Universidad Nacional Autónoma de México

Facultad de Estudios Profesionales Campus Acatlán

Análisis del comportamiento de clientes en una empresa mediante modelos de aprendizaje automático para la predicción del Churn Rate y segmentación de clientes con Python

T E S I S

QUE PARA OPTAR POR EL GRADO DE:

Licenciado en Matemáticas Aplicadas y Computación

PRESENTA:

Carlos Espadin Medina

TUTOR:

Mtra. Jeanett López García

Ciudad de México, 2024

A la Facultad de Ingeniería y a la Universidad, por la formación que me han dado.
Es gracias a ustedes que es posible el presente trabajo.
En verdad, gracias.
Yo.

Reconocimientos

También quisiera reconocer a ... por ...CONACYT, PAPIIT / etc. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Declaración de autenticidad

Por la presente declaro que, salvo cuando se haga referencia específica al trabajo de otras personas, el contenido de esta tesis es original y no se ha presentado total o parcialmente para su consideración para cualquier otro título o grado en esta o cualquier otra Universidad. Esta tesis es resultado de mi propio trabajo y no incluye nada que sea el resultado de algún trabajo realizado en colaboración, salvo que se indique específicamente en el texto.

Carlos Espadin Medina. Ciudad de México, 2024

Resumen

This is where you write your abstract ... Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Presentación

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1.2. Objetivo

Este trabajo tiene por objetivo ... Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1.3. Motivación

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1.4. Planteamiento del problema

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1.5. Metodología

Se tiene un objetivo principal, y para llegar a él Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1.6. Contribuciones

La principal contribución de este trabajo es Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1.7. Estructura de la tesis

Este trabajo está dividido en XX capítulos. Al principio se encuentra

Finalmente se encuentra la parte de

Capítulo 2

Inteligencia Artificial y Aprendizaje Automático

En este capitulo se presentará la teoría acerca de las herramientas de Machine Learning (ML) y metodologías utilizadas para el caso de estudio abordado en este trabajo, así como los dos enfoques del ML, estos son el aprendizaje supervisado, siendo del tipo clasificación binaria, y aprendizaje no supervisado para el caso de la segmentación de clientes (clustering).

Además abordaremos temas previos y de suma importancia en toda implementación de ML, como la obtención de la información, preparación de los datos, visualización, etc.

2.1. Preparación de los datos y recolección de los datos.

Una parte crucial en toda implementación de Machine Learning es la calidad de los datos, y es algo que cuando se hacen implementaciones sobre información real cobra mucha relevancia, ya que en términos generales es deber del científico de datos a cargo, asegurarse que dicha información sea de calidad.

Ya sea asegurarse del cumplimiento de los supuestos que cada modelo de machine learning demanda ó bien, asegurarse que la calidad de la información sea la adecuada para poder obtener las mejores estimaciones y evitar problemas de Under-fitting o Over-fitting.

En la mayoría de las implementaciones de Machine Learning necesitamos recolectar la información desde alguna fuente de origen, ya sea una base de datos, data lake o cual sea la fuente de la que provengan nuestros datos, por lo cual sera de vital importancia tener total conocimiento acerca del sistema de origen de la información. En consecuencia parte del pipeline que se va a preparar comprende la extracción de la información y en algunos casos preparar un primer procesamiento que nos asegure trabajar con el nivel de especificación que deseamos.

Para el caso particular presentado en este trabajo, es necesario asegurarnos que el nivel máximo de detalle en nuestros datos sean clientes, osea que cada campo y registro debe hacer referencia a información de los clientes, como su historial de compras, año de registro, etc. Y por lo tanto un primer paso sería validar que estemos trabajando con registros únicos.

En consecuencia es importante conocer todos los detalles posibles de los datos con los que vamos a trabajar, tanto en lo concerniente al negocio o a las propiedades inherentes de los datos, esto sera mejor abordado en el siguiente apartado.

2.1.1. Datos Faltantes y Outlier

Esta sección aun no se desarrolla por completo.

2.1.2. Análisis Exploratorio

De acuerdo con (Bruce et al., 2020) el análisis exploratorio de datos fue propuesto inicialmente por John W. Tukey en 1977, en un inicio se concebida como una breve inspección de las característica de los conjuntos de datos. Este análisis consiste en desentrañar todas las características útiles de los datos que estamos usando, ósea, una radiografía estadística de los datos.

Y a pesar de la antigüedad esta técnica se ha sabido mantener y adaptarse a las necesidades de nuestros tiempos, en la época de los macrodatos el análisis exploratorio es fundamental en cada implementación de aprendizaje automático que realicemos.

2.1.2.1. Tipos de datos

Conocer el tipo de datos con los que vamos a trabajar es quizás el primer y más importante de los pasos a ejecutar cuando se empieza un análisis. Es importante remarcar que el marco general sobre el que se suele trabajar en ciencia de datos es una estructura de datos llamada *rectangular data*, básicamente es una matriz de $N \times M$, donde N representa el número de columnas y M el número de filas.

En particular Python nos ofrece varias versiones para poder trabajar con este tipo estructuras de datos, aunque la librería Polar nos ofrece una muy buena alternativa para trabajar con *rectangular data* en este trabajo en particular decidí enfocarme solamente en Pandas y los *dataframes*.

Aclarado este ultimo punto, me centrare en hablar de los tipos de datos y la importancia que tiene en el análisis exploratorio de los datos, en general los tipos de datos están clasificados en dos tipos importantes categóricos y numéricos.

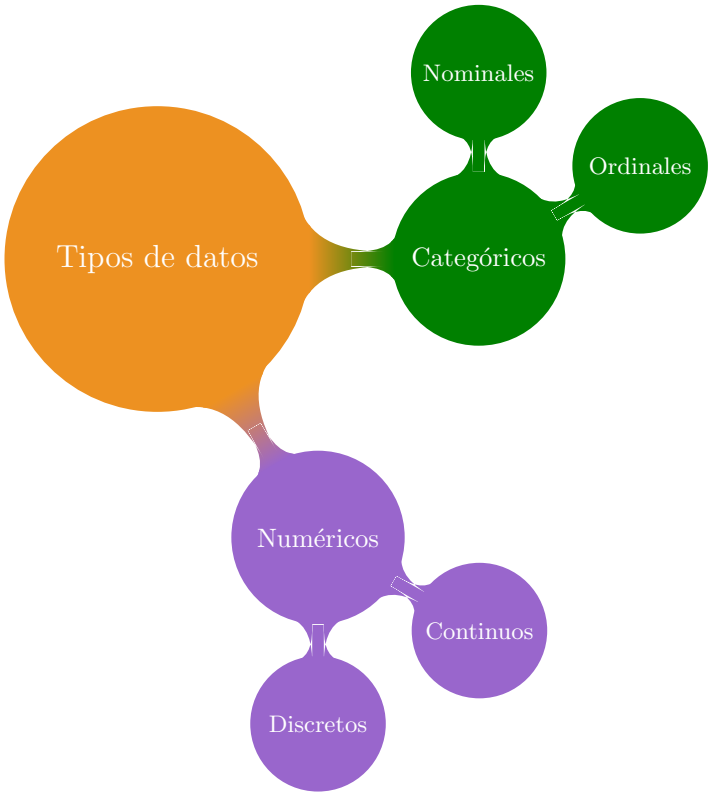
Bruce et al. (2020, p. 3) define a los datos categóricos como "datos que solo pueden adoptar un conjunto de especifico de valores que representan un conjunto de categorías posibles", en términos más coloquiales son datos cualitativos que pueden agruparse en categorías, de este tipo se desprende dos subtipos de datos categóricos; nominales, también llamados binarios, esto debido a que cuentan con unicamente dos categorías,

por ejemplo verdadero o falso, hombre o mujer, etc, y los datos ordinales, los cuales pueden ser agrupados en más de una categoría, como estados de la república, nombres de empleados, etc.

Del mismo modo y como su nombre lo indica, los datos numéricos son un conjunto de valores que expresan una cantidad, esta puede ser dinero, volumen, días, etc. En computación y en matemáticas siempre hay dos tipos generales de conjuntos numéricos con los que solemos trabajar, estos son aquellos conjuntos que son fácilmente contables, más em particular que toman valores enteros unicamente (discretos) y el conjunto de valores que no se limitan unicamente a números enteros, y pueden abarcar cualquier valor dentro del conjunto de los números reales (continuos).

En esta ultima parte cabe aclarar que hay que distinguir la definición de continuidad que nos proporciona la teoría matemática sobre los números reales y lo presentado en este trabajo, ya que como es bien sabido existe cierta limitante computacional para abordar la continuidad tal cual y como es concebida en la matemática tradicional, por lo cual aun que sean llamados datos continuos, entenderemos a estos como valores no necesariamente enteros.

Figura 2.1: fig. Tipos de datos



Una vez repasado la importancia de conocer el tipo de datos con los que nos podemos encontrar, es de suma importancia poder reconocer en nuestro conjunto de datos que

tipos tenemos, dos de las formas más comunes de visualizar esto en Python es atreves de pandas. La primera 2.1 nos proporciona unicamente el listado de todos las columnas que contiene el dataframe df y el tipo de dato de cada cada uno. Por otro parte 2.2 nos proporciona lo mismo, pero con el agregado de contar el número de registros no nulos dentro de cada columna.

```
df.dtypes
```

Listing 2.1: Ejemplo de conversión de fechas

```
df.info()
```

Listing 2.2: Ejemplo de conversión de fechas

Por lo tanto es bastante importante asegurarse que los campos que tenemos correspondan al tipo que en teoría deberían de tener, por ejemplo, si contamos con un campo que sea el id del producto que más compran los clientes o incluso el identificador de cada cliente, y esos datos son un tipo de dato numérico, valdría la pena considerarlos como datos categóricos y más del tipo ordinal, esto para evitar sacar conclusiones equivocadas sobre estos tipos de datos, una forma de cambiar el tipo de dato es la que se muestra en fragmento de código 2.3.

```
df.astype({
    'product_id': 'category',
    'customers_id': 'category'
  }
)
```

Listing 2.3: Ejemplo de conversión de tipo de datos

Previo a comenzar a hablar sobre el análisis univariado, bivariado, etc. Vale la pena mencionar que estas conversiones son a criterio, y que dependiendo como se obtengan los datos estos errores puedan prevenirse.

2.1.3. Visualización de datos

Esta sección aun no se desarrolla por completo.

2.1.3.1. Estadística Descriptiva: Análisis Univariado

En esta sección comenzamos a analizar las variables haciendo uso de la estadística descriptiva, (Mukhiya and Ahmed, 2020) menciona que la estadística descriptiva nos permite realizar un breve resumen acerca de nuestros datos y así poderlos entender con

mayor claridad. Estos resúmenes pueden ser presentado gráficamente o por medio de una representación numérica.

Así que comenzaremos a estudiar la distribución de nuestros datos, las medidas de tendencia central y dispersión de nuestras variables. Realmente esta parte es bastante simple, ya que Python y Pandas nos permiten ver estos estadísticos de manera sencilla que podemos ver en el código 2.4, la función *describe* integrada en Python nos permite realizar este análisis para variables categóricas y numéricas, haciendo uso del parámetro *include*, simplemente tenemos que especificar este parámetro como *[np.number]* y *object** para variables del tipo numéricas y categóricas respectivamente.

```
df.describe(include='all')
# Output:
               bp_id    Temporada  SO  ...  Linea_Negocio
count              660.0        660   660 ...            660
unique             149.0          2   3 ...              7
top             1001368.0         PV   3CR ...             FG
freq                26.0        348   622 ...            224
mean                 NaN        NaN   NaN ...            NaN
min                  NaN        NaN   NaN ...            NaN
25%                  NaN        NaN   NaN ...            NaN
50%                  NaN        NaN   NaN ...            NaN
75%                  NaN        NaN   NaN ...            NaN
max                  NaN        NaN   NaN ...            NaN
std                  NaN        NaN   NaN ...            NaN

```

Listing 2.4: Ejemplo Summary

De aquí viene la importancia de verificar que tengamos el tipo de dato deseado para cada variable, por lo cual es importante establecer una conversión haciendo uso de 2.3.

2.1.3.2. Estadística Descriptiva: Análisis Bivariado

2.1.3.3. Estadística Descriptiva: Análisis Multivariado

2.1.4. Reducción de dimensiones

2.1.4.1. PCA

2.1.5. Conjunto de entrenamiento y de validación

2.1.5.1. Tratamiento de clases desbalanceadas

Según Rodriguez (2018)

2.2. Aprendizaje Supervisado

2.2.1. Clustering

2.2.1.1. Kmeans

Cumplimiento de supuesto

2.2.1.2. Clustering GMM

Cumplimiento de supuesto

2.3. Aprendizaje No Supervisado

2.3.1. Clasificación Binaria

2.3.1.1. Regresión Logistica

Cumplimiento de supuesto

2.3.1.2. K Nearest Neighbors

Cumplimiento de supuesto

2.3.1.3. Random Forest

Cumplimiento de supuesto

2.4. Implementación de los modelos

2.4.1. Ajuste de los modelos

2.4.2. Metricas de Ajuste (Evaluación)

Capítulo 3

Abandono de clientes.

En este capítulo abordaremos el concepto de abandono de clientes, al ser este uno de los fenómenos que deseamos analizar y predecir, es de suma importancia comprender el concepto de abandono, esto para tener completo entendimiento sobre el comportamiento de los clientes y el fuerte impacto que puede tener el conocer el abandono dentro de una organización. En adelante nos referiremos al abandono como *churn* y a la tasa de abandono como *churn rate*.

La forma que usaremos para calcular la tasa de abandono en este trabajo sera la dada por 3.1, como podemos apreciar esta expresión considera cierto enfoque temporal, más adelante abordaremos el porque.

$$\text{Churn Rate} = \frac{\text{Número de clientes en el ultimo periodo}}{\text{Total de clientes}} - 1 \tag{3.1}$$

De hecho si tomamos la razon entre el numero de clientes en el ultimo periodo o en el actual y el total de clientes tendremos la tasa de retención de clientes. Entendiendo que la tasa de retención es el complemento de la tasa de abandono.

3.1. Definición de *churn*

De hecho Gold (2020, p. 36) define el origen de la palabra *churn* en el término *churn rate*, este termino hace referencia a la proporción de clientes que abandonan la empresa en un periodo determinado.

Más en especifico el *churn* es un estado u estatus que se le da a un cliente habitual de una empresa o servicio cuando este da por terminada la relación comercial, este concepto ha sido acuñado más recientemente por aquellas empresas que prestan servicios de suscripciones, como pueden ser plataformas de entretenimiento digital, proveedores de servicios de telecomunicaciones (internet, telefonía y/o televisión), etc. Sin embargo el *churn* no es únicamente aplicable a este tipo de compañías.

Ya que solo necesitamos que la empresa en cuestión oferte uno o más productos de los cuales tenga consumidores habituales para poder estudiar el *churn* en toda su cartera

de clientes y más en especifico poder abordar distintas estrategias para la retención de clientes.

En general el estudiar el comportamiento de los clientes puede ser un reto distinto dependiendo de la empresa e industria en la cual el científico de datos o analista de datos se encuentre, por lo cual es necesario tener pleno conocimiento sobre las reglas de negocio en cuestión. Ya que cada caso de estudio necesitará de enfoques distintos para poder entender de mejor manera el *churn* y cómo combatirlo.

De hecho ese debería ser el propósito principal de estudiar el *churn*, poder proporcionar una radiografía certera sobre el comportamiento de los clientes en una empresa, esto eventualmente permitirá a los líderes dentro de la organización poder tomar decisiones basadas en datos, y es tarea del científico de datos proveer la información lo más clara y precisa posible.

3.2. Metricas y causalidad del *churn*

Pese a que existen numerosas estrategias que podrían ayudar a disminuir la tasa de abandono y por lo tanto combatirlo, cabe resaltar que no nos enfocaremos en explicar dichas estrategias, más bien, nos enfocaremos en como aprovechar las razones subyacentes al *churn* para poder definir buenas métricas [1] que ayuden a evaluar el comportamiento de los clientes, y así poder usar estas mismas métricas en una eventual predicción del *churn*, el cual es propósito final de este trabajo.

Una definición de métrica para clientes puede ser la provista por Gold (2020, p. 51) "*cualquier medición que realice sobre todos los clientes individualmente*". Por lo cual es crucial encontrar las mejores metricas para monitorear a los clientes, de hecho Gold (2020) menciona las siguientes características importantes que debe tener una buena métrica; debe ser fácil de entender para la empresa, deben estar asociada con el *churn* y la retención, segmenta a los clientes de manera que facilita las intervenciones dirigidas a disminuir el *churn rate* y sobre todo es útil para múltiples áreas de la empresa (marketing, soporte, etc.)

En el capitulo anterior discutimos sobre la importancia de seleccionar correctamente las características para implementar *modelos de clasificación binaria*, mencionamos diversas técnicas de reducción de dimensiones basadas en criterios estadísticos, sin embargo, hay que considerar otras pautas a la hora de escoger dichas dimensiones si lo que deseamos es predecir el *churn*, estos juicios alternativos obedecen más a las reglas del negocio o empresa. Claro siempre hay que analizar la relación que pudieran tener con el *churn*.

[1] En este caso cabe aclarar que otros sinónimos con los nos referiremos a las metricas son características, dimensiones o variables predicadoras.

3.3. Como calcular el *churn*

Anteriormente expusimos la diversidad de enfoques del *churn*, que hay empresas dedicadas a ofertar servicios de suscripciones o que simplemente tienen una cartera de clientes habituales, pese a todo esto no todas las empresas tienen un indicador de abandono, de hecho en la mayoría de los casos lo primero sea determinar de manera correcta cuando un cliente a abandonado.

Para ellos en nuestro conjunto de datos asignaremos como *churn* a una columna cuyos registro sea true o false, esta fungirá como una bandera de abandono donde el valor true indica que un cliente abandono y False el caso contrario, como construir esta columna sera parte importante de este trabajo.

En el caso de las empresas que ofrecen suscripciones puede no aplicar, ya que saber cuando un cliente ha abandonado la compañía es más sencillo, podemos simplemente asignar el valor de True en la columna *churn* a todos aquellos que no cuentan con una suscripción activa y como False a los casos contrarios, y como mencionamos anteriormente, hablaremos da como construir el indicador de *churn*, para usar esta forma de determinar el *churn* es necesario contar con una variable que nos indique el tiempo desde la ultima transacción del cliente, y aun que no contemos con este dato podríamos determinarlo fácilmente de la siguiente manera:

```
1  fecha_ref = pd.Timestamp(datetime.now().date())
2  df['Tiempo_ultima_compra'] = df.groupby('bp_id')['Date_Last'].
     transform(lambda x: np.abs((fecha_ref - x.max()).days))
3
```

Listing 3.1: Ejemplo del calculo de fecha desde la ultima compra

Donde '$fecha\_ref$' es la variable que almacena la fecha del sistema cuando se ejecuta el programa, otra alternativa sería asignar a '$fecha\_ref$' un día en especifico, ejemplo el fin de mes. Por otra parte necesitamos una columna auxiliar en nuestro dataframe para realizar el calculo final que es el tiempo desde la ultima compra, esta columna es '*Date_Last*', esta contiene la ultima fecha de compra del cada cliente y en base a esta calcularemos el tiempo trascurrido desde la ultima compra, tal y como se muestra en el código 3.1.

Una vez determinada la columna $'Tiempo\_ultima\_compra'$ conviene analizar la distribución de los valores para dicha variable, esto nos permite ver el comportamiento general de cada individuo[1], en este caso, podemos visualizar el histograma de la variable en cuestión (figura 3.1) en la cual podemos apreciar que gran parte de los clientes no sobrepasan los 125 días de inactividad de compra, y de hecho podemos notar una linea vertical de color roja que indica el percentil 75 [2] de la variable '*Tiempo_ultima_compra*'

[1]En el capitulo 2 repasamos el análisis univariado y sus implicaciones teóricas, en esta sección nos enfocaremos unicamente en las aplicaciones.

[2]Para este ejemplo, decidí tomar el percentil 75, sin embargo la elección de percentil es a criterio

lo cual nos indica que los clientes con un tiempo de inactividad demasiado alto son realmente pocos.

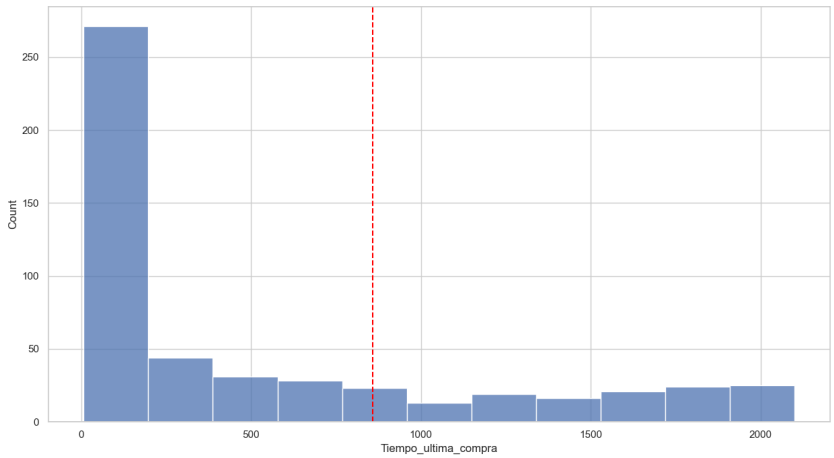


Figura 3.1: Histograma de Tiempo desde la ultima compra

Como sugerencia, conviene que incluyamos una columna en nuestro dataframe que contenga el percentil que deseamos tomar como condición para determinar el abandono, puedes seguir el ejemplo que se muestra en el código 3.2 como ejemplo de como realizar el calculo del k-esimo[1] percentil.

```
df['percentile_k'] = np.percentile(df['Tiempo_ultima_compra'], k)
```

Listing 3.2: Ejemplo de como determinar el percentil n

Una vez hecho el análisis del comportamiento de las ultima compra registrada de cada clientes y de haber calculado el percentil es hora de determinar la columna *churn* dentro de nuestro dataframe, por lo cual nos apoyaremos del código 3.3.

```
df['Churn'] = np.where(df['Tiempo_ultima_compra'] > df['percentile_k']
  ],True, False)
```

Listing 3.3: Ejemplo de como determinar el churn.

del científico de datos y esta fuertemente ligado a las caracteristicas de los datos con los que estemos trabajando, así que tómese este valor como lo que es, un ejemplo.

[1]Donde k es el valor del percentil a determinar.

Por ultimo, en la figura 3.2 podemos apreciar la proporción del *churn* en los clientes a analizar, y el resultado de la codificación anterior, en la gráfica de pie podemos notar que existe cierto desbalance en la proporción de clientes abandonadores y no abandonadores, recordemos que a posteriori esto puede ser un problema en la implementación de modelos de clasificación binaria, lo cual fue abordado en el capitulo 2.

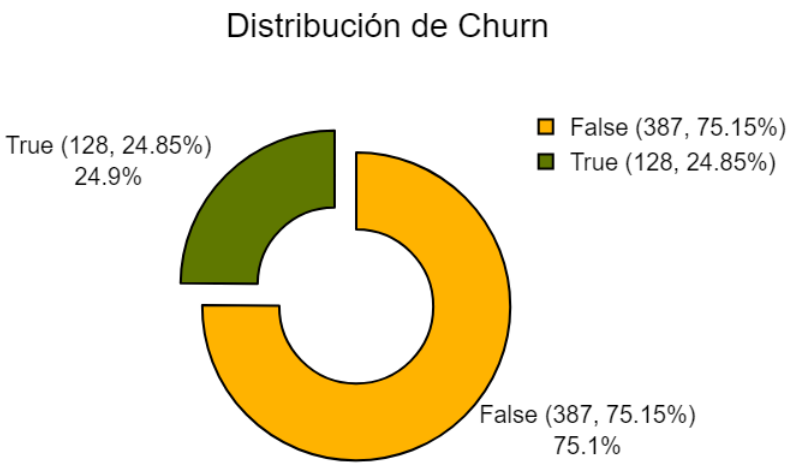


Figura 3.2: Gráfica de tarda para el abandono de clientes.

En estas ultimas secciones he abordado a profundidad el concepto de *churn*, el como analizar la actividad de compra de los clientes nos ayuda a comprender a como determinar el *churn*, por lo cual en la siguiente sección realizaremos un tipo de análisis más enfocado al comportamiento histórico de los clientes y ver cual ha sido el *churn* y retention rate.

3.4. Análisis de Cohortes y la tasa de abandono

Analizar el comportamiento del abandono de clientes a lo largo del tiempo nos puede ayudar a encontrar conductas que a simple vista suelen pasar desapercibidas. Por lo tanto cabe aclarar el significado de cohorte, (Gold, 2020) lo define como un grupo de individuos que son similares, y de hecho lo que nos interesa en este trabajo es estudiar esa similitud a lo largo del tiempo para poder encontrar patrones en el comportamiento del abandono de clientes.

Por lo cual los grupos que definiremos serán periodos de tiempo, agruparemos aquellos clientes que tuvieron actividad durante dichos periodos, en particular para este trabajo definiremos los cohortes como meses, pero bien podrían ser semanas, años, etc. El objetivo de identificar estos cohortes es tener mayor claridad de la actividad mensual de los clientes y poder hacernos las siguiente preguntas. ¿Cuantos clientes activos tiene una empresa por mes? ¿Cuantos clientes nuevos han adquirido? y ¿Cuantos clientes han perdido?

Podemos estudiar el comportamiento de estos cohortes usando diferentes metricas, ya sea por el numero de suscripciones a un servicio, volumen de compra, valor de compra o numero de clientes activos, este ultimo es la que usaremos en este trabajo, ya que esto nos permitirá relacionarlo con un concepto previamente visto, el *churn rate*.[1]

Hacer este análisis en python es tarea sencilla, primero es necesario tener la fecha de compra o fecha de facturación contenida en nuestro datafra, cabe aclarar que este ejercicio podría hacerse antes de determinar la variable de abandono, esto para evitar tener la correcta dimensión del dataframe con el que deseamos trabajar, ya que el nivel de detalle que deseamos tener para realizar este análisis es por fecha de compra, consumo, producto y clientes o usuarios. Aclarado lo anterior, podemos comenzar a revisar los pasos a seguir para hacer este análisis:

Duda: ¿Sera mejor mostrar esto como pseudocódigo?

1. Ordenar las observaciones de nuestro dataframe por fecha y id de usuario.[2]
2. Crear una variable que almacene el desplazamiento de cada cohorte en meses posteriores, esta variable nos ayuda a identificar los periodos de desplazamiento posterior a la adquisición de los clientes.
3. Determinar el mes de adquisición del cohorte de clientes.
4. Contar los clientes activos mensuales de cada cohorte y sus desplazamientos posteriores.
5. Del conteo final hay que tomar los clientes iniciales de cada cohorte y dividir todos los conteos por el número de clientes iniciales.

A continuación mostraremos la implementación de dicho análisis, el primero de los pasos esta ilustrado en el código 3.4

```
1  df.sort_values(by=['Fecha', 'bp_id'], inplace=True)
```

Listing 3.4: código del paso 1 del analisis de cohorte

[1] Antes de continuar tenemos que aclarar que hay distintos tipos de enfoques para agrupar los clientes, estos enfoques son inherentes al negocio o empresa en el cual se quiera implementar, además esta fuertemente ligado a la información que tengamos disponible, por lo cual hay que tener en cuenta estas limitantes antes de llevar acabo el análisis de cohortes.

[2] Es necesario que la variable que contenga la fecha sea del tipo de datetime para poder obtener una salida correcta en este paso y en los posteriores.

Y es que como bien lo indican las instrucciones simplemente tenemos que ordenar nuestro dataframe haciendo uso de la API de pandas.

Ordenar los datos de esta manera nos asegura que los cohortes que creemos tengan coherencia cronológicamente, esto porque en el paso 2 y 3 para determinar el mes de adquisición del cohorte y los meses de desplazamiento nos basamos en los meses de actividad de los clientes, en el código 3.5 creamos la columna *CohortMonth* la cual corresponde al mes de adquisición del cohorte, mientras que *InvoiceMonth* es el mes de compra cada cliente.

```
def get_month(x): return dt.datetime(x.year, x.month, 1)

df['InvoiceMonth'] = df['Fecha'].apply(get_month)
grouping = df.groupby('bp_id')['InvoiceMonth']
df['CohortMonth'] = grouping.transform('min')
```

Listing 3.5: código del paso 2 del analisis de cohorte

De forma auxiliar y previo obtener el resultado final, necesitamos crear una función que extraiga los valores enteros de los campos *CohortMonth* y *InvoiceMonth*, en el código 3.6 mostramos la definición de dicho método.

```
def get_date_int(df, column):
  year = df[column].dt.year
  month = df[column].dt.month
  day = df[column].dt.day
  return year, month, day
```

Listing 3.6: código para el método get_date_int

Análogamente para mejorar la forma en que vemos la información obtenida en el paso 3 nos apoyamos de la función *get_date_int*, el código 3.7 nos ayudará a crear una columna llamada CohortIndex, esta columna como su nombre es un conjunto de indices cuyo fin es representar los cohortes, osea nos permite tener una mejor visualización del desplazamiento de cada grupo de clientes.

```
invoice_year, invoice_month, _ = get_date_int(df, 'InvoiceMonth')
cohort_year, cohort_month, _ = get_date_int(df, 'CohortMonth')
years_diff = invoice_year - cohort_year
months_diff = invoice_month - cohort_month
df['CohortIndex'] = years_diff * 12 + months_diff + 1
```

Listing 3.7: código para la obtención de la columna *CohortIndex*

Por otra parte, para obtener el paso 4 debemos hacer uso de una pivot table agrupar los datos por *CohortMonth* y *CohortIndex*, así como agregar para obtener el resultado final, en el código 3.8 podemos observar como llevar acabo esta implementación. La utilidad de usar una tabla pivote para mostrar el resultado es porque gracias a esta forma podremos visualizar de mejor manera el desplazamiento desde el momento de adquisición para cada cohorte hasta la ultima actividad.

```
grouping = df.groupby(['CohortMonth', 'CohortIndex'])
cohort_data = grouping['pd_id'].apply(pd.Series.nunique)
cohort_data = cohort_data.reset_index()
cohort_counts = cohort_data.pivot(index='CohortMonth',
columns='CohortIndex',
values='pd_id')
```

Listing 3.8: código para la obtención de la columna *CohortIndex*

Finalmente para poder obtener la matriz de taza de retención tenemos que llevar acabo el paso 5, la implementación la mostraremos en fragmento de código 3.8 se crea la variable *cohort_sizes* que toma la primera columna, esta representa el número de clientes iniciales de cada cohorte, una vez hecho esto crearemos la tabla pivote final llamada *retention*, la misma es resultado de dividir toda la pivot *cohort_counts* por *cohort_sizes*, tal y como se muestra en la ecuación 3.1, de hecho ya podemos explicar el porque la ecuación del *churn rate* considera el numero de clientes por periodos de tiempo, y es que esta es la mejor forma de analizar la tasa de retención.

```
\# Dependencias
import seaborn as sns
import matplotlib.pyplot as plt
\# Calulo de la matriz de retencion
cohort_sizes = cohort_counts.iloc[:,0]
retention = cohort_counts.divide(cohort_sizes, axis=0)
retention.round(3) * 100
\# Visualizacion de matriz de retencion.
plt.figure(figsize=(35, 30))
plt.title('Retention rates')
sns.heatmap(data = retention,
yticklabels=df['CohortMonth'].sort_values().apply(lambda x: x.strftime
  ("%Y %b, %d")).unique(),
annot=True,
linewidths=.2,
fmt='.0%',
```

```
16    cmap='coolwarm',
17    vmin=0,
18    vmax=1)
19    plt.show()
```

Listing 3.9: código para la obtención de la columna *CohortIndex*

La gráfica 3.3 nos muestra el resultado final de las anteriores codificaciones, en los registros correspondientes a la primera columna podemos observar que la taza de retención es siempre del 100 % esto se debe a que son los clientes que tienen su primer actividad de compra dentro de ese cohorte, de tal forma que los registros de las columnas posteriores representan la actividad de los clientes después de su adquisición, por ejemplo, el cohorte enero de 2022 en el segundo mes[1] (febrero de 2022) podemos observar una retención del 76 % de los clientes iniciales y para el ultimo indice podemos notar que se mantuvieron 64 % de los clientes que iniciaron en enero del 2022, esto quiere decir que en **Fecha final** se perdió el 36 % de los clientes ingresados en enero de 2022, esto dicho grupo de clientes tiene un comportamiento de compra habitual y guardan cierta fidelidad.

Similarmente, los clientes adquiridos en agosto de 2022 tienen un comportamiento similar hasta 6 meses después de la adquisición, sin embargo es a aparir del mes 7 que este comportamiento baja al 40 %. El general la tasa de retención de clientes adquiridos en cada cohorte es bajo , ya que para el mes más reciente de cada cohorte podemos notar una baja tasa de retención, lo cual indica que la empresa ha perdido más clientes de los que ha ganado en los últimos meses.

[1] Ya que cada numero es una representación simplificada de cada cohorte, es importante tomar en cuenta que dicho numero se toma apartir del cohorte inicial, por ejemplo, para el cohorte enero de 2022 el numero 1 es igual al mismo mes, en cambio el numero 2 significa febrero de 2022, de forma similar el numero 3 es equivalente a marzo del 2022 y así sucesivamente hasta llegar al ultimo mes, por consecuencia ultimo mes solo tendrá datos hasta el indice 1, el penúltimo contiene información hasta el indice 2, etc.

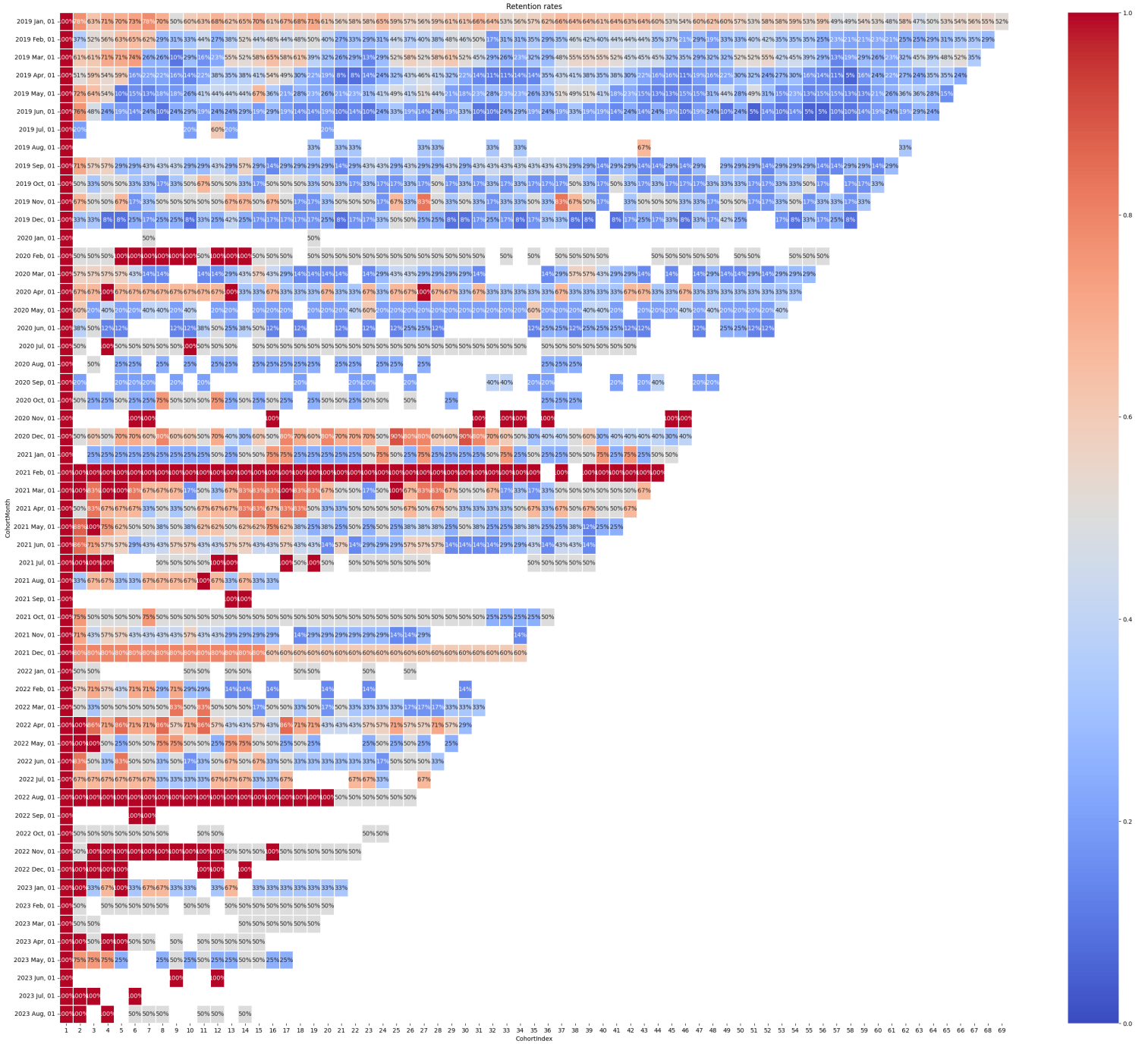


Figura 3.3: Matriz de Cohortes.

3.5. Implementación de modelos de ML para predecir el abandono

Una vez entendido el concepto de retención y abandono de clientes y las tasas y el análisis de cohorte, comenzaremos por mostrar la implementación de algoritmos de Machine Learning que nos permitan realizar predicciones del abandono. Recordemos que definimos la columna churn (abandono) como una variable de tipo binaria, donde true indica que un cliente abandono y false el caso contrario, en este caso nuestra variable objetivo sera la columna churn.

En esta sección nos apoyaremos de las herramientas vistas en la sección 2 y más en especifico llevaremos acabo la implementación de modelos de clasificación binaria para poder predecir las etiquetas de la variable churn, por lo cual no profundizaremos mucho en la teoría detrás, sino nos enfocaremos unicamente en los resultados obtenidos y la interpretación de los mismos.

Para este trabajo compararemos los resultados de distintos modelos de clasificación binaria, tomando en cuenta las metricas de evaluación como precisión, exactitud, etc, haciendo uso tanto de herramientas visuales y resúmenes estadísticos veremos las desventajas y ventajas de cada modelo, esto eventualmente nos conducirá a determinar el modelo que mejor realice la clasificación.

Antes de empezar con el análisis comparativo, es conveniente aclarar que los modelos que en adelante analizaremos ya fueron optimizados y entrenados en el capitulo anterior, por lo cual en esta sección tomaremos como punto de partida lo visto en el capitulo anterior.

3.5.1. Evaluación del modelo.

Previo a la profundización en cada modelo y sus resultados veremos como evaluar el rendimiento de cada modelo, en esta sección utilizaremos como ejemplo el modelo de regresión logística, por lo cual lo visto de aquí en adelante es fácilmente reproducible a cualquier otro modelo. Partiremos de las predicciones con el conjunto de datos de validación, así nos apoyaremos de la matriz de confusión y curva roc para evaluar nuestros modelos de forma visual y numérica.

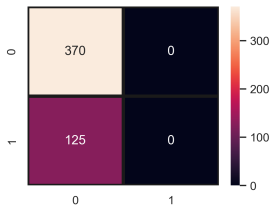


Figura 3.4: Matriz de confusión para el modelo de regresión logística.

En la gráfica 3.4 podemos apreciar el numero de aciertos (verdaderos positivos y negativos) y el numero de errores (falsos positivos y negativos) cometidos por el modelo. La representación gráfica anteriormente mencionada es la matriz de confusión asociada al modelo de regresión logística, de hecho esta gráfica nos permite encontrar la mayoría de las metricas para evaluar nuestro modelo. Por ejemplo, la ecuación 3.2[1] nos proporciona la *accuracy* o exactitud, esta métrica nos indica el porcentaje de acierto que tuvo el modelo. Python nos facilita el calculo de esta métrica, el código 3.10 muestra un ejemplo de como calcular la accuracy haciendo uso del sklearn.

Aquí pienso colocar el análisis de la matriz de este modelo, decidí no añadirlo por que puede cambiar.

$$\text{accuracy} = \frac{TP + TN}{(TP + TN + FP + FN)} \tag{3.2}$$

Sin embargo Peterson (2023) menciona que la accuracy usualmente no es una buena métrica para evaluar modelos cuando tenemos clases desbalanceadas en nuestra variable objetivo, lo cual es nuestro caso, ya que como mencionamos anteriormente la variable *churn* tiene más registros marcados como *false*, aun que en el capitulo anterior explicamos la forma en que lidiamos con el desbalance de las clases, explicaremos por que no siempre es conveniente optar por la exactitud, esto porque puede encubrir un incorrecto funcionamiento del modelo, por ejemplo, la matriz 3.4 podemos notar que el total de las observaciones que inicialmente fueron clasificadas como false son mayores a las clasificados como true, el problema es que aunque el porcentaje de aciertos en la clase con dominante es medianamente aceptable, siento mayor a 90 %, la clase minoritaria muestra un menor rendimiento, **insertar porcentaje** esta cifra indica el mal rendimiento para los registros clasificados como true.

Y es que aun que la clase true sea minoritaria, nos interesa tener la mayor cantidad de predicciones correctas para que en un futuro poder predecir correctamente los clientes que están abandonaran la compañía. Por lo tanto recomendamos optar por otras metricas para tener una visión mas completa del rendimiento de nuestro modelo.

```
1 from sklearn.metrics import accuracy_score,confusion_matrix,
    classification_report, precision_score, recall_score, f1_score
2 print("Accuracy: {}".format(accuracy_score(y_true=valid['y'],y_pred=
    valid['y^_lr'])))
3 print("Precision: {}".format(precision_score(y_true=valid['y'],y_pred=
    valid['y^_lr'])))
4 print("Recall: {}".format(recall_score(y_true=valid['y'],y_pred=valid[
    'y^_lr'])))
```

[1]Las variables TP, TN,FP y FN son el numero de casos true positives, true negatives, false positives y false negatives respectivamente.

```
print("F1: {}".format(f1_score(y_true=valid['y'],y_pred=valid['y^_lr']
 ])))
```

Listing 3.10: código para calculo de metricas

Algunas de las metricas a las que podemos recurrir son precision (precisión), recall (sensibilidad o recuperación) y F_β. Aun que existe muchas otras metricas, nosotros nos enfocaremos solo en estas. La precisión (ecuación 3.3) es el número de aciertos, predicciones correctas, entre el total de casos positivos captados por el modelo, osea del total de casos clasificados como true, cuales son realmente true, mientras que la sensibilidad (ecuación 3.4) es el numero de predicciones correctas, entre el total de casos positivos, es decir es la proporción de casos clasificados como true.

$$\text{precision} = \frac{TP}{(TP + FP)} \tag{3.3}$$

$$\text{recall} = \frac{TP}{(TP + FN)} \tag{3.4}$$

Hay distintas ventajas de usar una u otra métrica para evaluar nuestro modelo, si nosotros decidimos inclinarnos por la recall, estaríamos dando mayor importancia a reducir la cantidad de errores cometidos por el modelo, esto nos indica que esta métrica no nos aporta información acerca de los falsos positivos, y lo mismo pasa con la presicion, ya que al querer reducir el numero de falsos positivos estaríamos ignorando el rendimiento del modelo frente los falsos negativos, por lo cual escoger una u otra es decisión del científico de datos a cargo, en el código 3.10 nos muestra como calcular estas dos metricas con ayuda de Python.

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot (\text{precision} + \text{recall})} \tag{3.5}$$

La métrica que nos permite encontrar un equilibrio entre el numero de FP y FN es el F-score, esta métrica combina tanto la precision y recall en un solo valor, la ecuación 3.5 nos permite calcular dicho valor. El parámetro de β nos da la posibilidad de otorgarle un nivel de importancia a la precision o recall. De hecho si el valor de β es igual a cero el F-score le da mayor importancia a la precision, mientras que cuando β es igual a 0.5 el $F - score$ le da mayor peso a la recall, por ultimo si β es igual a 1, obtenemos el F_1-score (ecuación 3.6) le damos igual importancia a ambas metricas, de esta forma podemos evaluar tanto el número de falsos positivos y negativos.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{(\text{precision} + \text{recall})} \tag{3.6}$$

De la comparación entre precision y recall, surge otra forma de evaluar modelos de clasificación binaria, esta métrica es la curva roc (Receiver Operating Characteristic), Géron (2017, p. 91) nos dice que *la curva ROC representa gráficamente la tasa de*

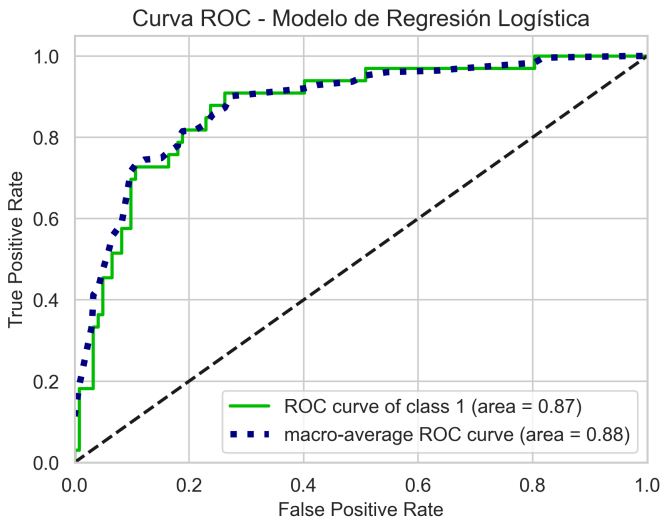


Figura 3.5: Curva ROC.

verdaderos positivos frente a la tasa de falsos positivos. La FPR es la proporción de casos negativos que se clasifican incorrectamente como positivos.

En la gráfica 3.5 podemos notar que el eje de ordenas esta etiquetado como True Positive Rate (TPR), mientras que el eje de abscisas esta etiquetado como False Positive Rate (FPR), donde las TPR puede ser también interpretado como la recall mientras que FPR suele interpretarse como el complemento de la especificidad (1- especificidad), (Géron, 2017) menciona que un buen clasificador se mantiene lo más posible alejado de la linea vertical punteada, dicha linea diagonal sirve más como una referencia y es también es llamada línea de no-discriminación.

Además de la curva roc, el área bajo la curva roc (comúnmente llamada AUC) es otra muy buena opción para medir el rendimiento de modelos de clasificación binaria, ya que un valor cercano a 1 de la AUC nos indicara que el contamos con un buen clasificador, este buen rendimiento se considera respecto a la especificidad y recall, es decir, estas dos metricas nos indica que nuestro modelo en cuestión dará una mejor calificación a la probabilidad de un registro positivo seleccionado al azar que a un registro negativo seleccionado de forma aleatoria.

De hecho este ultimo punto deja en evidencia que existe cierto inconveniente al evaluar nuestros modelos con la auc solamente, ya que la curva roc puede encubrir desbalances entre clases de la variable objetivo, Chugh (2023) plantea que la curva roc tiende a pasar por alto el desequilibrio entre clases, ya que la tasa de FP es una métrica que solo contempla a la clase negativa, lo que significa que se espera que el cambio en FP sea proporcional al cambio en FP+TN (todos los casos negativos), esto puede derivar en tener poca sensibilidad a cambios en la distribución de las clases de la variable objetivo.

Partiendo de este ultimo punto y recordando que en nuestro caso la información que

estamos analizando tiene cierto desequilibrio entre clases de la variable a predecir, sin embargo no todo esta perdido, ya que existe otra métrica que nos proporciona mayor información que la auc y que no pasa por alto el desequilibrio en entre clases, esta métrica es la curva precision-recall, en palabras de Chugh (2023) *una curva ROC es similar a la curva PR (Precision Recall), pero traza la tasa de verdaderos positivos (TPR) frente a la tasa de falsos positivos (FPR) para diferentes umbrales... La PR es comparativamente más informativa. Esto se debe a que la curva P-R proporciona información más significativa sobre la clase de interés en comparación con la curva ROC.*

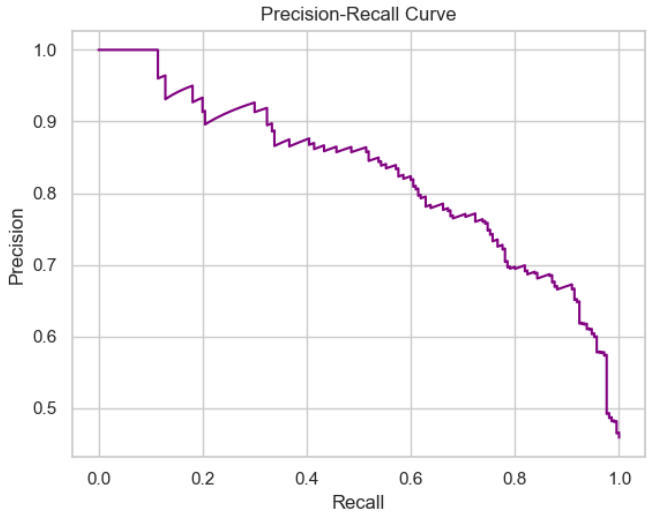


Figura 3.6: Curva PR.

Por lo tanto la curva PR nos ayudara a evaluar de mejor manera nuestros datos dando mayor importancia a los registros positivos pero sin dejar completamente de lado los negativos, encontrando así un equilibrio no arbitrario entre uno y otro. Esto porque uno de los grandes inconvenientes de elegir la auc sobre la PR es que en nuestro caso al tener mayor proporción de registros clasificados negativamente (osea como no abandono) hace que la tasa de FP sea demasiado pequeña, en comparación de la PR que al considerar la a la precision y por ende el rendimiento de nuestro modelo en los registros positivos.

3.5.2. Comparación del rendimiento entre modelos.

Capítulo 4

Customers Segmentations

4.1. Importancia

4.2. Analisis RFM

4.3. Implementación de modelos de ML para predecir Customer Segmentation

[(Lamport, 1986)]

Capítulo 5

Análisis, interpretación y discusión de los resultados

Capítulo 6

Coclusiones

Apéndice A

Código/Manuales/Publicaciones

A.1. Apéndice

Apéndice

Bibliografía

Bruce, P., Bruce, A., and Gedeck, P. (2020). *Estadistica Practica para ciencia de datos con R y Python*. Anaya Multimedia. 6

Chugh, V. (2023). Precision-recall curve in python tutorial. https://www.datacamp.com/tutorial/precision-recall-curve-tutorial. 26, 27

Gold, C. (2020). *Fighting Churn with Data: The science and strategy of customer retention*. Manning. 13, 14, 17

Géron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. O'Reilly. 25, 26

Lamport, L. (1986). *LaTeX: A Document Preparation System*. Addison-Wesley. 29

Mukhiya, S. K. and Ahmed, U. (2020). *Hands-On Exploratory Data Analysis with Python*. Packt Publishing. 8

Peterson, M. (2023). *Marketing Analytics: Predicting Customer Churn in Python*. DataCamp. 24

Rodriguez, D. (2018). El problema de desequilibrio de clases en conjuntos de datos de entrenamiento. https://www.analyticslane.com/2018/07/04/el-problema-de-desequilibrio-de-clases-en-conjuntos-de-datos-de-entrenamiento/#google_vignette. 10