron el total de sesiones en los diferentes días por cada usuario (totalSesiones). A continuación se muestra un ejemplo de la agregación:

| IDUSUARIO | totalDays | totalSesiones |
|---|---|---|

| | | |
|---|---|---|
| 6868841.0 | 1 | 12 |
| 6868844.0 | 1 | 4 |

Posteriormente se realizó un left join con la tabla usuarios (utilizando el ID usuario como llave) para agregar las dos variables cuantitativas calculadas.

Además de los cálculos mencionados, también se han construído otras variables a partir de la tabla de sesiones. Partiendo de la fecha de las sesiones se han calculado los deltas (en días) entre la primera visita y la última de de cada usuario y por último se guardó la fecha mínima (es decir la fecha de la primera visita) y la fecha máxima (la fecha de la última visita) y se anexaron a la tabla de usuario utilizando el mismo protocolo anteriormente descrito.

### 5.2.3 Tabla de usuarios

La tabla usuario es sobre la que se viene construyendo la tabla final para la fase de modelaje a continuación se describen algunas variables construidas y calculadas a partir de los datos originales.

Se calculó una variable a partir de los deltas entre la fecha de registro (campo original de la tabla de usuarios) y la fecha mínima o fecha de la primera sesión (campo traído desde la tabla de sesiones), dicha diferencia se expresa en días y por lo tanto es una variable cuantitativa. Dicha variable recibió el nombre de "**deltDateRegisP_consumo**"

Se realizó un procedimiento similar pero esta vez tomando la fecha máxima o fecha de la última sesión de cada usuario, generando un campo de iguales características. Se le ha asignado a dicha variable el nombre de "**deltDateRegisU_consumo**".

A partir del dominio del mail se realizó un procedimiento de separar cadenas de caracteres, de esta forma se construyó un campo solo con la última parte del dominio de registro de cada usuario, dicha variable se llamó "**FinalPartDominio**" y como valores de ejemplo tenemos: ".com", ".gov" o ".edu". Desde este campo se realizó un conteo del número de caracteres de cada parte final de los dominios, esperando que los dominios de email reales tengan entre 2 y 3 caracteres, esta variable lleva por nombre "**LenFinalPartDominio**".

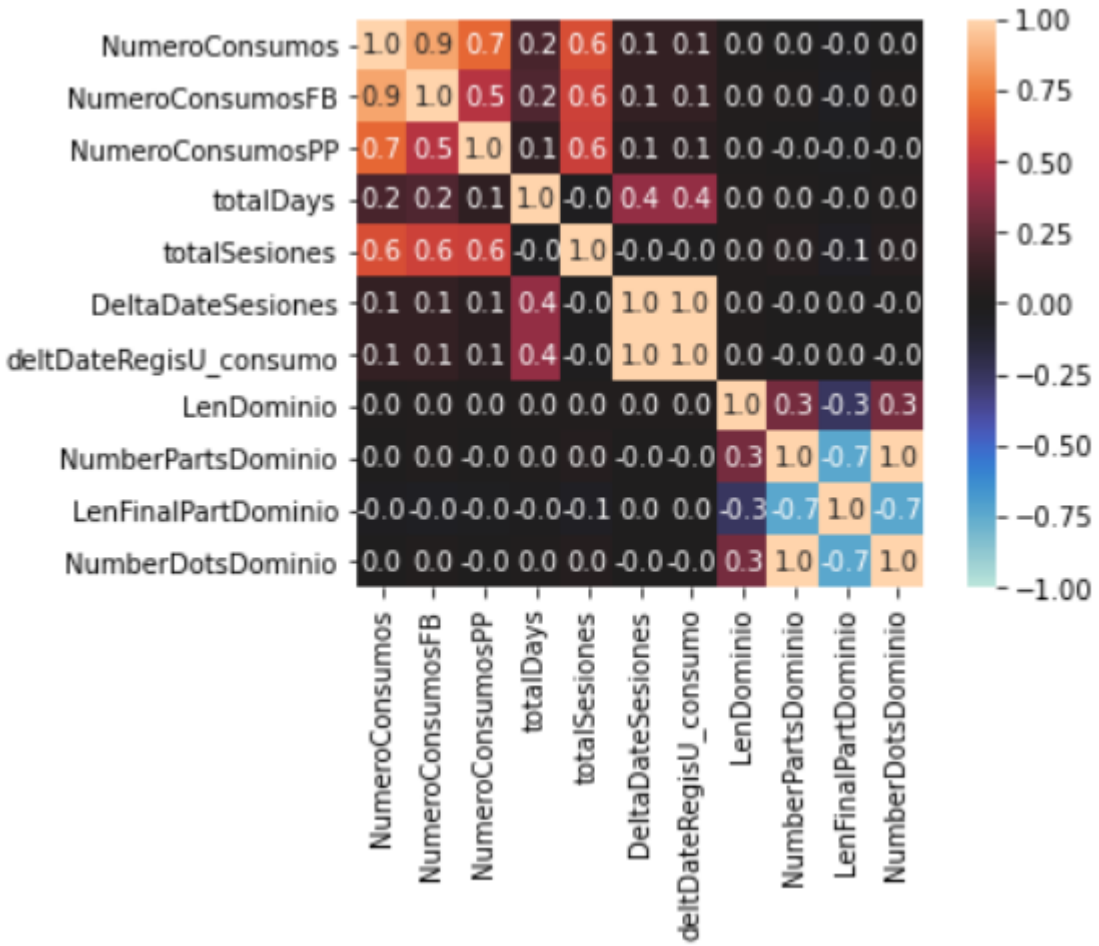
Operando con las cadenas de caracteres se contaron el número de partes que contiene cada dominio de los correos de registro, se contaron el número de veces que el punto "." aparece dentro de cada registro en dicho campo, de esta manera se generó un campo cuantitativo de nombre "**NumberPartsDominio**", de forma similar se contaron el número de puntos "." de cada dominio: "**NumberDotsDominio"**

Las variables categóricas **USU_CIUU**, **USU_ESTADO**, **FEC_CLIENTE** y **USU_DEPARTAMENTO** serán excluidas de los análisis posteriores debido a que presentan más del 80% de los valores pérdidos. Además de lo comentado con anterioridad se realizaron imputaciones sobres los campos "USU_TIPO" y "USU_TAMANIO" asumiendo que los valores ausentes a desconocido y no jurídica. Se observaron valores ausente o perdidos también en los campos de Numero de consumos totales, consumos ficha básica y perfil promocional, estos fueron imputados por cero "0" asumiendo la ausencia de consumos.

Luego de las operaciones comentadas, se realizó un procedimiento de One Hot Encoding sobre las variables categóricas: "TIPOUSUARIO", "CANAL_REGISTRO", "BONDAD_EMAIL", "USU_TIPO", "USU_TAMANIO". De esta manera se arman variables binarias con valores 0 y 1 por cada nivel de cada variable cualitativa mencionada.

## 5.3 Estudio de multicolinealidad

Para determinar multicolinealidad y las altas correlaciones entre predictores (solamente entre variables cuantitativas) se construyó una matriz de correlaciones y posteriormente se calcularon los índices VIF.




Se observan altas correlaciones entre predictores como el número de consumos, números de consumos de ficha básica y el número de consumos de perfil promocional (algo de esperarse teniendo en cuenta que dichas variables representan una combinación lineal). El mismo fenómeno descrito se observa también sobre las variables asociadas al número de partes del dominio y el número de puntos dentro del dominio. Analizando las puntuaciones VIF que se presentan:

| feature | VIF |
|---|---|
| NumeroConsumos | 6.343303 |
| NumeroConsumosFB | 4.550930 |
| NumeroConsumosPP | 2.299398 |

| | |
|---|---|
| totalDays | 1.271264 |
| totalSesiones | 1.824611 |
| DeltaDateSesiones | inf |
| deltDateRegisU_consumo | inf |
| LenDominio | 1.117649 |
| NumberPartsDominio | 1652.441625 |
| LenFinalPartDominio | 2.151788 |
| NumberDotsDominio | 685.587441 |

Se decidió descartar del conjunto de datos final a las variables:

- **NumeroConsumos**
- **DeltaDateSesiones**
- **deltDateRegisU_consumo**
- **NumberPartsDominio**
- **NumberDotsDominio**

Por lo tanto la estructura final de la tabla es la siguiente:

| Column | Non-Nul | Dtype |
|---|---|---|
| IND_CLIENTE (Target) | 142736 | int64 |
| NumeroConsumosFB | 142736 | float64 |
| NumeroConsumosPP | 142736 | float64 |
| totalDays | 142736 | int64 |
| totalSesiones | 142736 | int64 |
| LenDominio | 142736 | int64 |
| LenFinalPartDominio | 142736 | int64 |

| | | |
|---|---|---|
| TIPOUSUARIO_PJ | 142736 | uint8 |
| TIPOUSUARIO_PX | 142736 | uint8 |
| CANAL_REGISTRO_DirectorioEspecializado | 142736 | uint8 |
| CANAL_REGISTRO_DirectorioPopular | 142736 | uint8 |
| CANAL_REGISTRO_SEM | 142736 | uint8 |
| CANAL_REGISTRO_SEO | 142736 | uint8 |
| BONDAD_EMAIL_Error_Temporal | 142736 | uint8 |
| BONDAD_EMAIL_Invalido | 142736 | uint8 |
| BONDAD_EMAIL_NoMail | 142736 | uint8 |
| BONDAD_EMAIL_OK | 142736 | uint8 |
| BONDAD_EMAIL_SPAM | 142736 | uint8 |
| USU_TIPO_EMPRESARIO INDIVIDUAL | 142736 | uint8 |
| USU_TIPO_ENTIDAD EXTRANJERA | 142736 | uint8 |
| USU_TIPO_ENTIDAD FINANCIERA O DE SEGUROS | 142736 | uint8 |
| USU_TIPO_ENTIDAD SIN ANIMO DE LUCRO | 142736 | uint8 |
| USU_TIPO_HOLDING | 142736 | uint8 |
| USU_TIPO_INDUSTRIA / COMERCIO | 142736 | uint8 |
| USU_TIPO_ORGANISMO ESTATAL | 142736 | uint8 |
| USU_TIPO_SOCIEDAD COMERCIAL/INDUSTRIAL | 142736 | uint8 |
| USU_TIPO_SOCIEDAD NO COMERCIAL | 142736 | uint8 |
| USU_TAMANIO_MC | 142736 | uint8 |
| USU_TAMANIO_MD | 142736 | uint8 |
| USU_TAMANIO_NoPJ | 142736 | uint8 |
| USU_TAMANIO_PQ | 142736 | uint8 |
| USU_TAMANIO_SD | 142736 | uint8 |

| | | |
|---|---|---|
| USU_TAMANIO_SD | 142736 | uint8 |

## 5.4 Reducción de dimensionalidad

Como paso previo a la ejecución de los algoritmos de reducción de dimensiones se desordenó el conjunto de datos tomando una muestra aleatoria del mismo tamaño. Luego de lo mencionado, se definió X e Y y utilizando el método fit_transform del MinMaxScaler de Scikit-Learn se pasan a la misma escala todos los valores de los predictores.

Una vez que se ejecutó la transformación comentada, se guardó un fichero csv con la información del conjunto de datos original: 31 variables predictoras, una variable a predecir y 142736 registros.

Se ejecutaron reducciones de dimensiones utilizando el PCA de Scikit Learn, el TSNE de la misma librería y UMAP. Para la reducción por PCA se utilizó como parámetro que se retuviera un 95% de la varianza explicada. En todos los algoritmos se utilizó el parámetro random state a 42 para replicar los nuevos conjuntos de datos. Cada conjunto de datos resultante de ejecutar los algoritmos de reducción de dimensiones fueron guardados en ficheros csv. A continuación se detalla el código:

```
### PCA
pca = PCA(n_components = 0.95,
          random_state=42)
pca.fit(X)
X_PCA = pca.transform(X)

dataPCA = pd.DataFrame(X_PCA)
dataPCA["Y"] = y
dataPCA.to_csv("dataProd/dataPCA.csv",
               index=False, sep="|")

### UMAP
UMAP = umap.UMAP(n_components=2,
                 random_state=42, n_jobs=-1)
X_UMAP = UMAP.fit_transform(X)

dataUMAP = pd.DataFrame(X_UMAP)
dataUMAP["Y"] = y
dataUMAP.to_csv("dataProd/dataUMAP.csv",
               index=False, sep="|")

### TSNE
tsne = TSNE(n_components=2,
            random_state=42, n_jobs=-1)
X_TSNE = tsne.fit_transform(X)

dataTSNE = pd.DataFrame(X_TSNE)
dataTSNE["Y"] = y
dataTSNE.to_csv("dataProd/dataTSNE.csv",
               index=False, sep="|")
```

El resultado de los procesos anteriormente descrito da origen a 4 conjuntos de datos principales:

1. El conjunto de datos original con los 32 predictores y la variable Y-
2. El conjunto de datos reducido por PCA con 15 predictores y la variable Y.
3. El conjunto de datos reducido por UMAP con 2 predictores y la variable Y.
4. El conjunto de datos reducido por TSNE a 2 predictores y la variable Y.

## 5.5 Balanceo del conjunto de datos

Como se ha descrito a lo largo del presente documento, las clases del conjunto de datos (compradores y no compradores) se encuentran desbalanceadas, siendo la clase de interés a predecir, los usuarios que terminaron una compra efectiva; estos representan apenas el 1.8% de los registros. Por lo tanto, como parte de los objetivos de la presente investigación, se probarán diferentes métodos o técnicas para la generación de muestras sintéticas que balanceen las clases antes de la fase de entrenamiento de los modelos.

Para llevar a cabo lo comentado se desarrolló una función que recibe un conjunto de datos y un nombre (cadena de carácteres) como sufijo y devuelve una lista de tuplas con diferentes conjuntos de datos balanceados con muestra sintética (en diferentes proporciones), a continuación se presenta el código de dicha función para luego describirla con profundidad:

```
def getOverSamples(data, sufix):

    """
    """
    X = data.loc[:, data.columns != 'Y']
    y = data['Y']

    ############
    ############ Random Over Sample
    ############

    ### Minority class to 15% on the other class

    ros_15 = RandomOverSampler(random_state=43,
                               sampling_strategy=.15)
    X_ros_15, y_ros_15 = ros_15.fit_resample(X, y)
    data_ros_15 = pd.DataFrame(X_ros_15)
    data_ros_15["Y"] = y_ros_15

    ### Minority class to 33% on the other class

    ros_33 = RandomOverSampler(random_state=43,
                               sampling_strategy=.33)
    X_ros_33, y_ros_33 = ros_33.fit_resample(X, y)
    data_ros_33 = pd.DataFrame(X_ros_33)
    data_ros_33["Y"] = y_ros_33

    ### Equal class
    ros_equal = RandomOverSampler(random_state=43,
                                  sampling_strategy=1.0)
    X_ros_equal, y_ros_equal = ros_equal.fit_resample(X, y)
    data_ros_equal = pd.DataFrame(X_ros_equal)
    data_ros_equal["Y"] = y_ros_equal


    ############
    ############ SMOTE
    ############

    X_smote_15, y_smote_15 = SMOTE(random_state=42,
```

```
41                              sampling_strategy=.15).fit_resample(X, y)
42          data_smote_15 = pd.DataFrame(X_smote_15)
43          data_smote_15["Y"] = y_smote_15
44
45          X_smote_33, y_smote_33 = SMOTE(random_state=42,
46                              sampling_strategy=.33).fit_resample(X, y)
47          data_smote_33 = pd.DataFrame(X_smote_33)
48          data_smote_33["Y"] = y_smote_33
49
50          X_smote_equal, y_smote_equal = SMOTE(random_state=42,
51                                  sampling_strategy=1.0).fit_resample(X, y)
52          data_smote_equal = pd.DataFrame(X_smote_equal)
53          data_smote_equal["Y"] = y_smote_equal
54
55
56          ############
57          ############ ADASYN
58          ############
59
60          X_adasyn_15, y_adasyn_15 = ADASYN(random_state=42,
61                                  sampling_strategy=.15).fit_resample(X, y)
62          data_adasyn_15 = pd.DataFrame(X_adasyn_15)
63          data_adasyn_15["Y"] = y_adasyn_15
64
65          X_adasyn_33, y_adasyn_33 = ADASYN(random_state=42,
66                                  sampling_strategy=.33).fit_resample(X, y)
67          data_adasyn_33 = pd.DataFrame(X_adasyn_33)
68          data_adasyn_33["Y"] = y_adasyn_33
69
70          X_adasyn_equal, y_adasyn_equal = ADASYN(random_state=42,
71                                      sampling_strategy=1.0).fit_resample(X,
72      y)
73          data_adasyn_equal = pd.DataFrame(X_adasyn_equal)
74          data_adasyn_equal["Y"] = y_adasyn_equal
75
76          return [
77                ("ROS 15% {}".format(sufix),data_ros_15),
78                ("ROS 33% {}".format(sufix),data_ros_33),
79                ("ROS Equal Class {}".format(sufix),data_ros_equal),
80
```

```
81              ("SMOTE 15% {}".format(sufix),data_smote_15),
82              ("SMOTE 33% {}".format(sufix),data_smote_33),
83              ("SMOTE Equal Class{}".format(sufix),data_smote_equal),
84
85              ("ADASYN 15% {}".format(sufix),data_adasyn_15),
86              ("ADASYN 33% {}".format(sufix),data_adasyn_33),
87              ("ADASYN Equal Class {}".format(sufix),data_adasyn_equal)
            ]
```

La función en un primer paso recibe como input un conjunto de datos y lo separa en dos objetos: X e Y, una vez hecho esto, sobre la matriz X, realiza varios oversampling con diferentes algoritmos y diferentes parámetros.

Los tres métodos de oversamplig que utiliza la función son el ROS o Random Oversampling, el SMOTE o Synthetic Minority Oversampling Technique y el ADASYN o Adaptive Synthetic; para cada método se ejecutan tres proporciones diferentes de balanceo de las clases: la primera al 15%, es decir la clase minoritaria pasará a representar el 15% de la clase mayoritaria, al 33% es decir la clase minoritaria equivale un 33% de la mayoritaria y por último igualando las clases, es decir misma cantidad de compradores y no compradores. Cada conjunto de datos es reconstruido sumándole el campo Y o target y se agrupan en tuplas dentro de una lista que será el valor de retorno de la función.

La función explicada se aplicó a 4 de los 3 conjuntos de datos principales obtenidos en el proceso de reducción de dimensionalidad:

```
1  dataFrames_PCA = getOverSamples(PCA_data, "PCA")
2  dataFrames_UMAP = getOverSamples(UMAP_data, "UMAP")
3  dataFrames_TSNE = getOverSamples(TSNE_data, "TSNE")
```

El resultado de aplicar la función para cada conjunto de datos principal nos da como resultado un total de 27 conjuntos de datos balanceados de diferentes maneras, sumando el conjunto original, tenemos un total de 28 de conjuntos de datos preparados para la fase de entrenamiento de los diferentes modelos de clasificación:

| Conjunto de Datos | Algoritmo de Balanceo | Proporción de Balanceo | | |
|---|---|---|---|---|
| | | 15% | 33% | 50% |
| PCA | ROS | 1 | 1 | 1 |
| | SMOTE | 1 | 1 | 1 |
| | ADASYN | 1 | 1 | 1 |
| UMAP | ROS | 1 | 1 | 1 |
| | SMOTE | 1 | 1 | 1 |
| | ADASYN | 1 | 1 | 1 |
| TSNE | ROS | 1 | 1 | 1 |
| | SMOTE | 1 | 1 | 1 |
| | ADASYN | 1 | 1 | 1 |
| Original | | | | |

## 5.6 Modelaje estadístico

Para la fase de entrenamiento se utilizaron los siguientes clasificadores:

- Regresión Logística
- Árbol de decisión
- Random Forest
- XGBoost
- AdaBoost

Se definieron los hiperparametros correspondientes a cada uno de los algoritmos dentro de un objeto diccionario de python y se guardaron en una lista. Una vez definidos los posibles parámetros se definieron 4 listas, una de modelos, una con los parámetros de los modelos, una con los nombres de los modelos y otra con los conjuntos de datos.

Una vez definidas las 4 listas, se desarrolló un proceso iterativo en el que se iba recorriendo la lista de conjunto de datos; en cada recorrido de la lista se ejecutaba un segundo loop donde se recorrían las listas de modelos, nombres y parámetros. En este segundo proceso iterativo se hacía una validación cruzada de cada algoritmo probando todos los parámetros correspondientes, además se guardaban los resultados en una tabla y se guardaba el modelo de mayor performance en cada validación cruzada. Por lo tanto por cada conjunto de datos se probó cada algoritmo de clasificación con una validación que probaba diferentes parámetros.

El resultado del loop descrito con anterioridad es un directorio con los modelos guardados, siendo estos los más eficaces en las diferentes métricas de evaluación propuestas, además se construyó una tabla con los resultados finales de la fase de entrenamiento.

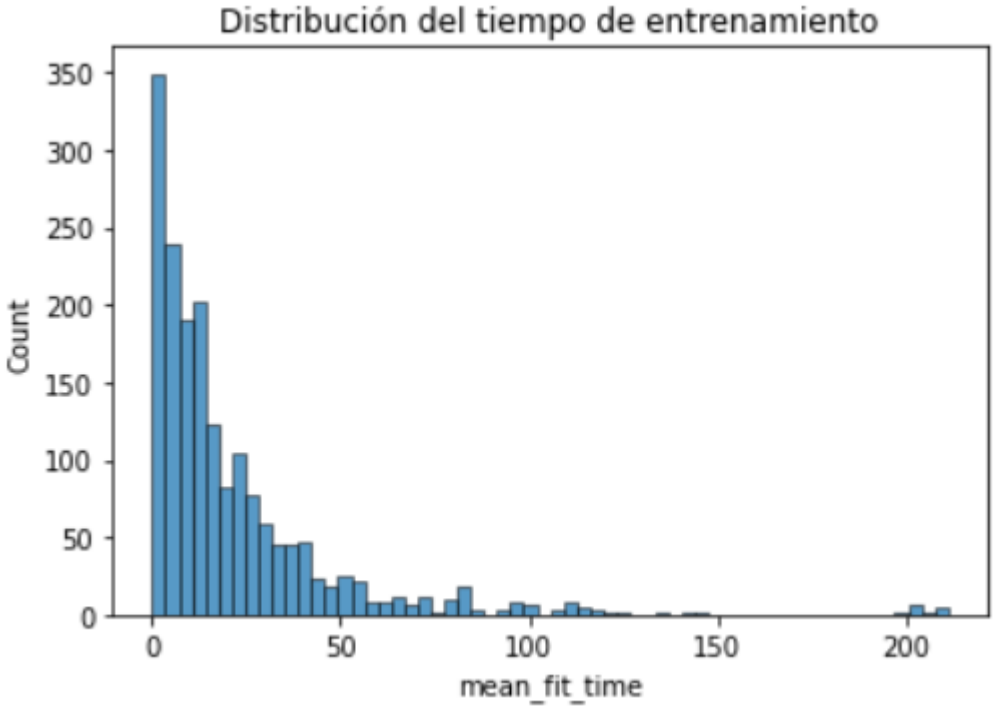
## 5.7 Análisis de resultados del entrenamiento

En total se entrenaron y probaron en la validación cruzada un total de 1792 modelos estadísticos de clasificación binaria, distribuidos de la siguiente manera:

| Model | Count Models |
|---|---|
| AdaBoost Classifier | 168 |
| Decision Tree Classifier | 112 |
| Logistic Regression | 168 |
| Random Forest Classifier | 448 |
| XG Boost Classifier | 896 |

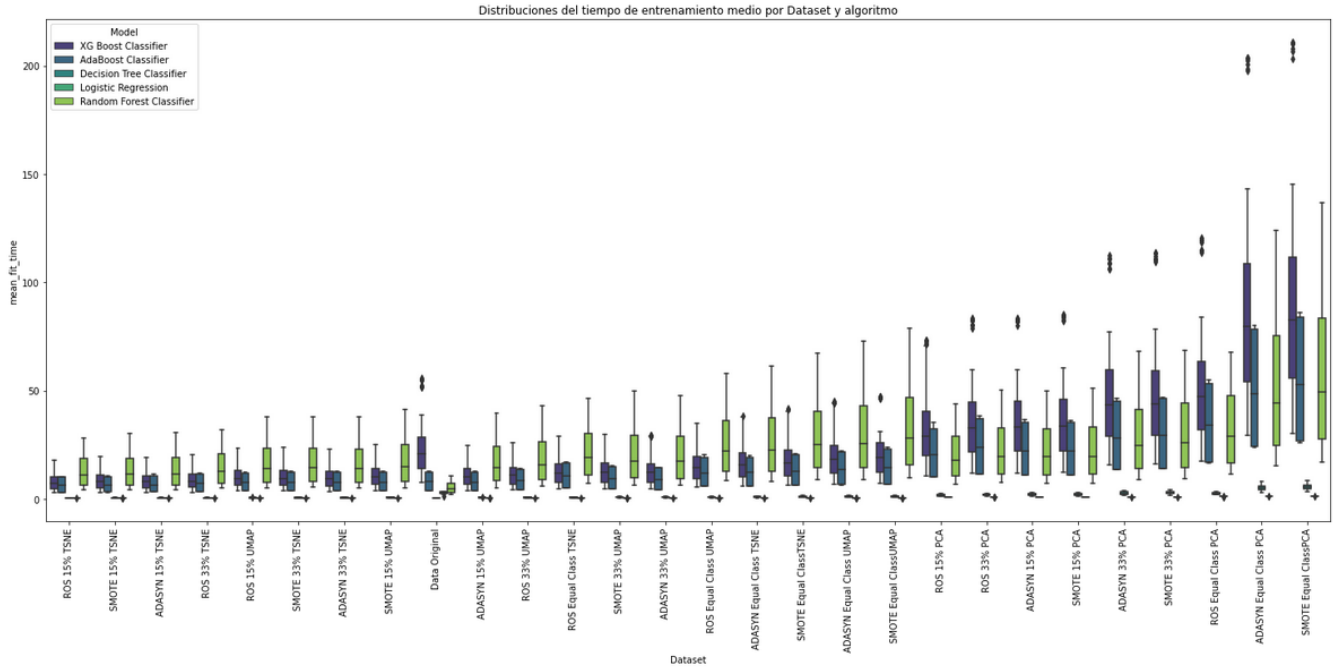
Se observa que los tiempo medios de entrenamiento corresponden a los modelos de regresión logística y los árboles de decisión. Por otra parte los tiempos medios de entrenamiento más altos corresponden a los modelos ensamblados: Random Forest y XG Boost.

A continuación se muestra la distribución de tiempos de entrenamiento teniendo en cuenta a todos los modelos probados en esta fase:




Se observa una distribución coleada hacia el cero, por lo tanto la mayoría de los modelos parece tardar menos de un minuto en el proceso de entrenamiento.

A continuación se comparan las distribuciones por algoritmo y conjunto de datos:



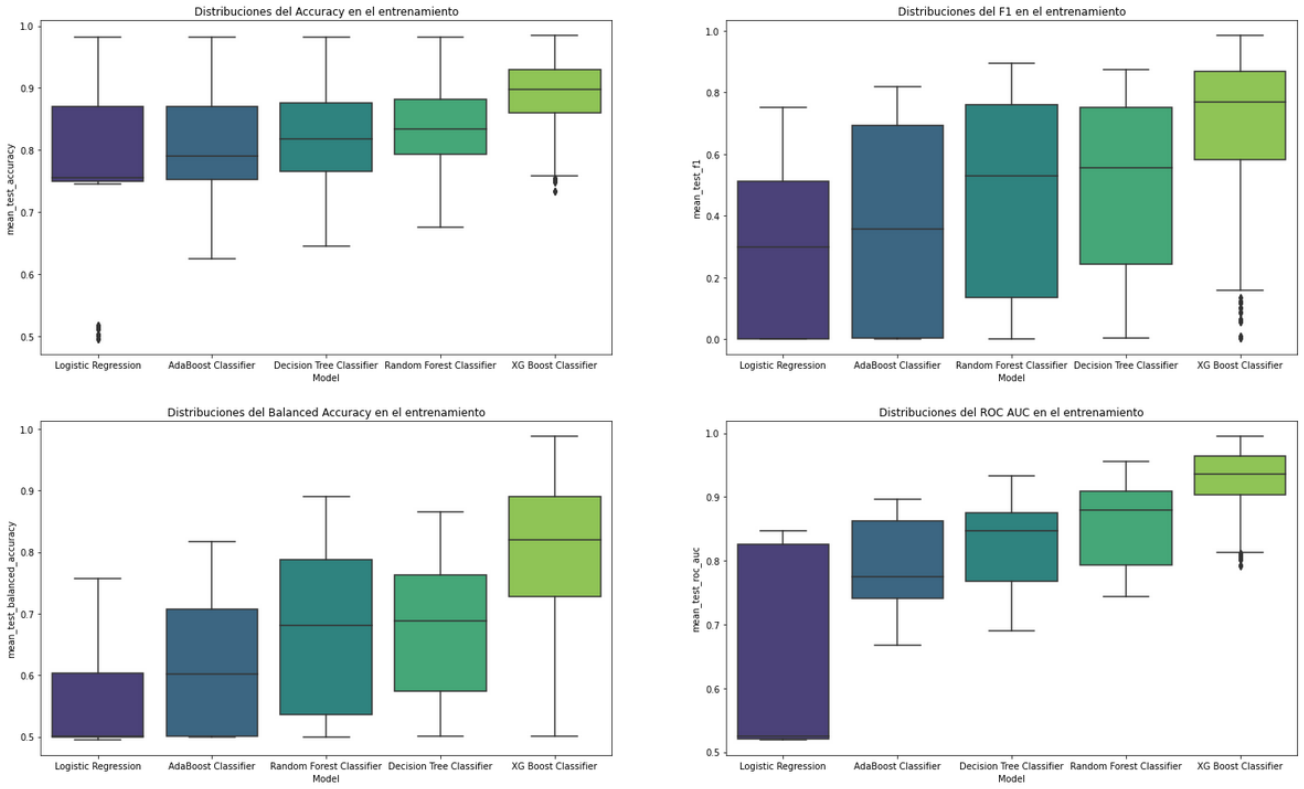

Se observa que las medianas de los tiempos de entrenamiento de los modelos ensamblados son mayores en la medida en la que el conjunto de datos contiene mayor número de muestras sintéticas, es decir en la medida que el conjunto de datos crece. Este efecto es mínimo en el caso de las regresiones logísticas y los árboles de decisión.

Con respecto a las métricas de evaluación de los diferentes modelos, a continuación se presenta una tabla resumen con los promedios:

| Model | Count Models | MeanFit_Time | Mean Accuracy | Mean F1 | Mean Balanced Accuracy | Mean ROCScore |
|---|---|---|---|---|---|---|
| AdaBoost Classifier | 168 | 16.9430 | 0.7992 | 0.3674 | 0.6147 | 0.7920 |
| Decision Tree Classifier | 112 | 1.4296 | 0.8209 | 0.4830 | 0.6669 | 0.8288 |
| Logistic Regression | 168 | 0.6153 | 0.7527 | 0.2789 | 0.5584 | 0.6325 |
| Random Forest Classifier | 448 | 23.9448 | 0.8324 | 0.4567 | 0.6615 | 0.8645 |
| XG Boost Classifier | 896 | 27.9187 | 0.8912 | 0.6949 | 0.7941 | 0.9282 |

Se observa que el promedio de puntajes en el F1 Score más alto, lo tiene la familia de modelos XGBoost, seguido por los árboles de decisión y el Random Forest. El score ROC más alto también lo tiene la familia de modelos XGBoost.

Consistentemente la familia de clasificadores XGBoost mantiene una performance elevada en las diferentes métricas de evaluación, como se demuestra en los siguientes gráficos de caja y bigote:




Los modelos XGBoost presentan una mediana mayor en la fase de entrenamiento en las distribuciones de F1 score, Balanced Accuracy y ROC Score.

## 5.8 Análisis de resultados de la validación

Para el proceso de validación se tomaron solo los modelos de mayor performance en relación al F1 score, dichos modelos fueron guardados como output de cada vez que se

ejecutó la validación cruzada. Se construyó una tabla con la localización del fichero donde fue guardado el modelo, su nombre y el dataset con el que fueron entrenados. A continuación se presenta una muestra de los registros de dicha tabla:

| FileLocation | ModelName | Model Dataset | Model Type |
|---|---|---|---|
| models/XGB_SMOTE 33% TSNE_.joblib | /XGB_SMOTE 33% TSNE | TSNE | XGB |
| models/AdaBoost_ADASYN Equal Class UMAP_.joblib | /AdaBoost_ADASYN Equal Class UMAP | UMAP | AdaBoost |
| models/LogisticRegresion_SMOTE Equal ClassUMAP... | /LogisticRegresion_SMOTE Equal ClassUMAP | UMAP | LogisticRegresion |
| models/LogisticRegresion_ADASYN 15% PCA_.joblib | /LogisticRegresion_ADASYN 15% PCA | PCA | LogisticRegresion |
| models/XGB_SMOTE 33% PCA_.joblib | /XGB_SMOTE 33% PCA | PCA | XGB |

Para la validación de cada modelo final, se tomaron los conjuntos de datos principales: PCA, UMAP y TSNE se diseñó una función que hacía el test de un algoritmo de entrada pasando como argumento la ubicación del fichero y el conjunto de datos utilizado en su entrenamiento. La función se encarga de cargar en memoria el modelo realizar las predicciones y calcular las métricas de evaluación.

Teniendo una lista de los diferentes modelos y los conjuntos de entrenamiento, se ejecutó el un proceso iterativo para calcular las métricas de cada algoritmo con el conjunto de datos sin balancear.

Obteniendo las métricas y sumando dichas listas a la tabla inicial de modelos tenemos la performance de cada algoritmo, a continuación los primeros registros de la tabla de resultados de validación:

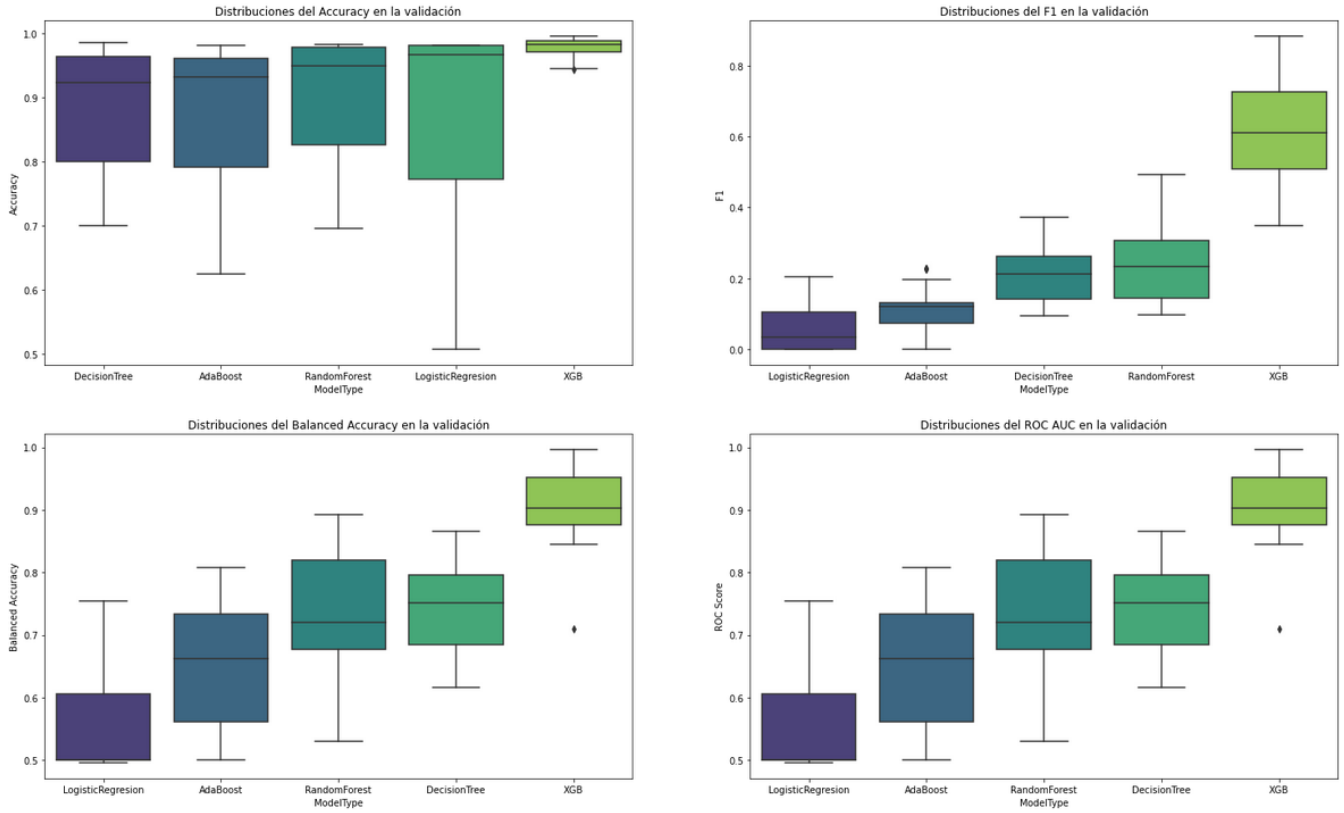
| FileLocation | ModelName | Dataset | ModelType | Accuracy | F1 | ROC Score | Balanced Accuracy |
|---|---|---|---|---|---|---|---|
| models/LogisticRegresion_ROS 15% PCA_.joblib | /LogisticRegresion_ROS 15% PCA | PCA | LogisticRegression | 0.9668 | 0.2020 | 0.6048 | 0.6048 |
| models/DecisionTree_ROS 15% PCA_.joblib | /DecisionTree_ROS 15% PCA | PCA | DecisionTree | 0.9620 | 0.3608 | 0.7772 | 0.7772 |
| models/RandomForest_ROS 15% PCA_.joblib | /RandomForest_ROS 15% PCA | PCA | RandomForest | 0.9815 | 0.4923 | 0.7391 | 0.7391 |
| models/AdaBoost_ROS 15% PCA_.joblib | /AdaBoost_ROS 15% PCA | PCA | AdaBoost | 0.9544 | 0.2245 | 0.6628 | 0.6628 |
| models/LogisticRegresion_ROS 33% PCA_.joblib | /LogisticRegresion_ROS 33% PCA | PCA | LogisticRegression | 0.9271 | 0.1775 | 0.6827 | 0.6827 |

Tal como se comentó en el apartado de entrenamiento, los modelos XGBoost presentan un promedio mayor en el F1 Score en general, así como en el balanced Accuracy y el Score ROC:

| ModelType | Accuracy | F1 | ROC Score | Balanced Accuracy |
|---|---|---|---|---|
| AdaBoost | 0.870670 | 0.107287 | 0.644264 | 0.644264 |
| DecisionTree | 0.880198 | 0.208305 | 0.741728 | 0.741728 |

| | | | | |
|---|---|---|---|---|
| LogisticRegresion | 0.851978 | 0.061906 | 0.560399 | 0.560399 |
| RandomForest | 0.897591 | 0.236654 | 0.732648 | 0.732648 |
| XGB | 0.976089 | 0.597998 | 0.906971 | 0.906971 |

La performance más pobre la tiene la familia de modelos de regresión logística y el AdaBoost en relación a las métricas comentadas. A continuación se presentan las distribuciones de cada métrica evaluada por algoritmo:



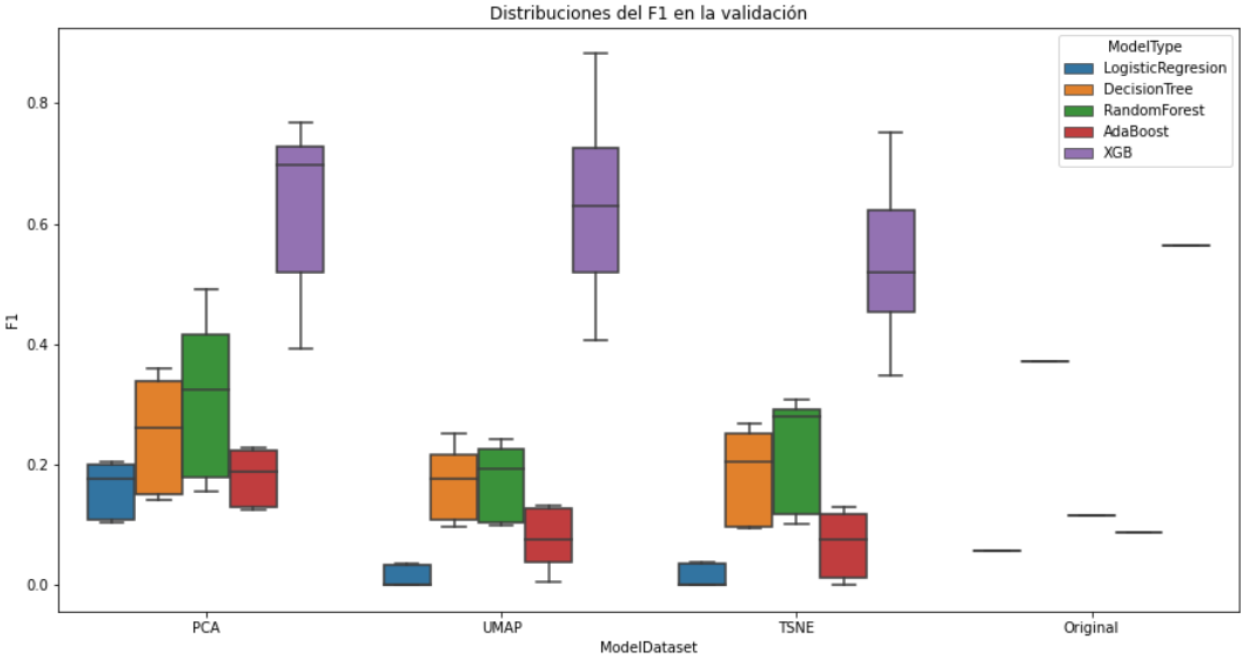

Se observa que hay modelos dentro de la familia XGBoost con una performance elevada siendo testeados con los conjuntos de datos desbalanceados; esto se nota al analizar el percentil 75 de la caja que corresponde al modelo ensamblado en el gráfico de F1 score,

donde se tiene puntajes por encima del 0.80. Estos puntajes elevados están presentes también en el Balanced Accuracy y el ROC Score.

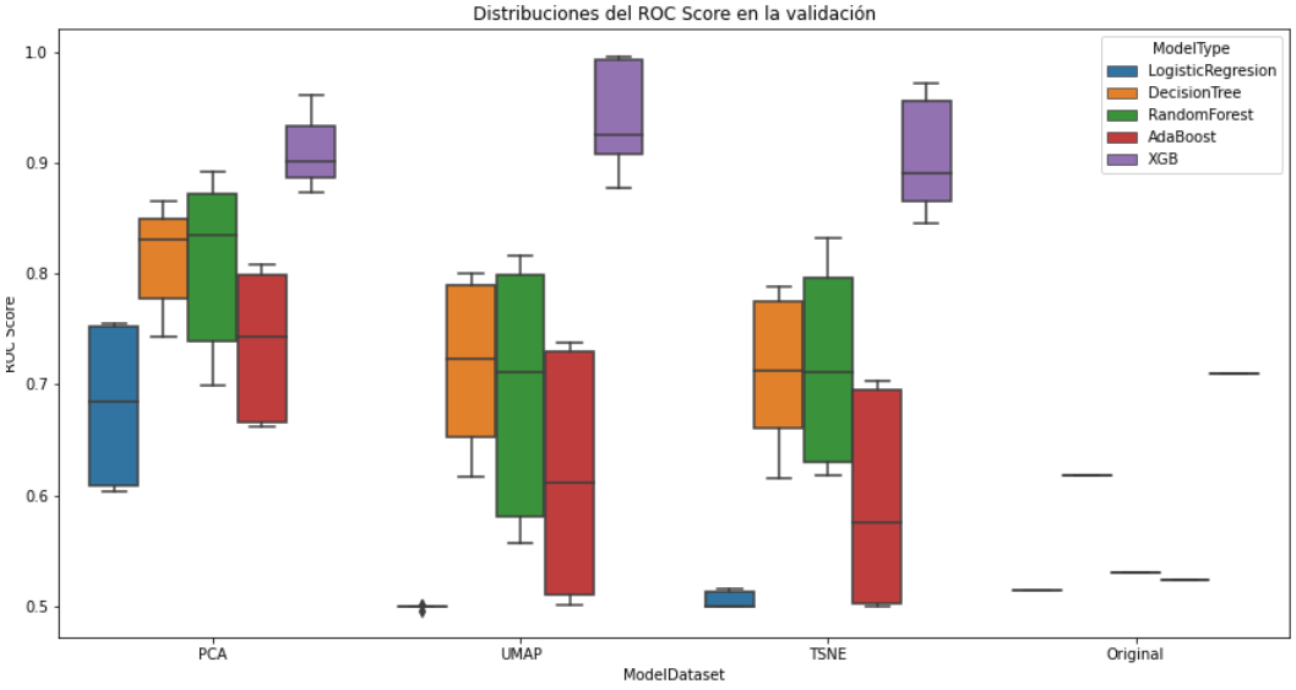
Comparando la performance pero por conjunto de datos tenemos rendimientos similares en promedio:

| ModelDataset | Accuracy | F1 | Balanced Accuracy | ROC Score |
|---|---|---|---|---|
| PCA | 0.913958 | 0.308528 | 0.790139 | 0.790139 |
| Original | 0.983871 | 0.239476 | 0.579586 | 0.579586 |
| UMAP | 0.878826 | 0.211537 | 0.691466 | 0.691466 |
| TSNE | 0.883291 | 0.207554 | 0.685292 | 0.685292 |

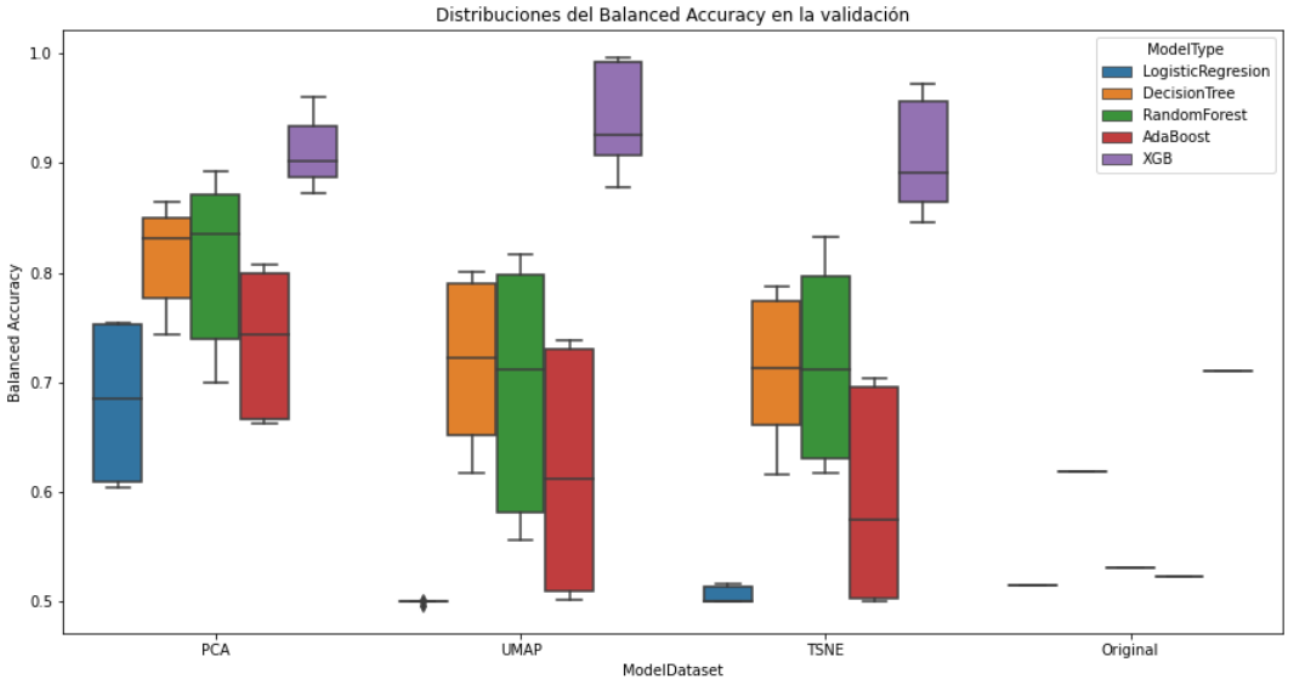
Aquellos modelos entrenados bajo el conjunto de datos que ha sido reducido por PCA parecen en promedio tener una ligera ventaja con respecto a los demás. Sin embargo si analizamos las distribuciones se notan modelos bajo condición de UMAP con excelentes resultados en el ROC score y el Balanced Accuracy:




La mediana más alta en este caso se encuentra en los conjuntos de datos con PCA.



En el caso del ROC Score y el Balanced Accuracy se nota que la performance de los modelos XGBoost bajo condición de entrenamiento UMAP presenta una mediana superior.



## 5.9 Evaluación de modelos finales

A partir de la tabla de modelos finales construida en el apartado anterior se presenta el top 10 de modelos, ordenando la tabla de mayor a menor a partir del puntaje en el F1, ROC Score y Balanced Accuracy:

| ModelName | Dataset | Model Type | Accuracy | F1 | ROC Score | Balanced Accuracy |
|---|---|---|---|---|---|---|
| /XGB_ROS 15% UMAP | UMAP | XGB | 0.9951 | 0.8824 | 0.9930 | 0.9930 |
| /XGB_ROS 33% UMAP | UMAP | XGB | 0.9927 | 0.8354 | 0.9963 | 0.9963 |
| /XGB_SMOTE 15% PCA | PCA | XGB | 0.9916 | 0.7673 | 0.8732 | 0.8732 |
| /XGB_ADASYN 15% PCA | PCA | XGB | 0.9913 | 0.7600 | 0.8736 | 0.8736 |
| /XGB_ROS 15% TSNE | TSNE | XGB | 0.9888 | 0.7524 | 0.9562 | 0.9562 |
| /XGB_ADASYN 33% PCA | PCA | XGB | 0.9893 | 0.728 | 0.8874 | 0.8874 |
| /XGB_SMOTE 33% PCA | PCA | XGB | 0.9892 | 0.7281 | 0.8888 | 0.8888 |
| /XGB_ROS Equal Class UMAP | UMAP | XGB | 0.9861 | 0.7251 | 0.9929 | 0.9929 |
| /XGB_ROS 15% PCA | PCA | XGB | 0.9860 | 0.6975 | 0.9334 | 0.9334 |

| | | | | | | |
|---|---|---|---|---|---|---|
| /XGB_ROS 33% TSNE | TSNE | XGB | 0.9828 | 0.6700 | 0.9676 | 0.9676 |

Se escogieron los dos primeros modelos de la tabla ordenada, el primero fue entrenado con el conjunto de datos UMAP con la clase minoritaria representando un 15% de la mayoritaria y el segundo con un 33%, ambos con el Random Over Sampling.

Los parámetros del modelo final 1 y el modelo final 2 son los mismos:

```
{'objective': 'binary:logistic',
 'base_score': 0.5,
 'booster': 'gbtree',
 'colsample_bylevel': 1,
 'colsample_bynode': 1,
 'colsample_bytree': 1,
 'eval_metric': None,
 'gamma': 0,
 'gpu_id': -1,
 'grow_policy': 'depthwise',
 'interaction_constraints': '',
 'learning_rate': 1.0,
 'max_bin': 256,
 'max_cat_to_onehot': 4,
 'max_delta_step': 0,
 'max_depth': 10,
 'max_leaves': 0,
 'min_child_weight': 1,
 'monotone_constraints': '()',
 'n_jobs': -1,
 'num_parallel_tree': 1,
 'predictor': 'auto',
 'random_state': 0,
 'reg_alpha': 0,
 'reg_lambda': 1,
 'sampling_method': 'uniform',
 'scale_pos_weight': 1,
 'subsample': 1,
 'tree_method': 'exact',
```

'validate_parameters': 1,
'verbosity': 0,
'criterion': 'gini'}

Analizando el reporte de clasificación del mode final 1:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 | 140121 |
| 1 | 0.80 | 0.99 | 0.88 | 2615 |
| accuracy | | | 1.00 | 142736 |
| macro avg | 0.90 | 0.99 | 0.94 | 142736 |
| weighted avg | 1.00 | 1.00 | 1.00 | 142736 |

Se observa un accuracy elevado, asociado al overfitting y a las clases desbalanceadas, sin embargo también se nota que el modelo es preciso detectando a los compradores (0.80) y mantiene un F1 score elevado de 0.88.

La matriz de confusión de este primer modelo nos indica que el algoritmo tiende a sobreestimar las compras, clasificando usuarios que no terminaron comprando:

| | **No Compra** | **Compra** |
|---|---|---|
| **No Compra** | 139455 | 666 |
| **Compra** | 24 | 2591 |

Se observa un comportamiento similar en el modelo final 2 en el reporte de clasificación, sin embargo es ligeramente menos preciso al momento de detectar los compradores
:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 0.99 | 1.00 | 140121 |
| 1 | 0.72 | 1.00 | 0.84 | 2615 |
| accuracy | | | 0.99 | 142736 |
| macro avg | 0.86 | 1.00 | 0.92 | 142736 |
| weighted avg | 0.99 | 0.99 | 0.99 | 142736 |

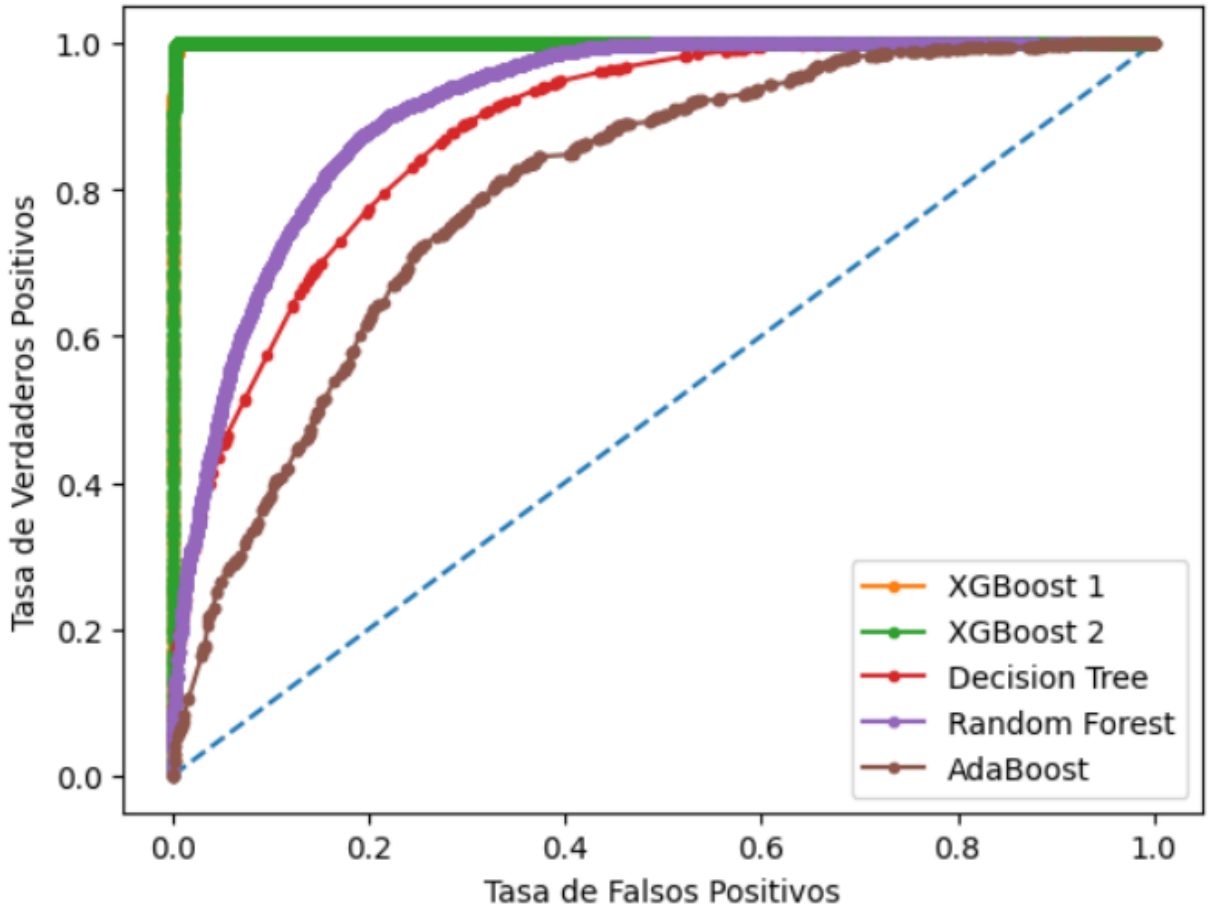
En la matriz de confusión se analiza que el modelo ha detectado todos los casos de compra sin embargo unos 1030 registros de no compradores fueron erróneamente clasificados:

| | **No Compra** | **Compra** |
|---|---|---|
| **No Compra** | 139091 | 1030 |
| **Compra** | 0 | 2615 |

Se cargaron algunos modelos entrenados bajo el mismo conjunto de datos: un árbol de decisión con UMAP y ROS al 15%, un random forest con ROS al 33% y UMAP y un AdaBoost al 33% y UMAP, se presentan las curvas ROC contra los modelos finales descritos con anterioridad:

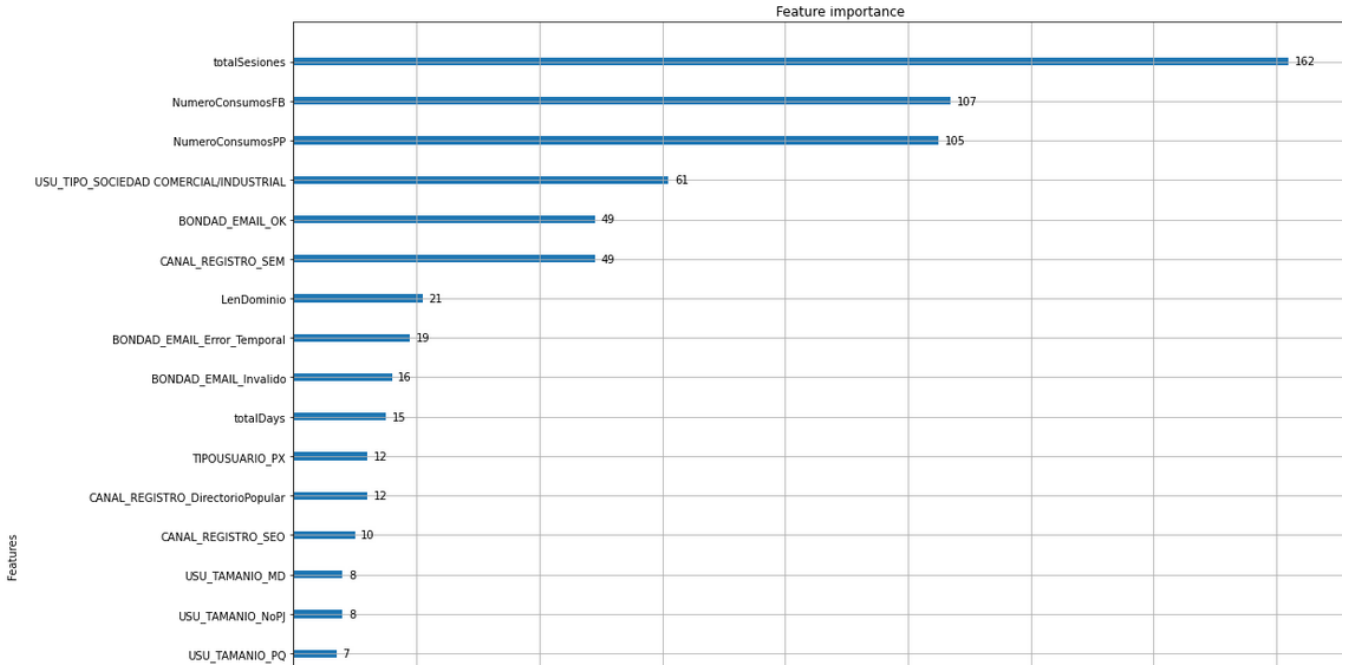
```
XGBoost 1: ROC AUC = 1.000
XGBoost 2: ROC AUC = 0.999
Decision Tree: ROC AUC = 0.881
Random Forest: ROC AUC = 0.915
AdaBoost: ROC AUC = 0.800
```

```
]: <matplotlib.legend.Legend at 0x7ffb04fc74d0>
```




## 5.10 Importancia de los predictores

Para entender la importancia de las variables se cargó en memoria el conjunto de datos original y se procedió a entrenar un modelo de XGBoost sin parámetros, a continuación se muestra un gráfico con la importancia que tiene cada predictor:




Los puntajes más altos de acuerdo al método **plot_importance** de la librería XGBoost nos indica que el número de sesiones, los consumos promocionales previos y si el usuario representa una sociedad comercial / industrial y por lo tanto una persona jurídica, son las variables que tienen un mayor peso para determinar a un posible cliente comprador.

# 6 - Conclusiones

El objetivo principal de la investigación fue construir un modelo por medio de la comparación de múltiples algoritmos que permita la correcta clasificación entre usuarios compradores y no compradores. Luego de un extenso ejercicio que implicó la comparación de algoritmos, transformaciones en el conjunto de datos y diferente procedimientos de oversampling, para la clasificación correcta de usuarios; se pueden establecer el siguiente grupo de conclusiones:

**Algoritmos**

Se observó un rendimiento pobre en algunos algoritmos como la regresión logística, sin embargo esperando que los modelos ensamblados tuviesen mayor robustez, también se vieron resultados pobres en el AdaBoost. Con respecto a este último modelo mencionado, se ha

notado que el algoritmo es altamente sensible a los problemas de clasificación desbalanceados y que la performance es mayor cuando las clases se encuentran igualadas en el conjunto de datos original.

Las familias de modelos que mostraron mayor performance en el proceso de validación fueron los árboles de decisión, el random forest y el XGBoost. Los primeros dos, en las distribuciones de los puntajes de F1, ROC Score y Balanced Accuracy; mantienen una mediana cercana. El XGBoost fue la familia de modelos más adecuada para el problema manteniendo puntajes F1 elevados en comparación al resto.

En el top 10 de modelos de mayor performance son todos XGBoost con conjuntos de datos reducidos por UMAP, PCA o TSNE; de esta lista mencionada, el de mayores prestaciones fue entrenado con un conjunto de datos reducido con UMAP y con oversampling ROS al 15%y tuvo como resultados de evaluación los siguientes:

- F1: 0.8824
- ROC Score: 0.9930
- Balanced Accuracy: 0.9930

Por otra parte, si bien el XGBoost fue el modelo con mayor performance en cada métrica de evaluación, es importante destacar que este es costoso, ejecutar una validación cruzada con la cantidad de parámetros que admite el algoritmo puede incluso llevar a un tiempo de entrenamiento por encima de los 30 min (en una validación cruzada), si a eso le sumamos el hecho de que se utilizaron 28 conjuntos de datos diferentes, el experimento se termina convirtiendo en algo computacionalmente muy costoso. Por lo tanto se recomienda utilizar una random search de parámetros para mitigar tal costo en investigaciones futuras.

**Transformaciones**

En la investigación se utilizaron principalmente 3 métodos para la reducción de dimensiones: PCA, UMAP y TSNE. Es importante destacar que la utilización de estos procedimientos puede implicar un importante ahorro en el costo computacional, debido a que la información es resumida y el volumen de datos a procesar es menor. A lo mencionado con

anterioridad es importante destacar que los menores tiempos de entrenamientos sobre conjuntos de datos reducidos con respecto al original.

Con respecto a la efectividad de los conjuntos de datos y algoritmos, el XGBoost mantiene un rendimiento similar en cada uno de los conjuntos reducido con las diferentes técnicas. En el caso de los árboles de decisión y los Random Forest, el PCA les favorece teniendo una mediana ligeramente más alta de F1 Score.

Por último, en relación a la reducción de dimensiones, es importante que el problema de dicha aproximación, es que las variables dejan de ser interpretables y por lo tanto no se termina de entender qué predictores originales mantienen una influencia sobre la compra en el comercio electrónico.

**Oversampling**

En este punto se observó sistemáticamente que al igualar las muestras el rendimiento del algoritmo es peor, es evidente que el overfitting es un problema común también en la clasificación binaria desbalanceada. Al utilizar los métodos de oversampling de forma que la clase minoritaria represente un 15% o un 33% con respecto a la clase mayoritaria, se nota una mayor performance de los modelos en general.

La aproximación más realista de lo anteriormente comentado, sería utilizando el 15%, debido a que el número de muestras sintéticas creadas será menor y se mantendrá cierto desbalance que ocurre dentro los datos del mundo real. El algoritmo más simple ROS, mantuvo los resultados más elevados utilizando el XGBoost.

**Variable predictoras**

Por último y en relación con los predictores, tomando en cuenta los puntajes que se asignan dentro del XGBoost; tres variables son fundamentales para predecir la compra por parte de un usuario: el número total de sesiones (a mayor número de sesiones probablemente se muestre un mayor interés sobre los servicios ofrecidos), el número de consumos promocionales del tipo Ficha Básica y Perfil Promocional que el usuario ha hecho, esto último tiene también sentido debido a que el usuario ya consume la información y por lo tanto está dispuesto a pagar por ella.

Otras variables que se muestra como predictores importantes: el usuario debería representar una sociedad comercial o industrial, el mail debe ser de un dominio válido y el registro ocurrió desde el SEM (este canal es más importante en la predicción que el SEO y por lo tanto vale la pena pagar para optimizarlo).

# 7 - Código Fuente

El código fuente de todo el proyecto, así como también las tablas de resultados pueden ser consultadas en el siguiente enlace que corresponde a un repositorio de github:

https://github.com/Vic16/TFM_PurchasePrediction

# 8 - Bibliografía

Albon, C. (2018). *Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning*. O'Reilly Media.

Bigon, L., Cassani, G., Greco, C., Lacasa, L., Pavoni, M., Polonioli, A., & Tagliabue, J. (2019). Prediction is very hard, especially about conversion. 10.48550/arXiv.1907.00400

Bingbing, X., Jitao, Z., & Xianyi, W. (2021). A prediction model of user buying behavior based on LSTM and SVM. *6th International Symposium on Computer and Information Processing Technology*, 26-31. 10.1109/ISCIPT53667.2021.00013

Bruce, P., Bruce, P. C., Gedeck, P., & Bruce, A. (2020). *Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python*. O'Reilly.

Cao, W., Wang, K., Gan, H., & Yang, M. (2021). User online purchase behavior prediction based on fusion model of CatBoost and Logit. *Journal of Physics: Conference Series*, *2003*(1). https://iopscience.iop.org/article/10.1088/1742-6596/2003/1/012011/meta

Chaudhuri, N., Gupta, G., Vamsi, V., & Bose, I. (2021). On the platform but will they buy? Predicting customers' purchase behavior using deep learning. *Decision Support*

*Systems*, *149*. https://www.sciencedirect.com/science/article/abs/pii/S0167923621001329

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321-357. https://www.jair.org/index.php/jair/article/view/10302/24590

Dou, X. (2020). Online Purchase Behavior Prediction and Analysis Using Ensemble Learning. *IEEE 5th International conference on cloud computing and big data analytics*. 10.1109/ICCCBDA49378.2020.9095554

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems* (2da ed.). O'Reilly.

He, H., Bai, Y., Garcia, E., & Li, S. (2008). ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. 10.1109/IJCNN.2008.4633969

Igual, L., Pujol, O., Seguí, S., Escalera, S., Dantí, F., Garrido, L., & Puertas, E. (2017). *Introduction to Data Science: A Python Approach to Concepts, Techniques and Applications*. Springer International Publishing.

Jia, R., Li, R., Yu, M., & Wang, S. (2017). E-commerce Purchase Prediction Approach By User Behavior Data. *international conference on computer, information and telecommunication systems*.

Lee, J., Jung, O., Lee, Y., Kim, O., & Park, C. (2021). A Comparison and Interpretation of Machine Learning Algorithm for the Prediction of Online Purchase Conversion. *Journal of Theoretical and Applied Eletronic Commerce Research*, *16*. https://doi.org/10.3390/jtaer16050083

Lemaítre, G., Nogueira, F., & Aridas, C. (2017). *Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning*. https://imbalanced-learn.org/stable/index.html

McInnes, L., Healy, J., & Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. https://arxiv.org/abs/1802.03426

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

Niu, X., Li, C., & Yu, X. (2017). Predictive Analytics of E-Commerce Search Behavior for Conversion. https://aisel.aisnet.org/amcis2017/DataScience/Presentations/7

Porcelli, A. M. (2020). La inteligencia artificial y la robótica: sus dilemas sociales, éticos y jurídicos. *Derecho Global. Estudios sobre Derecho y Justicia*, *VI*, 49-105. https://doi.org/10.32870/dgedj.v6i16.286

Rojo Muñoz, S. (2022). *Predicción de potenciales compradores en un ecommerce* [white paper para el TFM].

Suchacka, G., Skolimowska-Kulig, M., & Potempa, A. (2015). A k-Nearest Neighbors Method for Classifying User Sessions in E-Commerce Scenario. *Journal of Telecommunications and Information Technology*, *3*, 64-69. https://yadda.icm.edu.pl/baztech/element/bwmeta1.element.baztech-40e29335-8f5f-4d8c-aa93-8c13a90d1b2d

Vanderplas, J. T., & VanderPlas, J. (2016). *Python Data Science Handbook: Essential Tools for Working with Data*. O'Reilly Media, Incorporated.

Wu, H., & Li, B. (2022). Customer Purchase Prediction Based on Improved Gradient Boosting Decision Tree Algorithm. *International Conference on Consumer Electronics and Computer Engineering*, 795-798. 10.1109/ICCECE54139.2022.9712779