UNIVERSIDAD TORCUATO DI TELLA

Master in Management + Analytics

Modelos predictivos de vendedores fraudulentos en plataformas de venta Online

Alumna: Mia Molina Abeniacar
Director: Agustín Gravano, PhD

Mayo, 2021

Resumen

Existe poca bibliografía que analice en profundidad la casuística de fraude en el mercado online cuando el fraudulento es el usuario vendedor. Durante el año 2020 y con una pandemia afectando al mundo entero, las formas de consumir y relacionarse entre la población se vio fuertemente alterada. Esto provocó un crecimiento exponencial de la digitalización de los pagos en muchas partes del mundo, tomando cada vez más protagonismo la experiencia de los usuarios a la hora de realizar pagos online. De esta manera, en esta tesis se aborda el problema de la detección de fraude en vendedores de venta online, y se logra plantear un modelo de aprendizaje automático, con información de los usuarios vendedores de una empresa que se dedica a proveer servicios como plataforma de pagos online. En este trabajo se describe el tipo de procesamiento de los datos los cuales se llevó a cabo para armar la base de datos. También se plantea una metodología para abordar el problema planteado que brinda resultados prometedores los cuales serán discutidos más adelante. A su vez, se entrenan varios modelos con el objetivo de buscar aquel que tenga un mejor poder predictivo. Finalmente se discuten cuáles son las recomendaciones prácticas para llevar a cabo lo propuesto en el presente trabajo.

Abstract

There is little bibliography that analyzes in depth the casuistry of seller fraud in the online market. During the year 2020 and with the entire world being affected by the pandemic, the ways of consuming and interacting among the population was strongly disturbed. This caused an exponential growth in the digitization of payments in many parts of the world, with the user experience as one of the main aspects to look after in online payments. In this sense, this thesis presents the problem of fraud detection in online sales by identifying the fraudulent sellers. This is done by training different machine learning models, with information from sellers of a company that works providing its services as a platform for online payments. In this work we introduce the type of data processing which was carried out to build the final database. We also present the methodology proposed to address the problem raised, which results in promising results which will be discussed later. In turn, several models are trained in order to find the one with the best predictive power. Finally, we discuss practical recommendations to carry out what is proposed in this thesis.

0. Índice

1. Introducción

Con el avance inminente de la tecnología como parte de nuestra vida diaria, las compras a través de internet crecen a pasos agigantados. La participación del comercio electrónico en el mercado de retail global está mostrando un aumento constante a lo largo de los años, lo que indica un cambio evidente en la preferencia del consumidor de las ventas online por sobre las presenciales.

Si miramos los números en cantidad de usuarios, Brasil es el mercado online más grande de Latino América, y el cuarto respecto al mundial.

De acuerdo a datos publicados por Statista.com, Brasil representa alrededor del 42% de todo el comercio B2C en América Latina. Para el año 2019 se estimó que aproximadamente 31.4 millones de personas compraron al menos una vez de manera online. Eso es más del 40% respecto a los 22.3 millones que se registraron el año anterior.

El método de pago más utilizado en el mercado online de Brasil son las tarjetas, representando un 59% de las transacciones y acumulando $14 miles de millones en ventas (Satista.com, 2019). Este método de pago se estima que pierda una porción de mercado respecto a otros métodos (Billeteras Digitales, Créditos, Digital Currency etc) para el 2021, disminuyendo a casi 47%. Aún así los pagos online con tarjetas siguen liderando la lista de métodos de pago.

En el presente trabajo estaremos enfocándonos en el mercado de Brasil, específicamente en el mercado de transacciones online realizadas. Nuestra base de datos fue construida con información entre los períodos Febrero a Mayo de 2021.

1.1 Problema

El 2020 fue un año en que la pandemia global Covid 19 afectó los patrones de consumo de un modo inesperado. Para Abril de dicho año, alrededor de un tercio de la población mundial, se encontraba en confinamiento en sus hogares preservándose del fatal virus (Ravelin.com, 2020).

El incremento de las compras en el ecommerce generó un crecimiento tanto en las transacciones de usuarios buenos, como así también la de aquellos usuarios fraudulentos.

La experiencia del usuario en la compra o venta es un factor muy importante a la hora de generar engagement en los clientes. Es por esto que cada vez más las empresas que proveen servicios de procesamiento de pagos online se preocupan por generar una experiencia libre de fraude. De esta manera, ¿Cómo podemos

predecir si los vendedores dentro de una plataforma de pagos son usuarios fraudulentos o no, y de esta manera mejorar la experiencia de los usuarios compradores?

En el mundo del ecommerce existen distintos tipos de fraude. Si bien en el mercado existen diferentes herramientas para detectar y frenarlo, en el presente trabajo estaremos trabajando en entender las características y detectar específicamente un patrón de fraude conocido como Seller Fraud (Fraude vendedor).

1.2 Distintos tipos de fraude en el Ecommerce

El fraude en el mundo de las transacciones online crece y se perfecciona a medida que las barreras de seguridad son cada vez más altas y sofisticadas. Existen distintas maneras de hacer fraude a la hora de comprar o vender un producto en internet, y los usuarios fraudulentos se dedican a encontrar cualquier hueco que el sistema tenga para poder explotarlo y buscar aprovecharse del mismo. A medida que el mundo de pagos online crece, los usuarios fraudulentos crecen a la par.

A continuación se presentan los tipos de fraude más comunes dentro del mundo de las transacciones online, vale aclarar que además de los que se exponen a continuación existen muchísimos otros perfiles fraudulentos tipificados.

1.2.1. Identidad Falsa o Robo de Identidad

El Robo de identidad ocurre cuando el estafador usa información personal de una víctima, sin ningún tipo de aprobación por esta parte, para realizar una acción criminal o para engañar a otra persona.

Identidad falsa se da en aquellas situaciones en la que el estafador crea una identidad falsa para fines criminales.

En ambos casos los usuarios fraudulentos se hacen de esta identidad para hacerse pasar por otra persona ilegítimamente.

1.2.2. Fraude Amistoso o Contracargo

El fraude amistoso puede adoptar distintas formas, pero típicamente involucra un comprador legítimo que compra un producto o servicio y luego reclama que no realizó la compra, o que solamente recibió una parte del pedido, para que de esta manera se pueda quedar con las mercancías sin necesariamente haber pagado por las mismas.

Los compradores que realizan este tipo de fraude hacen las compras con tarjetas de crédito y una vez recibido el producto presentan un reclamo por pérdida o bien un

contracargo (desconocimiento del pago) a través del banco, con la intención de recibir un reembolso por la compra.

1.2.3. Fraude Tarjeta

El fraude tarjeta refiere generalmente a cualquier transacción fraudulenta que busque hacerse de dinero. La transacción fraudulenta puede realizarse para hacerse de bienes o servicios o para obtener fondos ilegalmente de una cuenta. Este tipo de fraude puede suceder simultáneamente al robo de identidad, pero también puede ocurrir en usuarios legítimos que realizan la transacción sin intención de pagar el producto o servicio.

La información de las tarjetas pueden ser adquiridas de distintas maneras, ya sea información comprada en la Dark Web, o información robada por la modificación o alteración de Cajeros Automáticos entre otros.

1.2.4. Testeadores de Tarjetas

Esta es la práctica de probar la validez de un dato de tarjeta de crédito en un sitio con planes de usar esa misma información en otro sitio web para cometer un fraude. El objetivo del estafador en este caso no es hacerse de bienes o servicios, si no de acceder a información legítima de datos de las tarjetas de crédito. Esto normalmente se utiliza para poder monetizar esta información luego.

En la práctica este tipo de fraude se suele hacer con bots en donde se automatizan los intentos de pagos y recopilan la información de aquellas transacciones que pudieron llevarse a cabo o que quedaron aprobadas.

1.2.5. Robo de cuenta

El robo de cuenta también conocido como ATO (Account Takeover) es una forma de robo de identidad en el que el fraudulento gana control de la cuenta del consumidor. Al hacerse de esta información el fraudulento puede tener acceso a información sensible (contraseñas, datos personales, etc.) y también contar con acceso a los fondos de las cuentas.

1.2.6. Phishing y Pharming

Ambos tipos de fraude son métodos para engañar al usuario a que acceda a sitios web falsos para que exponga su información personal. Phishing hace referencia a cuando el usuario es engañado para que entregue información sensible a los defraudadores. Pharming por el otro lado ocurre cuando por ejemplo, modifican las entradas DNS haciendo creer a los usuarios que están siendo redirigidos a una página web que en realidad fue creada para robar datos.

1.2.7. Fraude Vendedor

A su vez, también existe fraude por parte de los vendedores. Hay distintas características del fraude vendedor, aquellas que normalmente suelen ocurrir en la práctica pueden resumirse de la siguiente manera:

- Cuando un vendedor crea una página web y comienza a vender productos normalmente por debajo del precio del mercado, llamando así la atención de los compradores. Los compradores se ven atraídos por los precios y el vendedor comienza a tener muchas ventas. En este caso el vendedor nunca entrega los productos y desaparece con el dinero.

- El segundo caso de fraude vendedor es cuando hay un cambio en el comportamiento del mismo. El vendedor no necesariamente empieza siendo fraudulento desde el principio. En este caso el vendedor comienza a transaccionar y entregar productos en tiempo y forma, generando así un perfil de vendedor sano y seguro. En la práctica se suele observar un quiebre en el comportamiento, que se da normalmente cuando los vendedores comienzan a tener aumentos fuertes en las ventas de un momento a otro. Al tener una buena reputación como vendedor, cuentan con mayor ventaja que aquellos vendedores nuevos, por presentar más seguridad y entregas de ventas pasadas.

1.3. Justificación

A medida que el mundo digital y la cantidad de compradores online crece, también crecen los usuarios fraudulentos. Para apostar y seguir creciendo en el mundo de las transacciones online es importante ofrecer un ecosistema de pagos seguros a los usuarios, y que de esta manera ayude a impulsar y acompañar las tendencias que promueven un aumento de las compras online. Muchas veces las transacciones en internet generan incertidumbre y desconfianza por el poco conocimiento o inseguridad de los usuarios. Cada vez es más importante ofrecer un ecosistema seguro para generar engagement en los usuarios y así promover el crecimiento de cada plataforma. La prevención del fraude en el mundo online genera un valor agregado a la experiencia del usuario tanto comprador como vendedor, imprescindible para que una plataforma pueda seguir creciendo.

A la hora de investigar, pudimos observar que la gran mayoría de la literatura está dirigida a prevenir y detectar el fraude que proviene por el lado del comprador, pero existe muy poca información disponible que trabaje la identificación del fraude vendedor en el ecosistema online.

Con el presente trabajo buscaremos ampliar el conocimiento referido a la identificación y prevención de fraude vendedor en el mercado de Brasil para las transacciones online realizadas con tarjetas.

1.4. Objetivo

Al crecer el mundo de transacciones online, los usuarios se encuentran cada vez más expuestos a distintos riesgos a la hora de realizar una compra. El objetivo del presente trabajo es desarrollar un modelo que prediga si el usuario vendedor es un usuario fraudulento. Para esto se utilizarán técnicas de machine learning con el objetivo de predecir y automatizar la detección del mismo.

El modelo que entrenaremos buscará predecir la probabilidad de que un usuario sea efectivamente un vendedor fraudulento. Al trabajar con la probabilidad de fraude, esto nos permitirá trabajar en el armado de cortes de riesgo y distintas variables para poder obtener valores de recall y precisión altos en nuestra predicción.

2. Datos y Metodología

La información utilizada para el presente trabajo fue provista por una empresa Latinoamericana dedicada a proveer servicios como el procesamiento de pagos online entre sus principales actividades. Por cuestiones de confidencialidad, la información fue anonimizada para proteger la información personal de los usuarios.

2.1. ETL

Para obtener la información que se utilizó en el presente trabajo, se recurrió a implementar un proceso de ETL para el armado del dataset. ETL hace referencia a un proceso de 3 pasos también conocido como Extract, Transform y Load.

1. Extract: la información es consultada y extraída del sistema de origen.
2. Transform: la información es limpiada, normalizada y procesada para adaptarse al sistema de destino.
3. Load: la información ya transformada es almacenada en el sistema de destino.

2.1.1. Extract

Si bien la empresa que nos compartió la información ya tiene incorporado un proceso de extracción de datos y transformación para que la información consultada a la base de datos mismo ya se encuentre estandarizada, la extracción utilizada para este trabajo consistió en la lectura de información proveniente de distintas bases de datos para el armado del dataset final.

La información de las distintas bases de datos fue extraída mediante queries en SQL.

Para armar el dataset de usuarios fraudulentos lo que hicimos fue definir el periodo dentro del cual se identificó al vendedor como fraude. Los datos extraídos traen información de 293293 usuarios en Brasil. Para armar el dataset final, lo que hicimos fue traer la información de las actividades del vendedor en la plataforma las 8 semanas previas a que se identificó como fraudulento.

2.1.2. Transform

Para la transformación de los datos trabajamos en Python. La información de SQL fue extraída y almacenada en distintos csv para poder agruparlos y limpiarlos en Python y poder armar los dataset finales. En el proceso de transformación de los datos se procedió a anonimizar los datos de usuarios para preservar la

confidencialidad de su información personal y se procedió a limpiar los datos para poder ser guardados todos dentro de un mismo dataset. El proceso de limpieza consistió en la eliminación de columnas repetidas y limpieza de cualquier tipo de información que sea identificable del usuario. El output de la transformación fueron los tres datasets finales que se procedieron a cargar y trabajar posteriormente.

2.1.3. Load

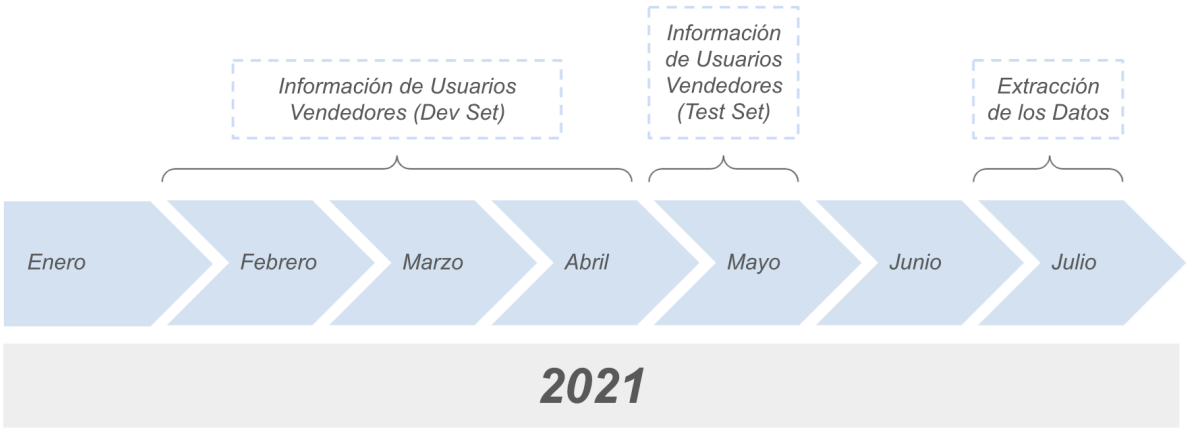
La carga de los datos para el posterior análisis se realizó desde R.

2.2. Descripción de los Datasets utilizados

En el presente trabajo contamos con 2 conjuntos de datos:

- Dev Set: La misma contiene datos de las ventas y performance de los usuarios durante 8 semanas. El Dev Set cuenta con información de vendedores identificados como fraudulentos en los meses de Febrero, Marzo y Abril del año 2021. A su vez, para la marca de usuarios no fraudulentos, se tomó en consideración aquellos usuarios vendedores que operaron al mismo tiempo que aquellos fraudulentos, pero que a la fecha de la extracción de los datos (Julio 2021) no fueron considerados Seller Fraud.
- Test Set: contiene el mismo tipo de información de vendedores para el mes de Mayo.

Gráfico 1: Cronología de la extracción de los Datos



Entender la información y saber leerla nos permite brindarle valor agregado e intuición a los modelos de inteligencia artificial.

Los datasets contienen información de las siguientes características del problema representadas por variables:

- Ventas realizadas
- Reclamos recibidos
- Contracargos notificados
- Periodicidad de retiros de dinero
- Antigüedad del vendedor en la plataforma
- Niveles de Aprobación - Rechazo Bancario - Rechazo por el equipo de Prevención de Fraude
- Tipos de integración utilizada para conectarse con la empresa para poder transaccionar
- Rubro de la empresa
- Precios promedios de los productos

2.3. Exploración de los Datos

La exploración de los datos es un paso fundamental en el proceso de entendimiento de los datos para armar un buen modelo predictivo de Machine Learning. El objetivo principal del análisis exploratorio en Machine Learning es ayudar a analizar los datos antes de hacer cualquier tipo de suposición. Esto puede ayudar a identificar errores obvios, como así también comprender mejor los patrones dentro de los datos, también permite detectar valores atípicos o encontrar relaciones interesantes entre las variables. El análisis exploratorio sirve también para garantizar que los resultados obtenidos en los entrenamientos de modelos sean válidos y puedan ser aplicables a los objetivos planteados desde las áreas comerciales o de negocio en la industria. El análisis exploratorio es un paso fundamental que permite obtener conocimientos sobre los datos, para luego usarse en un análisis o modelado de datos más sofisticado.

Como primer paso en la exploración de datos, se realizóun chequeo de características de las variables, las cuales se pueden describir a continuación

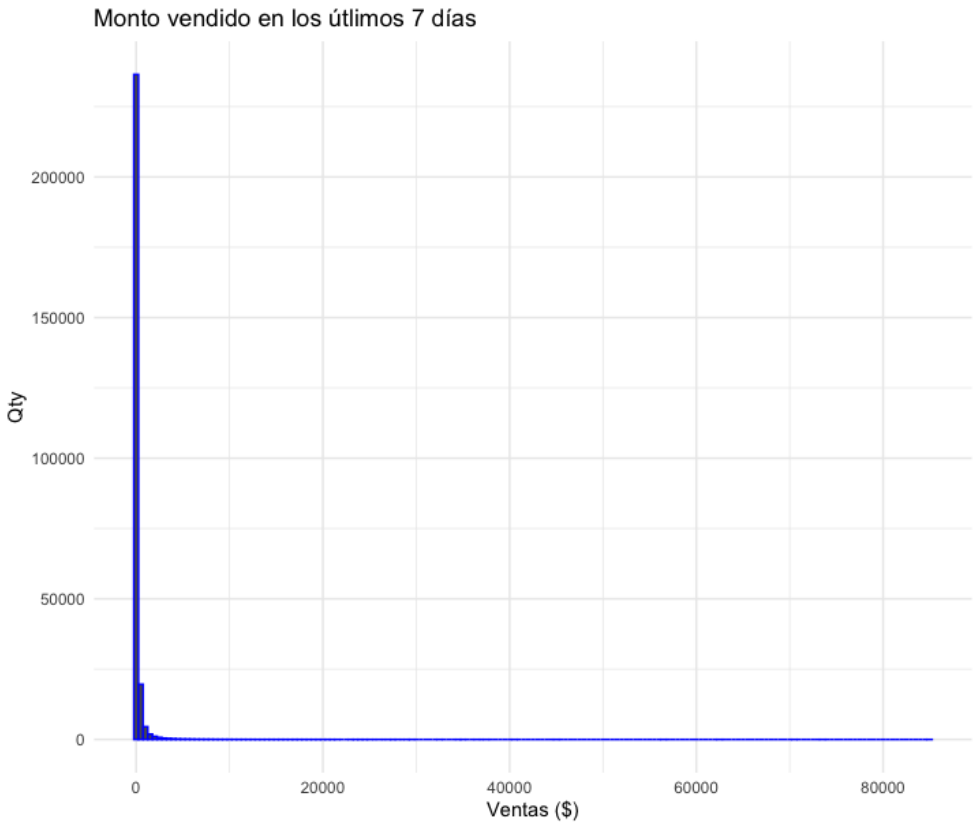
- Clases de Variables

  A la hora de entender el dataset lo primero que hicimos fue entender el tipo de variables que teníamos. El dataset contiene un total de 153 variables que explican las características del problema presentadas en el párrafo anterior, dentro de las cuales en su gran mayoría son variables numéricas a excepción de la variable *is_fraude* (booleano - variable a predecir), *site_id* (string), *industry_id* (string) y *fecha_de_registracion* (IDate).

- Variables con distribución asimétrica

  En aquellas variables con distribución asimétrica lo que suele pasar es encontrar datos atípicos que son numéricamente distantes del resto de los datos. Resulta muy importante tener identificados estos valores ya que existen algoritmos de Machine Learning y métricas estadísticas que resultan muy sensibles frente a estos valores. En nuestro dataset pudimos observar que contábamos con muchas de nuestras variables con distribución asimétrica. Por ejemplo, en el gráfico 1 se muestra la distribución de la variable tpv_7d, la cual mira el monto vendido por los usuarios en los últimos 7 días.

Gráfico 2 - Ejemplo de Variable con distribución asimétrica



- Valores Faltantes

  La variable *cus_industry_id*, la cual hace referencia al rubro en el que se desempeña la cuenta vendedora, fue una de las variables con mayor porcentaje de valores faltantes (98%). El motivo que justifica esta falta de completitud, es que el rubro es una variable autodeclarada por cada usuario a la hora de dar de alta su cuenta vendedora, cabe aclarar que en el proceso de alta de usuario la declaración del rubro de la empresa no es un dato obligatorio.

  A su vez, las variables de reclamos y tickets también presentaron varios datos faltantes. Una posible explicación detrás de este tipo de datos faltantes se puede dar en aquellos casos donde los vendedores no hayan tenido ventas por ticket o ningún tipo de reclamos sobre sus ventas. Si bien esto no parecería ser un problema de conceptualización de los datos, los valores inexistentes plantean un problema a la hora de entrenar un modelo. Más adelante, en la sección de Ingeniería de Variables se procede a explicar qué fue lo que se hizo para trabajar este problema.

- Variables Constantes

  Las variables constantes también representan una contradicción a la hora de entrenar modelos, ya que no aportan ningún tipo de variabilidad o tendencia sobre la variable a predecir. En nuestro dataset la única variable que se identificó que resulta constante es el *site_id.* Dado que todos los datos son de vendedores de Brasil, esto explica que resulte una variable constante, por lo cual se descartó del dataset.

- Alto porcentaje de 0’s

  Otro aspecto que nos es útil tener en cuenta, son aquellas variables con niveles altos de observaciones iguales a 0. En nuestro caso, las variables con mayor porcentaje de 0 se concentran en los niveles de contracargos y reclamos, lo cual tiene sentido ya que nuestra gran mayoría de observaciones son de vendedores sanos, que se podría decir que sus niveles de fraude reportado son menores.

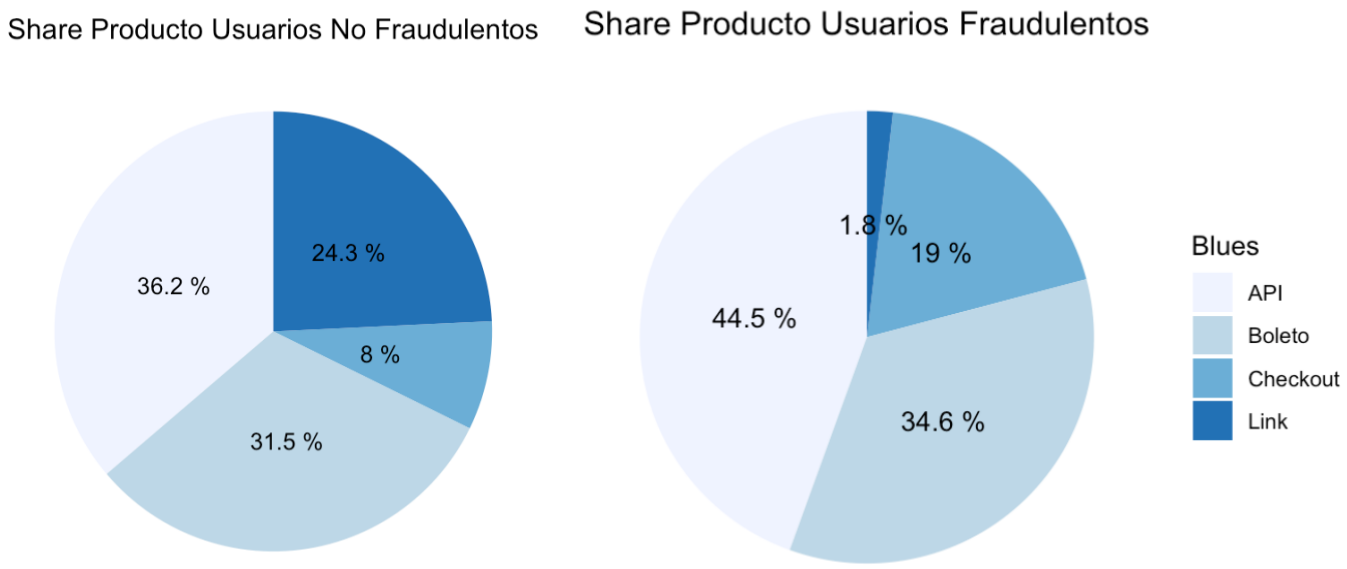
A continuación mostraremos el análisis de cómo interactúan y correlacionan distintas variables, las cuales en base al tipo de fraude con el que estamos trabajando creemos que serán aquellas con mayor poder predictivo.

<u>Ventas según tipo de Integración :</u>

Existen distintas maneras en las cuales los vendedores pueden integrarse con la plataforma para procesar pagos:

- API: en estos casos la conexión con el servicio de procesamiento de pagos se hace a través de una API.
- Checkout: Los vendedores se integran a través de una solución ofrecida por el mismo prestador del servicio del procesamiento de pagos, en donde a la hora de que el comprador quiere pagar, lo redirige a un checkout de la empresa que provee el servicio y realiza el pago allí.
- Link: es una solución mayormente usada cuando el vendedor no cuenta con una página web. En este caso el vendedor crea un link y se lo envía al comprador con el monto que debe pagar.
- Boleto: son pagos realizados con efectivo desde un canal de cobranza extrabancario.

Gráfico 3: Participación por tipo de integración



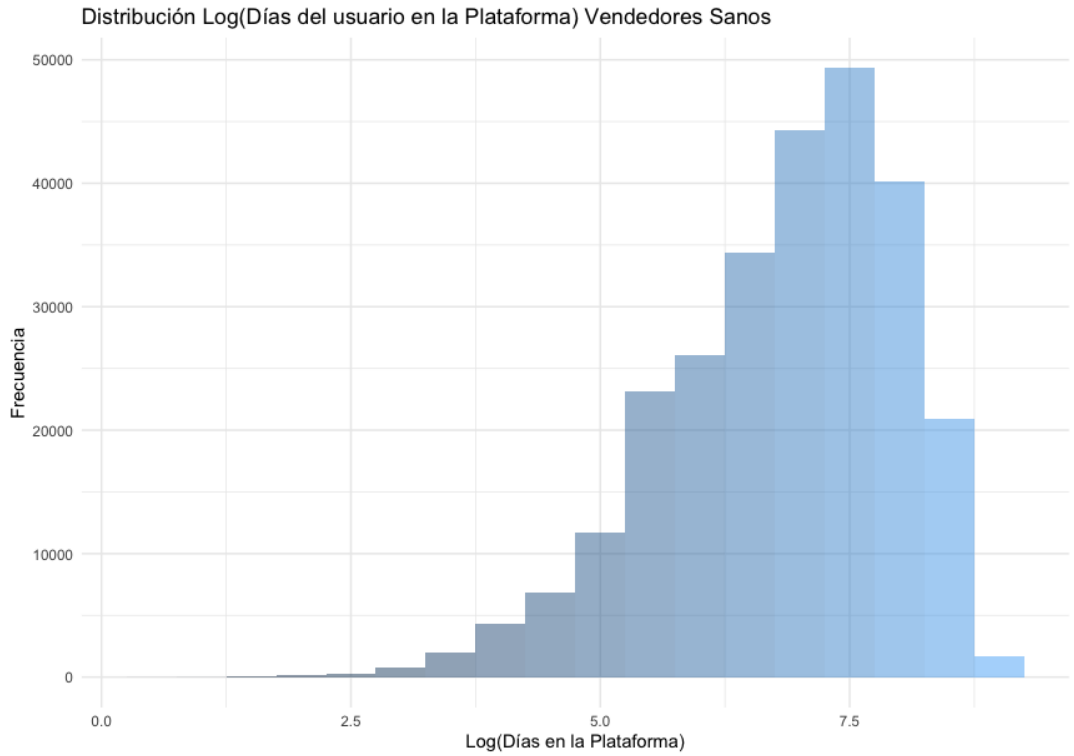
En el gráfico 3 se muestra como se componen las ventas de los usuarios respecto de los distintos tipos de integración que existe en la plataforma. Lo que podemos observar en el gráfico 3 es que en aquellos usuarios sanos demuestran estar homogéneamente distribuidos entre los distintos productos, a excepción de Checkout quien tiene un porcentaje menor. Por el contrario, en el caso de aquellos usuarios fraudulentos, existe una gran participación de ventas realizadas a través de

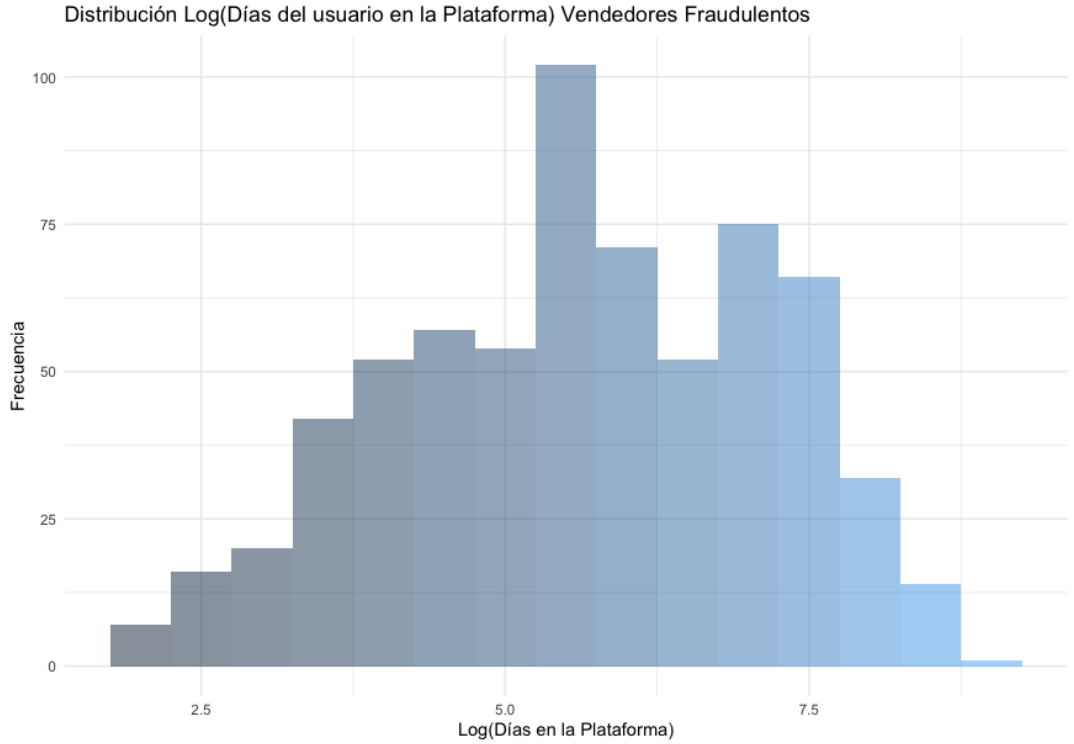
API y de Boleto. Entender a través de qué tipo de integración el usuario está vendiendo nos puede dar un mayor entendimiento de la operatoria de ventas. Vale destacar que los pagos a través de boleto, por ser un tipo de pago presencial, no permite la posibilidad para el comprador de realizar un contracargo. Es por eso que en el análisis de detección de fraude resulta importante ver las métricas y notificaciones no solo de los contracargos recibidos, sino también de los reclamos que lleguen a la plataforma sobre los vendedores.

Edad de usuarios en la Plataforma:

En el gráfico 4 se muestra la distribución del logaritmo de la edad del usuario en la plataforma. Como podemos ver en el gráfico 4, la edad de los usuarios sanos en la plataforma es bastante mayor que aquellos de los usuarios fraudulentos. Esto se puede explicar ya que entendemos que aquellos usuarios con mayor trayectoria e historial en la plataforma tienen menor probabilidad de intentar hacer fraude, comparado a aquellos que son nuevos y comienzan a hacer fraude desde el comienzo. Esto no significa que siempre se cumpla que los usuarios con más antigüedad en la plataforma sean completamente seguros, es menos probable que un vendedor sea fraudulento y que cuente con un historial sano de ventas en la plataforma.

Gráfico 4: Edad de usuarios en la Plataforma



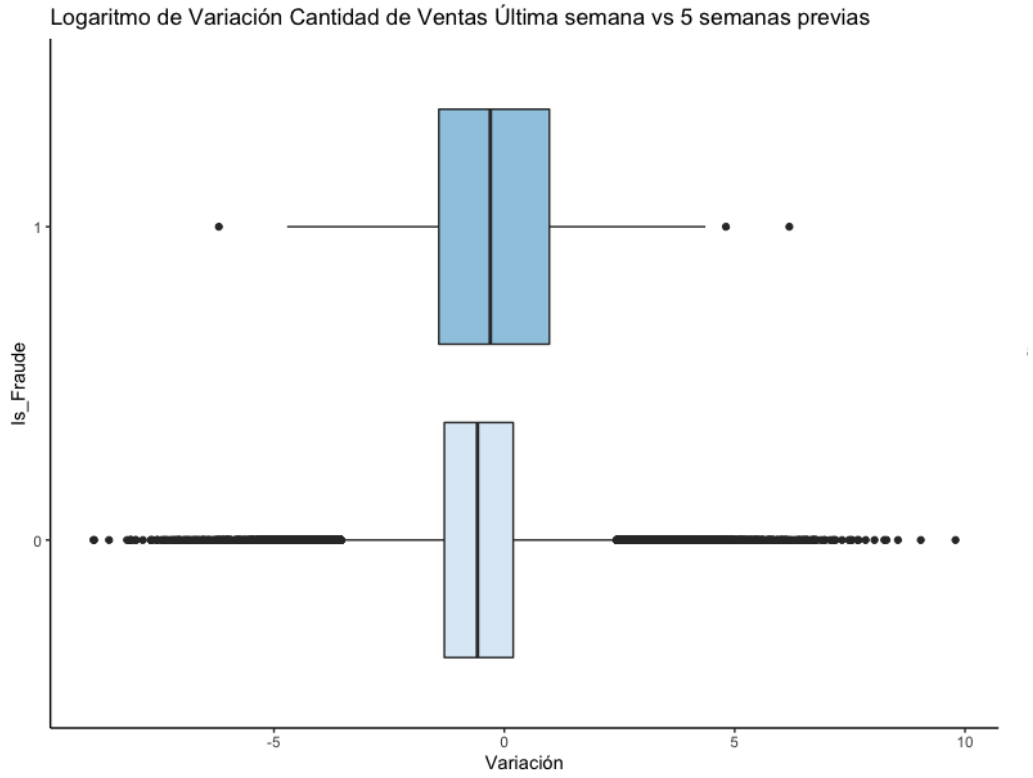


La edad de los vendedores resulta ser una variable que permite segmentar con un mayor nivel de eficiencia, el fraude vendedor. Como vemos en los gráficos anteriores, la edad del vendedor en la plataforma resulta ser una variable que permite distinguir con mayor probabilidad a usuarios fraudulentos cuando estos son nuevos o recientes en la plataforma. Un ejemplo de esto resulta cuando los vendedores fraudulentos a medida que se le van bloqueando las cuentas, proceden a crear nuevos usuarios para seguir operando hasta que sean bloqueados nuevamente. La falta de información e historial de ventas que evidencie la entrega de productos por parte de los vendedores resulta en un mayor riesgo de encontrarse frente a una situación de estafa.

Variación de ventas:

En el gráfico 6 se muestran un boxplot que presenta cómo varía el logaritmo de la cantidad de ventas realizadas de los usuarios fraudulentos y no fraudulentos la última semana respecto a 5 semanas antes.

Gráfico 5: Variación de Ventas



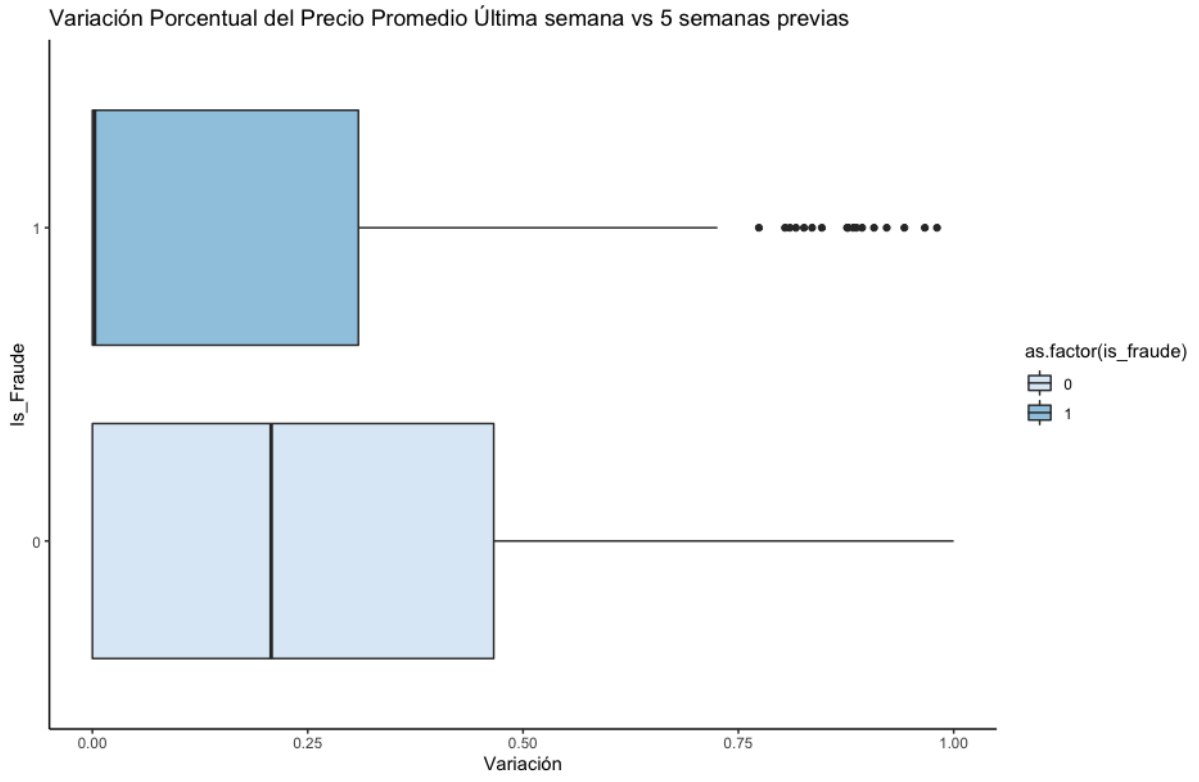
Como se planteó en el capítulo 1 de este trabajo, una de las características que se han observado en el pasado y que puede llegar a explicar el comportamiento

fraudulento en los vendedores, es un aumento en la cantidad de ventas realizadas. A diferencia de lo que podríamos haber supuesto, la variación en la cantidad de ventas que tienen los vendedores, no aumenta en el tiempo. Si bien existen valores por fuera del rango con un 95% de confianza, estadísticamente hablando hay una mayor concentración de todos los usuarios en un nivel de variación 0%. Es decir, que la gran mayoría de los usuarios fraudulentos, a diferencia de lo que hubiéramos pensado, no presenta diferencias en los niveles de ventas registrados en el periodo analizado.

Variación en el precio de venta promedio:

En el gráfico 7 se muestra un boxplot en el cual se puede observar como varía el precio promedio de los vendedores fraudulentos y no fraudulentos la última semana respecto a 5 semanas antes.

Gráfico 6: Variación de precio promedio



Como se mencionó en el párrafo anterior, otra de las características que pueden darse en este tipo de fraude es que los vendedores fraudulentos ofrezcan precios por debajo de la competencia. Si bien acá no podemos comparar los precios con respecto a la competencia porque no contamos con la información del tipo de

producto que vende cada usuario, no se observa que los vendedores adopten una estrategia de baja de precios en el tiempo, es decir los usuarios mantienen los precios estables. Al igual que sucedió con la variación de las ventas, no se observaron variaciones en los niveles de precio promedio ofrecidos por los usuarios fraudulentos con respecto a aquellos usuarios sanos.

2.4. Metodología

Métodos de *Resampling*

Los métodos de resampling son una herramienta indispensable a la hora de entrenar modelos de Machine Learning. Los métodos de resampling son herramientas muy útiles con múltiples beneficios en Machine Learning para lograr obtener modelos más precisos, en la selección de modelos finales y para la optimización de parámetros de dichos modelos. Estos consisten en seleccionar distintas muestras de un training set y entrenar el modelo de interés con cada muestra para obtener mayor precisión sobre los datos desconocidos.

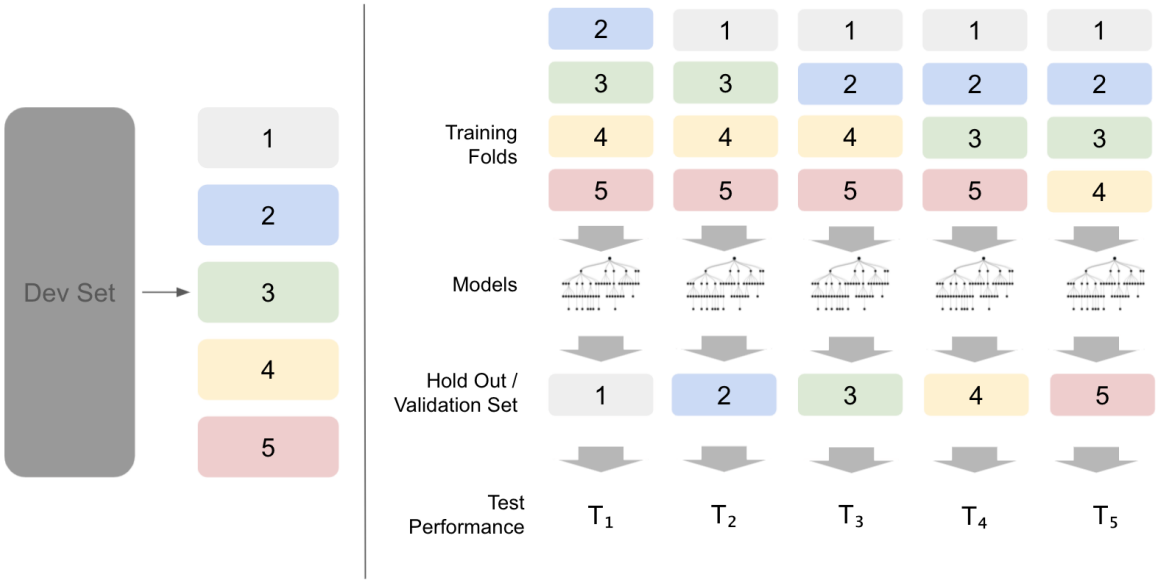
En el presente trabajo desarrollamos el enfoque de Cross Validation con k folds. Dicho enfoque trabaja con un parámetro k que refiere a la cantidad de grupos en los cuales se va a dividir aleatoriamente los datos. "*This approach involves randomly dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining k−1 folds"* (Gareth, Witten, Trevor, Tibshirani, 2013).

La metodología de k-fold cross validation es usada principalmente para estimar la performance del modelo sobre datos desconocidos, o datos sobre los cuales no fue entrenado. Es decir, utiliza una muestra limitada de los datos para estimar cómo funciona el modelo a la hora de hacer predicciones sobre datos que no fueron utilizados para el entrenamiento del modelo. Esta metodología es popularmente utilizada debido a la facilidad de su entendimiento, y por presentar resultados más próximos a los obtenidos en los datos de Test.

El procedimiento de la metodología de k-fold cross validation puede resumirse de la siguiente manera:

1. Se ordenan aleatoriamente los datos
2. Se separan los datos del punto 1 en los k-folds definidos en folds, procurando mantener aproximadamente la misma cantidad de observaciones en cada fold
3. Para cada grupo se toma un fold a modo de Validación
4. Los folds restantes se utilizarán para entrenar el modelo
5. Una vez entrenado el modelo, se valida la performance con el fold que se dejó a modo de Validación
6. Se estima la performance del modelo usando la media de los puntajes de evaluación de cada modelo

Gráfico 7: Ejemplo de Cross Validation con 5 Folds



Es importante destacar que cada observación al ser asignada a un grupo particular no puede moverse de grupo durante el procedimiento, de esta manera se puede asegurar que cada observación pueda ser utilizada como Validación una sola vez, y utilizada en el grupo de entrenamiento k-1 veces.

En el presente trabajo se procedió a trabajar con una metodología de resampling de 5 folds para el entrenamiento de los modelos en los distintos algoritmos. Con esta técnica se buscó tener una predicción más precisa, que nos diese una mayor certeza sobre los posibles resultados en los datos de Test. De esta manera, pudimos alcanzar métricas en el entrenamiento que se asemejan lo más posible con respecto a los resultados obtenidos en el testing de los modelos.

3. Modelos

3.1. Modelo Baseline - Regresión Logística

Para la elección del modelo baseline, elegimos una regresión logística por ser un algoritmo simple, rápido de entrenar, y útil para nuestro tipo de problema a predecir (clasificación). Los modelos de regresión logística resultan ser fáciles de interpretar, dado que los coeficientes de los modelos explican la relación que existe entre la variable independiente con respecto a la variable a predecir. Para ser más precisos, si tenemos una variable independiente con un coeficiente de 0.0055, entonces un aumento de una unidad (ceteris paribus) está asociada a un aumento del logaritmo de la variable dependiente por 0.0055 unidades. (Gareth, Witten, Trevor, Tibshirani, 2013).

Si bien la regresión logística es un modelo más restrictivo que otros algoritmos, nos resulta conveniente para tener de modelo baseline y a partir del mismo, buscar mejoras en las predicciones con otros algoritmos más sofisticados.

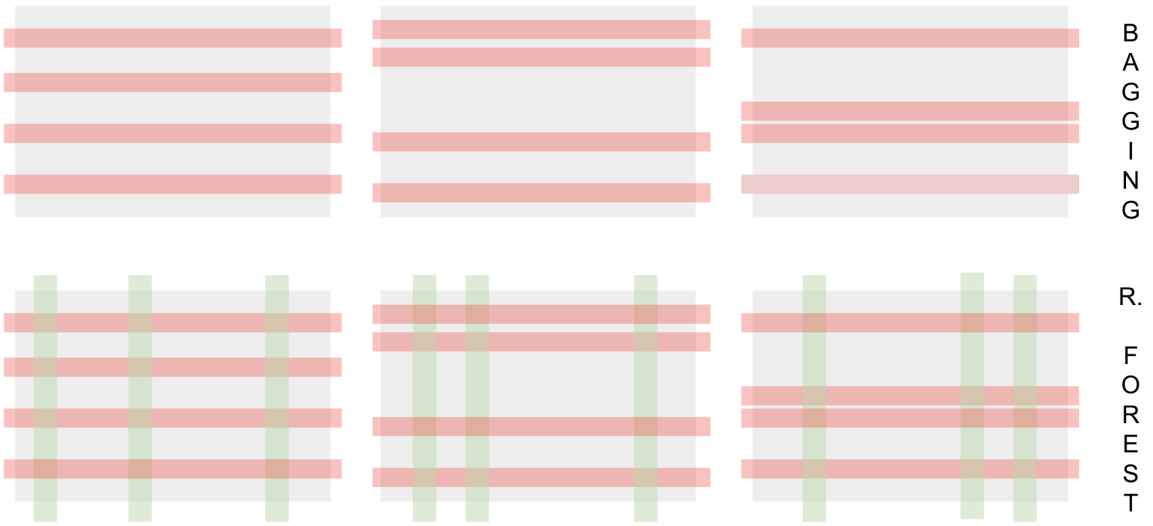
3.2. Random Forest

El algoritmo de Random forest es presentado como una herramienta muy poderosa, agrupando distintos aspectos de otros modelos como árboles de decisión y bagging.

El random Forest se puede resumir como un algoritmo que combina 3 estrategias.

Random Forest = Arboles + Bootstrap ’decorrelado’ + Ensamblaje

Gráfico 8: Ejemplo de sampling de Random Forest



Como se ilustra en el Gráfico 8, el algoritmo de Random Forest se presenta como una herramienta poderosa, que permite combinar no solo entre observaciones (filas), sino también sobre distintos conjuntos de variables (columnas). El bagging busca promediar el nivel de ruido que pueden a llegar a tener los datos, en busca de promediar el ruido de distintos modelos aproximadamente insesgados, y de esta manera buscar reducir la varianza.

Random Forest presenta una ventaja sobre los árboles Bagging, mediante un pequeño ajuste que decorrelaciona los árboles. Al igual que en Bagging, en Random Forest se crean distintos árboles de decisión en muestras bootstrap (muestrear con reposición sobre el conjunto de entrenamiento), pero cuando se crean estos árboles, cada vez que se hace un split en un árbol, un conjunto aleatorios de *m* variables son elegidas como candidatas a hacer el split del nodo.

A la hora de entrenar un modelo de Random Forest, existen distintos hiperparámetros que definen la estructura y potencia que puede alcanzar el algoritmo.

Dado que el Random Forest es un algoritmo que se basa en una estructura de árboles, los hiperparámetros que definen la estructura y profundidad del árbol y que estaremos trabajando en optimizar en busca de mejorar la performance del modelo, son los siguientes:

- *mtry* = El hiperparámetro *mtry* indica el número de variables seleccionadas aleatoriamente como candidatas en cada split del árbol. El número de variables predictoras en cada split es aproximadamente la raíz cuadrada del número total de predictores ($m = \sqrt{p}$) (Hastie, T., Tibshirani, R., & Friedman, J. (2008)).

- *maxnode* = La variable *Maxnode* indica la cantidad máxima de nodos terminales los árboles pueden tener. A mayor cantidad de nodos terminales, el árbol es más profundo y con una mayor posibilidad de overfittear los datos.

- *ntree* = La variable *Ntree* fija la cantidad de árboles que el algoritmo va a estar entrenando. A medida que la cantidad de árboles a entrenar crezca, la probabilidad de overfitting sobre los datos también será mayor.

- *nodesize* = El hiperparámetro *nodesize* indica la cantidad mínima de observaciones que deben encontrarse en los nodos terminales en los árboles entrenados. A diferencia de los hiperparámetros anteriores, la probabilidad de overfittear los datos aumenta a medida que el *nodesize* es cada vez menor.

3.3. XGBoost

XGBoost hace referencia a Extreme Gradient Boosting métodos presentados por Friedman (2001). Dicho algoritmo resulta muy eficiente en la reducción del sesgo y la varianza de un modelo. Gradient Boosting es un enfoque en el cual los nuevos modelos creados para predecir los errores o residuos de modelos anteriores. Cada modelo es luego asignado un nivel de peso para poder hacer la agregación de modelos y dar la predicción final.

El algoritmo de XGBoost también es reconocido por su velocidad a la hora de entrenar modelos, y la escalabilidad en este tipo de escenarios.

El concepto del Boosting dentro del XGboost hace referencia a la generación de distintos modelos que se entrenan de manera secuencial, en donde el modelo busca aprender sobre los errores del modelo anterior y generar un nuevo modelo en base a esas observaciones, con un mayor poder predictivo sobre los datos.

Durante el entrenamiento, los parámetros de cada modelo son ajustados de manera consistente buscando el mínimo de una función objetivo, la cual se puede definir a la hora de comenzar el entrenamiento. Dado que en el presente trabajo tenemos un problema de clasificación, utilizamos las funciones objetivo del área bajo la curva (AUC), y el área bajo la curva precision-recall (AUCPR).

Cada modelo es puesto a competir con el anterior. En aquellos casos donde el nuevo modelo presenta mejores resultados, se continúa con el mismo para seguir trabajando. En el caso en el que el modelo nuevo tenga peores resultados, se regresa al modelo anterior. Este proceso se repite hasta que la mejora que se obtenga de un nuevo modelo sea prácticamente la misma sobre el modelo anterior, lo cual nos indica que hemos encontrado el mejor modelo posible, o cuando se llega a la cantidad máxima de iteraciones definidas al comienzo del entrenamiento.

En el caso del XGBoost los hiperparámetros que estaremos optimizando para encontrar el mejor modelo, son los siguientes:

- nrounds = El hiperparámetro *nround* mira la cantidad de árboles el cual el modelo va a entrenar. A medida que la cantidad de cantidad de nrounds es cada vez mayor, también lo será la probabilidad de overfittear sobre los datos. El algoritmo se entrenará hasta que la mejora entre modelos sobre la predicción sea la misma, o hasta que se alcance el número de nrounds especificado.

- maxdepth = *Maxdepth* mira la profundidad máxima de cada árbol el cual el modelo entrena. El maxdepth es otro hiperparámetro que a medida que crece

más se ajusta a los datos, con posibilidad de overfittear en aquellos árboles muy profundos.

- eta = El *eta* es el learning rate y se limita a valores entre 0 y 1. A medida que menor sea el eta, más lento será el aprendizaje del modelo. A medida que el eta sea más bajo acompañado por un número alto de nrounds, mayor será la probabilidad de overfittear.
- gamma = El hiperparámetro *gamma* indica cual es la mínima reducción del error para que el árbol de entrenamiento pueda generar un corte en una hoja del árbol.
- col_sample = En este caso el *col_sample* indica qué proporción de las variables se van a muestrear y considerar en cada árbol entrenado.
- min_child_weight = El *min_child_weight* indica cual es la cantidad mínima de observaciones que se debe tener para considerar un corte en el árbol. Similar a node_size en el caso de Random Forest, a valores más bajos mayor es la probabilidad de sobreestimar los datos.
- sub_sample = El *sub_sample* me indica la proporción de submuestras de las instancias de formación. Establecerlo en 0.5 significa que XGBoost muestreará al azar la mitad de los datos de entrenamiento antes de cultivar árboles. y esto evitará el sobreajuste. La submuestra ocurrirá una vez para cada iteración.

3.4. Evaluación de los modelos

Con el objetivo de medir la performance de los modelos y así poder evaluar cuál es el más adecuado a la hora de predecir los usuarios fraudulentos, decidimos utilizar la métrica de ROC y F1 Score para hacer esta comparación.

3.4.1. AUC-ROC

La ROC (Receiver Operator Characteristic) es una métrica utilizada para la medición de las predicciones en problemas de clasificación binarios. La curva puede ser obtenida al graficar las distintas combinaciones de Tasas de Verdadero Positivo (TPR), contra las tasas de Falso Positivo (FPR) utilizando distintos umbrales. A medida que la curva es más cercana a la diagonal que conecta los dos vértices del cuadrado, menos capacidad de predicción tiene el modelo. El AUC (Area Under the Curve) es la métrica que mide la habilidad de un clasificador para distinguir entre las

clases y se utiliza como resumen de la curva ROC. El AUC, como lo dice el nombre, mide el área debajo de la curva ROC. A medida que el AUC sea más cercano a 1, mayor es la capacidad de separar las distintas clases.

La curva ROC nos indica visualmente qué tan bien se desempeña nuestro clasificador de aprendizaje automático, y es una de las métricas de evaluación más importantes para examinar el desempeño de cualquier modelo de clasificación. Cuando el AUC es igual a 1 esto nos indica que el modelo es capaz de distinguir perfectamente entre la clase positiva y la clase negativa.

En aquellos casos donde el valor de AUC varía entre 0.5 y 1 es donde existe el error de Falso Positivo o Verdadero Negativo. Estos errores pueden ser minimizados o maximizados, dependiendo del umbral que se decida aplicar. El peor de los casos sucede cuando el AUC es 0.5, ya que nos indica que el modelo no tiene capacidad discriminatoria entre las clases.

Finalmente, también existe un posible escenario donde el AUC esté cerca de 0. En estos casos el modelo está separando de manera casi perfectamente inversa las clases, es decir, el modelo predice la clase negativa como una clase positiva y viceversa de manera perfecta.

3.4.2. F1 Score - Precision & Recall

Al igual que la AUC-ROC el F1 Score también aparece como una métrica comúnmente utilizada en problemas de clasificación con clases desbalanceadas.

El F1-Score es una métrica de evaluación, que se utiliza en Machine Learning para expresar el rendimiento en modelos de clasificación. Esta métrica proporciona información combinada sobre la Precision y Recall de un modelo. Esto significa que un puntaje F1 alto indica un valor alto tanto para el Precision como para el Recall. Generalmente, la puntuación F1 se usa cuando necesitamos comparar dos o más algoritmos de aprendizaje automático para los mismos datos.

Para armar la métrica de F1 Score, primero debemos definir las variables que la componen. Las mismas se pueden construir a partir de una Matriz de Confusión.

$$F1\ Score\ =\ 2\ *\ \frac{(Precision * Recall)}{(Precision + Recall)}$$

		Predicted class		Total instances
		+	−	
Actual class	+	TP	FN	P
	−	FP	TN	N

La métrica de Precision hace referencia a qué tan preciso es nuestro modelo. Mide de aquellas observaciones que fueron marcadas como Positivos por nuestro modelo sobre la variable independiente (en nuestro caso usuarios Fraudulentos), cuántas efectivamente eran Positivas.

$$Precision = \frac{TP}{(TP + FP)}$$

La métrica de Recall mide cuántas observaciones del total de positivos, nuestro modelo puede capturar correctamente.

$$Recall = \frac{TP}{(TP + FN)}$$

De esta manera, el F1 Score combina ambas métricas y resulta más abarcador en aquellos casos donde necesitamos balancear el Precision y el Recall, sobre datasets de clases desbalanceadas (muchas observaciones que entran en Negativos). El F1 score es una métrica de evaluación definida como la combinación armónica de Precision y Recall. Es una medida estadística de la exactitud de un modelo.

3.5. Ingeniería de Variables

La ingeniería de variables es un paso fundamental a la hora de entrenar modelos de Inteligencia Artificial.

Esta práctica responde a la construcción de nuevas variables que surgen a partir de la transformación de las variables preexistentes en nuestro dataset. La transformación de la información ayuda a generar mayor escalabilidad, esto también nos permite explorar relaciones lineales a partir de conexiones no necesariamente lineales entre un feature y la variable a predecir, lo cual puede generar que el proceso de aprendizaje de los modelos sea más rápido y fácil (Nargesian, Fatemeh & Samulowitz, Horst & Khurana, Udayan & Khalil, Elias & Turaga, Deepak. (2017)).

La ingeniería de variables en Machine Learning es mucho más que seleccionar apropiadamente las variables y transformarlas. Es un proceso que no solo prepara

el dataset para que pueda ser compatible con los algoritmos que se utilizarán, sino también mejora la performance que pueden llegar a tener estos modelos. Es importante darle a los modelos variables relevantes, para así reducir la complejidad de los algoritmos.

En el presente trabajo utilizamos distintas técnicas de Ingeniería de Variables las cuales se pueden resumir a continuación:

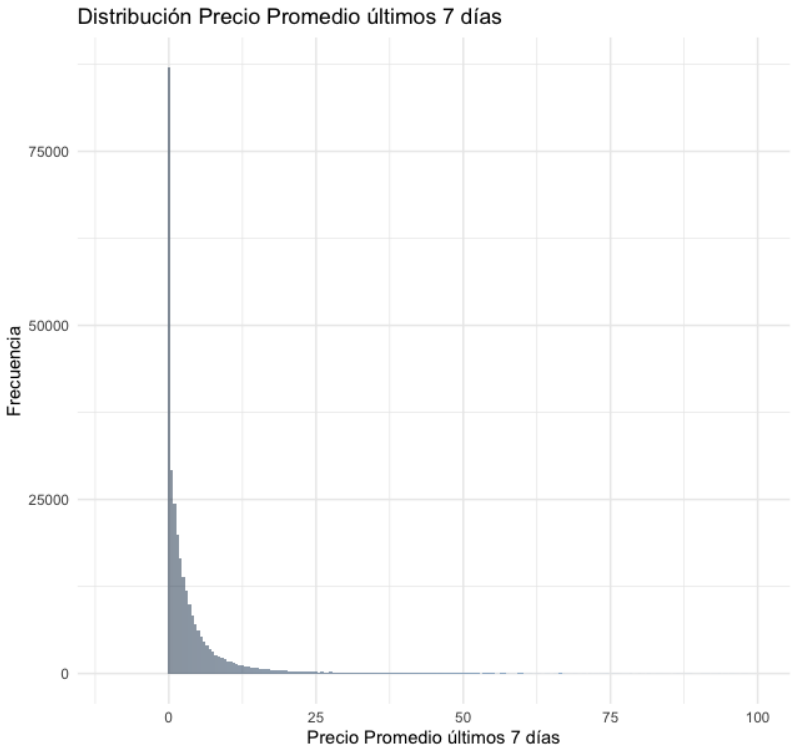
- Ratios Semanales: A partir del dataset original procedimos a armar distintos ratios semanales por usuario que se suelen utilizar en la industria del prevención de fraude. Entre estos se encuentran: Ratios de Aprobación, Ratios de Rechazo, Ratios de Contracargos y Ratios de Reclamos entre otros.

- Variaciones porcentuales: Trabajamos con variaciones porcentuales que miran cómo varían las principales variables durante las 8 semanas en las cuales insume información nuestro dataset. Con estas variaciones porcentuales lo que buscamos es entender si la variabilidad de alguna variable en particular puede darnos algún tipo de evidencia sobre el tipo de comportamiento de los usuarios fraudulentos. Por ejemplo, se creó la variable *v_apro_final_s1_s5,* la cual muestra la variación porcentual entre la aprobación final que tuvo el usuario en sus ventas en la última semana con respecto a 5 semanas antes.

- Logaritmos: al hacer el análisis exploratorio de variables, nos encontramos con que teníamos en el dataset 150 variables con una fuerte dispersión en su distribución lo cual podía llegar a afectar la precisión de nuestros modelos estadísticos. El tratamiento que decidimos adoptar, por considerarlo lo menos invasivo en la representatividad de los datos, fue aplicar logaritmo sobre aquellas variables que presentaban valores de este tipo para utilizar solamente en el modelo Baseline (Regresión Logística) por ser un algoritmo sensible a las escalas de las variables. Para el resto de los modelos se priorizó utilizar los datos en sus escalas originales, ya que son algoritmos que no se ven afectados por este problema.

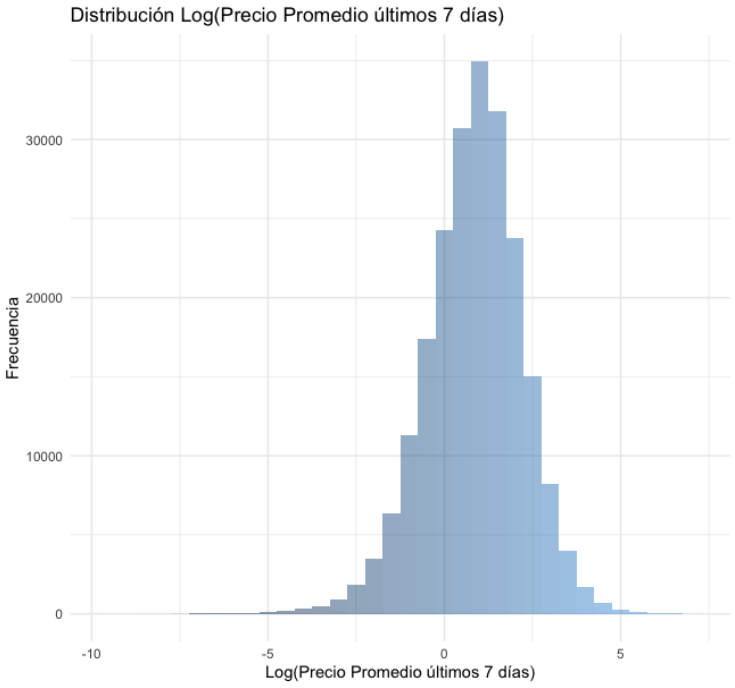
  En el gráfico 10 se puede observar un ejemplo de cómo se comporta la variable de precio promedio de venta de los últimos 7 días, la cual presenta una distribución asimétrica y cómo se ve afectada luego de aplicado el logaritmo.

Gráfico 9: Ejemplo variable sin Logaritmo aplicado



* Los límites del eje x en el primer gráfico fueron limitados en el gráfico para poder tener una mejor lectura de la distribución de la variable. Los valores máximos de la variable que miraba el precio promedio de los últimos 7 días llegaban a valores superiores a 1300 USD.

Gráfico 10: Ejemplo variable con Logaritmo aplicado



4. Resultados

4.1. Variables

A la hora de la definición del set de variables a trabajar, se tomó la decisión de quitar del análisis aquellas variables que contenían más de un 99.9% de 0's en su distribución por ser consideradas poco representativas. En total se eliminaron 68 variables, de las cuales 14 contenían información de contracargos, cuando las 54 restantes eran variables relacionadas con reclamos recibidos por el vendedor.

4.2. Performance

4.2.1 Modelo Baseline

Los parámetros utilizados para entrenar el modelo baseline fueron los siguientes:

- *alpha* = 0 (ridge). El *alpha* es un parámetro que introduce una restricción de presupuesto sobre los coeficientes
- *nfolds* = 5. El parámetro *nfolds* determina la cantidad de folds con los cuales se entrena el modelo.
- *type_measure* = "auc". El *type_measure* define cual es el parámetro de medición el cual se está queriendo maximizar, en este caso se utilizó el AUC el cual se utiliza para problemas de regresión logística de 2 clases.
- *family* = "binomial". El parámetro *family* es un objeto que se utiliza en modelos de glm y es utilizado para indicar el tipo de modelo a entrenar, en el caso de regresiones logísiticas se utiliza la familia binomial.
- *lambda* = exp(seq(10, -10 , length = 100)) En todos los casos los lambdas presentaron valores muy cercanos a 0, con un desvío estándar también cercano a 0. El *lambda* es un hiper parámetro que controla el trade off entre sesgo y varianza, al resultar en un valor muy cercano a 0 esto nos indica que el modelo pondera más el sesgo.

Dado que el lambda arrojado por los modelos en el entrenamiento presentó valores muy cercanos a 0, podemos inferir que todos nuestros modelos de baseline pueden presentar un overfitting de los datos, ya que con un lambda cercano a 0 el modelo será más complejo y aprenderá sobre las particularidades de los datos de entrenamiento.

4.2.2. Modelo Random Forest

Para el entrenamiento del Random Forest se procedió a optimizar los hiperparámetros en busca de la combinación que produzca los mejores resultados. Para el entrenamiento del modelo se utilizó la metodología de Cross Validation con 5 folds.

Para la optimización de hiperparámetros, se hizo uso de la técnica de Random-search con 4 distintos hiperparámetros, entre ellos:

- *mtry* = 27.
- *ntree* = {300,500}.
- *nodesize* = {50,60}.
- *sample_rate* = {0.55, 0.75}
- max_depth = {10,30}

La técnica de Random Search se presenta como un algoritmo que selecciona de manera aleatoria distintas combinaciones de hiperparámetros dentro de los límites que son puestos manualmente.

La mejor combinación de hiperparámetros resultó ser

- mtry = 27
- ntree = 149
- nodesize = 60

4.2.3. Modelo XGBoost

Para el entrenamiento del XGBoost se procedió a entrenar con el dataset seleccionado y optimizar los hiperparámetros en busca de la combinación que produzca los mejores resultados. Para el entrenamiento del modelo se utilizó la metodología de Cross Validation con 5 folds.

Para la optimización de hiperparámetros, se hizo uso de la técnica de Random-search con 7 distintos hiperparámetros. Se entrenaron 20 modelos con hiperparámetros aleatorios dentro de los siguientes rangos:

- nrounds = {20,60}.
- maxdepth = {1,20}.

- eta = {0.0025,0.1}.
- gamma = {0,1}.
- col_sample = {0.25,1}.
- min_child_weight = {10,30}.
- sub_sample = {0.5,1}.

La mejor combinación de hiperparámetros resultó ser

- *nrounds* = 33
- *maxdepth* = 17
- *eta* = 0.03542
- *gamma* = 0.33998
- *colsample* = 0.37718
- *minchildweight* = 22.49428
- *subsample* = 0.73198

4.3. Resultados en el Test Set

A continuación se muestran los resultados obtenidos sobre el conjunto de Train con los tres modelos entrenados: Regresión Logística, Random Forest y XGBoost.

Tabla 1: Resultados Cross Validation

	Modelo Baseline	Random Forest	XGBoost
AUC	0.956	0.995	0.989
F1 Score	0.241	0.519	0.449
Precision	0.167	0.212	0.400
Recall	0.429	0.883	0.511

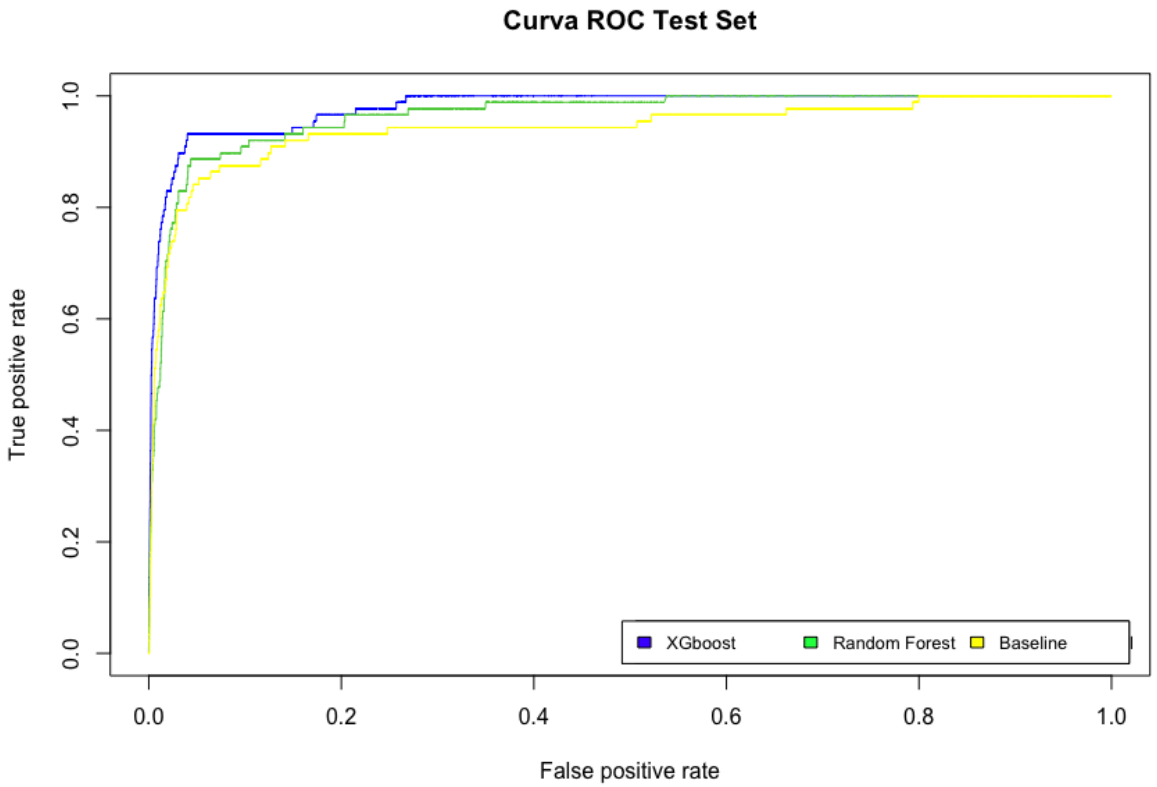
En ambos algoritmos tuvimos una performance por encima de los modelos baseline que se plantearon en un principio. Los resultados en los datos de Test, en base a las métricas elegidas para la medición, son los siguientes:

Tabla 2: Resultado en datos de Test

	Modelo Baseline	Random Forest	XGBoost
AUC	0.941	0.966	0.979
F1 Score	0.305	0.371	0.370
Precision	0.224	0.183	0.405
Recall	0.477	0.523	0.341

Al graficar las curvas ROC de cada modelo obtuvimos el siguiente gráfico:

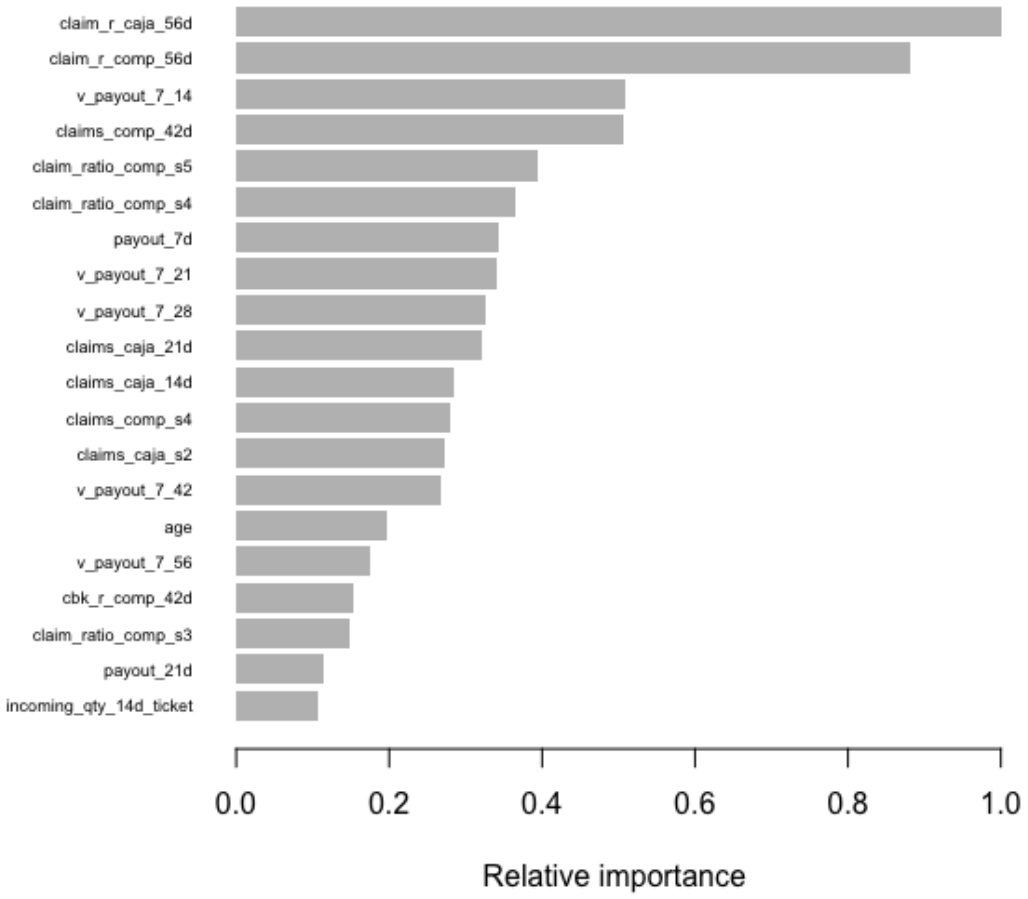
Gráfico 11:Curvas ROC



A partir de los resultados obtenidos, podemos coincidir en que tenemos una mejor performance en los datos de test con el modelo de XGboost, el cual presenta mejor poder de predicción en todas las métricas propuestas para la evaluación de los modelos.

En términos de importancia de variables en el modelo final, podemos ver en el gráfico 12 que aquellas variables que contienen información de los reclamos tienen una mayor participación en el top de las variables con mayor poder predictivo. También aparecen variables que miran el comportamiento de los retiros por parte de los vendedores en los últimos 7 días. Cabe destacar que la edad de los vendedores aparece también como una variable que resulta importante para la segmentación de poblaciones en el problema planteado.

Gráfico 12: Feature Importance



5. Conclusiones y Recomendaciones

5.1. Limitaciones y posibles mejoras futuras

El gran desafío de este trabajo consistió en el manejo de una base de datos muy grande. El primer obstáculo tuvo lugar a la hora de la Extracción de los datos, lo cual requirió un periodo de tiempo mayor al planeado en un principio.

Dada la cantidad de información procesada, una posible mejora a futuro sería utilizar una mayor ventana de tiempo a la hora de la extracción de información. En el presente trabajo se extrajo información de vendedores solamente en 4 meses del año, cuando la posibilidad de obtener una mejor captura de los patrones de venta podría verse favorecido si se incorporara información de por lo menos los últimos dos años a la hora de extraída la información. Para así poder captar las oscilaciones anuales de las distintas temporadas (Black Friday, Cyber Monday, etc.) las cuales podrían llegar a sesgar los resultados de incorporar pocos meses de información.

Por otro lado, a la hora de comenzar con el planteo de la estructura del trabajo se había propuesto entrenar un modelo Random Forest y una Red neuronal. El tercer obstáculo apareció cuando nos dimos cuenta de la limitación tecnológica con la cual contábamos para poder entrenar una Red Neuronal lo suficientemente profunda para aplicar en nuestra investigación. Dada esta limitación tecnológica fue que se decidió entrenar un XGBoost, algoritmo destacado por su poder predictivo como así también su velocidad y eficiencia de cómputo en el entrenamiento. En versiones futuras de este trabajo, y en caso de contar con el hardware lo suficientemente poderoso para poder entrenar una red neuronal profunda con una base de datos lo suficientemente grande, se podría evaluar la performance de una RN en busca de mejoras en las métricas de predicción para este tipo de problemas.

5.2. Recomendaciones

Los años 2020 y 2021 han demostrado alterar todas las convenciones sociales y acelerar el proceso de digitalización del dinero en el mundo entero. Parte de este cambio radical tuvo impacto en las distintas empresas que a partir de esto salen a cubrir las demandas y necesidades de un público que comienza a confiar y necesitar cada vez más las compras y transacciones de manera online. Dado este escenario, resulta importante para las distintas empresas de venta online poder ofrecer un nivel de confianza alto a sus clientes sobre la seguridad de su servicio. La seguridad a la hora de pagar online se vuelve un factor clave para poder ofrecer un servicio de calidad.

El punto central de este trabajo fue crear un modelo que permita identificar aquellos vendedores fraudulentos en una plataforma de ventas online. Teniendo en cuenta que la detección de fraude no es una ciencia exacta, y en donde los patrones de fraude se van transformando y adaptando a los distintos huecos que se encuentren en el sistema, una recomendación es que el reentrenamiento de los modelos destinados a prevenir el fraude se realicen con una frecuencia por lo menos anual para poder actualizar los distintos patrones. Otra recomendación para futuros entrenamientos consiste en evaluar los distintos niveles de rebalanceos de la clase minoritaria, los cuales probaron ser una técnica fundamental en el trabajo presentado.

Este modelo se planteó para que sea un modelo que se ejecute de manera offline. Es decir, no necesita actuar de manera online a la hora que se realiza un pago si no que puede correrse de manera periódica, semanal por como está planteada la recolección de información, para poder evaluar a los vendedores activos. Sumado a esto, sirve como un modelo que no interfiere o afecta los procesos online que corre la empresa, sino que realiza un feedback por detrás de los usuarios.

5.3. Aplicaciones Prácticas

El punto central de este trabajo fue crear un modelo para identificar, dentro del marco de la empresa que provee el dataset, a los vendedores fraudulentos. A partir de los resultados obtenidos, la recomendación para la implementación de este modelo sería correrlo de manera offline detectando aquellos usuarios fraudulentos, y enviando los casos a revisar manualmente para terminar de diagnosticar si es un vendedor que se quiere mantener en la plataforma o no.

Dado el caso, también se podría aplicar distintos umbrales en los resultados del modelo, para crear distintos cortes con distintos niveles de riesgo. Los casos a enviar a revisar podrán ser ordenados por la probabilidad de fraude que defina el modelo, priorizando revisar aquellos que presentan un mayor riesgo de ser vendedores fraudulentos.

Sin dudas que reducir el fraude a través de la detección y bloqueo de vendedores fraudulentos impacta enormemente no solo en la experiencia de los compradores, sino también en la reputación de seguridad de la empresa. De incorporar esta herramienta se podrá acelerar el proceso de identificación de vendedores fraudulentos lo cual impactará en una reducción de la monetización del fraude en la plataforma online.

5.4. Conclusión

En la gran mayoría de la bibliografía estudiada al comienzo del trabajo, poco se hablaba sobre cómo identificar **vendedores** fraudulentos. Las investigaciones realizadas hasta el momento, se centran casi con exclusividad en analizar y proponer nuevas metodologías orientadas hacia el tipo de fraude comprador.

Al comienzo de este trabajo, cuando se empezó a pensar en qué se quería obtener con esta investigación, nos dimos cuenta la gran diferencia de profundidad con la cual la bibliografía disponible abarcaba cada casuística de fraude en las ventas online. Visto esto, nos vimos motivados a profundizar la investigación sobre la casuística de Fraude Vendedor. En los resultados finales pudimos destacar que las variables más importantes en nuestro modelo final se concentraban en tres áreas: Reclamos recibidos por parte de los vendedores, Retiros realizados por los vendedores en los últimos 7 días y Edad del vendedor en la plataforma. Esto nos da un indicio para futuros estudios o análisis, sobre cuáles son las variables que podríamos seguir explotando a la hora de buscar segmentar distintos comportamientos en los vendedores de plataformas online.

El resultado de este trabajo es un modelo que predice la probabilidad de identificar un vendedor fraudulento con la información de sus ventas en las últimas 8 semanas. Este modelo fue creado usando un XGBoost clasificador con una performance de 0.979 AUC-ROC y un F1 Score de 0.37.

6. Bibliografía

[1] Hastie, T., Tibshirani, R., & Friedman, J. (2008) The Elements of Statistical Learning. Springer Series in Statistics.

[2] Gareth, J., Witten, D. Hastie, T. & Tibshirani, R. (2013) An introduction to Statistical Learning with Applications in R.

[3] Muñoz, L., Mazón, J. N. & Trujillo, J. (2011) ETL Process Modeling Conceptual for Data Warehouses: A Systematic Mapping Study. Latin America Transactions, IEEE (Revista IEEE America Latina)

[4] Saputra, Adi & Suharjito, Suharjito. (2019). Fraud Detection using Machine Learning in e-Commerce.

[5] Renjith, Shini. (2018). Detection of Fraudulent Sellers in Online Marketplaces using Support Vector Machine Approach. International Journal of Engineering Trends and Technology.

[6] Niu, Xuetong & Wang, Li & Yang, Xulei. (2019). A Comparison Study of Credit Card Fraud Detection: Supervised versus Unsupervised.

[7] Raghavan, Pradheepan & Gayar, Neamat. (2019). Fraud Detection using Machine Learning and Deep Learning.

[8] Nargesian, Fatemeh & Samulowitz, Horst & Khurana, Udayan & Khalil, Elias & Turaga, Deepak. (2017). Learning Feature Engineering for Classification.

[9] Weng & Poon (2006). A New Evaluation Measure for Imbalanced Datasets. School of Information Technologies, University of Sydney, Australia.

[10] Navarro, J.G (2020) Number of unique online shoppers in Brazil from 2017 to 2019. Statista.com.

[11] Fraud.net. Fraud Dictionary.

[12] F. Charte Ojeda (2014). Análisis exploratorio y visualización de datos con R.

[13] Nargesian, Fatemeh & Samulowitz, Horst & Khurana, Udayan & Khalil, Elias & Turaga, Deepak. (2017). Learning Feature Engineering for Classification. 2529-2535.

[14] Bergstra J. & Bengio Y. (2012). Random Search for Hyper-Parameter Optimization. Departement d'Informatique et de recherche opérationnelle - Université de Montréal.

[15] Friedman, J. H. (2001), "Greedy Function Approximation: A Gradient Boosting Machine." The Annals of Statistics, Vol.29, No.5, pp. 1189-1232.