UNIVERSIDAD TORCUATO DI TELLA

Master in Management + Analytics

Predicción de Churn en Fintech:

Una estrategia de retención integradora que utiliza algoritmos de Machine Learning con el objetivo de eficientizar el uso del presupuesto de Marketing

Sasha Sierchuk

Mayo 2022

Tutor: Maria Eugenia Irala, Msc in Management + Analytics

Resumen

La competencia entre empresas Fintech en Argentina está creciendo y el principal problema que afrontan es la pérdida de usuarios. Esto lleva a que las empresas tengan que hacer inversiones muy elevadas no sólo en adquirir usuarios, sino también en retenerlos y así prevenir que abandonen la plataforma.

Teniendo esta problemática en mente, el objetivo de esta tesis es desarrollar una estrategia para eficientizar el gasto en la retención de usuarios sin generar variaciones negativas en la métrica de retención. Para esto, vamos a hacer uso de técnicas de Aprendizaje Automático para poder predecir cuál es la probabilidad de que un usuario abandone la plataforma. Con esta probabilidad, vamos a probar una nueva estrategia de retención con el fin de mejorar la estrategia actual de Tap, una Fintech con principal foco en el pago de servicios. El proyecto propuesto pretende establecer una metodología de trabajo que ayude a entender el valor de los resultados de un modelo de Machine Learning en la práctica. En particular, se combinará un modelo predictivo con la implementación de herramientas de Marketing.

Como resultado de este estudio, se logró generar una estrategia que resultó en un ahorro del costo de retención del 60% sin notar impactos negativos significativos en la retención. Creemos que la metodología expuesta en este trabajo no solo le agregará valor a la empresa de la cual obtuvimos los datos y con la cual trabajamos, sino que podrá ser aplicada por otras empresas similares dentro de la industria Fintech al enfrentar una problemática tan importante como es hoy la baja retención de usuarios.

Abstract

Competition among Fintech companies in Argentina is growing and the main problem they face is the loss of users. This leads to companies having to invest large amounts of money, not only in acquiring users, but also in retaining them. therefore preventing them from abandoning the platform.

With this problem in mind, this thesis objective is to develop a strategy that makes user retention cost more efficient, without generating negative variations in retention metrics. For this purpose, we will make use of Machine Learning techniques to predict the probability of a user leaving the platform. Considering this probability, we will test a new retention strategy in order to challenge the current strategy of Tap, a Fintech company whose main focus is the payment of services. The proposed project aims to establish a working methodology that helps to understand the impact generated by the application of a Machine Learning model in a real business. In particular, a predictive model will be combined with Marketing tools implementations.

As a result of this study, we were able to develop a strategy that resulted in a retention cost reduction of 60% without significant negative impacts on retention. We believe that the methodology presented here will not only add value to the company from which we obtained the data, but can also be applied by similar companies within the Fintech industry when facing such an important problem as low user retention.

Índice

Índice de Tablas

Índice de Figuras

1. Introducción

1.1. Contexto

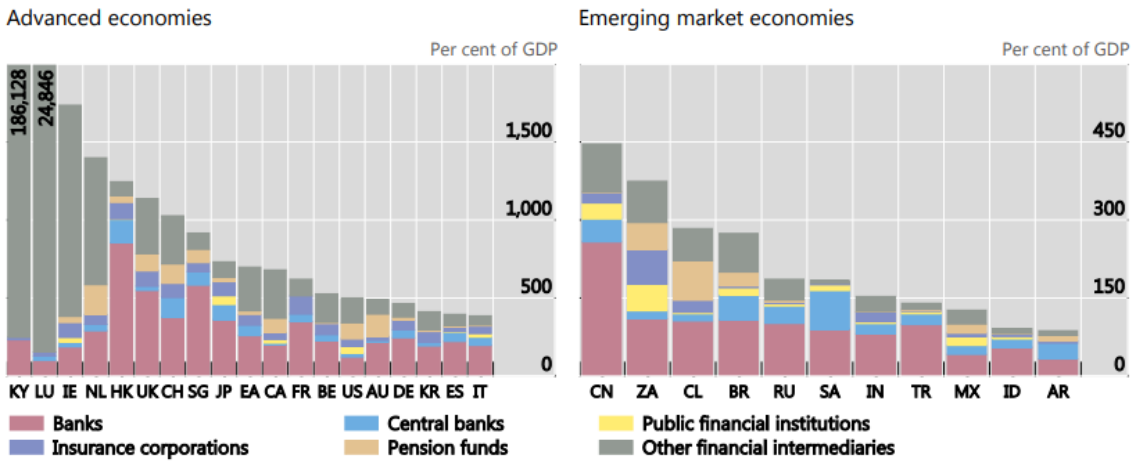
1.1.1 Ecosistema Fintech en Argentina

Fintech (del inglés, “financial technology”) hace referencia a una industria que aplica nuevas tecnologías a actividades financieras y de inversión con el propósito de simplificar las operaciones y facilitar el acceso a productos y servicios. Esta industria comenzó a desarrollarse en Argentina en 2017 y desde entonces ha experimentado un crecimiento exponencial al punto de llegar a ocupar el tercer puesto regional en el sector.

Según la Cámara Argentina de Fintech, las firmas del sector se pueden agrupar en nueve verticales, de acuerdo con la actividad que desarrollen: pagos digitales, que representan el 27% del total; créditos (21%) proveedores tecnológicos (12%); servicios fintech B2B, es decir, entre empresas (11%); blockchain y criptoactivos (11%); inversiones (7%); insurtech o seguros (5%); financiamiento colectivo (4%); y seguridad informática (2%) (Melisa Reinhold, 2022).

El sector emprendedor ve un gran potencial en el mercado argentino debido a sus bajos niveles de educación y actividad financiera en relación a otros países. Esto se puede apreciar en los siguientes gráficos que muestran la composición de los sistemas financieros de algunos países como porcentaje del PBI (Producto Bruto Interno). A la izquierda se encuentran los países con economías avanzadas y a la derecha, los países con economías emergentes. Como se puede ver, Argentina ocupa la última posición entre los 30 países que se encuentran representados, del lado de las economías emergentes.

Figura 1. Composición del sistema financiero por país (Financial Stability Board, 2019)



[1] Assets invested in foreign jurisdictions may distort these ratios.

Sources: Jurisdictions' 2018 submissions (national sectoral balance sheet and other data); IMF *World Economic Outlook*; FSB calculations.

El ecosistema Fintech en Argentina está conformado actualmente por 330 compañías, de las cuales, el 35% surgió durante el período de la pandemia. Esto es atribuible a que en un contexto de confinamiento las soluciones digitales que eliminaban la necesidad de cualquier tipo de trámite presencial cobraron una mayor relevancia en la población, acelerando el crecimiento del sector.

1.1.2 Machine Learning en Fintechs

La utilización de modelos de Machine Learning en la industria Fintech no es algo novedoso, habiendo muchos casos de aplicación y literatura en la materia. Hoy en día los modelos más comunes son los de predicción de *churn*, predicción de fraude y modelos de recomendación de productos. La aplicación exitosa de estos modelos puede resultar en grandes ventajas competitivas, ya que sus algoritmos predictivos habilitan nuevas formas de reducir costos y aumentar ventas. Este trabajo se enfocará en los modelos de predicción de *churn* por lo cual es importante repasar las siguientes definiciones:

- *Churn*: es una palabra derivada de cambio (change). Hace referencia a la terminación de un contrato (Lazarov, V., & Capota, M. 2007). En este proyecto, lo vamos a usar para hacer referencia a usuarios que dejaron de usar la plataforma Fintech hace más de 30 días.
- Retención: es la cantidad de usuarios que permanecen en la aplicación o plataforma de un periodo a otro.

Al consultar la literatura relacionada con esta temática, encontramos varios artículos de modelos de *churn* para bancos, pero poca información sobre modelos para Fintechs. La bibliografía que hace referencia a esta temática es “Customer Churn prediction for FinTech Companies using Artificial Neural Networks”. Este artículo desafía los modelos de Machine Learning para las Fintech y crea un modelo de redes neuronales artificiales para predecir los usuarios inactivos. El principal problema que vemos es que no tiene una implementación práctica para refutarlos, sino que los evalúa en base a su métrica de performance (*accuracy*) y los considera fuera de las expectativas esperadas. Por lo tanto, entendemos que no revela fundamentos suficientes para refutar los modelos de Machine Learning.

Otros textos relacionados con la temática son los modelos de *churn* para bancos. En este rubro, que es muy similar al de Fintech, podemos encontrar mayor cantidad de bibliografía. Aquí vemos la preponderancia de los Modelos de Árboles sobre otras pruebas y también encontramos que algunos trabajos hablan sobre cómo esto puede tener un impacto en el negocio, pero no implementan pruebas para comprobarlo. El más relevante

en este aspecto es el *paper* con el título de "Propension to customer churn in a financial institution: a machine learning approach" donde se remarca la *performance* de los modelos de *Random Forest* por sobre los modelos de *k-nearest neighbors*, *elastic net*, Regresión Logística y Vectores de Soporte (*Support Vector Machines*). Para demostrar el impacto en el negocio, estiman una pérdida proyectada que pueden predecir con el modelo. Ese número representa un 10% de los resultados operativos de los mayores bancos de ese país. Por lo tanto, sustenta la afirmación de que el *churn prediction* es muy importante para prevenir estas pérdidas y tiene un gran impacto en los resultados de negocio.

Por lo mencionado anteriormente, concluimos se encuentran escasos estudios relacionados directamente con la industria Fintech y en su mayoría no involucran estrategias claras de implementación de los resultados del modelo en el negocio. A partir de este relevamiento vemos una oportunidad de mejorar la implementación y mediciones del modelo en la práctica, además de la utilización de datos asociados de manera directa con la industria Fintech.

1.1.3 Herramientas de Marketing para Retención

En Marketing existen dos áreas de trabajo principales que pueden hacer uso directo de estos modelos para mejorar métricas de retención.

Por un lado, puede mencionarse el área de Marketing digital. Esta ayuda a la generación de demanda haciendo uso de internet en servicios digitales mediante medios interactivos y creativos innovadores (Sahai, S., Goel, R., Malik, P., Krishnan, C., Singh, G., & Bajpai, C., 2018). Es la encargada del armado de campañas y seguimiento de resultados en distintos canales como Facebook Ads, Google Ads, Twitter, etc. Dentro de estas herramientas encontramos campañas de *retargeting*, que consisten en impactar a usuarios que ya han interactuado con la plataforma con el objetivo de que vuelvan a transaccionar o prueben diferentes productos. Una de las variables determinantes del éxito de este tipo de campañas es la definición de la audiencia objetivo. Aquí es donde los modelos de predicción de *churn* son aplicables.

Por otro lado tenemos al área de CRM (Customer Retention Management), que se define como un conjunto de procesos diseñado para crear valor mutuo entre la aplicación y los usuarios. Estas interacciones ocurren a través de tres canales principales: push notifications, email marketing y mensajes in-app (comunicación dentro de la aplicación) (Kotarba, M., 2016). Una estrategia muy recurrente para mejorar la métrica de retención es la creación de flujos con cupones (vouchers por un monto de dinero determinado para usar en la aplicación).

Una distinción importante entre Marketing Digital y CRM es el modelo de costos. En las campañas de *retargeting* de Paid Media el costo es variable según la cantidad de conversiones objetivo conseguidas. En contraste, los costos de CRM son semifijos, varían según la cantidad de usuarios de la aplicación de manera escalonada, aunque, en la práctica, sucede a intervalos largos de tiempo por lo que también se puede considerar como un gasto fijo en un período de análisis acotado. Entonces el costo marginal de una nueva campaña es cero y por esto las estrategias de retención deben empezarse por CRM para lograr la mayor eficiencia en el tiempo.

1.1.4 Introducción a Tap

Tap es una compañía Fintech que surge en 2019 con el fin de ayudar a que la población argentina sea incluida en el sistema financiero y poder solucionar en pocos minutos desde el celular los típicos trámites de pago de servicios. Según Tomas Mindlin, el CEO y fundador de Tap: "Hoy en día el 50% de los pagos de servicios se realizan con efectivo y de manera presencial. Esto es ineficiente tanto para las personas como para las empresas".

Dado este contexto, Tap comienza ofreciendo pago de servicios, recargas de celular, recargas del medidor prepago de Edenor[1] y pagos con código QR a través de su aplicación. A medida que fue creciendo, fue incorporando nuevos productos como una tarjeta prepaga y el pago con QR interoperable (se puede pagar desde cualquier aplicación).

En el momento que surge Tap, ya existían varios competidores que brindaban soluciones similares de pago de servicios. Los principales son Mercado Pago y Ualá. El principal foco de negocio de Mercado Pago son los comercios donde se puede pagar con código QR y el de Ualá, la tarjeta prepaga. Por esta razón, Tap ve la oportunidad de enfocarse en el pago de servicios y poder ganar su lugar en el mercado.

Hoy en día Tap cuenta con una base total acumulada de más de 500.000 usuarios que se crearon su billetera virtual. El mes de Abril cerró con 80.000 usuarios activos y tiene proyecciones de seguir creciendo a futuro. Consideramos usuarios activos a aquellos que paguen un servicios, hagan una recarga, paguen con la tarjeta o tengan una transacción con código QR.

[1] Edenor (Empresa Distribuidora y Comercializadora Norte) es una empresa privada argentina que tiene por objeto social la prestación del servicio de distribución y comercialización de energía eléctrica dentro de la zona noroeste de la Ciudad de Buenos Aires y 20 partidos del conurbano bonaerense. Para más información se puede visitar el siguiente URL: https://www.edenor.com/institucional/nosotros/quienes-somos

1.2. Problema

Dado el alto volumen de empresas en la industria Fintech, podemos afirmar que se trata de un mercado sumamente competitivo. Esto deriva en dos problemas principales que enfrenta Tap al igual que sus competidores.

El primero es el aumento del costo de adquisición por usuario. Este aumento se debe a que la demanda es finita y las 330 compañías están compitiendo por adquirir los mismos usuarios. Un ejemplo claro de este fenómeno son las campañas de Marketing Digital, como Google Ads. El comportamiento de este algoritmo se puede profundizar en la página web de soporte *Google Ads Help* en la sección llamada *'Understanding bidding basics'*. El mecanismo de publicidad es por subastas y lo que hace es cada vez que un lugar para anuncios se disponibiliza, evalúa todos los anuncios que tiene disponibles para mostrar y selecciona uno. Esta selección se da principalmente por el precio que cada compañía - que quiere mostrar un anuncio - está dispuesta a pagar, entre otros factores.

El segundo es el empeoramiento o baja en las tasas de retención de usuarios. Esto se explica por la elevada cantidad de descuentos de adquisición. Estos descuentos, como bien describe el nombre, son utilizados con el propósito de la adquisición de nuevos usuarios en la plataforma o aplicación. En consecuencia, los usuarios suelen usar la aplicación para utilizar estos descuentos y luego siguen buscando descuentos en otras aplicaciones.

Dado que el costo de retener un cliente es significativamente más bajo que adquirir uno nuevo -en Tap es un 80% menor-, la métrica de *churn* pasa a ser un objetivo muy importante para Marketing y para el negocio en sí. Las compañías suelen concentrar muchos esfuerzos en bajar la tasa de *churn* dado que impacta directamente en los resultados y hace más escalables a los negocios

Para entender cómo vemos la retención, es importante en primer lugar entender a qué nos referimos cuando hablamos de *cohort* o cohorte. Una cohorte es un grupo de usuarios que comparten una característica común. El análisis de cohortes observa las métricas de retención de esos usuarios a lo largo del tiempo.

Figura 2. Análisis de *cohortes* de activación usando datos de Tap

Mes de vida / Mes de activación	0	1	2	3	4	5	6
ago-21	100%	54%	43%	39%	37%	35%	33%
sep-21	100%	49%	41%	38%	35%	33%	32%
oct-21	100%	46%	41%	36%	34%	34%	32%
nov-21	100%	42%	35%	31%	31%	28%	
dic-21	100%	49%	40%	39%	35%		
ene-22	100%	54%	47%	41%			
feb-22	100%	53%	41%				
mar-22	100%	45%					
abr-22	100%						

En este gráfico podemos ver la métrica de retención según el mes en el que se activó el usuario por primera vez. Las filas muestran una línea de tiempo desde el mes de agosto del 2021 hasta el mes de abril del 2022. Cada columna representa la cantidad de tiempo que ha transcurrido (en meses) desde que los usuarios hicieron su primera transacción. Un ejemplo de lectura del gráfico es que el 47% de los usuarios que adquirimos en Febrero no volvieron en Marzo.

La mejora que vemos en el tiempo con esta métrica se debe principalmente a la suba de inversión en retención. No obstante, si bien la retención subió en un 10% en el periodo de noviembre a diciembre, la inversión en retención subió en un 50%.

Como vemos en la primera columna de la Figura 2, la mayor pérdida de usuarios se da en el mes siguiente a su adquisición. En promedio, se pierde un 50% de usuarios que por alguna razón deciden no operar más por la plataforma. Esto trae grandes pérdidas a nivel de negocio dado que el costo de adquisición para usuarios que se quedaron dos meses en la aplicación se duplica y nos puede llevar a grandes pérdidas por usuario. Por esta razón creemos que puede ser muy útil la implementación de herramientas que puedan mitigar esta baja tan significativa para el negocio.

1.3. Objetivo

Como explicamos anteriormente, hoy en día el problema de la pérdida de usuarios después del primer mes de adquisición es una de las grandes preocupaciones del negocio, y en particular, de Tap. Para mitigar esta caída hoy en día se destina un 30% de la inversión de Marketing a todos los usuarios que cumplen 30 días sin volver a transaccionar.

En primer lugar, queremos encontrar un modelo que con el input de las *features* (variables descriptivas del comportamiento de los usuarios) que seleccionamos pueda predecir de la mejor manera qué usuarios se volverán inactivos. Con inactivos, nos referimos a que cumplan más de 30 días sin transaccionar. Para esto vamos a entrenar una serie de modelos de Machine Learning supervisados. Estos modelos son los que nos permitirán aprender de los patrones existentes en los datos y así poder generar predicciones sobre datos nuevos. Nos vamos a quedar con aquel que nos arroje una mejor métrica de *performance* en la estimación.

En segundo lugar, vamos a llevar a cabo un *A/B testing* con el propósito de comparar la estrategia actual de retención con la nueva estrategia en la que se van a ver aplicados los resultados del modelo. Este experimento será el método de validación del impacto que podemos generar en el negocio con la implementación del modelo en la práctica.

Teniendo en cuenta estos dos pasos, el objetivo final de este proyecto va a consistir en poder generar una nueva estrategia de retención integradora, que tome como input a la predicción que nos arroja el modelo de Machine Learning. Esta predicción nos va a permitir enfocar el presupuesto en usuarios que tienen una probabilidad de volverse inactivos alta y no desperdiciar dinero en usuarios que tienen una probabilidad baja de abandonar la plataforma. Por lo mencionado anteriormente, definimos que el objetivo de negocio es generar un ahorro en el presupuesto de Marketing sin que tenga un impacto en la retención.

Sabemos por la literatura consultada que los modelos de Machine Learning para atacar este problema tienen un gran impacto en cualquier industria. Cuando lo vemos en la industria de los bancos la bibliografía también nos muestra un impacto muy fuerte de la aplicación de estos modelos en el negocio.

Creemos que esta metodología no solo le agregará valor a la empresa sobre la cual armamos el proyecto, sino que podrá ser aplicada por otras empresas similares dentro de la industria Fintech al enfrentar una problemática tan importante como es hoy la baja retención de usuarios.

2. Datos

2.1. Bases de datos

Los datos fueron provistos por Tap, una aplicación Fintech que comenzó en el año 2019 y hoy en día cuenta con 80K usuarios activos y una base de más de 500K billeteras creadas.

En este proyecto se quieren utilizar variables para detectar patrones que nos permitan predecir la probabilidad de *churn* al mes siguiente de su activación. Por lo tanto, el primer paso para iniciar este modelo consiste en crear la base de datos integral con las características de cada usuario con el fin de usar esas variables como *input* al modelo predictivo.

Para iniciar la extracción de datos para el modelo, lo primero fue plantear las variables relevantes para la predicción. Esta investigación requirió de reuniones con cada equipo de la empresa para identificar otras variables relevantes adicionales a las contempladas por el equipo de Marketing (medio de adquisición, día de adquisición, comportamiento del usuario dentro de la App, etc.). En base a los inputs, se procedió con el desarrollo de la *query* para extraer el *dataset* de la empresa.

La empresa cuenta con una base que llamamos Data Lakehouse, dado que contiene una combinación entre Data Lake y Datawarehouse. Está conformado por tres capas de tablas llamadas L1, L2 y L3.

La primera capa, replica los datos de origen sin ninguna transformación. A esto se lo llama *Data Lake* ya que los datos pueden ser almacenados en cualquier tipo de formato y no se aplica ningún tipo de cambio. Ante estos tipos de datos, es necesario aplicar transformaciones para que estos puedan ser consumidos y utilizados con más facilidad. Al proceso de extracción de datos necesario para realizar dichas transformaciones se lo denomina *ELT (Extract, load, transform)*.

La segunda capa consiste en datos modelados y estructurados para optimizar las consultas. Para la creación de esta, utilizamos un proceso llamado Data warehouse que ensambla y administra datos de varias fuentes distintas con el objetivo de facilitar las consultas a la empresa (Stephen R. Gardner, 2002). El proceso que se utiliza en este caso se llama *ETL (Extract, transform, load)* y consiste en transformar la *data* previo a almacenarla. Las ventajas de las tablas *ETL* son que: son de muy fácil acceso, consolidan distintas fuentes de datos que se encuentran en L1, transforman los datos en grupos de información específicos de negocio y brindan facilidad a nivel consultas, análisis, reportes y toma de decisiones.

La tercera capa contiene tablas específicas para cálculos que después necesitan ser trasladados a herramientas de visualización. Vamos a ahondar en la L1 y L2 dado que en este trabajo no hacemos uso de la L3.

En primer lugar, para el armado de la *query* de este proyecto se utilizaron las tablas de la capa L1, ya que las tablas de la capa L2 no contenían toda la información necesaria para predecir si un usuario se iba a volver inactivo. Sin embargo, el volumen de datos resultó en tiempos de procesamiento excesivamente elevados que podían producir grandes demoras en el proyecto

Por esta razón, tomamos la decisión de usar la base de datos que obtenemos luego del proceso de Datawarehause y sacrificar ciertos datos dado que el tiempo para la ejecución del modelo era limitado y era necesario agilizar la consulta. De esta forma, al deshacernos de algunas columnas, se redujo el tiempo de procesamiento a minutos, manteniendo un volumen muy elevado de datos.

Para la creación de la *query* final, usamos las siguientes tablas:

- Tabla *transactions*

  Esta tabla cuenta con el detalle de todas las transacciones de los usuarios. Con este término nos referimos a cualquier movimiento de dinero que se ejecutó dentro de la aplicación.

- Tabla *account*

  Esta tabla cuenta con todos los detalles que tiene que completar el usuario a la hora de la creación de cuenta.

- Tabla *card_public*

  Esta tabla cuenta con los datos de asociaciones de tarjetas de los usuario dentro de la aplicación

- Tabla *appsflyer_events*

  Esta tabla cuenta con el detalle de eventos que se disparan cuando el usuario hace distintos tipos de acciones dentro de la aplicación. En este caso la estamos usando para ver qué pantallas visitan los usuarios.

El detalle de las tablas con sus respectivas columnas se puede ver en el Apéndice. En esta sección, pueden encontrar una breve introducción a cada una de las columnas.

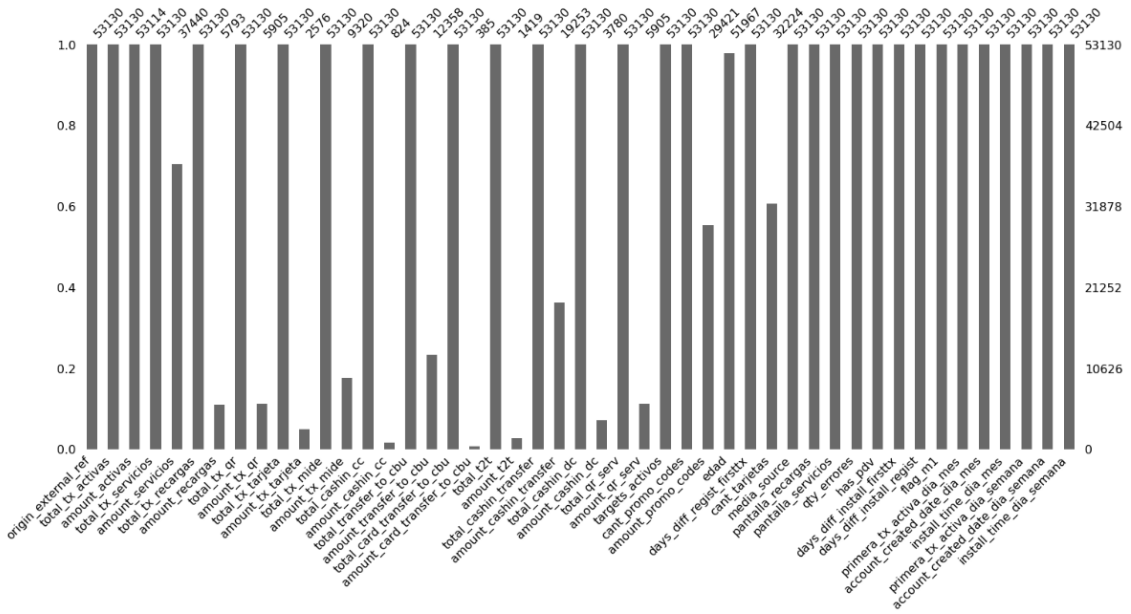
Para entender la información con la que contamos es importante saber que tenemos tres grupos de variables. En primer lugar, las variables de eventos que nos permiten entender cuántas veces se disparó cada pantalla o acción determinada por usuario. En segundo lugar, encontramos las variables de pagos que se representan como una sumatoria de distintos *ids* de transacciones por usuario. Las transacciones las podemos encontrar en su totalidad y el monto total transaccionado. Por último, tenemos las variables de fecha que nos ayudan a entender el ciclo de vida del usuario. Con estos tres grupos de variables creamos el *dataset* final.

2.2. Análisis exploratorio

En esta sección vamos a investigar las distribuciones y relaciones entre las variables para entender en profundidad el *dataset* con el que contamos y que esto nos ayude a preparar los datos para el modelo.

La información con la que vamos a hacer el análisis de datos considera el período de Octubre del 2021 a Febrero del 2022. Tomamos los primeros 20 días de vida del usuario para ver su comportamiento dentro de la aplicación. En la base tenemos 53,130 filas que corresponden cada una a un usuario diferente, y 47 variables, siendo algunas de ellas categóricas y otras numéricas.

Figura 3. Cantidad de datos no nulos por variable.

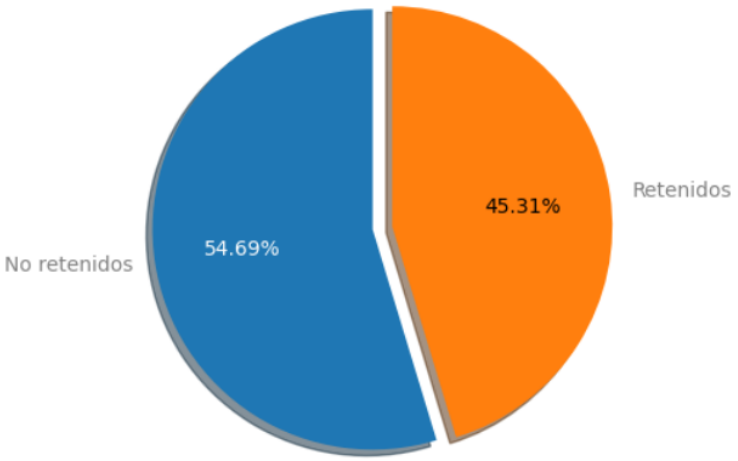


En primer lugar, analizamos la cantidad de datos faltantes que tiene el dataset para asegurarnos que no afecten la predicción. En este caso, contamos con 53,130 datos faltantes. Esto no es algo que afecte la calidad de los datos, pero sabemos que la mayoría de

modelos de Machine Learning no pueden trabajar con datos faltantes. Por lo tanto, es posible que no podamos correr el modelo si no aplicamos alguna técnica para reemplazar los valores. Para poder aplicar algún cambio es importante entender en qué consiste cada variable que tiene valores nulos y ver qué cambios se pueden aplicar en estas para que representen de la mejor forma el valor.

Por lo que podemos ver en el gráfico, estos datos faltantes corresponden a variables numéricas y se dan cuando no se observan transacciones determinadas en los usuarios en las variables de monto transaccionado. Por lo tanto, se decidió reemplazar por cero (0) dado que este valor expresa de la mejor manera que el usuario no tuvo transacciones y, por ende, no generó ningún valor para tales variables. Otra variable que tiene *missing values* es la de Cantidad de tarjeta, en este caso también decidimos reemplazarlos por 0 dado que esto ocurre cuando el usuario no tiene ninguna tarjeta asociada y creemos que es la mejor forma de representarlo.

Figura 4. Distribución de usuarios de la variable que vamos a predecir (churn)



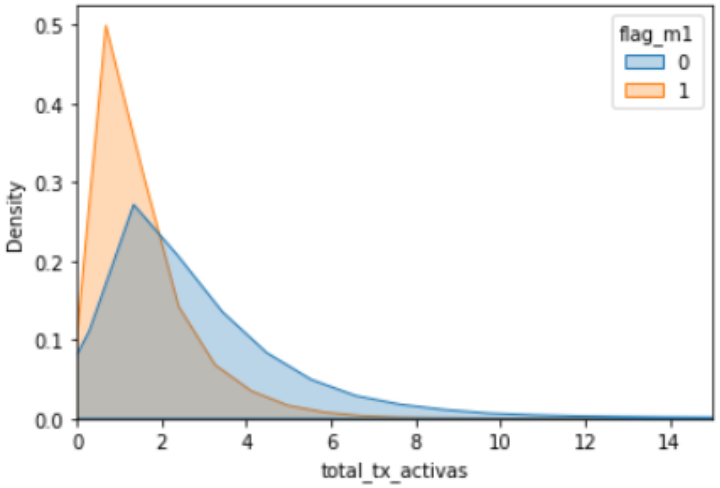
En segundo lugar, revisamos la distribución de los datos en base a la variable que queremos predecir. Tenemos un 45% de usuarios que vuelven a transaccionar a los 30-60 días después de su última transacción en su primer mes de vida y un 54% que no vuelven. Viendo estos números, podemos afirmar que los datos están balanceados. Esto es un factor muy importante para el modelo dado que, si los datos están desbalanceados, las métricas de performance pueden no ser las mejores, ya que predecir la clase minoritaria se torna más complicado.

En el artículo *"The class imbalance problem: A systematic study", Japkowicz, N., & Stephen, S. (2002)* se comprueba el problema que traen los datos desbalanceados en los resultados del modelo. Lo que remarca es que cuanto mayor es el grado de desequilibrio de clases, mayor

es la complejidad del concepto y cuanto menor sea el tamaño total del conjunto de entrenamiento, mayor será el efecto de los desequilibrios de clase en los clasificadores sensibles al problema.

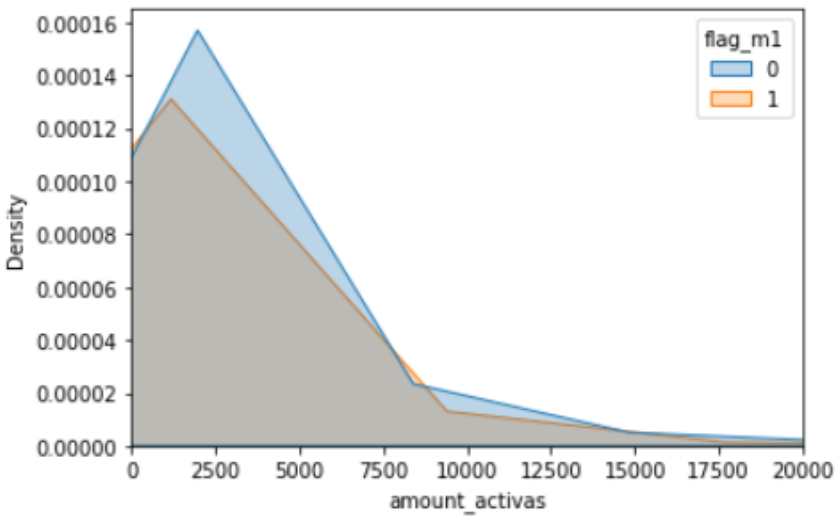
Los gráficos que vienen a continuación son gráficos de densidad. Estos se usan para entender las distribuciones de los valores de una variable y poder armar comparaciones entre distribuciones. En este caso todas las comparaciones se armaron en base a la variable que queremos predecir. El 1, que se encuentra en naranja, nos va a informar la distribución de los usuarios que se volvieron inactivos, mientras que la azul, los que no se volvieron inactivos.

Figura 5. Gráfico de densidad de la variable transacciones activas segmentado por usuario *churn* o no *churn*.



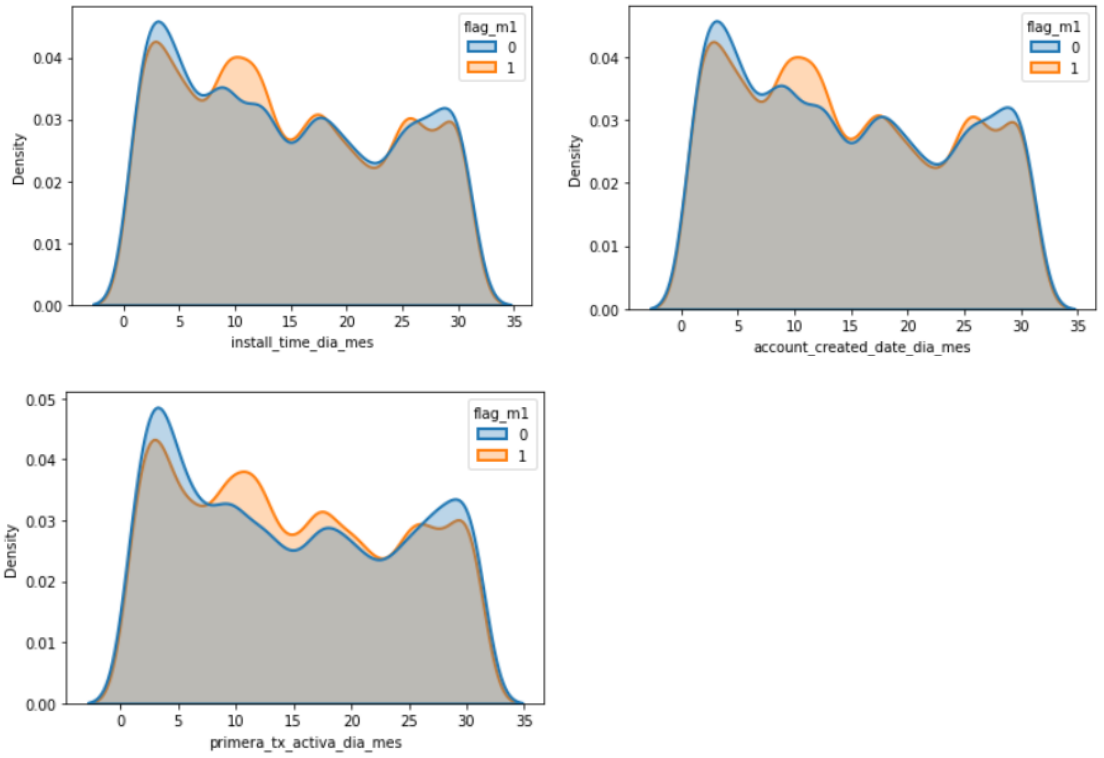
En la Figura 5, se muestra un gráfico de densidad de la variable de cantidad de transacciones activas en 20 días después de su primera transacción. Lo que podemos observar es que los usuarios que no vuelven tienen mayor densidad entre 0 y 2 transacciones. Los usuarios que vuelven a usar la aplicación tienen una densidad más amplia, aunque el pico se ve en las dos transacciones. Otro factor que consideramos importante es la densidad en la primera transacción. Vemos que en el caso de usuarios que vuelven tienen muy baja densidad y que los usuarios que no vuelven tienen su pico en una transacción. Por lo que podemos asumir que hay muchos de los usuarios que no vuelven que solo hacen la primera transacción y no vuelven a usar la aplicación

Figura 6. Gráfico de densidad de la variable *amount_activas* segmentado por usuario *churn* o no *churn*.



En la Figura 6, vemos un gráfico de densidad de la variable cantidad de dinero de transacciones activas en 20 días después de su primera transacción. Lo que podemos observar es que los usuarios que no vuelven tienen su pico de mayor volumen en números menores a 2500, mientras que los usuarios que vuelven tienen el pico en 2500. Lo que podemos afirmar es que el momento pico de los usuarios que vuelven es mayor al momento pico que los usuarios que no vuelven.

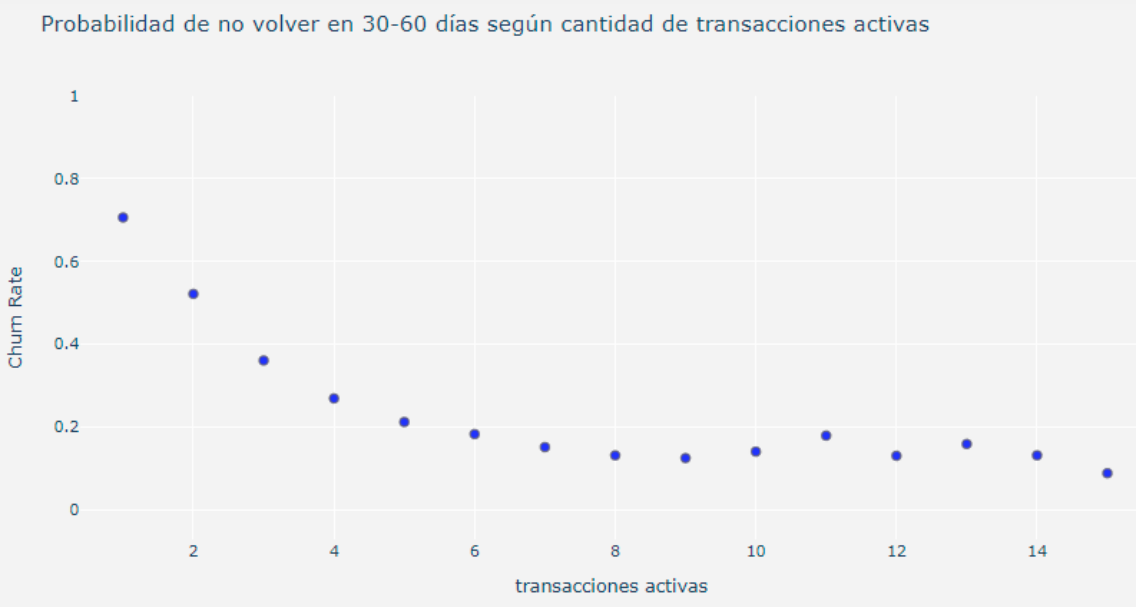
Figura 7. Gráfico de densidad de la variable día del mes de instalación, creación de cuenta y primera transacción segmentado por usuario *churn* o no *churn*.



En la Figura 7, mostramos los gráficos de densidad del día del mes del install, creación de cuenta y primera transacción activa. Podemos notar que las tres siguen una distribución muy similar dado que el tiempo promedio desde que el usuario se instala hasta que hace su primera transacción es de dos días. Además vemos un pico en los primeros días del mes, esto viene dado por la estacionalidad de los pagos de servicios, los cuales suelen pagarse a principio de mes cuando la gran mayoría de las personas cobran su salario. Otro factor importante de estos gráficos es que los que no vuelven tienen una distribución un poco más elevada de 5 a 15 días que los que vuelven y en el caso de la fecha de activación esto se repite de 15 a 31 días.

Los gráficos a continuación nos muestran la probabilidad que tiene un usuario de no volver en 30-60 días dependiendo de diferentes tipos de variable. Esto lo que nos dice es la cantidad de unos (usuarios que no volvieron) sobre el universo total de usuarios. Nos sirve para entender el impacto que tienen las diferentes variables en que un usuario se vuelva inactivo o no.

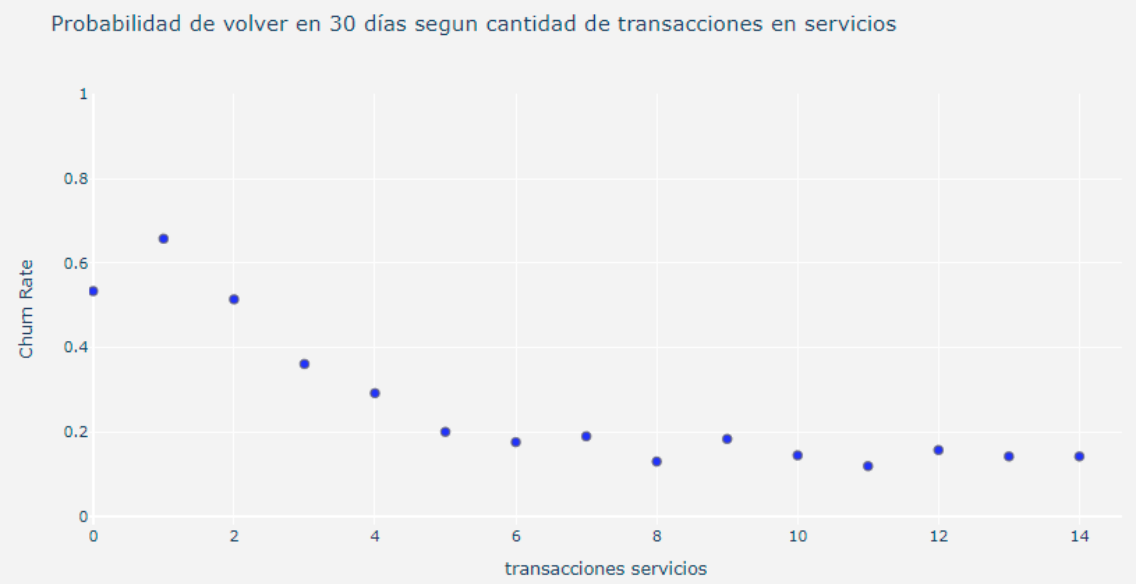
Figura 8. Gráfico de probabilidad de ser usuario *churn* en 30-60 días según la cantidad de transacciones que tiene un usuario.



En la Figura 8, podemos ver como va disminuyendo la probabilidad de no volver de un usuario a medida que van incrementando la cantidad de transacciones totales activas que tiene en los primeros 20 días de vida. Este patrón de baja se mantiene hasta que el usuario llega a seis transacciones con una probabilidad del 20% de no volver y luego se mantiene alrededor de esta probabilidad.

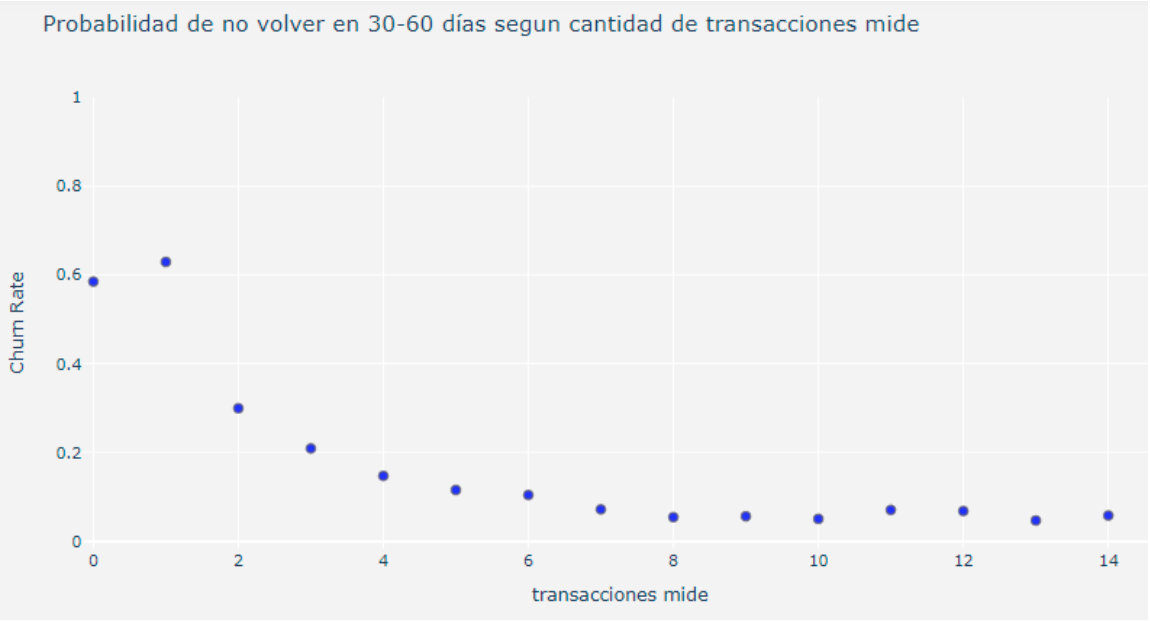
Esté gráfico nos muestra la importancia de incentivar a que los usuarios, en primer lugar, paguen todos los servicios que tienen que pagar en el mes con Tap, dado que eso ya nos llevaría a una probabilidad más alta de que el usuario vuelva. No obstante, esta estrategia podría no ser suficiente. Si nuestro objetivo es llegar a las seis transacciones va a ser importante que pueda hacer uso de otras funcionalidades que tiene la aplicación para alcanzar ese 20% de probabilidad de no volver.

Figura 9. Gráfico de probabilidad de ser usuario *churn* en 30-60 días según la cantidad de transacciones que tiene un usuario en servicios.



En la Figura 9 puede observarse un comportamiento muy similar a la tendencia del total de las transacciones activas dado que son el 80% de los usuarios totales los pagadores de servicios. En este gráfico, se alcanza el 20% en cinco transacciones y luego queda constante alrededor de este valor. Como mencionamos anteriormente, es muy importante poner el foco en que el usuario siga pagando sus otros servicios con Tap.

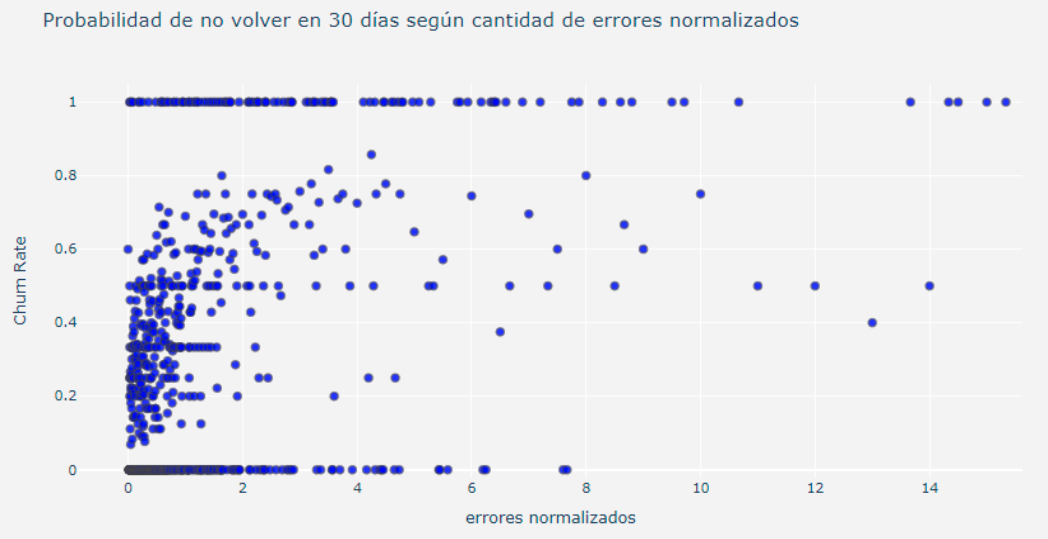
Figura 10. Gráfico de probabilidad de ser usuario *churn* en 30-60 días según la cantidad de transacciones que tiene un del producto MIDE.



En la Figura 10, podemos ver la probabilidad que tiene un usuario de no volver dado la cantidad de recargas del producto MIDE que hizo este. Este producto es el medidor prepago de Edenor que se puede recargar mediante el número de medidor en la App. Lo que podemos ver en el gráfico es que ya con dos transacciones de este producto puede observarse una probabilidad 23pp más baja que la media, que se encuentra en 52%. Por lo tanto sabemos que estos son usuarios que valoran la aplicación. Este grupo alcanza el 20% con la tercera transacción y luego sigue bajando hasta llegar a 5% en la octava transacción y se mantiene. Eso podría explicarse por la poca competencia que tiene la empresa en este producto.

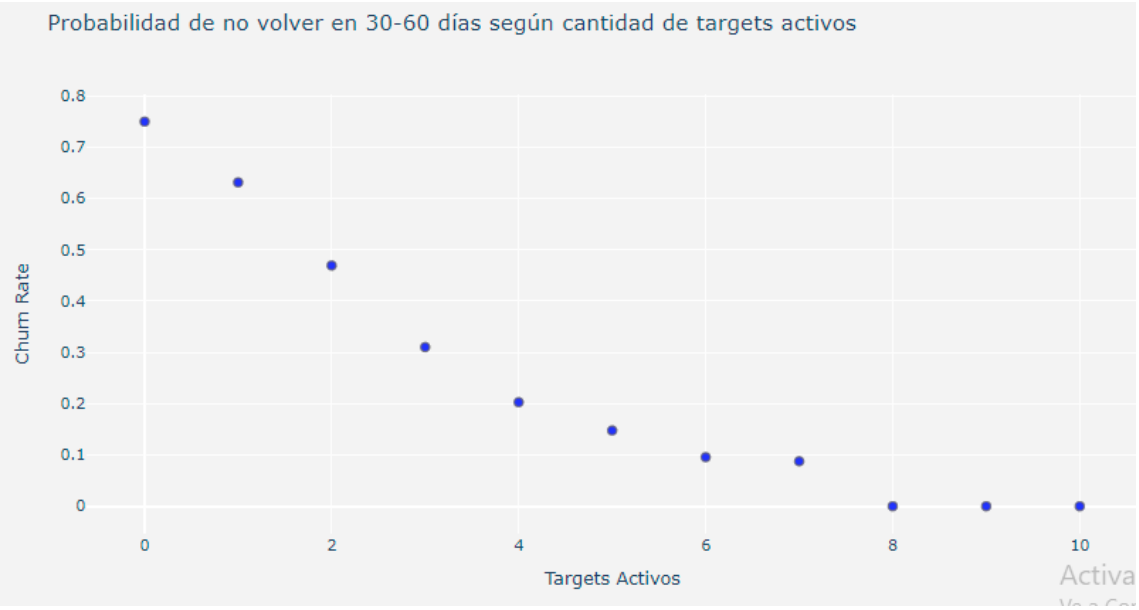
Lo que nos llama mucho la atención en este gráfico es la diferencia que hay entre tener una sola transacción de MIDE o tener dos. Esta diferencia es mayor a un 50%. Lo que nos dice esto es que hay un importante volumen de usuarios que paga solo una vez y se va. Esto es muy probable que suceda por los descuentos de adquisición que Tap ofrece, por lo tanto vemos importante modificar la promoción para que el descuento se otorgue con dos transacciones y ver si logramos mejorar la retención.

Figura 11. Gráfico de probabilidad de ser usuario *churn* en 30-60 días según la cantidad de pantallas de errores totales en la App.



En este gráfico podemos ver que hay una leve relación entre la cantidad de errores y la probabilidad de no volver en 30-60 días. Esto nos dice que a medida que más errores se disparan es más probable que no vuelvas a transaccionar en la aplicación. Esta variable fue normalizada dado que a mayor cantidad de errores, seguramente hayas pasado más tiempo en la aplicación. Lo que decidimos fue dividirlo por la cantidad de veces que un usuario entró a las pantallas de servicios y recargas. Generamos una variable con el nombre de errores_norm y borramos la variable qty_errors para que no afecte en el modelo.

Figura 12. Gráfico de probabilidad de ser usuario *churn* en 30-60 días según la cantidad de targets pagos.



Por último, tenemos el gráfico de cantidad de targets activos. Con targets nos referimos a servicios distintivos pagados por el usuario, como por ejemplo, un usuario que pagó Edenor y también Aysa[2] va a tener 2 targets. Parece ser un factor más importante que la cantidad de transacciones totales dado que ya con 4 targets distintos se alcanza un 20% de probabilidad mientras que en la Figura 3 se alcanza en 6 transacciones.

2.2.1 Conclusiones análisis exploratorio

Este análisis fue muy útil para el entendimiento de datos. Pudimos encontrar *insight* sobre variables que van a poder ayudar a mejorar la métrica de retención. Las más relevantes de las que pudimos analizar son: la cantidad de transacciones que un usuario tiene en MIDE (medidor prepago de Edenor) y la cantidad de targets que paga un usuario. A su vez, se pudieron detectar dos accionables que pueden derivarse de este análisis.

Por un lado, podría aumentarse el share de usuarios MIDE destinando más presupuesto en adquisición. Es importante remarcar que los descuentos de adquisición en Tap se otorgan a partir de que el usuario llegue a dos pagos, dado que estos usuarios tienen un 50% menos de probabilidad de volverse inactivos.

Por otro lado, es importante armar estrategias para poder aumentar el *cross-selling* entre servicios y con otros productos para conseguir mayores números de retención. Para esto, una forma simple de hacerlo es entender qué otros servicios pagan los usuarios que viven en el mismo sector y armar comunicaciones distintas por sector con el fin de que paguen todos los servicios que tienen disponibles.

Estas dos iniciativas son fáciles de implementar y nos pueden ayudar en el corto plazo a mejorar la métrica de retención. A mediano plazo, vemos más valioso la creación de un modelo que pueda hacer un análisis de todos los patrones presentes en las variables (no solo de algunos) para generar una estrategia integral más eficiente.

3. Metodología

3.1 - Tecnicas de Machine Learning

Para predecir la probabilidad que tiene un usuario de volverse inactivo en el mes siguiente de su activación primero tenemos que definir qué modelos de Machine Learning vamos a utilizar y qué métricas de performance calcularemos.

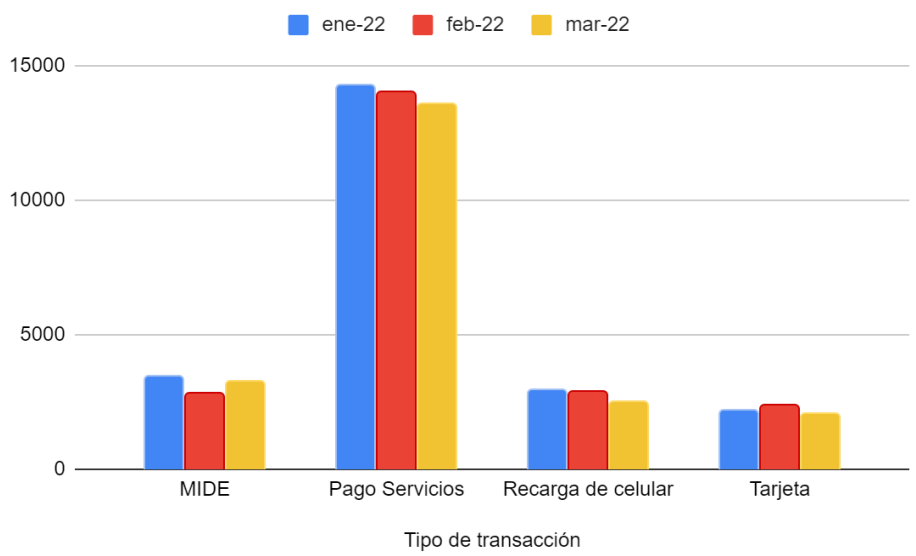
[2] Agua y Saneamientos Argentinos (AySA) es una empresa pública argentina dedicada a la prestación de servicio de agua corriente y cloacas.

En este caso, vamos a estar construyendo modelos de clasificación binaria. Seleccionamos estos modelos dado que estamos queriendo abordar un problema que requiere de un output cualitativo, que va a ser binario. Vamos a tener un 1 si el usuario no volvió a usar la aplicación y un 0 en caso contrario.

Para comenzar, es importante definir un horizonte de tiempo durante el cual vamos a evaluar las distintas variables que definen al usuario. Esto es importante dado que nos va a proveer la información con la que contamos y es determinante para la performance del modelo. No puede ser un tiempo corto, dado que necesitamos información del usuario para poder predecir, pero tampoco puede ser un tiempo largo dado que ya no vamos a tener las mismas herramientas para poder atraerlo. Nuestra selección fue estudiar al usuario durante los primeros 20 días de vida para poder predecir si este va a volver entre 30 y 60 días. Elegimos este horizonte dado que en Tap es el momento en el cual se pierde la mayor cantidad de usuarios.

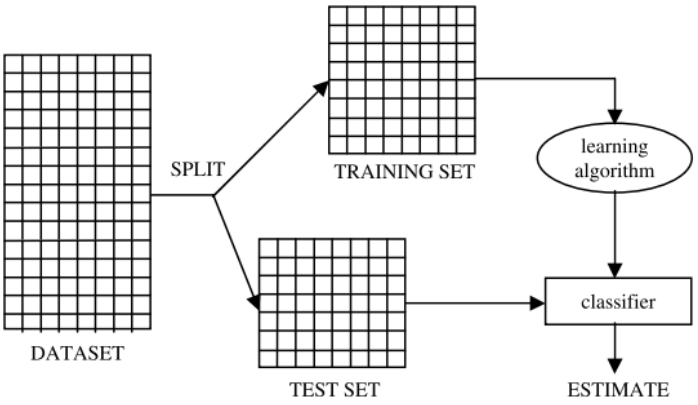
Dentro de Tap, un usuario puede transaccionar en distintos productos como pagos de servicios, recargas de celular, recargas MIDE y pagos con QR. Es por esto que otra decisión importante que tenemos que tomar es si correr el modelo con todos los datos en conjunto, o separarlo por producto. En este caso, lo que decidimos fue tomar a todos los usuarios de la aplicación, sin importar el producto que transaccionan. Esta decisión la tomamos dado que el volumen de usuarios con los que contamos hoy en los otros productos no eran lo suficientemente grandes para poder armar un modelo que generalice lo suficientemente bien. Esto se puede ver en la Figura 13. Hoy en día, el 85% de nuestra base de usuarios solo paga servicios. En el siguiente gráfico, podemos ver la cantidad de usuarios nuevos por producto de Noviembre a Febrero.

Figura 13. Distribución de usuarios nuevos por producto



El método que elegimos para la medición es el de *Train and Test*. Lo que hacemos es, previo a correr el modelo, separamos en grupo de *train* (conjunto de entrenamiento) y grupo de *test* (conjunto de prueba). En primer lugar, usamos el grupo de train para construir el clasificador del modelo. Luego, lo usamos para predecir la clasificación para el test set. Medimos la performance sobre ese grupo y sabemos cuántas de esas fueron correctas sobre el total. Esta métrica se llama accuracy y puede ser usada como una métrica de performance del modelo. (Bramer, Max. Principles of data mining, 2007)

Figura 14. Test and training set (Bramer, Max. Principles of data mining, 2007)



En este caso decidimos correr tres modelos base llamados Regresión Logística, Random Forest y LightGBM. Luego de entrenar estos tres modelos, vamos a evaluar la performance de cada uno y definir el modelo final que va a ser el que tenga la mejor performance según la métrica que definimos de comparación. A partir de haber seleccionado el modelo de mejor performance, buscaremos mejorar aún más su performance a partir de utilizar técnicas de selección de hiperparámetros con *cross validation*.

3.1.1 - Regresión Logística

Como primer modelo, decidimos correr una Regresión Logística. Este modelo surge de la necesidad de adaptar la regresión lineal para que sea válida para variables categóricas. Sea X el conjunto de variables independientes y p(X) la probabilidad condicional de que Y=1 dado X, podemos afirmar que el problema que tiene la regresión lineal es que cada vez que una línea recta se ajusta a una respuesta binaria que se codifica como 0 o 1 (como nuestra variable *flag m1*) en principio siempre podemos predecir p(X) < 0 para algunos valores de X y p(X) > 1 para otros (a menos que el rango de X sea limitado).

Para evitar este problema, en la Regresión Logística se usa la siguiente función:

$$p(X) = \frac{e^{\beta 0+\beta 1X}}{1+e^{\beta 0+\beta 1X}} \quad (1)$$

La función logística (1) siempre producirá una curva en forma de S, por lo tanto nunca va a traer probabilidades por debajo del cero o por arriba del uno.

Si distribuimos y aplicamos logaritmo de los lados nos queda la siguiente función:

$$log(\frac{p(X)}{1-p(X)}) = \beta 0 + \beta 1X \quad (2)$$

En la Regresión Logística, como podemos ver en la primera ecuación, no se ve una relación lineal entre p(X) y X. Por lo tanto, β1 no corresponde al cambio en p(X) asociado al aumento de una unidad en X. El valor de variación de p(X) debido a un cambio de una unidad en X depende del valor actual de X. Pero independientemente del valor de X, si β1 es positivo, entonces el aumento de X se asociará con el aumento de p(X), y si β1 es negativo, el aumento de X se asociará con la disminución de p(X). Y el porcentaje de cambio en p(X) por una unidad en X va a depender del valor actual de X.

Las ventajas de la Regresión logística son que se puede aplicar de una manera sencilla y que es fácil de interpretar dado que los parámetros explican la dirección y la intensidad de la importancia de las variables independientes sobre la variable dependiente. Las desventajas de este modelo son que no puede ser aplicado en problemas de clasificación no lineal (osea, supone un problema con clases linealmente separables), que se requiere una selección adecuada de las variables a usar y que no tiene ninguna optimización sobre estas (Danny Varghese, 2018).

Por las desventajas nombradas anteriormente, los modelos de Regresión Logística se utilizan principalmente como una herramienta de inferencia y análisis de datos, donde el objetivo es comprender el papel de las variables de entrada. Decidimos usar este modelo como base ya que sabemos que no es el mejor modelo para predecir.

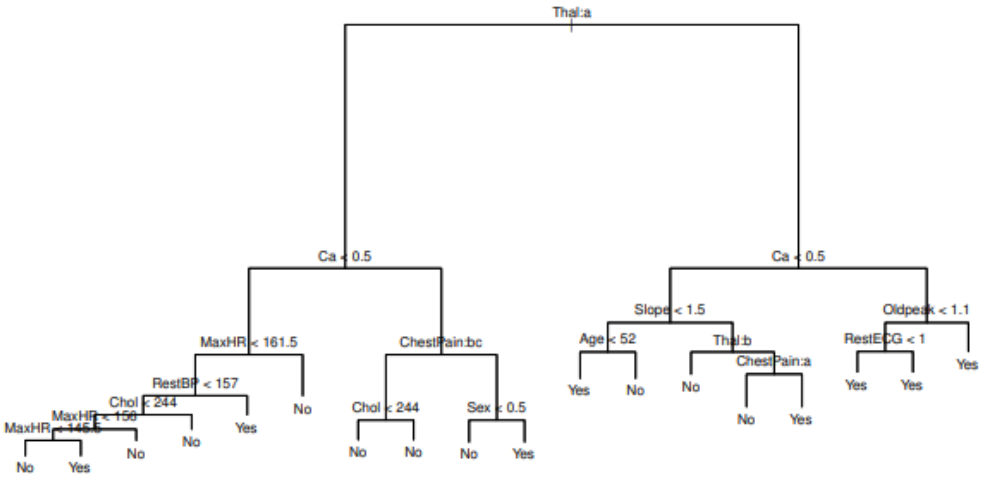
3.1.2 - Modelos de árboles

Para poder entender los modelos de Random Forest y LightGBM primero es necesario entender cómo funcionan los modelos de árboles, dado que estamos hablando de dos modelos que están dentro de la misma familia de algoritmos. Inicialmente, el modelo comienza a buscar un atributo con la mejor información obtenida en el nodo raíz y divide el árbol en subárboles. De manera similar, el subárbol se separa recursivamente siguiendo la misma regla. La partición se detiene si la hoja alcanza el nodo o no hay ganancia de

información. Una vez que el árbol está creado, las reglas se pueden obtener recorriendo cada rama del árbol y entendiendo las separaciones (Quinlan, 1993, 1996).

En el ejemplo de la Figura 13 podemos ver como la primera partición se da en la variable Ca <> 0.5. De un lado tenemos los que tienen esa variable mayor a 0.5 y del otro los menores. Esto se va a seguir separando por otras variables hasta que se llegue al punto de corte determinado.

Figura 15. Representación de árboles de decisión



Como criterio para hacer esta separación binaria se usan el Índice de Gini o la métrica de entropía. Ambas métricas suelen ser bastante similares y su función va a ser evaluar la calidad de las separaciones que va haciendo el árbol. Esta métrica va a definir la separación de las ramas con el objetivo de que las ramas sean lo más puras posibles para el objetivo de aprendizaje.

Para comprobar lo que llamamos pureza, podemos usar la métrica de entropía. La entropía es lo que nos permite medir la "incertidumbre" contenida en un conjunto de datos de entrenamiento, debido a la presencia de más de un clasificador posible.

Si hay K clases, podemos denotar la proporción de instancias con clasificación i por pi para i = 1 a K. El valor de pi es el número de ocurrencias de clase y dividido por el número total de instancias, es un número entre 0 y 1 inclusive. La entropía del conjunto de entrenamiento se denota por E. Se mide en 'bits' de información y se define mediante la fórmula:

$$E = -\sum_{i=1}^{K} p_i \log_2(p_i)$$

El valor de −pi log2(pi) es positivo para los valores de pi mayor que cero y menor que 1. Cuando pi = 1 el valor de −pi log2(pi) es cero. Esto implica que E es positivo o cero para todos los conjuntos de entrenamiento. Toma su mínimo valor (cero) si y sólo si todas las instancias tienen la misma clasificación, en la que caso solo hay una clase no vacía, para la cual la probabilidad es 1.

El "método de entropía" de selección de atributos es elegir dividir el dataset en el atributo que da la mayor reducción en la entropía (promedio), es decir, el que que maximiza el valor de la ganancia de información. Con ganancia de la información, nos referimos a la diferencia entre la entropía del dataset inicial y la entropía del dataset con la división en determinado atributo

En los árboles de decisión calcular la entropía es muy útil dado que ayuda a discernir qué variables generan una menor entropía si partimos la muestra inicial a partir de estas. La ganancia de información, derivada del cálculo de la entropía, permite conocer la homogeneidad de estas submuestras y terminar así mejorando las posibles predicciones.

El proceso de aprendizaje consiste en iterar en todos los nodos evaluando una posible separación y seleccionando el que tiene menor índice de Gini o entropía según lo que definamos.

Lo que se destaca al trabajar con modelos de árboles es lo siguiente: son muy fáciles de explicar, son similares a la toma de decisiones de los humanos, se pueden mostrar/entender en un gráfico con facilidad y algunos pueden soportar variables cualitativas sin la necesidad de la transformación en dummys, como por ejemplo el XGBoost.

Las desventajas son que es tan simple que no tiene buenos resultados en la práctica, en comparación con otros modelos y que pueden ser poco robustos. Si generamos árboles muy profundos, un cambio chico en el dataset puede generar cambios grandes en el output del modelo.

3.1.3 - Random Forest

Una forma de mejorar la performance del modelo de árboles es considerar sólo una parte de las observaciones y generar árboles individuales diferentes. Esto fue introducido por Ho (1995) con el nombre de random-subsample y continuado por Breiman (2001) que presentó formalmente el modelo de Random Forest.

Este modelo es un 'ensemble tree-based learning algorithm' es decir, un algoritmo que promedia las predicciones sobre muchos árboles individuales. Lo que hace es armar una

colección de árboles, intentando reducir la correlación entre ellos. Construimos un número de árboles de decisión en muestras de entrenamiento bootstrap. Con bootstrap, nos referimos a un número de subconjuntos de datos del mismo tamaño entre sí que son extraídos con reposición, esto permite que las muestras puedan estar en distintos subconjuntos repetidas veces. En el siguiente gráfico puede verse un ejemplo de la metodología Bootstrap. Dado que toda la información sale del mismo sitio, los modelos que entrenamos están fuertemente correlacionados.

Figura 16. Ejemplo de Boostrap

Al construir estos árboles de decisión, cada vez que se considera una división en un árbol, una muestra aleatoria de *m* variables predictoras se elige como candidatos divididos del conjunto completo de *p* variables predictoras. La división puede usar solo una de esas m predictoras. Podemos pensar en este proceso como una decoración de los árboles, lo que hace que el promedio de los árboles resultantes sea menos variable y por lo tanto más confiable.

3.1.4 - LightGBM

El modelo llamado LightGBM propone una mejora en comparación con otros modelos de árboles. Sus principales ventajas son el menor uso de memoria, una mejora en la velocidad y en la eficiencia del modelo y generalmente produce mejores resultados.

Este modelo utiliza dos tecnologías innovadoras para lograr estos mejores resultados. En primer lugar, utiliza lo que se llama Gradient-based One-Side Sampling, con el objetivo de conseguir un buen balance entre reducir el número de instancias de datos y poder mantener buenos resultados. Esto lo hace quedándose con todas las instancias con altos gradientes y haciendo un muestreo aleatorio en las instancias con gradientes cortos. Para compensar la influencia de la distribución de datos introduce una constante como multiplicador para las instancias de gradientes cortos. En segundo lugar, utiliza *Exclusive Feature Bundling* con el objetivo de reducir de manera eficiente la cantidad de *features*. Lo que hace es diseñar un algoritmo que pueda agrupar las distintas features en una llamada *exclusive feature bundle*. Este algoritmo construye los mismos histogramas de las *features*

individuales para las *features* agrupadas. Hace que la complejidad del modelo pase a ser *#data * #bundle*, que va a ser menor a la anterior que era *#data * #feature*. Reduciendo así los tiempos del modelo sin afectar su performance.

Aplicando estas variantes al modelo de gradient boosting decision trees, LightGBM ha demostrado una velocidad para el proceso de training de 20 veces más alta logrando mantener la misma accuracy (Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., & Liu, T. Y., 2017).
En la primera iteración de modelos, comparamos Regresión logistica, Random Forest y LightGBM. En la Sección 4.1 donde hablamos sobre la performance de los modelos, podemos ver que el que mejor performa es el LightGBM. Por lo tanto, vamos a entrar más en profundidad aplicando posibles mejoras al modelo base.

Para esto, es importante entender algunos conceptos. En primer lugar, vamos a introducir el término de hiperparámetroslos. Estos son parámetros cuyos valores controlan el proceso de aprendizaje y determinan los valores de los parámetros del modelo que termina aprendiendo de un algoritmo de aprendizaje. Estos son valores opcionales que si uno no se los asigna el modelo va a tomar una predeterminado.

En segundo lugar, es importante entender a qué nos referimos cuando hablamos de overfitting. Esto sucede cuando el modelo se ajusta en exceso a las particularidades de los datos de entrenamiento, y consecuentemente tiene mala performance en el testeo. Por lo contrario, nos referimos a underfitting cuando se trata de un algoritmo demasiado rígido que no captura patrones relevantes, consecuentemente tiene mala performance en entrenamiento y testeo.

En este modelo vamos a experimentar con equilibrar el desbalance de las clases, optimizar hiperparametros para encontrar el mejor valor a implementar con el objetivo de controlar el overfitting del modelo y regularizar para disminuir la varianza de este. A continuación vamos a estar hablando de la métrica que nos definió la selección del modelo de LightGBM como mejor performance e indagar sobre los distintos pasos del modelo mencionados en este párrafo.

En la siguiente tabla se pueden ver los hiperparamétricos que vamos a estar usando para intentar mejorar la performance del modelo base y el significado de cada uno de ellos. Estos hiperparámentros ayudan a controlar el overfitting y a disminuir la varianza del modelo.

Tabla 1. Hiperparámetros *LightGBM*

Hiperparámetro	Descripción
num_leaves	Setea el máximo número de nodos por árbol. Se usa para limitar la complejidad de los árboles
min_child_sample	Número mínimo de datos en una rama. Parámetro muy importante para evitar overfitting
max_depth	Limita la profundidad máxima para el árbol.
min_child_weight	Mínima suma de hessian en una rama
subsample	Usa bagging para seleccionar aleatoriamente parte de la data sin remuestreo.
subsample_freq	Frecuencia de bagging
colsample_bytree	Selecciona aleatoriamente una parte de los features para cada observación
learning_rate	Shrinkage rate
n_estimators	Número de iteraciones de boosting
max_bin	Máximo número de clusters que los valores de las futures pueden ser separados
reg_alpha	Agrega una penalidad que es igual a la suma del valor absoluto del coeficiente
reg_lambda	Agrega una penalidad que es igual a la suma de la raíz cuadrada del coeficiente
min_split_gain	La ganancia mínima para realizar el split

3.2 - Evaluación de modelos

3.2.1 Métrica de performance de modelos

Nuestra métrica seleccionada para la evaluación de nuestros modelos de machine learning fue el área bajo la curva de ROC (AUC-ROC). Esta es muy común para presentar resultados de problemas de decisión binarios. Lo que hace es relacionar la tasa de verdaderos positivos (TPR = $\frac{TP}{P}$) con la de falsos positivos (FPR = $\frac{FN}{N}$) (Davis, Goadrich, 2006).

Llamamos verdadero positivos al porcentaje de usuarios correctamente etiquetados como positivos. En este caso, serían todos los usuarios que predecimos que no iban a volver y realmente no volvieron dividido el universo total de usuarios que predije que no iban a

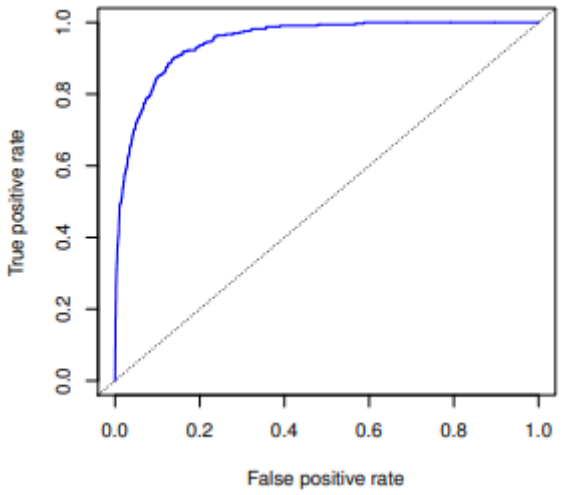
volver. Por lo contrario, llamamos falsos negativos al porcentaje de usuarios incorrectamente etiquetados como negativos. En este caso, serían todos los que predije que iban a volver y no volvieron sobre el total de usuarios que predije que no iban a volver.

Esta métrica muestra el rendimiento general de un clasificador resumido en todos los umbrales posibles. Cuando hablamos de umbral, nos referimos al valor de probabilidad en el cual uno va a decidir empezar a validar los usuarios como positivos o negativos. Este es un punto de corte que por default se suele tomar en 0,5 pero no tiene que ser necesariamente así.

La métrica de ROC va a tomar valores en un rango de de [0 ; 1], cuanto mayor sea el área bajo la curva implica una mejor performance del modelo (Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, 2021).

La curva se obtiene graficando las diferentes combinaciones del ratio de los verdaderos positivos en el *eje y* el ratio de los falsos positivos en el *eje x*. Un valor de 0.5 es el mínimo esperado, dado que este sería igual a predecir de forma azarosa el resultado. Este valor se da cuando los falsos positivos se igualan con los falsos negativos, se puede ver en la línea gris representada en el gráfico. El máximo valor es 1 donde estaríamos produciendo de forma perfecta, esto se da en el caso donde no tenes ningún falso positivo, todos los valores estarían bien predichos.

Figura 17. Curva de ROC



En el paper *The use of the area under the ROC curve in the evaluation of machine learning algorithms*, (Bradley, 1996) se hace una comparación, corriendo seis modelos diferentes, entre la métrica de AUC-ROC contra la métrica de accuracy (la probabilidad de tener una respuesta correcta). En esta comparación llegan a la conclusión de que la curva de ROC

muestra una mayor sensibilidad en el análisis de la varianza, un error estándar que disminuyó cuando las dos AUC-ROC y el número de muestras de prueba suben, independencia del umbral de decisión e invariante a la clase a priori a las probabilidades de clase.

Por otro lado, como mencionamos en la Figura 4 de la Sección 2.2, contamos con datos balanceados. Cuando miramos la métrica de AUC-ROC lo que vemos es cuanto es el total del area bajo la curva, sin pensar en la forma en la que está formada. Pero esto solo funciona para datos balanceados. Si contamos con datos no balanceados, tenemos que tener en cuenta la forma que adquiere la curva en representación del verdadero positivo y falso positivo, por lo tanto usaríamos la métrica de Weighted AUC (Weng, C. G., & Poon, J.,2008).

Por lo mencionado anteriormente, creemos que la métrica de AUC-ROC es la mejor métrica para entender qué modelo podría funcionar mejor en la práctica y definirlo como modelo final.

3.2.2 Optimización de hiperparametros

La optimización de hiperparámetros es sumamente importante y tiene un impacto muy fuerte en la performance del modelo. Este proceso consta en probar distintos valores de hiperparametros para poder definir cual es el que mejor performance va a tener en el modelo.

Las dos estrategias que pensamos para este proceso fueron 'grid search' y 'random search'. La primera lo que hace es probar cada combinación posible con cada valor prefijado, pero es muy costoso computacionalmente. Random search lo que hace es dar un rango que definimos de posibles valores para cada parámetro y aleatoriamente selecciona un valor que esté en el rango. Luego, se entrena el modelo con estos valores aleatoriamente seleccionados y se compara la performance de cada uno para determinar cuál es el mejor modelo.

En este caso, la estrategia para encontrar los mejores parámetros que decidimos utilizar fue la de 'random search'. Esta decisión la tomamos debido a que se demostró que es más eficiente en términos de la baja de tiempo computacional y en la mayoría de los casos muestra mejores resultados (Bengio, Bergstra, 2012).

Con la estrategia de optimización de hiperparámetros tenemos dos objetivos principales: controlar el overfitting y regularizar el modelo con el fin de disminuir la varianza. Con

estos dos objetivos en mente, hicimos la selección de los hiperparametros que íbamos a optimizar. Se pueden encontrar en la Tabla 4.

Para definir el rango con el cual íbamos a probar los hiperparámetros lo que hicimos fue tener en cuenta los valores de estos en el modelo base y contemplar que estos rangos los involucren y tengan valores por arriba y por abajo de este. Los hiper parámetros y los rangos que fueron probados mediante esta estrategia de random search fueron los siguientes:

Tabla 2. Pruebas de Hiper Parámetros *LightGBM*

Hiperparámetro	Rango	Valor optimo
num_leaves	[2; 50]	24
min_child_sample	[0; 30]	5.5
max_depth	[-1; 20]	19
min_child_weight	[1; 10]	5
subsample	[0.1; 1.5]	10
subsample_freq	[0; 16]	0.8
colsample_bytree	[0.1; 1.5]	40
learning_rate	[0.0005, 0.005, 0.05, 0.1, 0.2, 0.3, 0.4, 0.6, 0.8]	0.05
n_estimators	[25; 1025]	175
max_bin	[10; 510]	30
reg_alpha	[0; 20]	1
reg_lambda	[0; 20]	3
min_split_gain	[0; 30]	0

3.2.3 Cross Validation

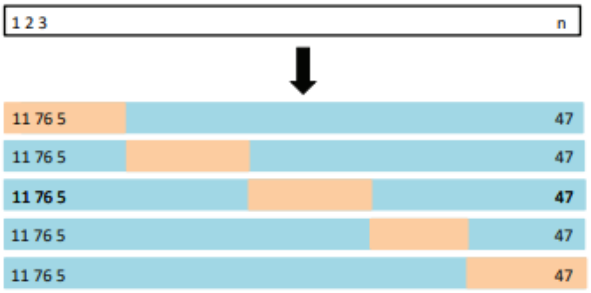
La validación cruzada es el proceso de medir la capacidad de generalización de diferentes modelos probándolos con nuevos datos, no vistos durante el entrenamiento, y luego eligiendo el más preciso en este conjunto de datos. Hay diferentes técnicas disponibles para realizar una validación cruzada. En este proyecto vamos a usar K-fold.

Esta técnica consiste en dividir aleatoriamente el set de datos en k clusters del mismo tamaño. El primer cluster es tomado como validation set y el modelo se entrena con k-1

clusters, esto corre repetidas veces y va cambiando el grupo que queda afuera como validation set. Al final del modelo tenemos k diferentes estimadores de la performance del modelo que son promediados para tener la performance final.

Este proceso lo usamos para comparar la performance de diferentes combinaciones de hiperparamentros para el mismo modelo. Esto lo hacemos así dado que si probamos muchos modelos diferentes con el mismo validation set lo que puede tender a pasar es que cometamos overfitting. Esta iteración en el validation set nos ayuda a evitar uno de los principales problemas de los modelos.

Figura 18. Ejemplo de *cross validation*



3.2.4 Ingeniería de atributos

La ingeniería de atributos es un paso muy importante para la performance del modelo dado que hay algunos formatos de datos que el modelo no soporta. Pero también hay datos numéricos que hay que entender si necesitan algún tipo de transformación para no generar un mal entendimiento en el modelo.

Lo que hicimos en este paso previo a correr el modelo fue lo siguiente:

- Transformación de fechas

  En este modelo contamos con 3 variables de fecha: *intall_time*, *fecha_creacion_cuenta* y *fecha_primera_tx*. Como el modelo no puede correr con estas variables, lo que decidimos es hacer una transformación en función de lo que creemos que más impacto puede tener en la predicción. Las dos cuestiones que creímos importantes para captar en el modelo son el día de la semana y el día del mes en el que se realizaron estas tres acciones (instalación de la aplicación en el celular, creación de la cuenta y la primera transacción). Lo vemos importante dado que sabemos que nuestro pico de transacciones se hace presente en los primeros días del mes y que

el día de la semana en el que más se transacciona dentro de la *App* es el lunes.

- Valores nulos

  Como vimos en la Sección 2.2 el dataset cuenta con 578.396 valores nulos en variables numéricas. Por lo tanto, decidimos reemplazar estos valores nulos por 0 dado que el 0 simula correctamente el comportamiento de los usuarios cuando esas variables están en nulo. En otras palabras, es un buen número que refleja el verdadero comportamiento de los usuarios.

- Transformación de variables categóricas

  El dataset cuenta con una sola variable categórica que es el media_source. Esto nos define por qué medio de marketing adquirimos al usuario. Creemos que es una variable con mucha importancia, dado que podemos tener canales que nos traen una mejor o peor calidad de usuarios en términos de métricas de activación y retención. Como el modelo no acepta variables categóricas lo que decidimos fue usar One-hot Encoding.

  Está técnica lo que hace es transformar los valores únicos de las filas de la variable categórica en columnas donde van a tener un 1 si contenían ese nombre y un 0 en el caso contrario. Esto es posible dado que tenemos solamente una variable con 20 valores únicos, dado que si contamos con más variables puede enlentecer el modelo el seguir agregando columnas. Esto permite una transformación simple y fácil de interpretar. Para aplicar esto usamos la función get_dummies[3] de pandas en Python.

- Normalización de variables:

  Teniendo en cuenta que el *qty_errores* es posible que aparezca más veces cuanto más usas la aplicación decidimos normalizar. Lo que hicimos fue dividir el valor de cantidad de errores sobre la cantidad de veces que el usuario entra a la pantalla de servicios y a la de recargas. Esto nos permite quitar el efecto de mayor uso de la app y ver un verdadero número aproximado del error.

3.2.5 Shap (*SHapley Additive exPlanations*) Values

[3]Documentación de get_dummies:
https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html

Entender por qué el modelo hace ciertas predicciones se vuelve tan importante como poder medir la performance del modelo. Sin embargo, las mejores métricas de performance se obtienen de modelos cada vez más complejos donde se vuelve más difícil la interpretación de los mismos. Para solucionar este problema, surge SHAP (*SHapley Additive exPlanations*), con el objetivo de asignarle a cada *feature* un valor de importancia para determinada predicción. (Lundberg, S. M., & Lee, S. I., 2017).

SHAP calcula el impacto de cada función en las predicciones realizadas por el modelo aprendido. En SHAP, dado una entrada $x = \left[x_1, .., x_p \right]$ y un modelo entrenado f, aproxima f con un modelo simple g que puede explicar fácilmente la contribución de cada valor de característica. El modelo g se puede formular de la siguiente manera:

$$g(z) = \phi_0 + \Sigma_{i=1}^{p} \phi_i z_i \quad (3)$$

Donde p es el número de features y z es una simplificación del input de x donde el valor de z correspondiente a las features usadas en la predicción es 1 y el valor de z correspondiente a las no usadas es 0. ϕ_i representa la contribución de cada una de las features, que se calcula con la siguiente ecuación:

$$\phi_i(f, x) = \Sigma_{z \subseteq x} \frac{|z|!(p-|z|-1)!}{p!} [f(z) - f(z/i)] \quad (4)$$

Esto es lo que llamamos SHAP value y es equivalente al valor de Shapley en la teoría de juegos. El valor de Shapley es un valor que representa la contribución de cada jugador cuando los jugadores cooperan en un juego con múltiples jugadores. Por lo tanto, SHAP calcula el valor de Shapley de cada característica como jugador en el modelo aprendido. El cálculo de los valores SHAP debe realizarse para todas las permutaciones de características, lo que requiere un tiempo exponencial. Sin embargo, se sabe que SHAP se puede calcular de manera eficiente para modelos con estructura de árbol. Dado que el modelo de aprendizaje utilizado en este estudio es un árbol potenciador de gradiente (LightGBM), se puede reducir el tiempo de cálculo del algoritmo SHAP (Futagami, K., Fukazawa, Y., Kapoor, N., & Kito, T. (2021)).

3.2.6 Experimentos

El arte de diseñar un experimento y el arte de analizar un experimento están estrechamente entrelazados y hay que estudiarlos uno al lado del otro. Al diseñar un experimento, se debe tener en cuenta el análisis que será realizado. A su vez, la eficiencia del análisis dependerá del experimento particular.

A la hora de crear un experimento, es necesario tener en cuenta el paso más importante, la planificación de este. Para entender mejor estos pasos y planificarlo nos basamos en el libro de *"Design and Analysis of Experiments of Angela Dean, Daniel Voss y Danel Draguljić".*

Los pasos a seguir para la planificación son:

1. Definir el objetivo del experimento: se recomienda hacer una lista de preguntas que se abordarán a lo largo del experimento. Esto nos va a ayudar a determinar las decisiones requeridas al momento de la verificación de resultados.
2. Identificar todas las fuentes de variación: una fuente de variación es cualquier cosa que pueda causar que una observación tenga un valor numérico diferente.
3. Elegir una regla para asignar los usuarios experimentales al experimento: especifica qué unidades experimentales se van a observar bajo qué tratamientos. La elección del diseño, que puede o no implicar a la etapa anterior dependiendo de todas las decisiones tomadas hasta ahora en la lista de verificación.
4. Especificar las medidas a realizar, el procedimiento experimental y las dificultades previstas: los datos (u observaciones) recopilados de un experimento son mediciones de una variable de respuesta. Por lo tanto, deben especificarse las unidades en las que se realizan las mediciones y estas deben reflejar los objetivos del experimento. Por lo general, hay dificultades imprevistas en la recopilación de datos, pero a menudo se pueden identificar tomando algunas medidas de práctica o ejecutando un experimento piloto.
5. Realizar un experimento piloto: Un experimento piloto es un pequeño experimento que involucra solo unas pocas observaciones. No hay conclusiones necesariamente esperadas de tal experimento. Se ejecuta para ayudar a completar la lista de verificación. Brinda la oportunidad de practicar la técnica experimental y de identificar insospechados problemas en la recogida de datos. Si el experimento piloto es lo suficientemente grande, también puede ayudar en la selección de un modelo adecuado para el experimento principal.
6. Especificar el modelo: El modelo debe indicar explícitamente la relación que se cree que existe entre la respuesta variable y las principales fuentes de variación que se identificaron en el paso dos.
7. Resuma el análisis: se deben destacar los puntos claves del análisis, incluyendo la hipótesis y los intervalos de confianza. Este paso determina los cálculos y verifica que el diseño sea adecuado para llegar al objetivo

Por lo mencionado anteriormente, entendemos que dedicar un poco más de tiempo a la planificación ayuda a garantizar que los datos puedan ser utilizados con la máxima ventaja.

Ningún método de análisis puede salvar un experimento mal diseñado desde sus comienzos.

La experimentación nos va a ser útil para entender si los modelos de Machine Learning de los que vamos a hacer uso, realmente generan un impacto para la empresa o no. Esto nos va a permitir ejecutar distintos experimentos con las probabilidades que nos brindan los modelos y evaluar si estamos generando una mejora en el negocio.

3.2.7 A/B Testing

Una de las estrategias más comunes de experimentación en Marketing es la llamada A/B Testing.

Las pruebas A/B se definen como un mecanismo para comparar dos versiones de una implementación o solución específica con el objetivo de poder descubrir cuál de los dos funciona mejor. Si bien se asocia con mayor frecuencia con sitios web y aplicaciones, el método tiene casi 100 años y es una de las formas más simples para tener un experimento aleatorio controlado.

En la década de 1920, el estadístico y biólogo Ronald Fisher descubrió los principios más importantes detrás de las pruebas A/B y los experimentos aleatorios controlados en general. El método de prueba ha ganado popularidad en las últimas dos décadas a medida que las empresas se dieron cuenta de que el entorno online es el más adecuado para ayudar a las diferentes áreas dentro de la empresa a responder preguntas como: "¿Qué es lo más probable que haga que los usuarios hagan clic? ¿O comprar nuestro producto? ¿O registrarse en nuestro sitio?". En la actualidad, este método se usa para evaluar todo, desde el diseño del sitio web hasta las ofertas en línea, los titulares y las descripciones de los productos (Amy Gallo, 2017).

Al asociarlo con empresas tenemos dos grandes áreas en las que se suele usar frecuentemente esta metodología. En el área de Marketing normalmente se generan dos versiones diferentes de un diseño y se evalúa cuál es el que mejor funciona para el cliente en base a un objetivo de Marketing planteado. Por diseño nos referimos al creativo de un email, push o *in App Message* (in-apps) o a una promoción donde vamos a buscar evaluar la recepción del usuario frente a estas diferentes opciones. En el área de Producto, generalmente, se crean dos versiones distintas de una determinada página o parte de la App y se prueba cual de ellas es la que muestra mejoras en la conversión que estén buscando perfeccionar.

La parte más difícil de un A/B test es determinar qué es lo que se quiere evaluar en un primer lugar. Un error que cometen algunas empresas es empezar a mover un gran número de palancas alrededor sin una planificación clara por adelantado para lo que están tratando de optimizar y qué es a lo que se va a impactar con estos cambios. En el libro "*A/B Testing: The Most Powerful Way to Turn Clicks Into Customers*" de Dan Siroker y Pete Koomen, se plantean los pasos a tener en cuenta a la hora de empezar con el armado de un A/B Test. Los pasos a seguir son los siguientes:

1. Definir el éxito: implica encontrar una métrica de éxito cuantificable para las preguntas que queremos responder con esta experimentación. Las métricas de éxito son los números específicos que se espera que sean mejorados por las pruebas.
2. Encontrar el cuello de botella: el objetivo de este paso es encontrar cual es el problema que se está intentando solucionar para el negocio con el experimento.
3. Construir una hipótesis: la hipótesis hace que las pruebas sean más informativas porque brindan un propósito específico ayudando a perfeccionar lo que se está buscando determinar. Si se ejecuta un experimento sin formar una hipótesis de antemano, puede recopilar información que va a ser útil anecdóticamente mientras se pierde la lección más profunda.
4. Hacer el experimento: lo que queda es ejecutar la prueba y hacer un seguimiento de que la misma se implemente correctamente.
5. Mostrar resultados: una vez que el experimento haya alcanzado una significancia estadística en los resultados podemos definir cuales de las variantes fue la que mejor performó en la práctica.

Dados estos pasos a seguir, buscamos distintas recomendaciones que hay que tener en cuenta a la hora de pensar cómo hacer un A/B Testing. Dado que el experimento lo vamos a hacer en una herramienta que se llama Braze, decidimos tomar sus sugerencias a la hora de implementarlo.

Braze es la herramienta líder en *Mobile Engagement Automation* a nivel mundial. Esta herramienta tiene varias funcionalidades pero nos vamos a concentrar en las dos más importantes de las que vamos a hacer uso. La primera es la de segmentación, que nos permite agrupar a los usuarios en *clusters* o grupos según características que definamos. La segunda es la herramienta de *canvas*, que se definen como flujos de distintas comunicaciones (emails, pushs o in-apps) que pueden ser enviadas a un segmento específico o también enviarse a partir de que se genera una acción específica en la aplicación.

Esta herramienta, en la sección de User Guide, en un artículo llamado *'Multivariate and A/B testing'* nos provee de cinco recomendaciones a la hora de hacer el A/B Testing:

- **Ejecutar la prueba con la mayor cantidad de usuarios posibles**: las muestras grandes aseguran que los resultados reflejen las preferencias del usuario promedio y es menos probable que se dejen influir por valores atípicos. Los tamaños de muestra más grandes también permiten identificar variantes ganadoras que tienen márgenes de victoria más pequeños.
- **Clasificar aleatoriamente a los usuarios en diferentes grupos de prueba**: la aleatorización está diseñada para eliminar el sesgo en el conjunto de prueba y aumentar las probabilidades de que los grupos de prueba tengan una composición similar. Esto asegura que las diferentes tasas de respuesta se deban a diferencias en sus mensajes en lugar de sus muestras.
- **Saber qué elementos está tratando de probar**: Las pruebas A/B permiten probar las diferencias entre varias versiones de dos estrategias distintas de comunicación. En algunos casos, una prueba simple puede ser más efectiva, ya que aislar los cambios permite identificar qué elementos tuvieron el mayor impacto en la respuesta. Otras veces, presentar más diferencias entre variantes le permitirá examinar valores atípicos y comparar diferentes conjuntos de elementos. Ninguno de los métodos es necesariamente incorrecto, siempre que tenga claro desde el principio lo que está tratando de probar.
- **Decidir cuánto tiempo se ejecutará la prueba antes de comenzar y no finalice su prueba antes de tiempo**: Los especialistas en marketing a menudo se ven tentados a detener las pruebas después de ver los resultados que les gustan, lo que sesga sus hallazgos. Es muy importante no cortar la prueba antes de lo que predeterminar desde un principio.
- **Si es posible, incluya un grupo de control**: incluir un grupo de control le permite saber si sus mensajes tienen un mayor impacto en la conversión de usuarios que no enviar ningún mensaje.

Este es el método más clásico en *Customer Retention Management*, donde las empresas suelen dividir a sus usuarios aleatoriamente y probar distintos mensajes, creativos o promociones en los email, pushs o in-apps que se le muestran. Estas versiones comparten objetivos de Marketing similares pero difieren en redacción y estilo.

Dado este contexto, para entender los experimentos es importante aclarar algunas métricas que se usan para medir el éxito de los mismos. En todas las pruebas es recomendable dejar un grupo de control, este es un porcentaje de la base de usuarios que

vamos a separar aleatoriamente y no aplicarles el tratamiento de interes. Luego de transcurrir el experimento, vamos a comparar los resultados contra este grupo para entender la incrementalidad real de la estrategia que estamos implementando. Siguiendo esto, vamos a llamar a los usuarios que reciben el experimento Target Group y a los usuarios que dejamos fuera Control Group.

Para la medición de estos experimentos, vamos a tener que evaluar distintas métricas. La primera importante a resaltar es el *Conversion Rate* (CVR) o, en español, ratio de conversión. Está métrica de conversión va a ser definida según el contexto y aplicación del experimento. Un ejemplo de conversiones puede ser hacer un click o una compra. En definitiva es realizar cierta acción que queremos medir. Está métrica la vamos a usar como métrica de comparación entre los distintos segmentos.

$$Conversion\ Rate\ (CVR) = \frac{Usuarios\ que\ convirtieron}{Total\ usuarios\ que\ impactados} \quad (5)$$

En segundo lugar, la métrica que vamos a querer medir es lo que llamamos la incrementalidad que se puede ver en la siguiente ecuación. Lo que nos va a decir es el porcentaje de mejora de nuestra estrategia implementada en comparación con el grupo de control.

$$Incrementalidad = \frac{(CVR\ TG - CVR\ CG)}{CVR\ TG} \quad (6)$$

Es importante tener en cuenta estas dos métricas de performance a la hora de evaluar los resultados de los A/B testings.

Por último, nos parece importante aclarar cómo revisamos la significancia estadística de estos experimentos. La misma herramienta mencionada anteriormente, Braze, nos permite correr los *tests* estadísticos que nos arrojan el nivel de confianza de cada uno de los resultados que uno desea evaluar. Para esta medición, la herramienta utiliza el 'Intervalo de confianza de Wilson'. Este intervalo propone una mejora al intervalo de aproximación normal.

El intervalo de aproximación normal se basa en la aproximación de la distribución del error sobre una observación distribuida binomialmente, con una distribución normal. Esta aproximación se basa en el Teorema Central del Límite y no es confiable cuando el tamaño de la muestra es pequeño o la probabilidad de éxito es cercana a 0 o 1.

El intervalo de puntuación de Wilson propone una mejora con respecto al intervalo de aproximación normal en múltiples aspectos. Fue desarrollado por Edwin Bidwell Wilson (1927). A diferencia del intervalo de aproximación normal, el intervalo de puntuación de

Wilson es asimétrico. Por lo tanto, no sufre problemas de sobreimpulso ni de intervalos de ancho cero que afectan al intervalo normal, y puede emplearse de forma segura con muestras pequeñas y observaciones sesgadas. (Lott, A., & Reiter, J. P. ,2020)

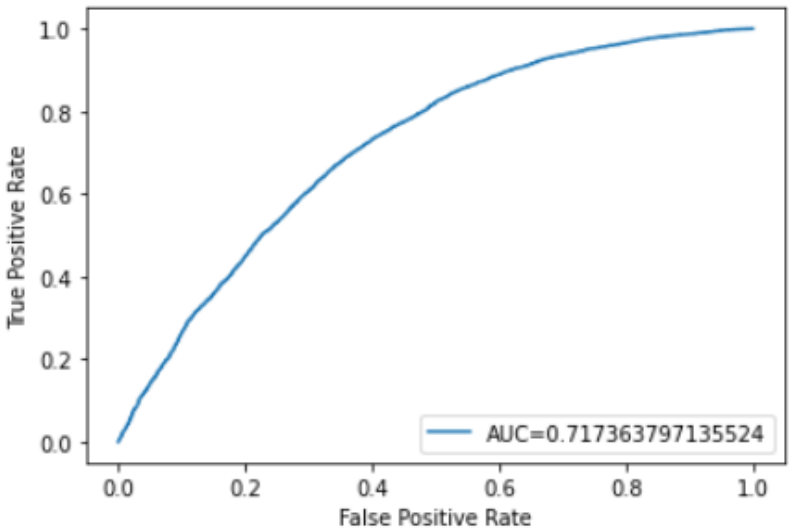
4. Resultados

4.1 - Performance de los Modelos

Como mencionamos en el inciso 3.1, en este proyecto entrenamos tres diferentes modelos básicos para definir en cual nos íbamos a terminar enfocando. Estos los consideramos modelos ingenuos, dado que no son complejos y hace uso de los parámetros por default que tienen los modelos.

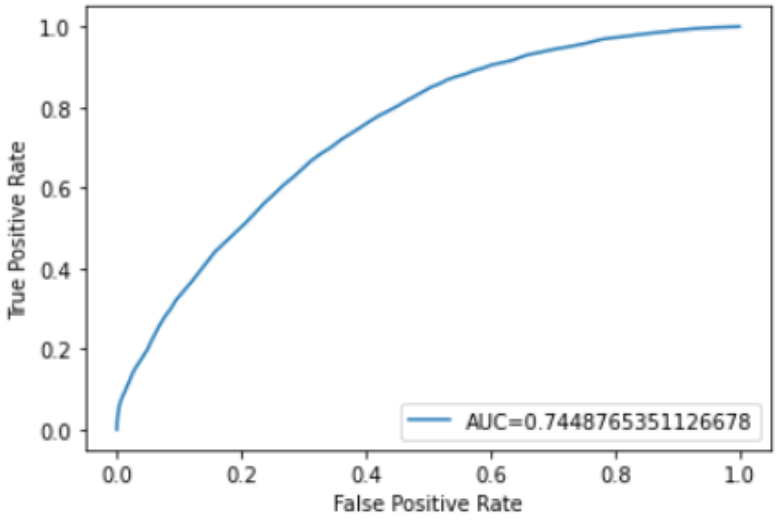
El primer modelo, es una regresión logística, lo seleccionamos como primer modelo base dado que sabemos que este asume que los datos son linealmente separables y en este tipo de problemas de clasificación suele tener una peor performance que los modelos de árboles. Como podemos ver en el siguiente gráfico la performance la medimos por la métrica de AUC (área bajo la curva de ROC) y fue de un 71%.

Figura 19. Curva de ROC - Modelo Regresión Logística



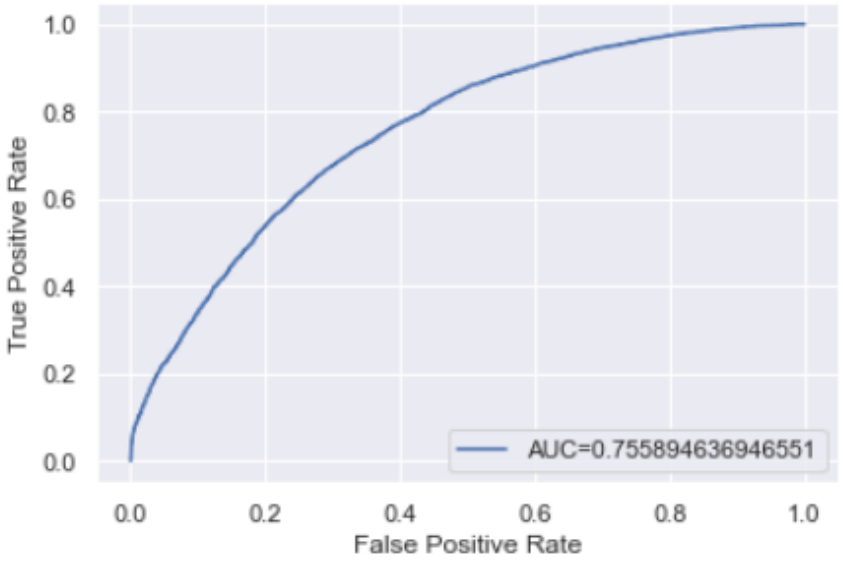
Dada la performance del modelo de Regresión logística, decidimos continuar con un modelo de árboles, también en su versión más simple, para intentar mejorarla. En segundo lugar, entrenamos un modelo base de Random Fores utilizando también cross validation para el testeo. La performance de este modelo superó en un 4% la performance del modelo de regresión logística. El AUC de este modelo es de un 74%.

Figura 20. Curva de ROC - Modelo Random Forest



Por último, decidimos entrenar un modelo de LightGBM. Donde los resultados superaron los de los primeros dos modelos. El AUC nos da un 75.6%. Supera en un 2% la performance del modelo anterior.

Figura 21. Curva de ROC - Modelo LightGBM



Dado estos tres resultados, decidimos seguir perfeccionando el modelo de LightGBM donde vamos a hacer balanceo de datos y pruebas de hiperparametros para poder controlar el overfitting y la varianza. Luego de estas pruebas vamos a definir cuál es el mejor modelo según el AUC y poner en práctica el modelo.

Tabla 3. Resultados modelos LightGBM

	Tipo de modelo	AUC	MEJORAS

1	LightGBM con hiperparámetros por *default*	0.756	
2	LightGBM con equilibrio de data no balanceada y hiperparametros por *default*	0.756	0.00%
3	LightGBM con *Cross Validation* con *Random Search* de hiperparámentros: *num_leaves, min_child_samples, max_depth y min_child_weight, subsample, subsample_freq, colsample_bytree, and max_depth, learning_rate, n_estimators, and max_bin, reg_alpha, reg_lambda y min_split_gain*	**0.757**	**0.13%**

En el segundo modelo, para equilibrar la data no balanceada, usamos un hiperparámetro de este modelo que se llama *'scale_pos_weight'.* Para calcular el valor que tenía que llevar este parámetro contamos todos los valores de flag_m1 = 0 sobre todos los valores de flag_m1 = 1. En este caso no vemos una mejora en la performance, dado que no contabamos con un alto desbalance de estas.

El modelo demostró un cambio con la prueba de hiperparametros y logramos una mejora de la performance de un 0.13%. En este, hicimos random search con los parámetros que se mencionan en la tabla y encontramos sus óptimos.

En la siguiente tabla pueden se puede ver entre qué rangos fueron probados los hiperparametros y los resultados óptimos que fue arrojando cada modelo. El que seleccionamos como mejor modelo, contempla los valores óptimos hasta el max_bin.

Tabla 4. Optimización hiperparámetros modelos LightGBM

Hiperparámetros	**Rangos**	**Valor óptimo**
num_leaves	[2; 50]	24
min_child_sample	[0; 30]	5.5
max_depth	[-1; 20]	19
min_child_weight	[1; 10]	5
subsample	[0.1; 1.5]	10
subsample_freq	[0; 16]	0,8

colsample_bytree	[0.1; 1.5]	40
learning_rate	[0.0005, 0.005, 0.05, 0.1, 0.2, 0.3, 0.4, 0.6, 0.8]	0.05
n_estimators	[25; 1025]	175
max_bin	[10; 510]	30
reg_alpha	[0; 20]	1
reg_lambda	[0; 20]	3
min_split_gain	[0; 30]	0

4.2 - Importancia de las variables

Para calcular esta métrica utilizamos el método de SHAP values explicado en la Sección 3.2.5. Para esto usamos una librería de python que nos simplifica la implementación de esta metodología para medir la importancia de las variables que se llama SHAP.

En el siguiente gráfico podemos ver representado en el eje *y* las primeras 20 variables con mayor importancia de todas las que seleccionamos anteriormente para el modelo. En el eje *x*, vemos con un gráfico de barras el shap value de las variables del eje y. Este representa una medida de la importancia de las variables.

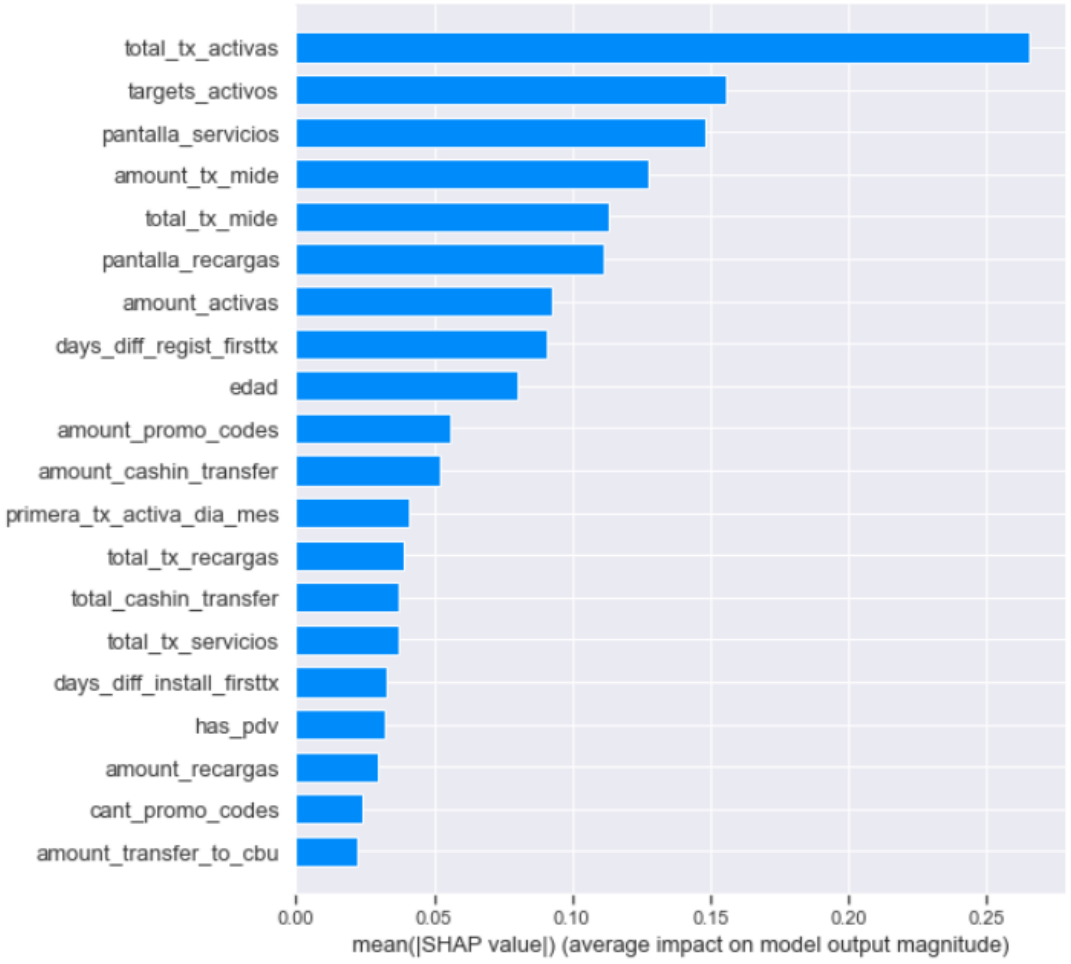
Podemos notar que la primera variable tiene un shape value un 66% frente a la segunda, la variable más relevante para explicar si el usuario va a ser un usuario churn es el total de transacciones activas. Con esta variable, tenemos en cuenta la cantidad de pagos de servicios, recargas, pagos con tarjeta o pagos con QR que tiene un usuario en particular, en sus primeros 20 días de vida.

En segundo lugar aparece una variable que cuenta la cantidad de servicios distintivos que pagó una persona en sus primeros 20 días de vida. En tercer lugar se encuentra la variable de *pantallas_servicios* que indica la cantidad de veces que el usuario entró a ver esa pantalla sin importar si terminó transaccionando o no. En cuarto y quinto lugar tenemos la cantidad de transacciones y monto transaccionado que tiene un usuario en Mide, como explicamos anteriormente, este es el producto con mayor retención en la aplicación y tiene sentido que si estamos ante un usuario que transacciona este producto sea mucho más probable que se quede en comparación con un usuario de servicios.

Viendo este gráfico lo que más nos llama la atención es la diferencia entre la importancia de las variables amount_promo_code y total_promo_codes. Esto lo que nos dice es que no es tan importante la cantidad de promociones que uno da sino el valor de las mismas, teniendo el doble de importancia el valor en la predicción.

Otro valor que nos llama la atención, es la edad del usuario. Creemos que dada la importancia de esta variable, es importante para el negocio entender a qué edad los usuarios es más probable que se queden para armar una estrategia de branding para llegar ese público determinado.

Figura 22. Importancia de las variables en el modelo

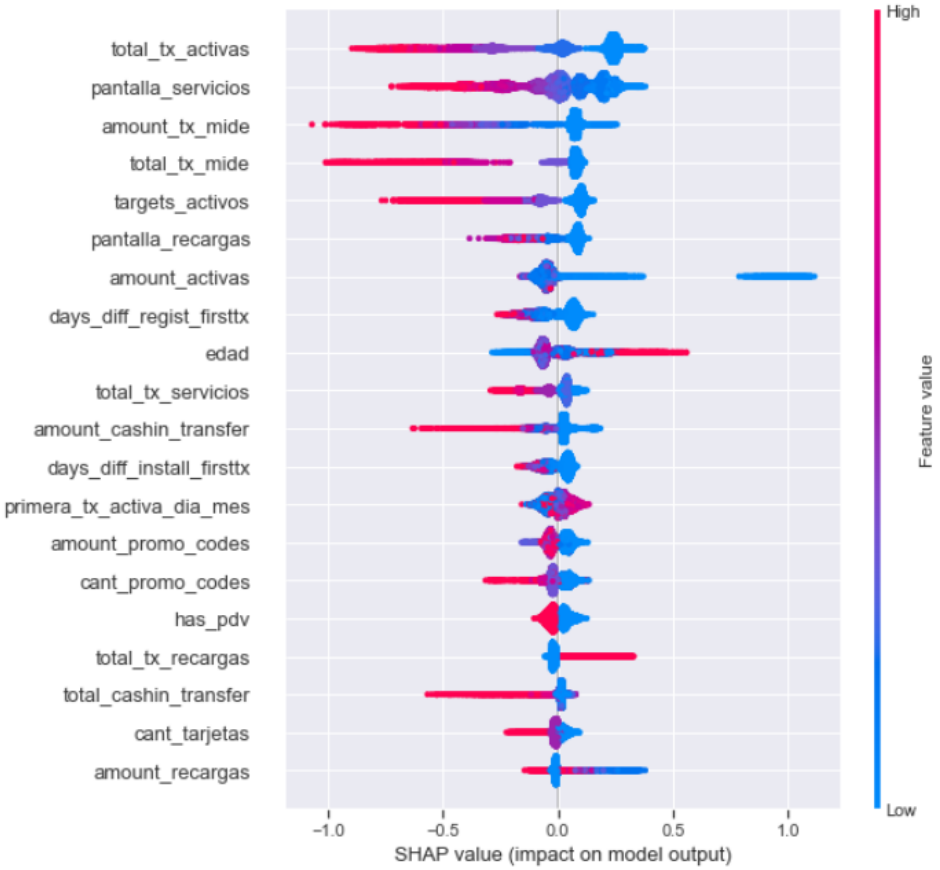


Otra figura interesante para la evaluación de los resultados fue la Figura 20. En esta podemos ver el orden de la importancia de variables determinado por el shap value en el eje y. Para cada variable, el color de los puntos está determinado por el valor de la misma. A valores altos se va a encontrar en rosa y a valores bajos se va a encontrar en azul. Un ejemplo claro es la cantidad total de transacciones, a medida que menos transacciones tenga el usuario, su shap value es mayor. Esto nos dice que menor cantidad de transacciones van a llevar a una más alta predicción de usuarios churn. Para poder terminar de entender este gráfico, otro ejemplo diferente es la variable de edad. En ese caso a valores mayores de edad tenes un shap value mayor. Por lo tanto a mayor edad el usuario va a tener mayor probabilidad de ser un usuario churn.

Otro valor que vemos importante resaltar es la importancia de la variable amount_promocodes . Esperábamos encontrar con que a mayor cantidad de dinero entregado por usuario en su primer mes en la aplicación, más chances tengas de ser un usuario churn. Dado que creíamos que había una gran masa de usuarios que solo se descargan la aplicación por el descuento y luego la dejaban de usar. Lo que vemos es lo contrario, cuanto más plata te doy en tu primer mes de vida mayor probabilidad tiene el usuario de quedarse.

Por último resaltamos el valor de recargas de celular como extraño. A mayor cantidad de este tipo de transacciones el usuario tiene un mayor shap value, es decir, mayor posibilidades de ser un usuario inactivo.

Figura 23. Shap values y su relación con el valor de las variables



Este gráfico nos parece fundamental para entender la relación que tienen los valores de nuestras variables en la predicción que estamos haciendo. Esta relación, nos hace interpretar mucho mejor al modelo y poder accionar sobre relaciones que no sabíamos que existían. Un ejemplo es poner más foco a los usuarios de recarga de celular que antes no los teníamos en cuenta para ponernos objetivos de retención. Por otro lado, al querer disminuir el presupuesto, entendemos que tenemos que ser muy cuidadosos con el que damos en los primeros días de vida porque eso puede implicar una baja importante de la retención al mes siguiente.

4.2.1 - Shap values analisis por usuario

Por último, vamos a experimentar con los shap values de las variables de algunos usuarios específicos que fueron falsos negativos y otros que fueron falsos positivos. Para poder armar esta experimentación primero necesitamos fijar el umbral de probabilidad. En este caso, decidimos fijarlo en 0,5, esta es la medida por default que viene incorporada al momento de etiquetar los datos utilizando el modelo entrenado. Por consiguiente, tomamos la decisión de seguir esa medida, debido a que deseamos realizar pruebas teniendo en cuenta el umbral más utilizado en la práctica en las primeras iteraciones de un modelo de Machine Learning. De esta forma, aquellos usuarios que tengan una

probabilidad mayor a 0,5 los asignamos como usuarios *churn* y a aquellos con probabilidades menores a 0,5 como que no son *churn*.

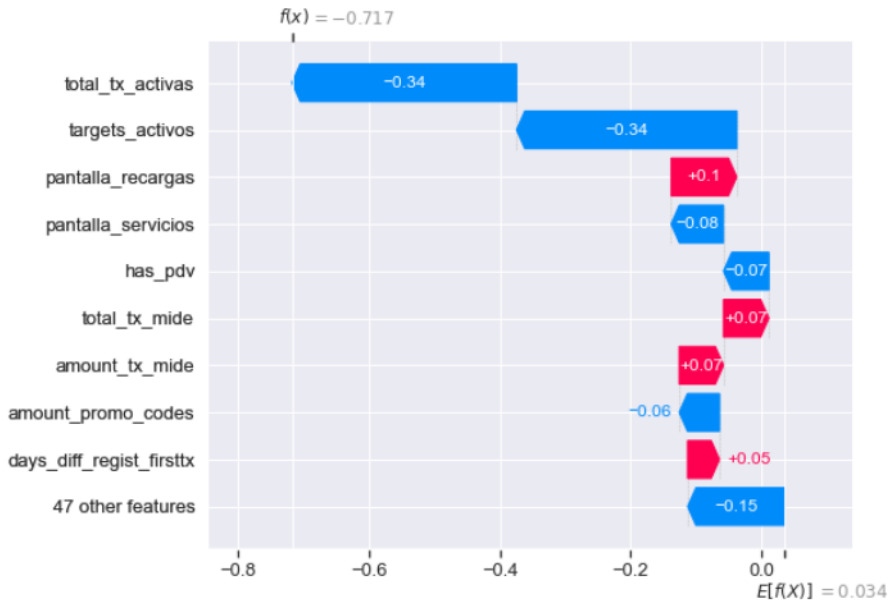
Esto nos va a ayudar a entender si puede haber algún patrón en los datos que confunda repetidas veces a las predicciones y tomarlo como input para el negocio.

Falsos positivos

En esta sección vamos a explorar más detalladamente algunos usuarios que en las pruebas de test nos dieron falsos negativos. Con esta denominación nos referimos a usuarios que el modelo predijo que se iban a quedar en la aplicación pero en realidad se convirtieron en usuarios churn. Estos gráficos nos muestran cuánto aumenta o disminuye cada una de las variables en la predicción de churn del usuario.

En primer lugar seleccionamos al usuario con el id igual a 51914b1d-7910-4379-969f-eba6b57099d6, que en el modelo arroja una probabilidad de volverse inactivo del 32%.

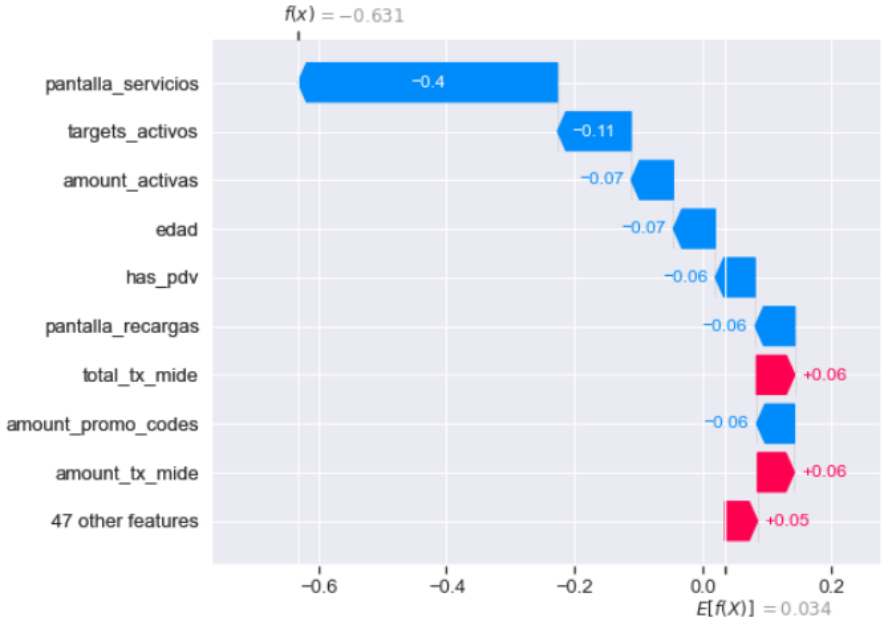
Figura 24. Shap values de variables para id: '51914b1d-7910-4379-969f-eba6b57099d6'



El valor de E[f(x)] = 0.034 es el valor promedio previsto de probabilidad de churn para el dataset completo. Este usuario tiene una baja probabilidad de ser churn por lo que nosotros consideramos que va a volver y luego no vuelve. Las dos variables que más lo empujan para abajo son la cantidad de transacciones y la cantidad de targets activos.

En segundo lugar seleccionamos al id '2a5485ad-04d1-4358-9eae-8b1029cf41b4', este usuario tiene una probabilidad de no volver del 34%. La principal variable que lo empuja para abajo en la probabilidad es la pantalla de servicios.

Figura 25. Shap values de variables para id = '2a5485ad-04d1-4358-9eae-8b1029cf41b4'



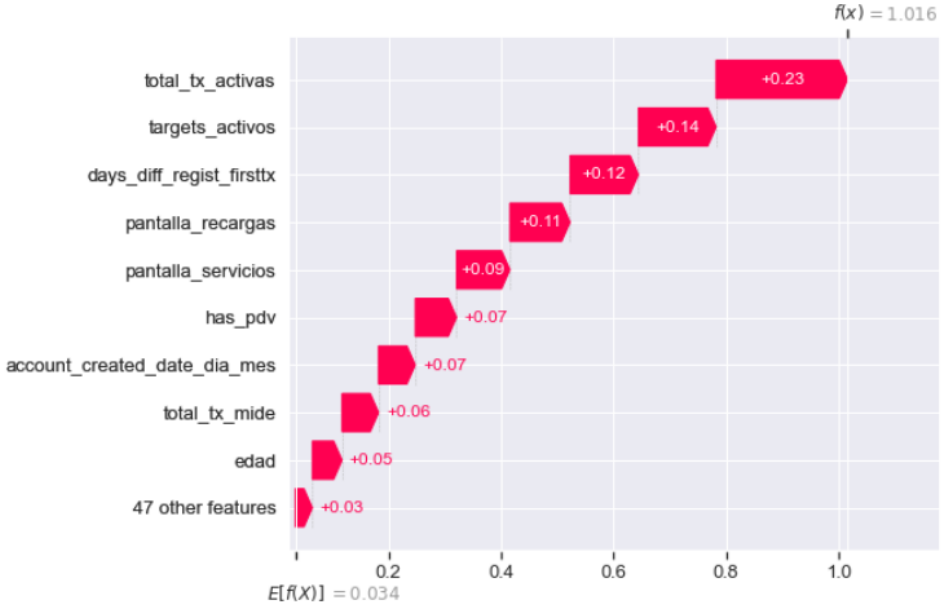
Seguimos evaluando distintos ids y vimos que las variables que más impacto tienen en la predicción son las de transacciones que al mismo tiempo son las que más impacto tienen en la predicción. No vimos un patrón concreto que pueda ser el causante de los falsos negativos, por lo que decidimos seguir incluyendo en el modelo todas las variables.

Falsos negativos

En este caso, vamos a evaluar a usuarios que predecimos que iban a convertirse en usuarios *churn* pero terminaron volviendo a transaccionar a la aplicación. En este caso son todos los que la probabilidad nos dio mayor a 0,5.

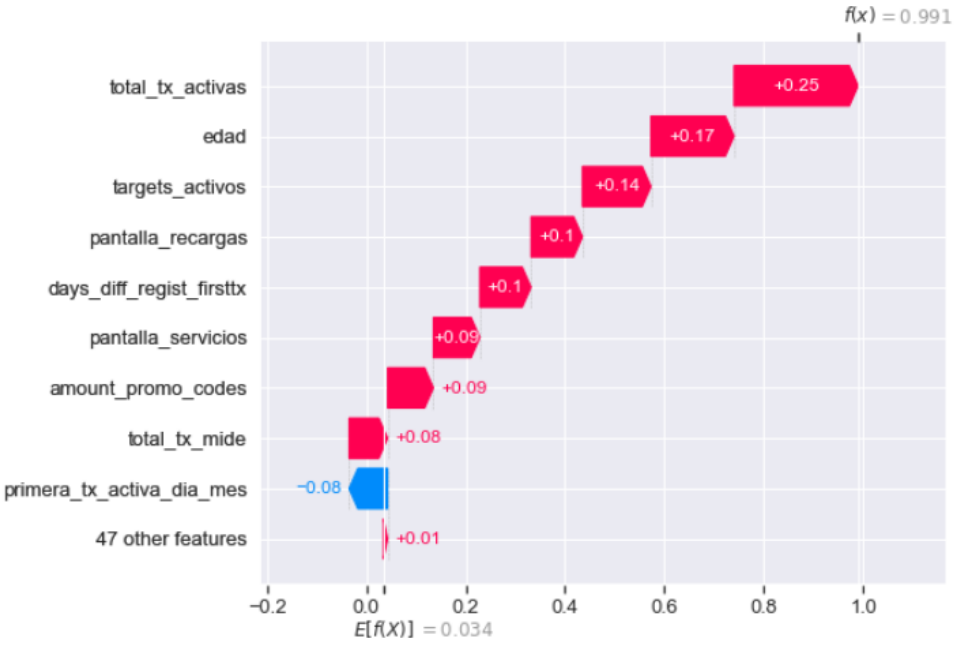
Empezamos con el caso del usuario con el siguiente id 'e0491965-d3c8-42e3-a48c-2f3396d155be'. Las dos variables que más influyen para predecir que el usuario no va a volver son la cantidad de transacciones activas y la cantidad de *targets* activos que tiene el usuario.

Figura 26. Shap values de variables para id = 'e0491965-d3c8-42e3-a48c-2f3396d155be'



En segundo lugar, seleccionamos el caso de un usuario que la probabilidad de que sea usuario churn nos da un 74%, pero luego vemos que vuelve a transaccionar el siguiente mes. A este usuario lo que más le empuja la probabilidad para arriba es el total de transacciones activas.

Figura 27. Shap values de variables para id = d1e7c4a1-dd9c-4216-b394-ed4e69dc3af5'



Sabemos que los falsos negativos son los más costosos para el negocio. Esto se da dado que estamos prediciendo que este usuario no va a ser *churn*, entonces decidimos no hacer nada al respecto y luego se termina volviendo un usuario inactivo. En este caso puede observarse que la variable que más influencia tiene en que la probabilidad termine en 70%

de transacciones activas. No obstante, no estamos dispuestos a eliminarla del modelo ya que es la variable más relevante cuando vemos la importancia de estas.

4.3 - Planificación del experimento

Para poder planificar el experimento de la implementación de los resultados que nos arroja el modelo en la práctica, primero es importante entender la estrategia actual que usa el equipo de Marketing de Tap con el objetivo de aumentar la retención. Esta información va a ser útil para luego poder entender desde que lugar se planteó la nueva estrategia a implementar.

4.3.1 Estrategia actual

En la actualidad, tenemos dos flujos prendidos con el objetivo de aumentar la retención. El primero busca prevenir que el usuario se vuelva inactivo y el segundo busca activarlo una vez que ya lo consideramos inactivo.

La campaña de prevención se dispara en el momento que el usuario cumple 15 días desde la última transacción y lo que hace es intentar que vuelva a transaccionar antes de que se vuelva inactivo. A esta estrategia la llamamos *prevent churning*. Consta en tres comunicaciones orgánicas: la primera informa sobre un descuento que Tap ofrece después del pago en la plataforma a partir del cual el usuario tiene la posibilidad de ganar hasta $200. Luego de dos días le recordamos que tiene que volver a pagar sus servicios y por último le contamos que también puede hacer recargas desde Tap. Creemos importante esta estrategia para intentar disminuir el gasto en usuarios que iban a volver orgánicamente.

La segunda campaña consiste en darle al usuario dos cupones en el lapso de una semana y se lanza cuando este cumple los 30 días de vida sin transaccionar. En primer lugar, le damos un cupón de 20% con tope de $200. Con esto, nos referimos a que en su próximo pago, les hacemos un descuento del 20% de lo que pague el usuario con un tope máximo de $200 de descuento. Luego de transcurrir 3 días sin transaccionar le ofrecemos un cupón de 40% con tope de $400. El objetivo de esta campaña es tratar de convertir al usuario con un monto más bajo antes de darle lo máximo que estamos dispuestos a ofrecer para que este se quede.

El objetivo de esta estrategia integral estaba en aumentar la métrica de retención de los usuarios. Para medir esta estrategia dejamos un grupo de control, explicado anteriormente en la Sección 3.2.7. Por lo que, para evaluar los resultados, vamos a comparar la conversión a transacción del grupo que recibió la comunicación contra los que no recibieron nada.

Tabla 5. Resultados: Estrategia actual retención

Resultados Marzo - 30 días				
Grupo	Usuarios impactados	Usuarios con transacciones	CVR	Incrementalidad
Target	7992	2321	29,04%	14,47%
Control	406	103	25,37%	

Estos resultados fueron obtenidos de la herramienta de Braze, donde se implementó este experimento. Esta herramienta nos permite definir un evento de conversión previo a iniciar el experimento y hace un monitoreo de los usuarios que realizan ese evento de conversión durante el periodo que uno le determine. Otra ventaja de está herramienta es que corre los chequeos estadísticos que se necesitan para validar los resultados (test de hipótesis), mencionados anteriormente en la Sección 3.2.7.

Como se puede ver en la Tabla 7, podemos ver los resultados completos de Marzo 2022 tomando como medición 30 días después de recibir esta campaña. Los resultados de esta estrategia fueron los siguientes: el ratio de conversión (CVR) del grupo de control era un 12% más bajo que el de la variante con cupón, por lo tanto con un 90% de confianza podemos afirmar que el grupo de control va a performar peor que la variante de cupón. La estrategia era incremental en un 14% con un 90% de confianza.

4.3.2 Estrategia del experimento propuesto

Habiendo entendido la estrategia actual de la cual hace uso la empresa para poder tener una mejora en las métricas de retención, vamos a proceder a explicar el paso a paso de cómo decidimos plantear esta estrategia integral, con el objetivo de poder probar el funcionamiento de un modelo de *churn prediction* en la práctica.

Para empezar, es importante entender cómo se generaron los diferentes segmentos para aplicar este experimento. Para la selección de usuarios, elegimos tomar a todos los que cumplieron entre 20 y 30 días de vida desde su primera transacción. Lo más exacto sería tomar solamente a los que cumplieron 20 días de vida cada día, pero el problema era que nos quedaba una base muy chica para hacer comparaciones y que los resultados lleguen a ser significativos. Por lo tanto, decidimos considerar toda la base entre 20 y 30 días de vida y mandar las comunicaciones que vamos a explicar a continuación.

Teniendo en cuenta esa estrategia, vamos a explicar paso a paso el experimento que planificamos para poder llevar nuestro modelo a la práctica y comprobar si funciona mejor que la estrategia actual. Para esta etapa, vamos a seguir los pasos explicados en la Sección 3.2.7 del libro de "*A/B Testing: The Most Powerful Way to Turn Clicks Into Customers*".

4.3.3 Definir el éxito

El éxito de este experimento va a estar en lograr generar un ahorro en la inversión de Marketing para retención y no perder usuarios. Definimos este objetivo, ya que decidimos enfocarnos en primer lugar en poder generar un ahorro para la empresa en usuarios que no valía la pena invertir y hoy se está invirtiendo.

Vamos a medir dos métricas fundamentales. En primer lugar, vamos a medir el ahorro en inversión por usuario. Esta se obtiene primero considerando la inversión total de cada campaña y dividiéndola por la cantidad de usuarios que terminaron transaccionando con esa campaña. Esta métrica la vamos a definir como la **métrica de éxito**, dado que es nuestro principal objetivo generar este ahorro.

La segunda métrica que vamos a considerar va a ser el ratio de conversión de los usuarios. Esta se obtiene dividiendo la cantidad de usuarios total que transaccionan sobre la cantidad de usuarios impactada con el experimento. A esta métrica la vamos a llamar **métrica de control**, dado que nuestro objetivo es que se mantenga estable ante los cambios generados con nuestro experimento.

El resultado que esperamos obtener con este experimento es que el ratio de conversión sea lo más similar posible entre las dos ramas pero que logremos un ahorro en la inversión significativo.

4.3.4 Construir una hipótesis

Para construir la hipótesis, en primer lugar es importante formular la pregunta que vamos a querer responder con este experimento. Para esto es importante tener en cuenta que nuestra métrica de éxito es el ahorro que este experimento puede generar en la inversión de Marketing.

La pregunta que formulamos es la siguiente: Pudiendo detectar patrones que identifican a estos usuario que van a volverse inactivos ¿Podemos generar un ahorro de inversión para este segmento de usuarios?

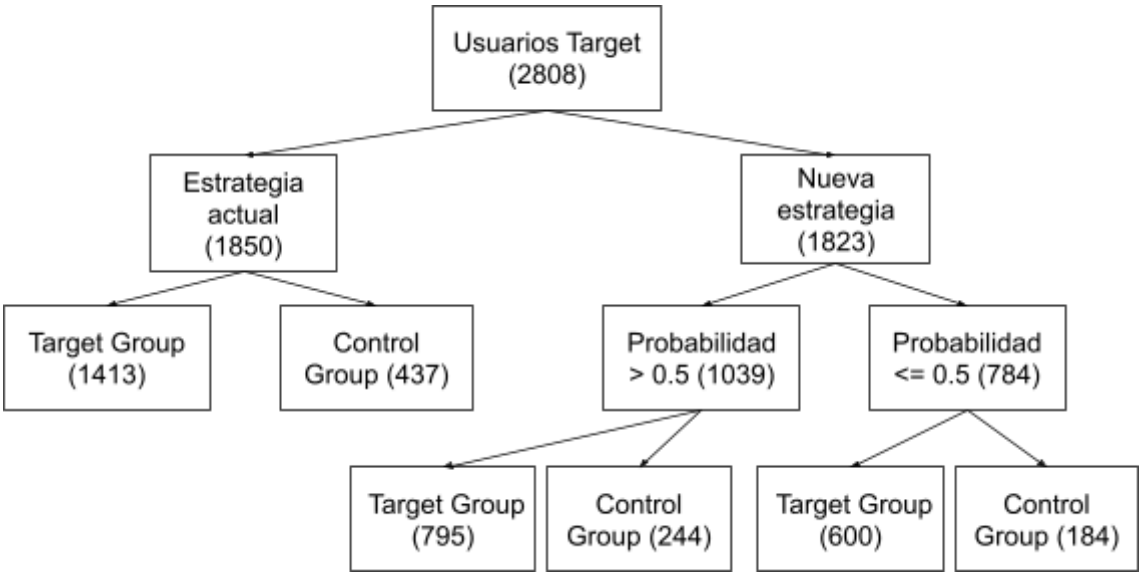
Dada esta pregunta la hipótesis que vamos a plantear para el experimento es la siguiente: *“Diseñar e implementar una estrategia de retención que tenga en cuenta la probabilidad que tiene un usuario de volverse churn con un modelo de Machine Learning va a generar un ahorro en la inversión de Marketing de la empresa”*

4.3.5 Hacer el experimento

Para la implementación del experimento, usamos una herramienta que se llama Braze. Esta herramienta permite, por un lado, cargar una base de datos con ids y un atributo determinado que les queramos asignar, en este caso, la probabilidad de volverse inactivo. Por otro lado, nos provee de una funcionalidad para armar diferentes variantes de comunicación y medirlas contra un grupo de control.

Para poder implementar este experimento, en primer lugar, segmentamos a todos los usuarios que cumplieron entre 20 y 30 días días desde su primera transacción. Luego, hicimos una división aleatoria de los 2808 usuarios en dos partes iguales. A la primera mitad les vamos a dar el mismo tratamiento que les dábamos anteriormente. La segunda mitad, la vamos a dividir en dos partes. El primer segmento está conformado por los usuarios que tienen una predicción de no volver mayor a 0.5 y el segundo los que tienen una probabilidad menor o igual a 0.5. Para el primer segmento vamos a usar el mismo tratamiento que dábamos a toda la base de usuarios, vamos a darle dos cupones consecutivos del mismo valor ascendente con tres días de diferencia entre uno y el otro. La segunda mitad, va a recibir comunicaciones orgánicas similar al que se le manda al usuarios entre los 15 y 25 días de vida.

Figura 27. Estructura del experimento



Este test va a durar 7 días y luego vamos a medir las métricas de ahorro y conversión para determinar el éxito de este experimento.

4.4 - Resultados del experimento

Los resultados decidimos evaluarlos en diferentes periodos de tiempo. Lo ideal siempre es evaluarlos la mayor cantidad de tiempo posible. Sin embargo,esto muchas veces se

dificulta con la implementación de diferentes campañas o con la salida de nuevas funcionalidades. Esto puede tener un efecto en la medición del experimento que no nos permita ver el verdadero impacto de este. Por esta razón, para evaluar la posibilidad de diferentes variaciones, vamos a medir el experimento a 10 y 30 días desde el primer día de implementación.

4.4.1 Resultados a 10 días post experimento

En primer lugar evaluamos el experimento 10 días después de la implementación de este. La idea es tener un primer panorama de los resultados. Recordando lo mencionado anteriormente, tenemos 2 variantes: la primera que recibe el mismo tratamiento con la estrategia actual de la empresa y la segunda que recibe la estrategia nueva. A la segunda la vamos a dividir en 2 variantes, aquellos que tienen una probabilidad mayor a 0.5 y los que tienen una probabilidad menor. Dado que a las 3 variantes les dejamos un grupo de control, vamos a poder evaluar la incrementalidad de estas estrategias en comparación a no recibir ninguna comunicación.

En la Tabla 8 podemos ver como la estrategia actual medida en los primeros 10 días después de impactar al usuario generó una incrementalidad de un 11% con un 78% de confianza. Esto quiere decir que mandar un cupón a todos los usuarios, sin importar la probabilidad que tenga de no volver que tengan, nos genera una variación incremental de los conversion rates en un 11%.

Tabla 6. Resultados: Usuarios que recibieron la estrategia actual 10 días

Resultados estrategia actual - 10 días					
Grupo	Usuarios impactados	Usuarios con transacciones	CVR	Incrementalidad	Confianza
Target	1413	418	29,58%	**11,44%**	**78,75%**
Control	437	116	26,54%		

En la Tabla 6, podemos ver como la estrategia nueva propuesta medida en los primeros 10 días después de impactar al usuario generó una incrementalidad de un 23% con un 95% de confianza. Esto quiere decir que mandar un cupón a los usuarios que sabemos que tienen una probabilidad mayor a 0.5 de volverse inactivos y mandarles solo comunicación a los que tienen una probabilidad menor a 0.5, nos trae un 23,4% más de usuarios. Esta estrategia duplica a la anterior, por lo que parece ser la mejor estrategia a implementar si lo que buscamos es incrementalidad.

Tabla 7. Resultados: Usuarios que recibieron la nueva estrategia

Resultados nueva estrategia - 10 días					
Grupo	Usuarios impactados	Usuarios con transacciones	CVR	Incrementalidad	Confianza
Target	1395	393	28,17%	**23,04%**	**95,19%**
Control	428	98	22,90%		

Para terminar de definir cuál es la mejor estrategia, es importante entender cuánto dinero se gastó en cada una de ellas y comprender cuál es la diferencia en términos de conversión generales. Dado que si traemos menos volumen de usuarios con la estrategia nueva, y es más incremental, igualmente no serían buenos resultados.

Por lo mencionado anteriormente, es importante tener en cuenta los resultados del experimento que se muestran en la Tabla 8. Podemos ver que la estrategia actual tiene un ratio de conversión 9% menor que la variante de la estrategia nueva. No obstante no hay evidencia suficiente para afirmar que el ratio de conversión haya cambiado o sea diferente con respecto a la estrategia actual.

Por otro lado, lo que sí podemos afirmar, es que estamos ahorrando un 59% del presupuesto de marketing en comparación con la estrategia anterior y que eso no nos perjudica el volumen de usuarios que retenemos.

Tabla 8. Resultados 10 días desde que se lanzó el experimento

Resultados test - 10 días						
Estrategia	Usuarios impactados	Usuarios con transacciones	CVR	Gasto total	Gasto por usuario	Ahorro por usuario
Nueva	1395	393	28,17%	44480	113	**-59,58%**
Actual	1413	418	29,58%	117040	280	

Con esta evaluación podemos empezar a ver positividad en nuestro modelo en las tres métricas que más nos importan: incrementalidad, conversión y ahorro. Sin embargo, vamos a tener que seguir evaluando los resultados a más días porque podemos estar viendo un adelanto de las transacciones en alguna de las variantes. Este adelanto se puede dar debido a que, si comparamos hacer una comunicación con respecto a no hacerla, los que no reciben esta comunicación puede ser que tarden más tiempo en transaccionar.

4.3.2 Resultados a 30 días post experimento

Para definir los resultados del experimento, decidimos evaluar la performance del experimento 30 días después de la implementación del mismo. Este análisis, no solo nos va a decir cual es la mejor estrategia a implementar, sino que también nos va a permitir medir la retención real que tuvimos en cada segmento, dado que los datos los medimos entre sus 30 y 60 días de vida en la aplicación.

En primer lugar, medimos la incrementalidad de la campaña que armamos con la estrategia actual que tiene la empresa. En la Tabla 9, los números de esta estrategia muestran que no parece haber una incrementalidad entre mandar estos cupones y el grupo de control, pero al tener un bajo nivel de confianza, no lo podemos confirmar.

Dado que esta incrementalidad no la podemos tener en cuenta por su nivel de confianza, decidimos tomar como válida tomar la evaluación del modelo anterior que se hizo, donde medimos todos los usuarios que entraron a la estrategia actual en marzo durante 30 días después de recibir el impacto y este resultó ser un 14% incremental. Estos resultados se pueden encontrar en la Sección 4.4.1.

Tabla 9. Resultados: Usuarios que recibieron la estrategia actual medida en 30 días

Resultados estrategia actual - 30 días					
Grupo	Usuarios impactados	Usuarios con transacciones	CVR	Incrementalidad	Confianza
Target	1413	628	44,44%	**-2,40%**	**31%**
Control	437	199	45,54%		

En segundo lugar, vamos a medir la incrementalidad de las campañas donde aplicamos la nueva estrategia. En esta, podemos afirmar con un 90% de confianza que hay una incrementalidad de un 12% en comparación a no mandar nada. Si comparamos esta estrategia contra los resultados de Marzo que nos dieron significativos, nos da una baja de dos puntos porcentuales en incrementalidad.

Tabla 10. Resultados: Usuarios que recibieron la nueva estrategia medida en 30 días

Resultados nueva estrategia - 30 días					
Grupo	Usuarios impactados	Usuarios con transacciones	CVR	Incrementalidad	Confianza
Target	1395	624	44,73%	**11,96%**	**90%**
Control	428	171	39,95%		

Por último, como vemos en la Tabla 11, podemos afirmar que ahorrando un 62% en el presupuesto de Marketing podemos generar una estrategia en donde se mantenga la métrica de conversión. Si vemos la significancia estadistica de el CVR, no podemos decir, con 95% de confianza que la variante nueva va a perforar peor que la actual. Por lo tanto no es significativa la diferencia que encontramos entre estas dos estrategias.

Tabla 11. Resultados finales 30 días después del lanzamiento del experimento

Resultados test - 30 días						
Estrategia	Usuarios impactados	Usuarios con transacciones	CVR	Gasto total	Gasto por usuario	Ahorro por usuario
Nueva	1395	624	44,73%	44480	71	**-61,75%**
Actual	1413	628	44,44%	117040	186	

4.3.2 Discussion

Este experimento se creó con el fin de poder usar las probabilidades que obtenemos de correr un modelo de Machine Learning con el *dataset* seleccionado para diseñar una estrategia más eficiente de retención.

Con lo planteado anteriormente, podemos concluir que los resultados del experimento son positivos. Podemos afirmar que esta estrategia, nos permite generar un ahorro de un 61% en el presupuesto sin ver un impacto negativo significativo en la retención del usuario. Por lo tanto, cumple con las expectativas esperadas y genera un ahorro que luego puede ser utilizado con el fin de mejorar la métrica de retención o aplicarlo a otras estrategias de adquisición de usuarios dentro de Marketing

5. Conclusiones

5.1 - Logros alcanzados con el proyecto

En este proyecto, lo que buscamos es implementar un modelo de Machine Learning para poder usar las predicciones de este como input para mejorar la estrategia actual de retención de una empresa llamada Tap que partenece al sector *Fintech*. Con mejorar nos referimos a eficientizar el gasto de retención que hoy en día maneja el area de Marketing. Para este objetivo, utilizamos los datos de la compañía llamada Tap.

Los datos fueron extraídos del Datawarehouse de Tap con un proceso de ETL (Extract Transform Load) donde logramos armar los datasets necesarios para poder tomarlo como input para los modelos de Machine Learning entrenados.

En el proceso de exploración pudimos encontrar distintos patrones en la información que nos permitieron entender las variables con las que estábamos trabajando en el modelo. A su vez, la detección de estos patrones a través del análisis exploratorio nos permitió armar una serie de accionables interesantes que podrían ser aplicados en el corto plazo para mejorar la retención.

El siguiente paso fue el desarrollo de modelos de Machine Learning. En esta etapa, decidimos armar tres modelos base y definir con cual íbamos a seguir experimentando. Los modelos que usamos fueron los siguientes: Regresión Logistica, Random Forest y LightGBM. Dado que el que mejor perforó fue el modelo de LightGBM decidimos usar random search con cross validation para la prueba de hiperparametros, con el propósito de mejorar la performance del modelo seleccionado. El modelo inicial que construimos arrojó un ROC-AUC del 0.71 y logramos una mejora con el modelo final de un 7% con un ROC-AUC final de un 0.76

Por último, diseñamos un experimento para entender cómo funciona este modelo en la práctica. Usamos las probabilidades arrojadas por el modelo para generar una estrategia que permita no gastar dinero en usuarios que sabemos que por su comportamiento de los primeros 20 días después de su primera transacción, van a volver a transaccionar en la aplicación.

Esta estrategia nos permitió ahorrar un 60% del presupuesto de Marketing, sin tener un impacto negativo en la retención de los usuarios. Por esta razón, vemos sumamente importante poder usar las probabilidades que nos arroja el modelo para poder ahorrar y en un futuro generar una estrategia que también nos permita mejorar las métricas de retención.

5.2 - Limitaciones y futuras posibles mejoras

Sabemos que los resultados de performance a los que se llegaron no son los mejores y que todavía se pueden perfeccionar. Estas mejoras las podemos hacer aplicando cambios en la base de datos que le damos como input al modelo o en el modelo en particular que seleccionamos.

Como mejora en la base de datos, podemos crear nuevas variables con la información obtenida que nos permitan mejorar la calidad de información que le brindamos al modelo. Un ejemplo podría ser crear una variable que sea el monto promedio del usuario o agregar otros formatos de las fechas como día del año. Otra mejora a la base de datos puede ser seguir indagando en campos que podrían ayudar a la predicción y sumarlos a las bases. A modo de ejemplo, podría agregarse el género y características del dispositivo móvil del usuario. No obstante, el enfoque de la tesis implicaba generar un modelo sencillo que contenga variables lo más generales posibles pero que tenga un impacto en la práctica y con el fin de poder ser extrapolables a otras compañías.

Con respecto a las mejoras del modelo, es importante aclarar el costo computacional que lleva este modelo. Se realizó en una computadora de un estudiante, con una memoria RAM de 8 GB y el modelo LightGBM con la optimización de los hiperparametros tardó un aproximado de 5 horas y 27 minutos. Por esta razón, decidimos no seguir haciendo más pruebas con otros hiperparámetros para este modelo. Esto se puede mejorar dado que existen plataformas como Google Colab que permiten montar un modelo en una notebook de Google y correrlo con GPU. Para poder industrializarlo, tendríamos que montarlo en la plataforma de MLOps de la organización que va a permitir lograr los objetivos de cómputo, monitorización, automatización y reproductividad de la ingeniería de *features*.

Por otro lado, también creemos que se pueden aplicar mejoras a la implementación de estos resultados. Como el principal foco de la empresa hoy es el ahorro, en el experimento nos enfocamos en disminuir esta métrica para los usuarios que tenían una probabilidad baja de convertirse en usuario *churn*. No obstante, sabemos que podemos experimentar con distintos umbrales o generar más grupos de división de los resultados con el fin de perfeccionar la estrategia de retención y poder no solo generar un ahorro sino que también lograr aumentos del ratio de retención. Un ejemplo de esto sería crear una campaña distinta con mayor inversión para los usuarios que tienen una probabilidad mayor a 0.75.

5.3 - Implementaciones para el negocio

Como mencionamos anteriormente, con el input del modelo de Machine Learning que generamos durante esta tesis, la empresa puede ahorrar un 60% del presupuesto de retención. Por esta razón, es importante que lo podamos automatizar para que este modelo se ejecute diariamente con los usuarios que cumplen 20 días desde su fecha de activación y podamos generar los segmentos necesarios en nuestra herramienta para lanzar estas comunicaciones correspondientes.

Además, creemos que es importante seguir indagando en la segmentación y ejecución de distintos experimentos para lograr no solo generar un ahorro sino también poder mejorar la métrica de retención con estas predicciones.

Por último, es fundamental destacar que aplicando modelos de Machine Learning logramos optimizar el presupuesto de Marketing de Tap. Armamos una metodología de trabajo que implementa un prototipo que el humano no hubiera podido replicar y lo lleva a la práctica de manera eficiente cumpliendo así con las expectativas esperadas para este proyecto.

Referencias

Amy Gallo (2017). A Refresher on A/B Testing. https://hbr.org/2017/06/a-refresher-on-ab-testing.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of machine learning research, 13(2).

Berry, M. J., & Linoff, G. S. (2004). Data mining techniques: for marketing, sales, and customer relationship management. John Wiley & Sons.

Danny Varghese. (2018). Comparative Study on Classic Machine learning Algorithms https://towardsdatascience.com/comparative-study-on-classic-machine-learning-algorithms-24f9ff6ab222.

Davis J, Goadrich M. 2006. The Relationship Between Precision-Recall and ROC Curves. Proceedings of the 23rd International Conference on Machine Learning. Association for Computing Machinery.

Financial Stability Board (2019) https://www.fsb.org/wp-content/uploads/P040219.pdf

Futagami, K., Fukazawa, Y., Kapoor, N., & Kito, T. (2021). Pairwise acquisition prediction with SHAP value interpretation. The Journal of Finance and Data Science, 7, 22-44.

Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani. (2013). An introduction to statistical learning : with applications in R. New York :Springer,

Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction

Huang, B., Kechadi, M. T., & Buckley, B. (2012). Customer churn prediction in telecommunications. Expert Systems with Applications, 39(1), 1414-1425.

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. Intelligent data analysis, 6(5), 429-449.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. Advances in neural information processing systems, 30

Kotarba, M. (2016). New factors inducing changes in the retail banking customer relationship management (CRM) and their exploration by the FinTech industry. Foundations of management, 8(1), 69.

Lott, A., & Reiter, J. P. (2020). Wilson confidence intervals for binomial proportions with multiple imputation for missing data. The American Statistician, 74(2), 109-115.

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. Advances in neural information processing systems, 30.

Malhotra, P., Patel, P., Shah, N., Veera, R., & Sanghavi, R. (2019). Customer Churn prediction for FinTech Companies using Artificial Neural Networks. i-Manager's Journal on Computer Science, 7(4), 46.

Melisa Reinhold (2022) Camara central de Fintechs Argentina. https://camarafintech.org/revolucion-fintech-el-fenomeno-que-cambio-la-forma-de-usar-el-dinero-y-las-finanzas-personales/ 2

Mitra, S., Pal, S. K., & Mitra, P. (2002). Data mining in soft computing framework: a survey. IEEE transactions on neural networks, 13(1), 3-14.

Quinlan, J. R. (1993). C4.5: Programs for machine learning. Morgan Kaufman Publishers.

Quinlan, J. R. (1996). Improved use of continuous attributes in c4.5. Journal of Artificial Intelligence Research, 4, 77–90., V., & Capota, M. (2007). Churn prediction. Bus. Anal. Course. TUM Comput. Sci, 33, 34.

Sahai, S., Goel, R., Malik, P., Krishnan, C., Singh, G., & Bajpai, C. (2018). Role of social media optimization in digital marketing with special reference to trupay. International Journal of Engineering & Technology, 7(2.11), 52-57.

Weng, C. G., & Poon, J. (2008, November). A new evaluation measure for imbalanced datasets. In *Proceedings of the 7th Australasian Data Mining Conference-Volume 87* (pp. 27-32).

Apéndice. Datos

Las tablas que se terminaron usando para el armado de las querys y los campos que se utilizarán como input del modelo son los siguientes:

- Appsflyer: esta es la plataforma que usamos para atribución de usuarios, también contiene todos los eventos que se disparan desde el front.
    - external_ref (identificador) → id de identificación
    - Media Source (categórica)→ medio por el cual adquirimos al usuario
    - Install time (fecha)→ fecha y hora de instalación
    - Pantalla_recargas (numérica) → cuentas veces entró a la pantalla de recargas
    - Pantalla_servicios (numérica)→ cuantas veces entró a la pantalla de servicios
    - qty_errors (numérica)→ cantidad de errores que se le dispararon en la aplicación
    - prueba_vida (numérica)→ binaria con 1 si hicieron la prueba de vida y 0 en caso contrario

Tabla 12. Datos de la tabla de appsflyer

external_ref	media_source	install_time	qty_errores	pantalla_servicios	pantalla_recargas	prueba_vida
1baceadd-8a25-4440-81b7-d15b7ecbf323	qr_code	2021-11-15 19:16:27.294	0	6	2	0
2f272af2-030b-4dd0-9ba7-bd4ca26d304d	organic	2022-01-13 02:41:25.147	3	10	0	0
a8a62211-dc13-44b9-9683-6f1a62fa0e66	restricted	2021-12-12 20:58:43.659	0	0	8	1
61d42fd4-49d7-4391-a11c-58f429eceeef	email	2021-12-05 21:58:38.921	1	5	0	0
50e857eb-2604-4c05-8018-e9b72c019f67	restricted	2021-11-29 19:55:40.288	57	13	7	0
a124347e-69fd-4ce1-8573-2835e8ea3419	af_app_invites	2021-12-02 23:02:48.229	2	5	0	1
a392c876-4ec9-4367-8cc1-6bdce9116a86	organic	2022-01-10 12:10:29.449	1	4	4	1
647a5caa-c7cb-4949-a22a-0dd87004019c	organic	2022-02-04 09:18:18.801	6	5	1	1
16e44735-a8bc-40ad-8b3b-04640aa2bf2e	organic	2021-10-28 16:51:49.222	1	5	0	0
e49e42b2-2f80-4dd5-9f13-02bebd2237eb	organic	2021-12-07 12:48:58.000	4	1	0	0
d13b8808-9f2e-460d-a067-b142c3a6645c	organic	2022-01-27 11:48:54.943	3	1	11	1
1e02e2c7-3c9a-42f9-b1ad-860dccf0eeb7	af_banner	2021-11-28 20:26:58.706	0	5	10	0
7a17d3cc-f31a-4b5c-82a0-94a3e26fa202	email	2021-12-08 15:37:50.723	6	6	0	0
92546769-adec-4c2e-8c46-fda1cefc6f02	googleadwords_int	2021-11-30 08:51:10.765	2	3	0	1
877126e1-4748-4256-9253-aa9bf04b2d6d	af_app_invites	2021-11-18 16:02:15.390	1	4	0	0
40798ae5-4415-4e9b-bb4a-72186a33bcf1	organic	2021-08-22 11:16:58.850	0	11	2	0
697185a4-ce80-422b-bb18-16a0e9c2334a	organic	2021-12-19 10:34:19.654	3	1	1	0
7e800eda-76ab-4ba5-8a0c-17f4a280efdd	organic	2021-09-14 17:19:21.991	0	0	1	0
919c2b38-4b3e-46dd-8481-3143fe1cb72c	organic	2022-01-06 09:05:47.351	3	7	0	1
de80cfc2-e0e3-4b03-842c-aaef918f170e	af_banner_home	2021-11-10 23:05:02.291	1	11	0	1

- Transactions: tabla que contiene todos los movimientos del usuario en la aplicación
    - external_ref → id de identificación
    - total_tx_activas (numérica) → suma de pagos de servicios, recargas, pagos con tarjeta y pagos con QR
    - amount_activas (numérica) → suma de dinero transaccionada en pagos de servicios, recargas de celular, recargas mide, pagos con tarjeta y pagos con QR
    - total_tx_servicios (numérica) → suma de pagos de servicios
    - amount_servicios (numérica) → suma de plata transaccionada en pagos de servicios
    - total_tx_recargas (numérica) → suma de recargas de celular
    - amount_recargas (numérica) → suma de plata transaccionada en recargas
    - total_tx_qr (numérica) → suma de pagos con QR
    - amount_tx_qr (numérica) →suma de plata transaccionada en pagos con QR
    - total_tx_tarjeta (numérica)→ suma de pagos con tarjeta
    - amount_tx_tarjeta (numérica) → suma de plata transaccionada en tarjeta
    - total_tx_mide (numérica)→ suma de recargas mide
    - amount_tx_mide (numérica)→ suma de plata transaccionada en mide
    - total_cashin_cc (numérica)→ suma de cashins con tarjeta de crédito
    - amount_cashin_cc (numérica)→ suma de plata transaccionada en cashin con tarjeta de crédito
    - total_transfer_to_cbu (numérica)→ suma de transferencias a cbu externo con dinero en cuenta

- amount_transfer_to_cbu (numérica)→ suma de plata transaccionada en transferencias a cbu con dinero en cuenta
- total_card_transfer_to_cbu (numérica)→ suma de transferencias a cbu con tarjeta
- amount_card_transfer_to_cbu (numérica)→ suma de plata transaccionada en transferencias a cbu con tarjeta
- total_t2t (numérica)→ suma de transferencias dentro de Tap
- amount_t2t (numérica)→ suma de plata transaccionada en transferencias dentro de Tap
- total_cashin_transfer (numérica)→ suma de cashins por transferencia
- amount_cashin_transfer (numérica)→ suma de plata transaccionada por cashin por trnsferencia
- total_cashin_dc (numérica)→ suma de cashins por tarjeta de débito
- amount_cashin_dc (numérica)→ suma de plata transaccionada en cashin con tarjeta de débito
- total_qr_serv (numérica)→ suma de pagos de servicios con QR
- amount_qr_serv (numérica)→ suma de plata transaccionada en pago de servicios con QR
- targets_activos (numérica)→ cantidad de servicios distintos que paga un usuario
- cant_promo_codes (numérica)→ cantidad de cupones promocionales que reciben
- amount_promo_codes (numérica)→ cantidad de plata en cupones promocionales que reciben

Tabla 13. Datos de la tabla de transacciones

origin_external_ref	total_tx_activas	amount_activas	total_tx_servicios	amount_servicios	total_tx_recargas	amount_recargas	total_tx_qr	amount_tx_qr	total_tx_tarjeta	amount_tx_tarjeta	total_tx_mide	amount_tx_mide	total_cashin_cc	amount_cashin_cc	total_transfer_to_cbu
00687730-7293-46fa-b	1	1747	0		0		0		1	1747	0		0		1
be058135-6d19-4a83-9	1	1746	1	1746	0		0		0		0		0		1
3b6ea0f7-038f-412a-8	1	400	0		0		0		0		1	400	0		0
bd5d2d8e-5b4e-4c16-	1	895	1	895	0		0		0		0		0		0
4329150e-d8cf-4793-8	1	2402	1	2402	0		0		0		0		0		0
e95f90db-548a-40f7-9	1	26	0		0		0		1	26	0		0		0
a8966ba7-289c-4508-b	1	300	0		0		0		0		1	300	0		0
291d80c3-57c6-4258-8	1	1400	1	1400	0		0		0		0		0		0
5dd54ccc-3fae-4b7b-a	2	800	0		0		0		0		2	800	2	800	0
bdfb6456-468a-4486-8	3	500	0		0		0		0		3	500	0		0
10875b90-81aa-495d-8	1	800	0		0		0		1	800	0		0		3
37e32266-1e6e-4ff4-b	2	1119	1	979	0		1	140	0		0		0		1
b919e75f-5974-456e-b	2	1053	0		0		0		2	1053	0		0		1
aca772fc-be64-48d5-a	2	9061	2	9061	0		0		0		0		0		0
7ea53aa1-1a5f-431e-a	1	350	0		1	350	0		0		0		0		26
cc457dc1-f617-46a7-b	2	1521	2	1521	0		0		0		0		0		0
2535d872-d457-4d18-8	2	500	0		0		0		0		2	500	0		0
fb2ed25c-6e75-4791-b	2	124	0		0		0		2	124	0		0		11
8e541cf0-8dfd-49af-82	1	874	1	874	0		0		0		0		0		0
ad69076a-272b-4b9e-a	2	2290	2	2290	0		0		0		0		0		0

origin_external_ref	amount_card_transfer_to_cbu	total_t2t	amount_t2t	total_cashin_transfer	amount_cashin_transfer	total_cashin_dc	amount_cashin_dc	total_qr_serv	amount_qr_serv	targets_activos	cant_promo_codes	amount_promo_codes	cant_tarjetas
00687730-7293-46fa-b		0		1	2000	0		0		1	0		0
be058135-6d19-4a83-9		0		1	1746	0		0		1	1	1000	0
3b6ea0f7-038f-412a-8		0		1	500	0		0		1	0		1
bd5d2d8e-5b4e-4c16-9		0		3	1000	0		0		1	1	358	1
4329150e-d8cf-4793-8		0		0		0		0		1	0		2
e95f90db-548a-40f7-9		0		8	450	0		0		1	0		0
a8966ba7-289c-4508-b		0		0		1	300	0		1	0		1
291d80c3-57c6-4258-8		0		0		0		0		1	1	400	1
5dd54ccc-3fae-4b7b-a		0		0		0		0		1	0		1
bdfb6456-468a-4486-8		0		2	500	0		0		1	0		0
10875b90-81aa-495d-8		0		2	33882	1	632	0		1	0		2
37e32266-1e6e-4ff4-b		0		1	1000	0		1	140	2	1	40	1
b919e75f-5974-456e-b		0		2	1054	0		0		1	0		0
aca772fc-be64-48d5-a		0		2	9062	0		0		2	1	70	0
7ea53aa1-1a5f-431e-a		0		12	7201	0		0		1	0		0
cc457dc1-f617-46a7-b		0		0		0		0		2	2	936	1
2535d872-d457-4d18-8		0		2	800	0		0		1	0		0
fb2ed25c-6e75-4791-b		0		9	16682	0		0		1	0		0
8e541cf0-8dfd-49af-82	505	0		0		0		0		1	1	350	1
ad69076a-272b-4b9e-a		0		2	2300	0		0		1	0		0

- Card public:
    - external_ref → id de identificación
    - cant_tarjetas (numérica)→ cantidad de tarjetas asociadas a su cuenta
- Dim account:
    - external_ref → id de identificación
    - creacion_cuenta (fecha)→ fecha de registro