Facultad de Ciencias Económicas y Empresariales
ICADE

ANÁLISIS DE PÉRDIDA Y RETENCIÓN DE CLIENTES EN PLATAFORMAS *OVER-THE-TOP*

Autor: Íñigo Gemperle Sánchez del Corral
Director: Lourdes Fernández Rodríguez

MADRID | Abril 2024

Por la presente, yo, Íñigo Juan Gemperle Sánchez del Corral, estudiante de E2 Analytics de la Universidad Pontificia Comillas al presentar mi Trabajo Fin de Grado titulado "Análisis de Pérdida y Retención de Clientes en Plataformas *Over-The-Top*", declaro que he utilizado la herramienta de Inteligencia Artificial Generativa ChatGPT u otras similares de IAG de código sólo en el contexto de las actividades descritas a continuación:

1. **Brainstorming de ideas de investigación:** Utilizado para idear y esbozar posibles áreas de investigación.
2. **Crítico:** Para encontrar contraargumentos a una tesis específica que pretendo defender.
3. **Metodólogo:** Para descubrir métodos aplicables a problemas específicos de investigación.
4. **Interpretador de código:** Para realizar análisis de datos preliminares.
5. **Corrector de estilo literario y de lenguaje:** Para mejorar la calidad lingüística y estilística del texto.
6. **Generador previo de diagramas de flujo y contenido:** Para esbozar diagramas iniciales.

Afirmo que toda la información y contenido presentados en este trabajo son producto de mi investigación y esfuerzo individual, excepto donde se ha indicado lo contrario y se han dado los créditos correspondientes. Soy consciente de las implicaciones académicas y éticas de presentar un trabajo no original y acepto las consecuencias de cualquier violación a esta declaración.

Fecha: 22/04/2024

Firma: 

RESUMEN

El presente trabajo está enfocado en analizar la tasa de rotación de clientes en plataformas de servicios de *streaming* en la industria del entretenimiento *Over-The-Top* (OTT). Diferentes técnicas de análisis de datos han sido empleadas para obtener una mejor comprensión del comportamiento de los usuarios de estos servicios. Se demuestra la eficacia de los modelos de predicción y clasificación utilizados mediante un contraste de hipótesis, que pretende validar la utilidad de estos, en el diseño de estrategias de retención de clientes de las empresas en este sector.

PALABRAS CLAVE

Pérdida de clientes, Retención de clientes, *Over-The-Top,* Modelos de predicción, Modelos de clasificación, *Streaming,* Contraste de hipótesis, *Churn*, Análisis de supervivencia.

ABSTRACT

This paper aims to analyze the customer churn rate of platforms in the streaming service Over-The-Top entertainment industry. Many data analysis techniques have been employed to get a better understanding of the behavior from the users of these services. It is demonstrated that this prediction and classification models can be used by means of a hypothesis test, to validate designs in customer retention strategies from different companies in this industry.

KEY WORDS

Customer churn, Customer retention, Over-The-Top, Prediction models, Classification models, Streaming, Hypothesis testing, Churn, Survival analysis.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

1.1. Alcance

1.1.1. Objetivos de la Investigación

El objetivo principal de este Trabajo de Fin de Grado (TFG) es entender la tasa de cancelación de clientes o suscriptores en diversas plataformas de entretenimiento, así como identificar aquellos que optan por mantener la suscripción. Se busca diseñar estrategias basadas en análisis de datos y modelos predictivos para retener a estos usuarios. También se investiga el sesgo en el género y las preferencias de uso que estos tienen.

Además, se analiza la evolución de la industria del entretenimiento y su impacto en el mercado, con un enfoque en plataformas como Netflix, HBO, Disney+, Twitch, entre otras. Se examina cómo las plataformas de *streaming* han transformado de manera significativa el panorama de los medios de comunicación, desafiando a la televisión convencional.

Estas plataformas están impulsando la producción de contenido libre de manera exponencial, forzando a que el resto de los medios tengan que adaptarse a este cambio. Además, se ha creado una nueva manera de monetizar el contenido, con suscripciones mensuales sin permanencia, lo que también está teniendo repercusión en los modelos de negocio de muchas empresas dedicadas al marketing y a la publicidad.

1.1.2. Hipótesis

Este TFG utiliza diferentes modelos de aprendizaje automático o *machine learning*. El desarrollo del trabajo trata de demostrar que estos se pueden emplear para diferentes análisis dentro de la industria del *streaming* y servicios de vídeo. Aunque se prueba que pueden llegar a obtener valores altos de precisión en distintos análisis, esto no es suficiente por sí solo. Para que estos modelos sean fiables y se puedan utilizar en cualquier contexto, se debe realizar un contraste de hipótesis que demuestre que éstos superan de manera significativa la precisión que presenta la regla de la mayoría. Estos conceptos se explican con mayor detalle en el apartado 4.1 del trabajo.

1.1.3. Asunciones

En un Trabajo de Fin de Grado como este se debe detallar cuales son las suposiciones que se toman para que el análisis cobre sentido. Como se explicará a lo largo del desarrollo, la base de datos utilizada podría estar más completa. Esto quiere decir, que para realizar este análisis se han aceptado ciertos factores. En primer lugar, se asume que los datos utilizados son representativos y plenos de una industria tan grande como los servicios de *streaming*. En segundo lugar, que los modelos de *machine learning* aplicados son los más apropiados. Por último, que las métricas utilizadas son las más adecuadas para evaluar el rendimiento de estos modelos. Es crucial comprender estas presunciones para entender el contexto completo del análisis y sus resultados, ya que han sido establecidas por la limitación de los recursos y alcance del estudio.

1.1.4. Restricciones

Para terminar de comprender el contexto del trabajo es fundamental valorar que existen distintos obstáculos. En primer lugar, existe como inconveniente el tiempo para completar todo el trabajo. Para cumplir correctamente el propósito que tiene este TFG se debería disponer de un plazo más largo, logrando así ahondar mejor en cada parte. En segundo lugar, existen limitaciones en cuanto a los recursos de adquisición y preparación de los datos. Por último, y la más importante, existe la restricción en la capacidad computacional para el entrenamiento y evaluación de los modelos. Aunque estos factores suelen estar presentes en cualquier análisis, cabe destacar su presencia e impacto en la planificación de este.

1.2. Presentación del Tema

La industria tecnológica ha estado creciendo exponencialmente durante las últimas décadas. Esto ha llevado a una evolución constante de numerosas industrias vinculadas a la tecnología, y a cómo están interconectadas. La televisión y otros medios de comunicación, como los periódicos o la radio, son canales que se han visto afectados en su manera de transmitir y llegar al público.

La televisión es uno de los medios de comunicación de masas más importantes. La sociedad tiende a depender de programas de televisión informativos, de noticias, o de

deportes, para el entretenimiento. Sin embargo, estos medios están decayendo en volumen de audiencia y relevancia.

La televisión y radio son los medios preferidos para las generaciones de mayor edad. Los *Baby Boomers* es la generación que más consume televisión por cable. Solamente un 9% de la audiencia (Wise, 2023) está comprendida en el rango de edad de los 18 a los 24 años. Esto quiere decir que las generaciones más jóvenes ya no están satisfechas con los medios de comunicación tradicionales, y que demandan algo más. Las nuevas generaciones solicitan conocer toda la información de manera más instantánea, e informarse en tiempo real, desde la comodidad de su teléfono móvil u ordenador.

El desarrollo de Internet tuvo su comienzo durante la Guerra Fría, con la creación de la teoría de conmutación de paquetes en el año 1.961. Esta teoría se basaba en que toda la información que se desprendía de un dispositivo era fraccionada para facilitar la transmisión. Esas fracciones fueron denominadas paquetes. A raíz de esta teoría, se estableció un sistema de comunicaciones descentralizado al año siguiente.

En 1.965, se consiguieron conectar dos ordenadores mediante una línea telefónica. Cuatro años después se lograron conectar cuatro ordenadores, y se nombró a esta interconexión Arpanet. En 1.989 se introdujeron recursos clave como *Hypertext Markup Language* (HTML) y *Hypertext Transfer Protocol* (HTTP), comúnmente utilizados para diseñar páginas web y transferir datos, seguidos por el lanzamiento de Internet al público en general en 1.991 (Telefónica, 2024).

Con este crecimiento, surgieron los primeros servicios y plataformas de *streaming* por Internet, en abril del año 1.995 (Colaboradores de Wikipedia, 2024). En consecuencia, surgieron los servicios *Over-The-Top* (OTT). Durante los últimos años, los servicios OTT han pasado de ser una opción poco utilizada, a una de las más consumidas en los sectores de la información y el entretenimiento.

Con la nueva era digital, el uso de Internet aumenta a diario, y con él, sus servicios. El servicio de *streaming* logró alcanzar un incremento del 91,7% («Caracterización del uso de algunos servicios *Over The Top* en España (Comunicaciones electrónicas y servicios audiovisuales)», 2015, p.9) hasta 2.013, y desde entonces su crecimiento no se ha detenido.

El *streaming* lleva incrementando su popularidad de manera considerable en los últimos años, lo que ha llegado a transformar este género en una de los más comunes para el contenido multimedia. Aunque el más conocido es el de las películas, el mundo de los *eSports* ha ganado muchísima audiencia. Una muestra de la gran relevancia que ha logrado esta industria es que en 2.022 contaban con un total de 3.200 millones de usuarios alrededor del mundo en todas las plataformas (Statista, 2023).

La industria está dividida en varios sectores. En primer lugar, el *streaming* de series y películas, en el que se encuentran empresas como Netflix, Amazon Prime o Disney+. Por otro lado, se hallan la transmisión de música y *podcast*, siendo las plataformas más conocidas Spotify, Apple Music y YouTube. Finalmente, está el servicio de videojuegos y torneos *eSports,* con Twitch y YouTube como las más destacadas.

En conclusión, esta industria ha mostrado un crecimiento exponencial a lo largo de los años. La pandemia dio un impulso a estas plataformas, que, sin duda, ha logrado captar la atención de millones de personas e inversores. Se estima que este crecimiento continúe, lo que la convierte en una industria excelente tanto para el entretenimiento personal, como el desarrollo económico.

2. DEFINICIÓN Y CONTEXTO DE LA INDUSTRIA *OVER-THE-TOP*

> El mercado OTT (*Over-The-Top*) se refiere a la distribución de vídeo, audio y otros contenidos multimedia a través de Internet directamente a los usuarios, prescindiendo de los canales de distribución tradicionales como la televisión por cable o por satélite. Las plataformas OTT ofrecen acceso a la carta a una amplia gama de contenidos, como películas, programas de televisión, eventos en directo y programación original (*Over the Top (OTT) Market Insights*, s.f.).

Muchas organizaciones desglosan los servicios OTT de distinta forma. Los dos principales se pueden clasificar en servicios audiovisuales y de audio. Ambos se definen como un servicio OTT que se distribuye sujeto a demanda a través de un dispositivo con acceso a Internet. El primero engloba plataformas como Netflix, Amazon Prime Video, Disney+ y YouTube. Y los de audio engloban plataformas como Spotify, Apple Music o YouTube Music Premium.

Los modelos de negocio de las plataformas OTT están organizados de manera diferente, en función de la demanda que tiene cada una. Sin embargo, todas ellas tienen en común que suelen estar basadas en un modelo de suscripción, y en algunos casos, algunas ofrecen un modelo *freemium*. Este modelo de negocio está basado en una estrategia por la que las empresas ofrecen su producto o servicio de manera gratuita a los usuarios, pero con algunas limitaciones, es decir con las funciones básicas. Se ofrece la posibilidad a los clientes de ampliar el servicio a un acceso premium, con todas las características y funcionalidades. Esta estrategia es muy común y suele ser beneficiosa para las empresas que la utilizan, ya que les permite ampliar su base de usuarios.

Muchas de estas plataformas sufren el problema de mantener los suscriptores durante largos periodos de tiempo, dado que muchas de ellas ofrecen un contenido similar. Para el usuario, es muy fácil pasarse de una a otra plataforma, y consumir el contenido de todas, por lo que es un mercado con una demanda y una oferta muy amplia. El *churn* es la tasa de abandono de suscripción, es decir, el porcentaje de usuarios que cancelan su membresía dentro de un periodo de tiempo determinado. Esta tasa será el objeto de estudio en este TFG.

La pérdida de clientes es un problema vivo y complejo dentro de esta industria, dado que los usuarios piden cada vez más contenido en directo y al momento. Por lo tanto, las plataformas tienen la complicación de mantener el ritmo que demandan los clientes. Es una métrica importante para los servicios OTT, ya que indican tanto la satisfacción de los consumidores, como la eficacia de la publicidad o del marketing que utiliza la empresa. Si una compañía tiene un *churn* alto, esto indica que los clientes no están satisfechos con el servicio, y que quizá hay mejores opciones en el mercado.

El uso de Internet difiere enormemente entre jóvenes y mayores. Las generaciones más jóvenes lo utilizan de una manera más intensa. “Las mayores tasas de penetración del servicio se encuentran entre los usuarios de edades comprendidas entre los 16 a 34 años de edad.” («Caracterización del uso de algunos servicios *Over The Top* en España (Comunicaciones electrónicas y servicios audiovisuales)» 2015, p. 11). Por lo tanto, las empresas deben ajustar su contenido y estrategias de mercado al público al que quieran dirigirse. Los adolescentes no tienen los mismos hábitos de consumo, ni las mismas preferencias que los adultos.

2.1. Estudio del Mercado

Los ingresos de la industria OTT y su cuota de mercado han crecido de manera exponencial, y continúan haciéndolo a diario. Al final del ejercicio correspondiente a 2.021, la industria OTT alcanzó un tamaño de mercado de 238 mil millones de dólares, y se espera que para finales del año 2.027 alcance los 476 mil millones (*The State Of OTT 2023 - Technologies & Key Trends Impacting Over-The-Top Media Brands*, 2023). Según Statista, el número de usuarios que las OTT esperan tener para 2.027 es de 4,2 mil millones (Statista, s.f.).

Por otro lado, los servicios de *streaming* han cambiado enormemente la manera de monetizar contenido en Internet. El sector que más dinero genera en esta industria es el de la publicidad. Se espera de este sector unos ingresos de 191,3 mil millones de dólares para finales de 2.024 (Statista, s.f.), lo que supondría más de la mitad de los ingresos totales de la industria.

Desde la pandemia por el Covid-19, la utilización de estos servicios se vio afectada por el confinamiento, ya que era la manera óptima para consumir contenido multimedia de entretenimiento. Estados Unidos es el país con el mayor valor de mercado, con un total de 76,7 mil millones de dólares en 2.023 (Statista, s.f.), que, comparado con el resto de los países, será el que más ingresos genere: 126,5 mil millones de dólares (Statista, s.f.).

2.2. Servicios de Plataformas de Vídeo

Las plataformas más populares que ofrecen contenido en vídeo son Netflix, HBO, Disney+, Amazon Prime Video, Apple TV, Twitch y YouTube. En este apartado se hace una comparación de las que disponen de un mayor número de usuarios, y por relevancia. Se han excluido HBO y Apple TV debido a que en España no son las plataformas más utilizadas. Ambas se consumen por la exclusividad de su contenido, pero los catálogos son menos extensos que el resto, y en el caso de la segunda, con pagos adicionales, lo que conlleva a una menor demanda.

Netflix es la plataforma pionera y líder en el mercado con unos ingresos de 33,7 mil millones de dólares en 2.023, lo que supone un 6,6% de aumento con respecto a 2.022. Además, cuenta con 238,3 millones de suscriptores alrededor del mundo (*Netflix Revenue and Usage Statistics (2024) - Business of Apps*, 2024). El éxito de Netflix se debe a su

amplio catálogo de películas, series y documentales, así como su interfaz intuitiva y fácil de usar. Sin embargo, el precio de la suscripción es algo relativamente mayor al resto de las plataformas.

Otra de las plataformas más populares es Disney+, que cuenta con un total de 150 millones de suscriptores globalmente en el último año. Además, tuvo unos ingresos de más de 8 mil millones de dólares en 2.023, lo que supuso un incremento del 13% con respecto al año anterior (*Disney Plus Revenue and Usage Statistics (2024) - Business of Apps*, 2024). El contenido de esta plataforma está más dirigido a niños y familias que buscan disfrutar de las franquicias clásicas de Pixar, Marvel, Star Wars o Disney. El catálogo es menos extenso que Netflix, pero dispone de un contenido de mayor calidad, y a un precio menor.

Amazon Prime Video tiene un modelo de negocio algo diferente al resto de plataformas, y este servicio viene incluido con la suscripción regular a Amazon Prime. El modelo que emplean es el de un catálogo variado con acceso libre a todos los usuarios, y un contenido de pago extra para comprar o alquilar películas y series concretas. Además, la interfaz es menos intuitiva que las plataformas anteriores.

Por otro lado, Twitch está más enfocada al contenido de videojuegos. Esta plataforma permite a los usuarios interactuar con los *streamers* o creadores de contenido en directo. Esto, y el modelo de negocio basado en las suscripciones y donaciones ha tenido gran éxito en la industria, posicionando la marca en el top de plataformas de videojuegos. YouTube, por otro lado, dispone de un servicio muy similar a Twitch, ofreciendo además la posibilidad de guardar el contenido en forma de vídeos. Es decir, que el contenido de Twitch no se mantiene subido, y el de YouTube sí.

3. METODOLOGÍA

En primer lugar, para la realización de este Trabajo de Fin de Grado es necesaria una base de datos confiable y relevante en el sector de la industria OTT. Este conjunto debe incluir información demográfica como la edad o el género, así como el tiempo de uso de las plataformas.

La base de datos que se ha utilizado necesitaba un proceso de limpieza y depuración para conseguir tener únicamente la información más relevante. De esta manera se logra

deshacerse de la que no es útil para el objetivo de este estudio. Se describen más adelante las técnicas utilizadas durante esta etapa, explicando cada paso del proyecto de manera clara y comprensible.

A continuación, se realizó un estudio exploratorio de los datos para comprender mejor la información, identificar algún patrón y visualizar las variables más relevantes. Este análisis incluye la búsqueda y el manejo de los valores nulos e irrelevantes de la base de datos, que se encuentran en la depuración de la información.

Después, se procederá al análisis del *churn* de los clientes en la industria OTT, es decir, la rotación o pérdida de clientes. Este análisis descriptivo es importante para que las empresas puedan estudiar y reducir su tasa de pérdida de clientes. Para ello, en este trabajo se hará un análisis profundo de dicha tasa, en la que se incluirán factores como la duración media de suscripción, o el número de veces que se llama a atención al cliente, como factor de satisfacción.

En resumen, la metodología de este trabajo incluye desde la selección y procesamiento de los datos, hasta su análisis exploratorio y predictivo. Esto proporciona una visión completa y en detalle del estudio en cuestión, proporcionando también estrategias prácticas para la retención de clientes en la industria OTT.

3.1. Descripción de los datos

Para la consecución del propósito que tiene este Trabajo de Fin de Grado, es necesario contar con una base de datos que contenga información relevante. En primer lugar, debe contener información correspondiente a la demografía de los usuarios, como puede ser la edad o el género. Por otro lado, referencias a los hábitos de consumo que tienen los suscriptores, como la frecuencia con la que utilizan la plataforma, la duración de cada sesión, o el tipo de servicio que tienen contratado.

En la mayoría de las ocasiones, es útil conocer si el usuario ha mantenido alguna interacción con el servicio de atención al cliente de la plataforma, poniendo alguna queja, o en general, saber qué problemas ha sufrido. La calidad de los datos, y la información que estos recojan, debe ajustarse a cada estudio, y variará en función de lo que se quiera investigar o demostrar.

La búsqueda del *dataset* se realizó en diferentes repositorios, como Data.gov, OpenML,

Data.world y Kaggle. Ésta concluyó en una base de datos que cumplía las características mencionadas anteriormente, extraída del repositorio Kaggle. Es la correspondiente a una plataforma de servicios OTT que ha sido anonimizada, con datos relativos únicamente a 2.020. Aunque lo ideal sería contar con datos completos desde que empezó a operar la plataforma, es la mejor información que se ha encontrado. Lo que se pretende con este estudio, es que las plataformas OTT puedan utilizar estos modelos de *machine learning* en sus operaciones para predecir la pérdida de clientes y poder trazar planes de acción sobre ello.

La base de datos original contaba con un total de 16 variables y 2.000 observaciones. Aunque para realizar un estudio como este, y que los resultados sean fiables, lo ideal sería contar con un conjunto de mayor volumen, este fichero era el más significativo.

La descripción de las variables se encuentra en la Tabla 1, que contiene 3 columnas respectivas a: número de la variable, nombre y breve descripción de lo que representa cada una.

Tabla 1. Descripción de las Variables

Número de variable	Nombre	Definición
1	year	Año
2	customer_id	Identificador del cliente
3	phone_no	Teléfono
4	gender	Género
5	age	Edad
6	no_of_days_subscribed	Número de días que lleva suscrito en 2.020
7	multi_screen	Tipo de servicio contratado: un dispositivo/varios
8	mail_subscribed	Suscrito a *newsletter* por correo
9	weekly_mins_watched	Minutos empleados por semana
10	minimum_daily_mins	Minutos mínimos de visualización al día
11	maximum_daily_mins	Minutos máximos de visualización al día
12	weekly_max_night_mins	Minutos de visualización por la noche
13	videos_watched	Películas/episodios totales vistos
14	maximum_days_inactive	Máximo de días inactivo
15	customer_support_calls	Llamadas a atención al cliente
16	churn	Sí el cliente se ha dado de baja o no

Fuente: (*Churn Modeling for OTT Platforms*, 2023)

Para el procesamiento de los datos, se ha utilizado el lenguaje de programación Python en la versión 3.11.5, en un Jupyter Notebook en la versión 6.5.4. Se han empleado

librerías como pandas, numpy, matplotlyb y sklearn, entre otras. Este lenguaje permite a los usuarios escribir y ejecutar código de programación de manera interactiva. Es muy útil para realizar análisis de datos estadísticos, ya que es muy conocido por su simplicidad y fácil uso. Las librerías son conjuntos de herramientas que sirven para realizar unas tareas en concreto. En primer lugar, pandas es usada comúnmente para la manipulación de datos estructurados.

Por otro lado, numpy es muy útil para trabajar con datos organizados en matrices y operar numéricamente con ellas. La librería Matplotlib permite crear visualizaciones como gráficos o diagramas, y, por último, sklearn para realizar análisis predictivos como la clasificación.

El análisis comienza procesando estas librerías en el cuaderno de trabajo, es decir, el cuaderno virtual en el que se ejecuta el código de Pyhton. Al cargarlas, se encontrarán disponibles para utilizarlas en cualquier momento. Posteriormente, se procede a imputar el conjunto de datos que se va a utilizar, que comúnmente se introducen en documentos con formato Excel o CSV. Una vez se ha establecido bien la información que se necesita es imprescindible comprobar que los datos se han almacenado de la manera correcta. Cada fila de estos datos debe corresponder a un cliente diferente, y cada columna representar una característica como la edad, el género, etc.

Un comando es una instrucción que se le da al ordenador en un lenguaje específico para que realice una acción concreta. El comando ‘datos.shape’ en Python se utiliza para mostrar el número de observaciones y variables que hay en el conjunto de datos. Esto se realiza para comprobar que el tamaño se ha cargado correctamente. Al ejecutarlo, se visualiza que existe un total de 2.000 observaciones y 16 variables, por lo que a partir de aquí se puede empezar a tratar los datos.

A continuación, se establece un comando adicional para que se muestre la estructura y la manera en la que está organizada la información, como se muestra en la Tabla 2. En esta hay disponibles 4 campos, el número de variable, su nombre, la cantidad de valores no nulos que contiene y el tipo de variable que es.

Tabla 2. Estructura de los datos

```
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   year                   2000 non-null   int64
 1   customer_id            2000 non-null   int64
 2   phone_no               2000 non-null   object
 3   gender                 1976 non-null   object
 4   age                    2000 non-null   int64
 5   no_of_days_subscribed  2000 non-null   int64
 6   multi_screen           2000 non-null   object
 7   mail_subscribed        2000 non-null   object
 8   weekly_mins_watched    2000 non-null   float64
 9   minimum_daily_mins     2000 non-null   float64
 10  maximum_daily_mins     2000 non-null   float64
 11  weekly_max_night_mins  2000 non-null   int64
 12  videos_watched         2000 non-null   int64
 13  maximum_days_inactive  1972 non-null   float64
 14  customer_support_calls 2000 non-null   int64
 15  churn                  1965 non-null   float64
dtypes: float64(5), int64(7), object(4)
```

Como se puede observar, las variables son de diferentes tipos: 'int64', '*object'* y 'float64'. La primera significa que todos los datos comprendidos en esa variable son números enteros, es decir, sin decimales. El 64 que aparece en el nombre hace referencia a los 64 bits de memoria que Python asigna a cada celda en la que se almacena el dato. El tipo '*object'* es una estructura en forma de cadena de texto, y 'float64' son variables que sí que tienen decimales, *float* en adelante.

El año, el identificador, la edad, el número de días suscritos, el número de minutos por la noche, los vídeos vistos y el número de llamadas a atención al cliente son variables de tipo entero. Por otro lado, están las variables de tipo *float*, que son los minutos de visualización a la semana, el mínimo y el máximo visto en un día, y el número de días inactivo. *Churn* aparece también como *float* porque el sistema le añade un 0 como decimal, cuando en realidad es de tipo binaria con un 1 cuando el cliente se ha dado de baja en la suscripción, y un 0 cuando no.

Hay cuatro variables de tipo carácter:

- Número de teléfono: identificado como carácter debido al guion que contiene entre los números.
- Género: masculino o femenino.
- *Multi_screen:* muestra un sí o no a si tiene contratado un servicio con más de un dispositivo.
- *Mail_subscribed*: muestra sí o no a si está suscrito a la *newsletter* por correo.

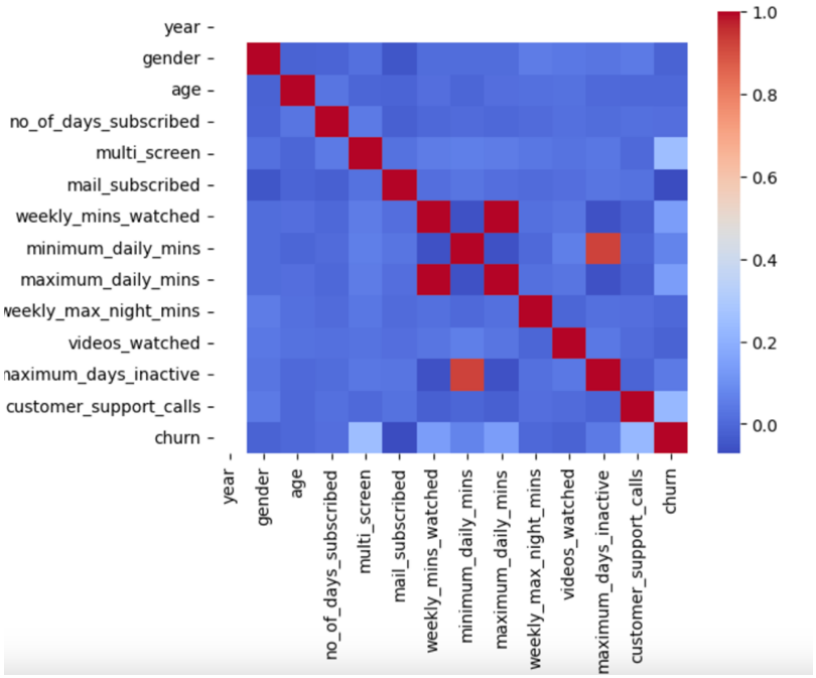
Además, más adelante (Figura 1) se ha representado un mapa de correlaciones, que se utilizan para entender la relación que tienen las variables entre sí. En un mapa de

correlaciones, se utiliza comúnmente colores y grosor diferentes para las variables que sí que están relacionadas de alguna manera y la fuerza que tiene esa correlación.

Como se observa en la Figura 1, existe una correlación entre '*weekly_mins_watched'* y '*maximum_daily_mins'*. Esto se muestra en que el nodo que representa esa relación se encuentra coloreado con un valor correspondiente a 1, lo que implica que cuanto mayor sea la cantidad de minutos que el usuario consume, mayor será la cantidad de minutos por semana. Puede parecer obvio a simple vista, pero es necesario hacer este tipo de comprobaciones.

Algo parecido ocurre con '*minimum_daily_mins*' y '*maximum_days_inactive*', puesto que, a mayor número de minutos vistos al día, menor será la cantidad de días que esté inactivo este usuario.

Figura 1. Mapa de Correlaciones



Se debe tener en cuenta que el mapa de correlaciones lo único que muestra es la correlación entre las variables, y no la causalidad. Aunque indica que dos variables están relacionadas, no garantiza que un cambio en una de ellas afecte en la misma dirección o magnitud a la otra. Por ejemplo, una correlación negativa o inversa indicaría que el aumento de una variable está asociado con la disminución de la otra, pero no explícitamente en la misma proporción. Este tipo de gráfico proporciona una visión global de las relaciones que existen entre las variables y puede mostrar tanto correlaciones

positivas como negativas, pero sirve únicamente como punto de partida para realizar un análisis más extenso y detallado. Es importante tener en consideración la dirección y la magnitud de la correlación.

Una vez entendidos bien los datos que existen y cómo están estructurados, se procede al procesamiento de éstos, limpiando y depurando la información, como se ha explicado anteriormente. En primer lugar, hay que decidir si todas las variables son necesarias y aplican a este estudio. Se puede prescindir de tres: el año, el identificador del suscriptor y el número de teléfono. El año es 2.020 para una todos los casos, y el identificador y número de teléfono sirven para que la empresa identifique a sus clientes, por lo que esta información no es relevante.

Por lo tanto, se procede a eliminarlas de la base de datos, y a ejecutar el comando '*head*(5)', el cual instruye al ordenador que muestre las cinco primeras observaciones. Esto es útil para comprobar que los datos que se han eliminado han sido extraídos correctamente. Según la información mostrada en la Tabla 3, las variables del año, el número de teléfono y el identificador, han sido eliminadas correctamente. Esta tabla también sirve como referencia a cómo está estructurado el contenido de los datos de cada observación.

Tabla 3. Muestra de los Datos

	gender	age	no_of_days_subscribed	multi_screen	mail_subscribed	weekly_mins_watched	minimum_daily_mins	maximum_daily_mins	weekly_m
0	Female	36	62	no	no	148.35	12.2	16.81	
1	Female	39	149	no	no	294.45	7.7	33.37	
2	Female	65	126	no	no	87.30	11.9	9.89	
3	Female	24	131	no	yes	321.30	9.5	36.41	
4	Female	40	191	no	no	243.00	10.9	27.54	

Aunque en la Tabla 3 toda la información parece estar en orden, es común que las bases de datos contengan valores nulos o vacíos, lo que puede suponer un problema a la hora de realizar análisis, dado que estos sesgan la información. La manera más rápida y sencilla de obtener qué observaciones están vacías es mediante el comando '.*isnull*()'. Con esta ejecución el cuaderno de trabajo muestra que existen tres variables con valores nulos, y estas son *gender*, *maximum_days_inactive* y *churn*. En la variable del género faltan un total de 24 observaciones, en el número de días inactivo faltan 28 y en *churn* 35.

El tratamiento de estos valores es crucial en el resultado final de este estudio dado que según se traten afectará a la precisión del modelo de una manera u otra. Estas tres

variables con valores nulos son factores importantes en este modelo, además de *churn*, que es la variable objetivo. Es por esto por lo que el tratamiento de estos datos ha sido detalladamente analizado y cuidado. En primer lugar, se mostraron los índices que ocupan estos clientes en el *dataset* en las tres variables, para ver si alguno compartía que tenía valores nulos, y buscar algún patrón. Finalmente se observó que ninguno coincidía, por lo que no existe ningún patrón aparente.

El número de observaciones que suelen tener los análisis de esta índole, suelen superar las 10.000. Por lo tanto, este caso no es de gran volumen, ya que contiene 2.000. Esto significa que un valor vacío tiene más peso en esta base de datos, por lo que hay que prevenir perder a los suscriptores que tienen valores nulos.

Es por esto por lo que la decisión ha sido imputar los datos en vez de eliminarlos directamente. Al imputarlos, se asegura que no se reduce el número observaciones, por lo que la información sigue siendo relevante. Existen multitud de opciones por las que sustituir esos valores vacíos, pero en este trabajo se ha optado la media y la moda. La primera opción se suele utilizar en datos de categoría numérica, y la moda es la preferida para datos categóricos, dado que no existe un término directo de media para este tipo.

Por lo tanto, los valores de la variable género se han sustituido por la moda, es decir, masculino. Por el contrario, los 28 valores nulos de la variable *maximum_days_inactive* han sido imputados con la media, ya que esta es de tipo numérico.

Por último, la decisión del tratamiento de los valores vacíos de la variable *churn* es la más crítica. La decisión del tratamiento de estos datos fue la de imputarlos con la moda. *Churn* es una variable numérica, pero de tipo binaria, es decir con valor (0,1), por lo que no se puede sustituir por la media. Si se hiciese, los valores quedarían sesgados, por lo que la decisión más apropiada fue imputarlos con la moda. Una vez hecho esto, se vuelve a comprobar que los datos han sido imputados correctamente, y que no se ha perdido ninguna observación.

A continuación, se muestra en la Tabla 4 una descripción de los datos depurados. Como se puede observar, se han eliminado del conjunto las variables de *year, customer_id* y *phone_no,* y todas tienen 2.000 observaciones lo que significa que no se ha perdido ningún dato, y que se puede proceder al análisis.

Tabla 4. Descripción de los Datos Limpios

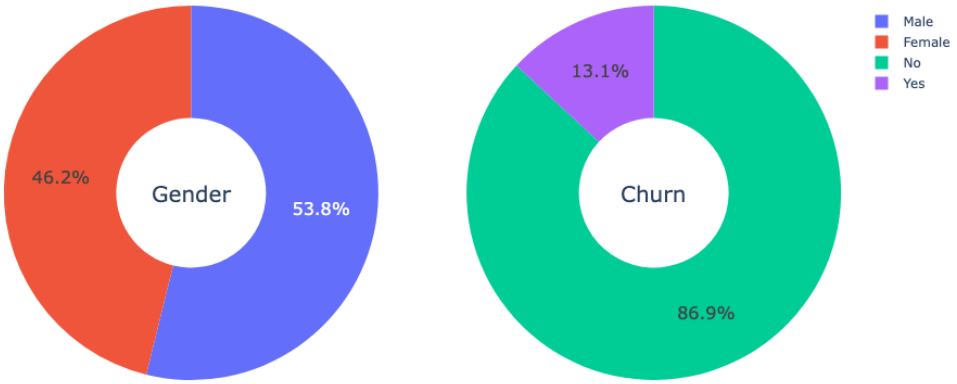
```
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   gender                  2000 non-null   object
 1   age                     2000 non-null   int64
 2   no_of_days_subscribed   2000 non-null   int64
 3   multi_screen            2000 non-null   object
 4   mail_subscribed         2000 non-null   object
 5   weekly_mins_watched     2000 non-null   float64
 6   minimum_daily_mins      2000 non-null   float64
 7   maximum_daily_mins      2000 non-null   float64
 8   weekly_max_night_mins   2000 non-null   int64
 9   videos_watched          2000 non-null   int64
 10  maximum_days_inactive   2000 non-null   float64
 11  customer_support_calls  2000 non-null   int64
 12  churn                   2000 non-null   float64
dtypes: float64(5), int64(5), object(3)
```

Posteriormente, se procede a un análisis y distribución de las variables. Primero se analiza la variable género y después el *churn* de los clientes. Esto se muestra en la Figura 2, lo que permite tener una visualización sencilla y clara de cómo están compuestas estas dos variables.

En la distribución de estas variables, se concluye que, del total de suscriptores, un 53,8% son hombres, y un 46,2% son mujeres. Esto permite a la plataforma OTT conocer mejor las características demográficas de sus clientes. La edad promedio de estos es de 39 años.

Por otro lado, la variable objetivo *churn*, muestra en su distribución que hay un total de 13,1% de suscriptores que se dieron de baja en 2.020, y que el 86,9% se quedaron. Según muestra la leyenda de la Figura 2, *male* representa el género masculino, *female* el género femenino, *no* los clientes que mantienen la suscripción y *yes* aquellos que sí se dan de baja.

Figura 2. Distribuciones de Género y *Churn*



3.2. Técnicas para el Análisis

Como se ha mencionado anteriormente, los modelos que se utilicen para el estudio dependerán siempre de los objetivos que se tengan. Para este caso, se ha considerado oportuno la utilización de los algoritmos que se mencionan a continuación. En primer lugar, se ha hecho uso de KNN o k vecinos más cercanos. Según IBM:

> El algoritmo de k vecinos más cercanos, también conocido como KNN o k-NN, es un clasificador de aprendizaje supervisado no paramétrico, que utiliza la proximidad para hacer clasificaciones o predicciones sobre la agrupación de un punto de datos individual. Si bien se puede usar para problemas de regresión o clasificación, generalmente se usa como un algoritmo de clasificación, partiendo de la suposición de que se pueden encontrar puntos similares cerca uno del otro. (*¿Qué Es El Algoritmo De K Vecinos Más Cercanos? | IBM*, n.f.)

En segundo lugar, se ha utilizado *Support Vector Classifier* (SVC) o *Support Vector Machine*. Es un algoritmo de aprendizaje supervisado que se utiliza en modelos de clasificación y regresión. Su objetivo principal es hallar la manera óptima de discernir entre distintos tipos de datos.

El uso de árboles de decisión también se ha implementado para este análisis. Es un modelo jerárquico que se utiliza como apoyo en la toma de decisiones y en sus consecuencias, en el que se representa una estructura parecida a un árbol en la que cada nodo es una característica distinta del conjunto de datos.

A continuación, se ha utilizado el modelo de clasificación Random Forest. Este modelo combina varios árboles de decisión, de tal modo que cada árbol depende de un vector aleatorio que se prueba de manera independiente. También, se ha usado un modelo de regresión logística, el más clásico para predecir. Este tipo predice un resultado de una variable categórica en función de las que se establezcan como predictoras.

Por último, se ha implementado un modelo de análisis de supervivencia. Esta técnica se utiliza cuando la variable de interés es el tiempo que existe entre el inicio de un evento hasta que éste ocurre. En este caso, se ha establecido como variable de tiempo el número de días que pasa el usuario inactivo, y como evento, el *churn*.

El uso de estos modelos en lugar de otros es simple. Cada uno de ellos aporta algo diferente, es decir, cada uno tiene fortalezas que los demás no. KNN es un modelo muy útil para identificar patrones en los datos no lineales, y en el caso de la industria OTT donde la retención de clientes es muy versátil, sirve para identificar rápidamente grupos con características similares, como usuarios que comparten ciertas preferencias en contenido o características demográficas y cuáles de éstas tienen impacto directo en el *churn.* Comprender e identificarlos correctamente es fundamental para entender los comportamientos de los usuarios y poder establecer estrategias personalizadas.

Por otro lado, SVC demuestra ser más eficiente en la separación de datos basada en las características, especialmente en aquellos datos más complejos. En el caso de las OTT, ayuda a establecer un límite claro entre los consumidores que abandonan y los que no, por lo que la predicción se hace mucho más ajustada. Para el caso del árbol de decisión y Random Forest, la interpretación de los datos es más intuitiva, por lo que las características por las que se define cada conjunto de usuarios son mucho más visuales. Esto facilita el proceso de identificación de patrones en la conducta de los clientes, y permite tomar mejores decisiones en cuanto a personalización.

Por último, el análisis de supervivencia muestra también de manera más visual las tendencias de los clientes. Además, es común incluir en el gráfico de dicha tendencia el tiempo que los usuarios permanecen suscritos, el cual suele medirse con la mediana. La mediana representa el valor que ocurre al cumplirse la mitad del tiempo que se ha establecido como variable.

Para concluir el análisis de estos modelos, se ha generado la curva ROC (*Receiver Operating Characteristic*) de cada uno de ellos. Esta curva se utiliza para conocer la capacidad discriminativa de cada modelo, lo que evalúa el rendimiento que tienen para clasificar. Al combinar los resultados en un único gráfico, se ofrece una visión global de todos en términos de eficacia. Esta visualización se mostrará más adelante, tras haber explicado detalladamente el procedimiento.

3.3. Procesamiento de Datos

El siguiente paso es el procesamiento de los datos. Los algoritmos de aprendizaje automático utilizados requieren que los datos sean numéricos, no categóricos. Para ello,

hay que convertir los categóricos del *dataset* en enteros, lo que se consigue con el siguiente código mostrado en la Figura 3:

Figura 3. Código Empleado

```
def object_to_int(dataframe_series):
    if dataframe_series.dtype=='object':
        dataframe_series = LabelEncoder().fit_transform(dataframe_series)
    return dataframe_series
```

Una vez hecho esto, se desea conocer la correlación que tiene cada variable con *churn*, dado que es la que se va a establecer como objetivo. Como se muestra en la Tabla 5, en cada fila se encuentra la correlación que tienen cada una de ellas con respecto a la variable *churn,* es por eso por lo que ésta tiene valor de 1. Es decir, como es el caso de *multi_screen* o *customer_support_calls* son las que tienen un índice de correlación más alto.

Esta correlación representa cómo cambian cada una de las variables y con cuánta fuerza, en relación lineal con lo que varíe *churn.* Cuanto más cercano a 1, más fuerte y positiva será la relación, y viceversa. En caso de ser cercano a 0, esto indica una correlación débil o inexistente.

Tabla 5. Correlaciones

churn	1.000000
multi_screen	0.258324
customer_support_calls	0.204774
weekly_mins_watched	0.162876
maximum_daily_mins	0.162874
minimum_daily_mins	0.066646
maximum_days_inactive	0.046348
age	0.011777
weekly_max_night_mins	0.006917
no_of_days_subscribed	0.002517
gender	-0.015122
videos_watched	-0.019314
mail_subscribed	-0.077705

Posteriormente, se define la variable *churn* como objetivo, creando dos conjuntos de datos diferentes, como se muestra en la Figura 4: la variable 'X' contiene todas las variables independientes, mientras que la 'Y' contiene únicamente la objetivo.

Figura 4. Código Asignación de Variables

```
x = datos.drop(columns = ['churn'])
y = datos['churn'].values
```

```
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.30, random_state = 40, stratify=y)
```

También se muestra en la Figura 4 cómo el conjunto de datos se divide en dos partes, un conjunto de entrenamiento y otro de prueba, o más conocido como *test* y *train set*. La partición de los datos que se hace en modelos de predicción y clasificación sirven para evaluar la capacidad que tiene un modelo para generalizar, es decir, la capacidad para predecir sobre datos que no se ven. Al separar los datos en un conjunto de entramiento y otro de prueba, se proporciona a los modelos una posibilidad de aprender relaciones entre las variables, y una vez ha aprendido, se evalúa el conjunto de prueba con esos datos que ha ido probando el conjunto de entrenamiento. El porcentaje de datos que se asignan a cada parte es importante para evitar el sobreajuste, es decir, para impedir que el modelo no sea capaz de generalizar datos nuevos.

Como muestra la tercera línea de la Figura 4, se ha hecho una partición de 70-30, es decir, un 30% de los datos se asignan a prueba, y un 70% a entrenamiento. Aunque la división más habitual suele ser 60-40, tras realizar numerosas pruebas se obtuvieron resultados más favorables separando los conjuntos en 70-30.

Además, es necesario establecer un valor que sirva de semilla para los datos. Dado que la división de las observaciones se hace completamente de manera aleatoria, con la fijación de la semilla se consigue controlar que se obtenga siempre el mismo resultado todas las veces que se ejecute el comando. El valor que se asigne es a elección del autor del proyecto. En la Tabla 4, se muestra que la semilla se ha fijado en 40 mediante el comando '*random_state* = 40*'*. De nuevo, este valor ha sido escogido debido a que con él se han obtenido los resultados óptimos del análisis probado.

4. ANÁLISIS DE RESULTADOS EN LA PÉRDIDA Y RETENCIÓN DE CLIENTES EN LA INDUSTRIA OTT

4.1. Análisis de resultados

En este TFG se emplean distintas técnicas de clasificación y predicción para obtener una visión global de la tasa de abandono de clientes en la industria OTT. Estos modelos se emplean a modo de maximizar la capacidad de predicción del análisis y poder obtener así información complementaria acerca de todos los rasgos que puedan influir en el estudio. El objetivo de utilizar cada modelo no es más que el de obtener una comprensión adicional desde distintos puntos de vista, y deben ser estudiados de manera conjunta.

En este apartado del trabajo, se procede a hacer un análisis de los resultados que se han obtenido con los modelos. Para cada uno, se ha representado a modo de tabla, un informe que lo explica. Las partes que componen este detalle son varias, y todas sirven para profundizar y tener una visión mejor. Sin embargo, para este análisis basta con fijarse en algunas de ellas.

Las partes que lo componen son: *precision, recall, f1-score,* y *support.* La primera corresponde a la capacidad del modelo para clasificar correctamente los datos que no pertenecen a una clase específica, por lo que un valor alto supondría una precisión a la hora de evitar clasificar de manera incorrecta. La segunda hace referencia a la sensibilidad que tiene el modelo a la hora de identificar los datos, por lo que un valor alto indica que el modelo los detecta de manera correcta. El *f1-score* combina la precisión con el *recall* mientras que *support* especifica el número de observaciones que pertenecen a cada clase. En el caso del último componente, se muestra que por cada modelo se hace uso de 600 observaciones, mientras que el total es de 2.000. Esto se debe a la partición que se hizo en el apartado anterior del 30%. Esto quiere decir que los modelos utilizan ese 30%, es decir, 600 de las 2.000 observaciones, para realizar las pruebas.

En cada algoritmo, las tablas representadas tendrán la misma estructura. Se observa en modo de filas un 1 y un 0, lo que representa las clases que se mencionaban anteriormente, lo que para este estudio concreto refleja los clientes que tienen un 1 o un 0 en el campo de *churn*.

En la parte inferior de los reportes se verán tres filas adicionales: *accuracy, macro avg* y *weighted avg.* La primera se refiere a la precisión de cada componente, y las otras dos al promedio macro y ponderado, respectivamente. La diferencia entre el macro y el ponderado es que el primero no toma en cuenta el tamaño de cada clase, y el ponderado sí.

En la explicación de cada modelo, se ha añadido el valor que tiene su AUC-ROC, el área bajo la curva ROC, la cual ha sido explicada anteriormente. Esta métrica es útil para conocer y evaluar el rendimiento del modelo, ya que representa la probabilidad de que el correspondiente modelo clasifique de manera correcta. Un valor cercano a uno representa perfección a la hora de discernir, y un valor cercano a 0,5 que el modelo no es capaz de clasificar.

En conclusión, toda la información que aparece en este informe es importante para el estudio de este TFG. Sin embargo, los factores que más se han tenido en cuenta son la precisión del modelo y su respectivo AUC-ROC.

Como muestra la Tabla 6, el primer modelo empleado es KNN. Éste ha obtenido una precisión del 87%, y un valor de 0,53 para su AUC-ROC. En conclusión, aunque el acierto de KNN es bueno, el área bajo la curva ROC no representa buena clasificación, por lo que sería más oportuno continuar probando otros algoritmos más adecuados para este problema en particular, y que es recomendable analizar otros modelos que mejoren la clasificación.

Tabla 6. Informe Clasificación KNN

KNN accuracy: 0.8683333333333333

	precision	recall	f1-score	support
0	0.87	0.99	0.93	521
1	0.50	0.06	0.11	79
accuracy			0.87	600
macro avg	0.69	0.53	0.52	600
weighted avg	0.83	0.87	0.82	600

La Tabla 7 muestra los datos obtenidos para el modelo *Support Vector Classifier* (SVC). Este modelo obtuvo valores parecidos, con una precisión del 87%. Sin embargo, el valor de la AUC-ROC es algo inferior, siendo 0,5, lo que concluye que el valor de exactitud de clasificación es igual de válida que para KNN, es decir, el área bajo la curva ROC es prácticamente inservible.

Tabla 7. Informe Clasificación SVC

SVM accuracy is: 0.8683333333333333

	precision	recall	f1-score	support
0	0.87	1.00	0.93	521
1	0.00	0.00	0.00	79
accuracy			0.87	600
macro avg	0.43	0.50	0.46	600
weighted avg	0.75	0.87	0.81	600

En la Tabla 8 se muestran los datos relativos al árbol de decisión. De nuevo, se ha obtenido una precisión en el modelo de 87%. A diferencia de los anteriores, éste ha obtenido un área bajo la curva ROC de 0,74, el mejor valor hasta ahora. El resultado sugiere que el árbol de decisión tiene mejor capacidad de discriminación que los

anteriores, por lo que esto implica una buena determinación en la clasificación de los datos, lo que hace que sea el modelo más recomendable hasta el momento.

Tabla 8. Informe Clasificación Árbol Decisión

Decision Tree accuracy is : 0.8683333333333333

	precision	recall	f1-score	support
0	0.93	0.92	0.92	521
1	0.50	0.56	0.53	79
accuracy			0.87	600
macro avg	0.72	0.74	0.73	600
weighted avg	0.87	0.87	0.87	600

Para una mejor representación, y un entendimiento más profundo del modelo de árbol de decisión, se muestra un ejemplo de cómo clasificaría las observaciones en el apartado 7.15 de los anexos. Los árboles de decisión suelen utilizarse mediante dos criterios diferentes: entropía y Gini. Ambas medidas se emplean para determinar la impureza con la que el árbol divide los datos en cada uno de los nodos o niveles. En la práctica, ambas medidas suelen dar resultados parecidos, pero siempre se debe escoger en función de las características de los datos, y de las preferencias que se tengan. Se ha establecido que el árbol utilice el criterio de Gini para medir la calidad de la división, el cual viene establecido por defecto. La elección de este criterio se debe a las pruebas que se realizaron con ambos criterios, en las que Gini obtuvo los mejores resultados.

Para el criterio escogido, cuanto mayor sea el valor de la impureza, la probabilidad de que ese elemento escogido de manera aleatoria esté mal clasificado, es mayor. Además, se ha establecido una profundidad máxima del árbol de 3 nodos, con el único fin de que sea legible y entendible. Este gráfico se encuentra en el apartado 7.15 de los anexos, y de él se pueden concluir varios factores. En primer lugar, que el número de llamadas que el cliente realiza a atención al cliente puede indicar insatisfacción con el servicio o una mala experiencia. En segundo lugar, que los que no están suscritos al correo electrónico de la empresa son más propensos a darse de baja.

Por último, que tanto el tiempo mínimo como máximo que los usuarios emplean en utilizar la plataforma, es crucial en el interés que tienen por continuar utilizándola. Sin embargo, cabe destacar que hay factores no incluidos en el árbol que son importantes, como por ejemplo el precio, y que se deben tener en cuenta.

El modelo Random Forest ha logrado obtener una excelente precisión del 93%, lo que supone el valor obtenido más alto de todos. Además, el área bajo la curva ROC es de 0,78, lo que implica ser el modelo con mayor efectividad de todos, debido a su alta capacidad predictiva. Se observan estos datos en la Tabla 9.

Tabla 9. Informe Clasificación Random Forest

0.9283333333333333

	precision	recall	f1-score	support
0	0.94	0.98	0.96	521
1	0.82	0.58	0.68	79
accuracy			0.93	600
macro avg	0.88	0.78	0.82	600
weighted avg	0.92	0.93	0.92	600

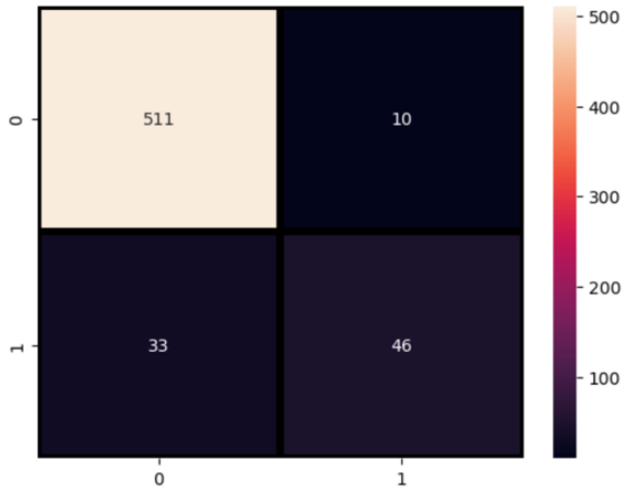
Para mostrar información adicional, se cumplimentará más adelante una matriz de valores con los falsos positivos y verdaderos positivos que el modelo Random Forest es capaz de predecir. Se ilustran cuantas observaciones el modelo es capaz de clasificar correcta e incorrectamente.

La matriz de confusión es una herramienta muy útil para obtener mayor contexto del modelo, ya que muestra de manera visual su rendimiento, y cómo éste ajusta y depura los datos. Suelen estar estructuradas en forma cuadrada, en la que en las filas de dicha matriz se encuentran las clases reales de los datos, mientras que en las columnas están las que predice el modelo en cuestión.

Por lo tanto, en este caso las filas y columnas representadas con 0 y 1, corresponden a darse o no de baja en la suscripción de la plataforma OTT, respectivamente. A modo de leyenda, se muestra en la parte derecha de la matriz una barra que indica la cantidad de observaciones que se clasifican dentro de cada sección.

Como muestra la Figura 5, 46 observaciones fueron clasificadas de manera correcta como verdaderos negativos. Además, el modelo acertó en la clasificación de 511 observaciones como verdaderos positivos. En conclusión, Random Forest es capaz de clasificar correctamente 557 observaciones de las 600 que trata en las pruebas, en su correspondiente clase. Esta tasa corresponde al 93% de precisión mencionada en el reporte de clasificación de la Tabla 9, lo que demuestra la buena capacidad del modelo en la predicción, lo que aporta información adicional.

Figura 5. Matriz de Confusión Random Forest



Por último, como presenta la Tabla 10 se realizó una regresión logística. Este modelo ha obtenido una precisión del 87%, lo que indica una capacidad igual de válida que los primeros modelos. Sin embargo, el valor para el área bajo la curva ROC es de 0,56, por lo que se concluye una competencia para discriminar muy limitada.

Tabla 10. Informe Clasificación Regresión Logística

Logistic Regression accuracy is : 0.865

	precision	recall	f1-score	support
0	0.88	0.98	0.93	521
1	0.46	0.14	0.21	79
accuracy			0.86	600
macro avg	0.67	0.56	0.57	600
weighted avg	0.83	0.86	0.83	600

Para una correcta interpretación de la curva ROC, se debe comprender que cada punto de la curva representa el rendimiento del modelo. Cuanto más se acerque la curva a la esquina superior izquierda en la gráfica, el rendimiento del modelo será mejor, puesto que esto indica que hay una tasa alta en la clasificación de verdaderos positivos y una tasa baja en falsos positivos.

Sin embargo, cuanto más cerca se encuentre a la curva de la diagonal que hay trazada en la gráfica, el modelo será menos discriminativo, por lo que no estará mejorando la clasificación, y sus predicciones no serán tan fiables.

A diferencia de otras métricas como la precisión, el área bajo la curva ROC tiene como ventaja que no está influenciada por el desequilibrio de clases en el conjunto de datos, es

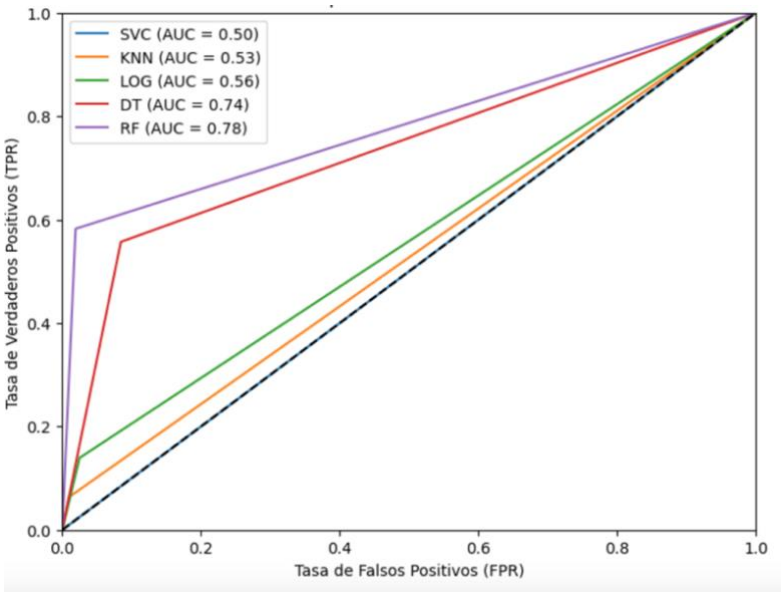
decir, que haya una clase mucho más frecuente que otra; ni por la elección que se hace en el valor del umbral de decisión, por lo que también es una herramienta válida en este estudio.

Hay que recordar que clases, en este caso, corresponde a darse de baja o no del servicio. Como se ha mencionado anteriormente, existe un desequilibrio dado que hay un 13,1% de cancelación de suscripciones frente a un 86,9% de continuidad, por lo que esta métrica es útil en este caso.

En la Figura 6 se encuentra la información agregada de todos los modelos utilizados, ordenados de menor a mayor por el área, como se aprecia en la leyenda. Según aparece, SVC representa el *Support Vector Classifier,* KNN muestra el algoritmo de k vecinos más cercanos, LOG representa el modelo de regresión logística, DT representa el árbol de decisión y RF Random Forest, siendo este último el más optimo.

En el eje horizontal, se recoge la Tasa de Falsos Positivos, y en el eje vertical la Tasa de Verdaderos Positivos, es decir, las observaciones que son negativos y se clasifican como positivos, y los que verdaderamente lo son y así se clasifican, respectivamente.

Figura 6. Curvas ROC para los Modelos de Clasificación



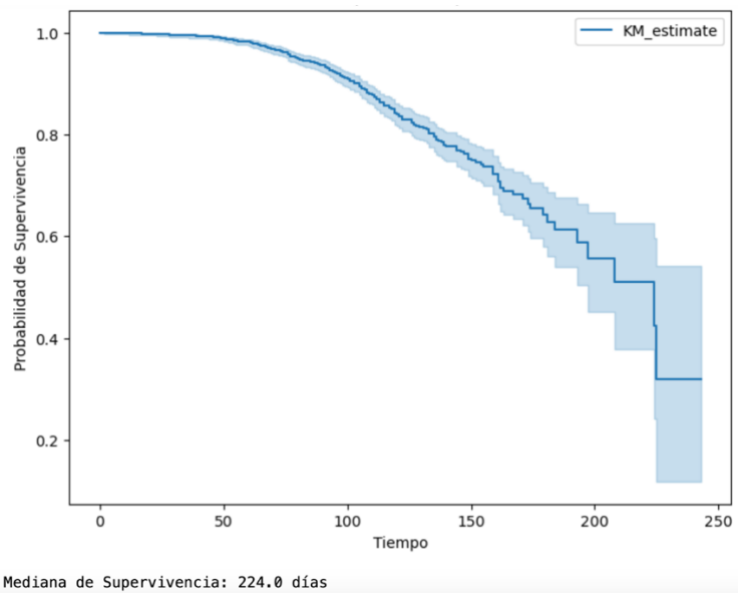
Por último, pero no menos importante, se realiza el análisis de supervivencia de Kaplan-Meier. Tal y como lo define IBM:

> El modelo de Kaplan-Meier se basa en la estimación de las probabilidades condicionales en cada punto temporal cuando tiene lugar un evento y en tomar el límite del producto de esas probabilidades para estimar la tasa de supervivencia en cada punto temporal (*IBM Documentation*, 2023).

Para aplicar este modelo se ha utilizado el paquete de Python *KaplanMeierFitter* de la librería *lifelines*. Dicho paquete, está especialmente diseñado para un análisis de supervivencia como el de este TFG. Para realizarlo, se debe establecer una variable de tiempo, y otra como factor de supervivencia.

Por un lado, la variable *no_of_days_subscribed* ha sido fijada como el tiempo que transcurre hasta que ocurre el evento, y por otro, *churn* se ha establecido como el factor de supervivencia o no en el modelo, es decir, de mantenerse suscrito. El gráfico correspondiente al análisis de supervivencia se encuentra representado en la Figura 7.

Figura 7. Curva Kaplan-Meier para Variable *Churn*



En el eje de abscisas, la gráfica muestra el tiempo correspondiente a los días que el consumidor está suscrito a la plataforma OTT a lo largo del año 2.020. En el eje de ordenadas, se muestra la probabilidad aproximada de supervivencia de cada usuario, siendo el valor cercano a 1 una probabilidad asegurada de sobrevivir, lo que termina mostrando la expectativa de que un cliente cancele la suscripción en función del tiempo que lleve como consumidor.

Adicionalmente, se ha agregado el valor de la mediana de supervivencia localizado en la esquina inferior izquierda de la Figura 7, que en este caso es 224 días. La mediana es un dato estadístico que se utiliza para conocer el punto en el que un conjunto de datos se divide en dos partes iguales, es decir, el punto medio. En este caso, se utiliza para conocer el tiempo en el que la mitad de las personas se han dado de baja del servicio. Visualmente, sería el punto de la gráfica en el que la curva de Kaplan-Meier cruza con la línea trazada horizontalmente en el 50% de probabilidad del eje de abscisas. Esta medida es útil para conocer rápidamente cuándo ocurre el evento.

En conclusión, este análisis es útil para conocer rápidamente información sobre los clientes de plataformas OTT. Para este caso, la mitad de los clientes se han dado de baja antes de los 224 días lo que indica que a medida que pasa el tiempo, la probabilidad de que los usuarios se den de baja aumenta. Cabe destacar que este análisis no identifica todos los factores y que pueden afectar al resultado, como es el caso del precio del servicio o de la competencia. Por lo tanto, es útil en este estudio para conocer mejor el comportamiento de los clientes, pero complementado como una herramienta más.

4.2. Contraste de hipótesis

Es importante destacar que el conjunto de metodologías avanzadas empleadas en este TFG sirve para aportar información de valor en la hipótesis de que son útiles para un estudio más avanzado. Sin embargo, no se termina de demostrar con certeza que los resultados obtenidos son fiables, ya que en ocasiones estos valores se han podido obtener por casualidad o azar.

Por lo tanto, es crucial argumentar las conclusiones obtenidas mediante un contraste de hipótesis. Se trata de una técnica estadística empleada comúnmente en empresas en la toma de decisiones basadas en datos. Consiste en formular dos hipótesis: una nula (H0) y otra alternativa (H1). La primera hace referencia a la afirmación que se va a probar en el contraste, y normalmente la que no genera efecto en el estudio; y la segunda es la que se quiere demostrar.

Para el caso de este TFG, se establece como hipótesis nula que los modelos empleados no superan significativamente la regla de la mayoría, y como alternativa que al menos uno de ellos sí que vence de manera relevante esta norma. Ésta se emplea normalmente para modelos de clasificación, y su funcionamiento es sencillo. Normalmente, la regla de

la mayoría es usada en el contexto de la clasificación de datos, y sirve para predecir la clase más común en el conjunto de entrenamiento para todos los casos del conjunto de prueba. Es decir, que se evalúa la distribución de clases que hay en el *train set* y se establece la que tenga mayor frecuencia de las dos.

Por lo tanto, suponiendo que en este caso el 60% de los casos en el conjunto de entrenamiento forman parte de la clase "*churn: Yes"* y el 40% restante a "*churn: No*", la regla de la mayoría predeciría la categoría mayoritaria para todos los casos del *test set*.

En conclusión, H0 establece que las precisiones de los modelos empleados en este análisis son iguales o menores a la precisión que se espera de la regla de la mayoría. Por otro lado, que H1 demuestra que al menos uno de los modelos tiene mayor precisión que la supuesta con esta ley.

Para el estudio del rendimiento de los modelos se pueden emplear diferentes métricas. En primer lugar, y la más común, la precisión. Es útil utilizarla en casos donde la distribución de clases está balanceada. En aquellas situaciones donde se quiera clasificar una variable objetivo de tipo binaria también es común emplear la precisión, la sensibilidad o la especificidad. Sin embargo, como se mostró anteriormente, *churn* tiene una distribución descompensada en los usuarios que sí se dan de baja y los que no, con un 13,1% y 86,9%, respectivamente.

Por lo tanto, en un caso como este lo mejor es utilizar una medida que ajuste de manera acertada esta descompensación. Una medida que cumple este objetivo es AUC-ROC, o área bajo la curva ROC. Con este parámetro, se aplicó la técnica de validación cruzada, la cual consiste en dividir los datos en subconjuntos llamados *folds* o pliegues para entrenarlos, para finalmente hacer un promedio de los resultados. El número que se establezca depende del tamaño del conjunto de datos y del balance de clases. Para este TFG se consideró oportuno determinar 10 *folds* para la validación cruzada, ya que es un valor común para un estudio de características similares a este. La principal ventaja del *cross-validation* es paliar el sobreajuste y una estimación más precisa.

Por último, se usó la prueba t de Student. Ésta es útil en el contexto de los contrastes de hipótesis, por lo que es una herramienta fundamental e imprescindible. Esta es la prueba que demuestra la evidencia mencionada anteriormente de que algún modelo supera significativamente la regla de la mayoría. El contraste se realiza mediante la comparación

de la ‘p’ obtenida, el cual se define como la probabilidad de que un valor estadístico calculado sea posible dada H0 cierta. El valor p utilizado de referencia comúnmente es 0,05. Por lo tanto, si el que se obtiene en el contraste de hipótesis es menor, se rechaza la hipótesis nula y se concluye que al menos uno de los modelos supera la regla de la mayoría.

En conclusión, el contraste realizado obtuvo un p-valor de 0,003, lo que indica una evidencia fuerte en contra de H0. Esto quiere decir que al menos uno de los modelos supera significativamente la regla de la mayoría. Los datos de la Figura 8 muestran los AUC-ROC promedio de cada modelo según el *cross-validation.* El modelo con mayor promedio es Random Forest, lo que corrobora el resultado de la superioridad de este sobre los demás algoritmos obtenido en el apartado anterior.

Figura 8. Promedios AUC-ROC de los modelos

```
AUC-ROC medio del modelo KNeighborsClassifier: 64.27%
AUC-ROC medio del modelo RandomForestClassifier: 85.23%
AUC-ROC medio del modelo DecisionTreeClassifier: 72.21%
AUC-ROC medio del modelo SVC: 68.28%
AUC-ROC medio del modelo LogisticRegression: 73.79%
El contraste de proporciones muestra que al menos uno de los modelos supera significativamente la regla de la mayor
ía.
```

El código completo utilizado en el contraste de hipótesis se encuentra en el apartado 7.16 de los anexos. Es importante destacar que en este análisis se han analizado los modelos KNN, Random Forest, árbol de decisión, SVC y regresión logística. Es decir, se ha excluido el análisis de supervivencia, debido a que este es el único incompatible con la validación cruzada.

El objetivo fundamental de este contraste es el de proporcionar evidencia estadística de que los modelos empleados en este TFG superan en rendimiento a una regla de la mayoría. Por lo tanto, esta comprobación ha sido fundamental en la validación de la utilidad y fiabilidad de los algoritmos, debido al resultado positivo obtenido que demuestra que pueden utilizarse en el estudio de *churn* de clientes en plataformas OTT.

5. CONCLUSIONES

La importancia de la industria de las plataformas y servicios de *streaming* continúa creciendo a diario en el mercado. Este sector ha abierto muchas puertas a millones de personas, tanto a nivel personal por entretenimiento, como profesional. Desde que

comenzó su crecimiento, este sector ha generado millones de puestos de trabajo, y ha desencadenado la creación de nuevas empresas en todo el mundo, además de atraer la atención de grandes inversores, lo que ha supuesto un aumento en el capital que controla.

Dentro de este sector, existen diferentes subsectores. Cada uno de éstos es completamente diferente a los demás, por lo que sus respectivas estrategias de mercado han de ser diferentes. En la preparación de este Trabajo de Fin de Grado, se realizaron búsquedas de estudios similares a este en repositorios científicos y de confianza, como EBSCO, Dialnet y Google Académico. El hallazgo fue bastante limitado dado que únicamente se encontraron dos que tuviesen los mismos objetivos, siendo uno de ellos muy genérico y poco extenso.

En primer lugar, es importante reconocer que el estudio tiene ciertas limitaciones. Por un lado, la base de datos con la que se ha trabajado no dispone de gran cantidad de observaciones ni variables. Esto hizo que el análisis fuese más riguroso a la hora de tratar la información, dado que cualquier variación alteraría la relevancia y el peso de cada dato.

Otra gran limitación, es que la información con la que se ha trabajado es correspondiente al año 2.020, y no se conoce a qué plataforma pertenece exactamente. Lo ideal hubiera sido disponer de datos históricos, y haber podido realizar un análisis más completo.

Por otro lado, se utilizó este conjunto debido a la escasez de *datasets* relacionados con la industria en los repositorios. Existen otras bases de datos relevantes, aunque con la información poco actualizada. Además, éstas no contenían la información necesaria para este TFG, sino que únicamente disponían de valores únicos respectivos a audiencia según la plataforma. Por lo tanto, es concluyente que la que se utilizó era la que mejor cumplía los requisitos de este análisis.

Cabe destacar que no se encontraron más que dos estudios similares a éste. Esto quiere decir que este análisis puede servir a futuras investigaciones para realizar exploraciones más profundas, ya que no existen exploraciones más extensas que los mencionadas. Dado el caso, se recomienda examinar nuevos modelos de aprendizaje automático, como por ejemplo series temporales, ya que cumplen una función similar al análisis de supervivencia. Para los fines que quería cumplir este Trabajo de Fin de Grado, los modelos utilizados cumplieron la función. Sin embargo, para análisis más profundos sería más conveniente la utilización de modelos predictivos más complejos.

Además, sería interesante la obtención de datos adicionales, como por ejemplo la satisfacción de cada cliente con el servicio que le ofrece la plataforma, para poder obtener más información relativa al por qué un cliente se queda, o se va.

Durante el TFG, se ha demostrado que con la utilización de los seis modelos empleados: KNN, SVC, árbol de decisión, Random Forest, regresión logística y análisis de supervivencia; se pueden realizar estudios relacionados con los clientes de cualquier empresa. El objetivo de este trabajo era analizar la retención de clientes que poseen empresas en la industria OTT. A raíz del análisis, también se pretende que se utilice como referencia para cualquier sector o empresa.

Se concluye que el modelo que ha obtenido los datos óptimos para analizar el *churn* de sus clientes es Random Forest, con una precisión del 93%, y un área bajo la curva ROC de 0,78. Además, como se demostró en el apartado de contraste de hipótesis, se concluye que estos modelos son útiles para realizar estudios de la rotación y pérdida de clientes en plataformas de la industria OTT. El código de Python utilizado para el TFG se aporta al final del documento en los anexos.

6. BIBLIOGRAFÍA

Capgemini, Sande, F. V., Belarbi, M., & Falkenberg, A.-K. (s/f). OTT Streaming Wars: Raise or Fold; How Data is Reshuffling the Cards of the M&E Industry.

Caracterización del uso de algunos servicios *Over The Top* en España (Comunicaciones electrónicas y servicios audiovisuales). (2015). En *Cnmc* (DTRAB/DP/0004/14). Comisión Nacional de los Mercados y la Competencia. https://www.cnmc.es/sites/default/files/1533234.pdf

Churn modeling for OTT platforms. (2023, 24 febrero). https://www.kaggle.com/datasets/santhoshvr97/ott-chrun-modeling-ott

Colaboradores de Wikipedia. (2024, 12 febrero). *Streaming*. Wikipedia, La Enciclopedia Libre. https://es.wikipedia.org/wiki/Streaming

Curry, D. (2020, 16 noviembre). Video streaming app revenue and usage statistics (2023). Business of Apps. https://www.businessofapps.com/data/video-streaming-app-market/

Customer churn prediction. (2021, 29 junio). Kaggle.com; Kaggle. https://www.kaggle.com/code/bhartiprasad17/customer-churn-prediction

Disney Plus Revenue and Usage Statistics (2024) - Business of Apps. (2024, 1 marzo). Business of Apps. https://www.businessofapps.com/data/disney-plus-statistics/

Devi, O. R., Pothini, S. K., Kumari, M. P., & Charan, U. N. S. (2023, May). Customer Churn Prediction using Machine Learning: Subcription Renewal on OTT Platforms. In *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)* (pp. 1025-1029). IEEE.

IBM documentation. (2023, 4 agosto). https://www.ibm.com/docs/es/spss-statistics/saas?topic=statistics-kaplan-meier-survival-analysis

Logic, G. (2023). The State of OTT 2023 - Technologies & Key Trends Impacting Over-The-Top Media Brands.

Netflix Revenue and Usage Statistics (2024) - Business of Apps. (2024, 7 febrero). Business of Apps. https://www.businessofapps.com/data/netflix-statistics/

Over the Top (OTT) market insights. (s.f.). https://www.mordorintelligence.com/industry-reports/over-the-top-market

¿Qué es el algoritmo de k vecinos más cercanos? | IBM. (s.f.). https://www.ibm.com/es-es/topics/knn

Churn modeling for OTT platforms. (2023, February 24). https://www.kaggle.com/datasets/santhoshvr97/ott-chrun-modeling-ott

Senthil, N. D., "OTT subscriber churn prediction using machine learning" (2023). *Electronic Theses, Projects, and Dissertations*. 1660. https://scholarworks.lib.csusb.edu/etd/1660

Statista. (2023, 11 julio). *Usuarios de vídeo OTT en el mundo 2017-2027*. https://es.statista.com/previsiones/1289336/usuarios-de-video-ott-en-el-mundo

Statista. (s. f.). *OTT Video - Worldwide | Statista Market Forecast*. https://www.statista.com/outlook/amo/media/tv-video/ott-video/worldwide

The State of OTT 2023 - Technologies & Key Trends Impacting Over-The-Top Media Brands. (2023). GlobalLogic. https://www.globallogic.com/wp-content/uploads/2023/03/state-of-ott-1.pdf

Telefónica. (2024, 14 febrero). Historia de Internet: ¿cómo nació y cuál ha sido su evolución? *Telefónica*. https://www.telefonica.com/es/sala-comunicacion/blog/historia-internet-como-nacio-evolucion/

Unidad de Competencia Económica. (2022). Servicios OTT Audiovisuales y de Audio. Org.mx. https://www.ift.org.mx/sites/default/files/estudio_de_servicios_ott.pdf

Wise, J. (2023, 13 noviembre). *How many people watch TV in 2024? (User statistics) - EarthWeb*. EarthWeb. https://earthweb.com/tv-users/

7. ANEXOS

7.1. Carga de librerías y conjunto de datos

En este apartado se muestra la carga de todas las librerías utilizadas, y la carga de la base de datos, así como una muestra de las primeras observaciones y el número total de variables.

```
import pandas as pd
import numpy as np
import missingno as msno
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import warnings
warnings.filterwarnings('ignore')
```

```
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from xgboost import XGBClassifier
from catboost import CatBoostClassifier
from sklearn import metrics
from sklearn.metrics import roc_curve
from sklearn.metrics import recall_score, confusion_matrix, precision_score, f1_score, accuracy_score, classificatio
```

```
datos = pd.read_csv('/Users/inigogemperlesanchezdelcorral/Desktop/ICADE/Quinto/TFG BA/ott_churn_model_dataset.csv')
```

```
datos.head()
```

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen	mail_subscribed	weekly_mins_watched	minimum_daily_mins	maximum_d
0	2020	100198	409-8743	Female	36	62	no	no	148.35	12.2	
1	2020	100643	340-5930	Female	39	149	no	no	294.45	7.7	
2	2020	100756	372-3750	Female	65	126	no	no	87.30	11.9	
3	2020	101595	331-4902	Female	24	131	no	yes	321.30	9.5	
4	2020	101653	351-8398	Female	40	191	no	no	243.00	10.9	

```
datos.shape
```

```
(2000, 16)
```

7.2. Estructura de los datos y extracción de las variables prescindibles

En este apartado se muestran la estructura que tienen los datos, y se eliminan del conjunto de datos las variables *customer_id, pone_no* y *year*, dado que no son necesarias en este análisis.

```
datos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 16 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   year                    2000 non-null   int64  
 1   customer_id             2000 non-null   int64  
 2   phone_no                2000 non-null   object 
 3   gender                  1976 non-null   object 
 4   age                     2000 non-null   int64  
 5   no_of_days_subscribed   2000 non-null   int64  
 6   multi_screen            2000 non-null   object 
 7   mail_subscribed         2000 non-null   object 
 8   weekly_mins_watched     2000 non-null   float64
 9   minimum_daily_mins      2000 non-null   float64
 10  maximum_daily_mins      2000 non-null   float64
 11  weekly_max_night_mins   2000 non-null   int64  
 12  videos_watched          2000 non-null   int64  
 13  maximum_days_inactive   1972 non-null   float64
 14  customer_support_calls  2000 non-null   int64  
 15  churn                   1965 non-null   float64
dtypes: float64(5), int64(7), object(4)
memory usage: 250.1+ KB
```

```
datos.columns.values
```

```
array(['year', 'customer_id', 'phone_no', 'gender', 'age',
       'no_of_days_subscribed', 'multi_screen', 'mail_subscribed',
       'weekly_mins_watched', 'minimum_daily_mins', 'maximum_daily_mins',
       'weekly_max_night_mins', 'videos_watched', 'maximum_days_inactive',
       'customer_support_calls', 'churn'], dtype=object)
```

```
# Quitar la columna customer_id, el número de telefono, y el año (son todos 2020)
# ya que para este estudio no se necesita
datos = datos.drop(['customer_id'], axis = 1)
datos = datos.drop(['phone_no'], axis = 1)
datos = datos.drop(['year'], axis = 1)
datos.head()
```

	gender	age	no_of_days_subscribed	multi_screen	mail_subscribed	weekly_mins_watched	minimum_daily_mins	maximum_daily_mins
0	Female	36	62	no	no	148.35	12.2	16.81
1	Female	39	149	no	no	294.45	7.7	33.37
2	Female	65	126	no	no	87.30	11.9	9.89
3	Female	24	131	no	yes	321.30	9.5	36.41
4	Female	40	191	no	no	243.00	10.9	27.54

7.3. Conteo de valores nulos e imputación de los datos

En este apartado se muestra que variables contienen valores vacíos, y cuantos. Además, se presentan los índices que ocupan esos valores nulos dentro de cada campo. Por último, están las imputaciones realizadas.

```
# Verificar qué columnas tienen valores nulos y contarlos
columnas_con_nulos = datos.columns[datos.isnull().any()]
valores_nulos_por_columna = datos[columnas_con_nulos].isnull().sum()

# Imprimir las columnas con valores nulos y la cantidad de valores nulos en cada una
print("Columnas con valores nulos:")
print(valores_nulos_por_columna)
```

```
Columnas con valores nulos:
gender                  24
maximum_days_inactive    28
churn                   35
dtype: int64
```

```
# Quiero ver lo indices para saber si hay alguno en común
datos[datos['gender'].isnull()].index
```

```
Index([   5,    6,   20,   21,   38,   64,   88,  107,  128,  152,  153,  197,
        198,  244,  245,  294,  345,  393, 1399, 1400, 1424, 1425, 1426, 1997],
      dtype='int64')
```

```
datos[np.isnan(datos['maximum_days_inactive'])].index
```

```
Index([   6,    7,   29,   30,   81,   82,  104,  128,  153,  154,  202,  203,
        204,  236,  237,  282,  283,  334,  385,  386, 1836, 1837, 1859, 1860,
       1861, 1874, 1880, 1998],
      dtype='int64')
```

```
datos[datos['churn'].isnull()].index
```

```
Index([  81,   82,  120,  121,  156,  157,  194,  195,  228,  290,  291,  364,
        365,  410,  451,  452,  495,  535,  536,  537,  538,  676,  677,  678,
        767,  768,  813,  875,  876,  910,  911,  915, 1993, 1994, 1995],
      dtype='int64')
```

```
# Sé que churn es la variable objetivo, y es importante no perder datos. Tras hacer varias pruebas, he decidido
# que lo mejor es imputar esta variable con la moda (debería ser con la media al ser numérica pero es variable binar
datos['churn'].fillna(datos['churn'].mode()[0], inplace=True)
```

```
# Imputamos la variable gender con la moda, ya que es categórica. A las numéricas, con la media
datos['gender'].fillna(datos['gender'].mode()[0], inplace=True) # Obtenemos el primer valor de la moda e imputarlos

# Imputamos la variable maximum_days_inactive con la media, ya que es una variable numérica
datos['maximum_days_inactive'].fillna(datos['maximum_days_inactive'].mean(), inplace=True)
```

7.4. Verificación de la correcta imputación

En este apartado se comprueba que los datos se han imputado correctamente, y que ninguna variable contiene valores vacíos.

```
# Verificar qué columnas tienen valores nulos y contarlos
columnas_con_nulos = datos.columns[datos.isnull().any()]
valores_nulos_por_columna = datos[columnas_con_nulos].isnull().sum()

# Imprimir las columnas con valores nulos y la cantidad de valores nulos en cada una
print("Columnas con valores nulos:")
print(valores_nulos_por_columna)

#Observamos que todos los valores nulos han sido rellenados. No perdemos ningún valor.
```

```
Columnas con valores nulos:
Series([], dtype: float64)
```

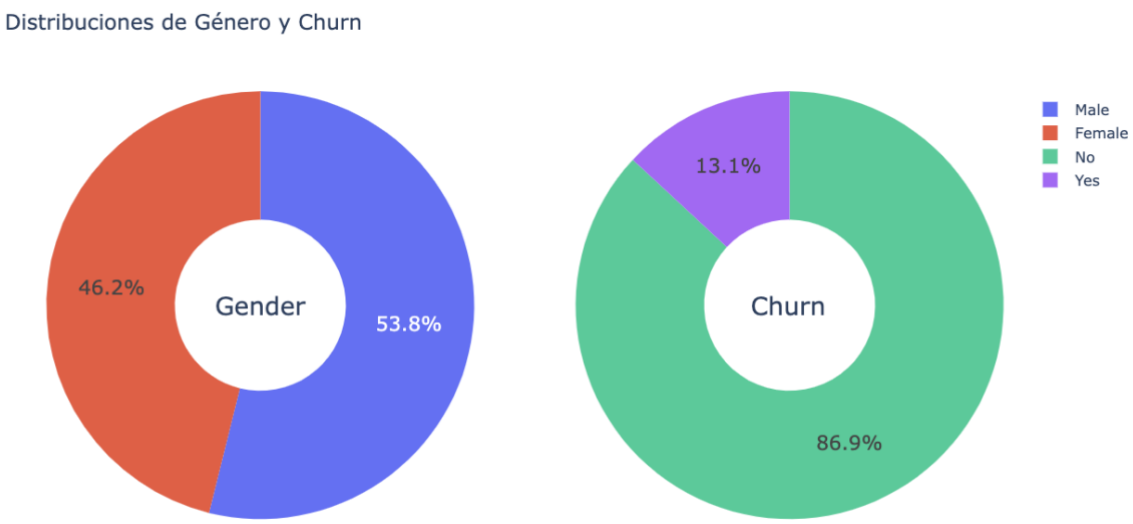
```
# Volvemos a comprobar que los valores no se han perdido. Es posible que las variables 3 y 4 salgan como tipo entera
# es por pasos más abajo y he ejecutado esto después para comprobar sin reiniciar el kernel
datos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   gender                  2000 non-null   object 
 1   age                     2000 non-null   int64  
 2   no_of_days_subscribed   2000 non-null   int64  
 3   multi_screen            2000 non-null   object 
 4   mail_subscribed         2000 non-null   object 
 5   weekly_mins_watched     2000 non-null   float64
 6   minimum_daily_mins      2000 non-null   float64
 7   maximum_daily_mins      2000 non-null   float64
 8   weekly_max_night_mins   2000 non-null   int64  
 9   videos_watched          2000 non-null   int64  
 10  maximum_days_inactive   2000 non-null   float64
 11  customer_support_calls  2000 non-null   int64  
 12  churn                   2000 non-null   float64
dtypes: float64(5), int64(5), object(3)
memory usage: 203.3+ KB
```

7.5. Distribución de las variables género y *churn*

Este apartado muestra un gráfico con las distribuciones de hombres y mujeres, y por otro lado, el porcentaje de usuarios que se dan de baja del servicio.

```
# Gráfico para visualizar % de hombres y mujeres y el % de churn

g_labels = ['Male', 'Female']
c_labels = ['No', 'Yes']

# Creamos gráficos utilizando el tipo de dominio para hacer un gráfico de tarta
fig = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'}, {'type':'domain'}]])
fig.add_trace(go.Pie(labels=g_labels, values=datos['gender'].value_counts(), name="Gender"),1, 1)
fig.add_trace(go.Pie(labels=c_labels, values=datos['churn'].value_counts(), name="Churn"),1, 2)

# Quiero que el gráfico sea de tipo 'donut'
fig.update_traces(hole=.4, hoverinfo="label+percent+name", textfont_size=16)

fig.update_layout(
    title_text="Distribuciones de Género y Churn",
    # Leyenda dentro de los gráficos
    annotations=[dict(text='Gender', x=0.175, y=0.5, font_size=20, showarrow=False),
                 dict(text='Churn', x=0.815, y=0.5, font_size=20, showarrow=False)])
fig.show()
```



7.6. Procesamiento de los datos

En este apartado, se muestra cómo se transforman los datos categóricos en tipo entero, y la correlación que tiene cada variable respecto a *churn*.

```
# Estoy convirtiendo los datos categóricos en enteros

def object_to_int(dataframe_series):
    if dataframe_series.dtype=='object':
        dataframe_series = LabelEncoder().fit_transform(dataframe_series)
    return dataframe_series
```

```
datos = datos.apply(lambda x: object_to_int(x))
datos.head()
```

	gender	age	no_of_days_subscribed	multi_screen	mail_subscribed	weekly_mins_watched	minimum_daily_mins	maximum_daily_mins	w
0	0	36	62	0	0	148.35	12.2	16.81	
1	0	39	149	0	0	294.45	7.7	33.37	
2	0	65	126	0	0	87.30	11.9	9.89	
3	0	24	131	0	1	321.30	9.5	36.41	
4	0	40	191	0	0	243.00	10.9	27.54	

```
# Quiero ver la correlación que tienen las variables con churn
plt.figure()
datos.corr()['churn'].sort_values(ascending = False)
```

```
churn                     1.000000
multi_screen              0.258324
customer_support_calls    0.204774
weekly_mins_watched       0.162876
maximum_daily_mins        0.162874
minimum_daily_mins        0.066646
maximum_days_inactive     0.046348
age                       0.011777
weekly_max_night_mins     0.006917
no_of_days_subscribed     0.002517
gender                   -0.015122
videos_watched           -0.019314
mail_subscribed          -0.077705
Name: churn, dtype: float64

<Figure size 640x480 with 0 Axes>
```

7.7. Asignación *churn* como variable objetivo y conjuntos de entrenamiento y prueba

A continuación, se presenta la asignación de los conjuntos de datos como *train y test set*.

```
# Establezco la variable churn como la objetivo
x = datos.drop(columns = ['churn'])
y = datos['churn'].values
```

```
# Divido los datos en conjuntos de entrenamiento y prueba // tras probar varios, 70-30 es el que mejor me da
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.30, random_state = 40, stratify=y)
```

7.8. Modelo KNN

A continuación, se realiza el algoritmo KNN y se muestran sus respectivos resultados.

KNN

```
knn_model = KNeighborsClassifier(n_neighbors = 11)
knn_model.fit(x_train,y_train)
predicted_y_knn = knn_model.predict(x_test)
accuracy_knn = knn_model.score(x_test,y_test)
print("KNN accuracy:",accuracy_knn)

print(classification_report(y_test, predicted_y_knn))
```

```
KNN accuracy: 0.8683333333333333
              precision    recall  f1-score   support

           0       0.87      0.99      0.93       521
           1       0.50      0.06      0.11        79

    accuracy                           0.87       600
   macro avg       0.69      0.53      0.52       600
weighted avg       0.83      0.87      0.82       600
```

```
from sklearn.metrics import roc_auc_score

# Calcula el AUROC
auroc_knn = roc_auc_score(y_test, predicted_y_knn)

print("AUROC:", auroc_knn)
```

```
AUROC: 0.5268471051288903
```

```
# Calculo los valores falsos positivos y verdaderos positivos
fpr_knn, tpr_knn, _ = roc_curve(y_test, predicted_y_knn)
```

7.9. Modelo SVC

A continuación, se realiza el algoritmo SVC y se muestran sus respectivos resultados.

SVC

```
svc_model = SVC(random_state = 1)
svc_model.fit(x_train,y_train)
predict_y = svc_model.predict(x_test)
accuracy_svc = svc_model.score(x_test,y_test)
print("SVM accuracy is:",accuracy_svc)

print(classification_report(y_test, predict_y))
```

```
SVM accuracy is: 0.8683333333333333
              precision    recall  f1-score   support

           0       0.87      1.00      0.93       521
           1       0.00      0.00      0.00        79

    accuracy                           0.87       600
   macro avg       0.43      0.50      0.46       600
weighted avg       0.75      0.87      0.81       600
```

```
# Calcula el AUROC
auroc_svc = roc_auc_score(y_test, predict_y)

# Imprime el valor del AUROC
print("AUROC:", auroc_svc)
```

```
AUROC: 0.5
```

```
# Calculo los valores falsos positivos y verdaderos positivos
fpr_svc, tpr_svc, _ = roc_curve(y_test, predict_y)
```

7.10. Modelo RF

A continuación, se realiza el algoritmo RF y se muestran sus respectivos resultados.

RF

```
model_rf = RandomForestClassifier(n_estimators=100, random_state=40)
model_rf.fit(x_train, y_train)

# Make predictions
prediction_test = model_rf.predict(x_test)
print (metrics.accuracy_score(y_test, prediction_test))

print(classification_report(y_test, prediction_test))
```

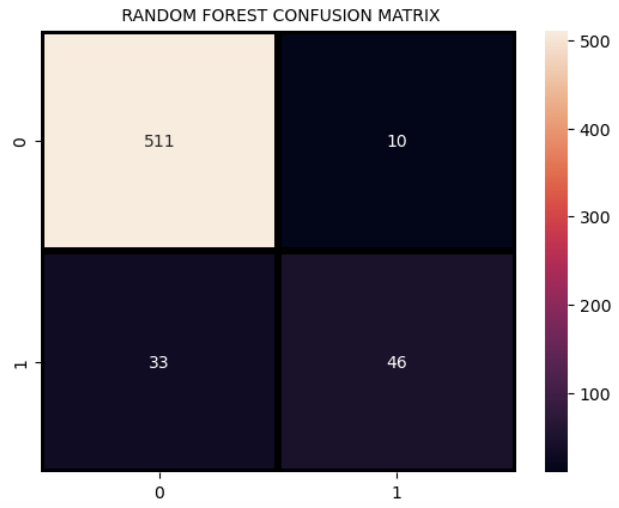
```
0.9283333333333333
              precision    recall  f1-score   support

           0       0.94      0.98      0.96       521
           1       0.82      0.58      0.68        79

    accuracy                           0.93       600
   macro avg       0.88      0.78      0.82       600
weighted avg       0.92      0.93      0.92       600
```

```
# Calcula el AUROC
auroc_rf = roc_auc_score(y_test, prediction_test)

# Imprime el valor del AUROC
print("AUROC:", auroc_rf)
```

```
AUROC: 0.7815423115236035
```

```
# Calculo los valores falsos positivos y verdaderos positivos
fpr_rf, tpr_rf, _ = roc_curve(y_test, prediction_test)
```

```
plt.figure()
sns.heatmap(confusion_matrix(y_test, prediction_test),annot=True,fmt = "d",linecolor="k",linewidths=3)

plt.title(" RANDOM FOREST CONFUSION MATRIX",fontsize=10)
plt.show()
```



7.11. Modelo regresión logística

A continuación, se realiza el algoritmo de la regresión logística y se muestran sus respectivos resultados.

Logistic Regression

```
logreg = LogisticRegression()
logreg.fit(x_train,y_train)
accuracy_logreg = logreg.score(x_test,y_test)
print("Logistic Regression accuracy is :",accuracy_logreg)

logreg_pred= logreg.predict(x_test)
report = classification_report(y_test,logreg_pred)
print(report)
```

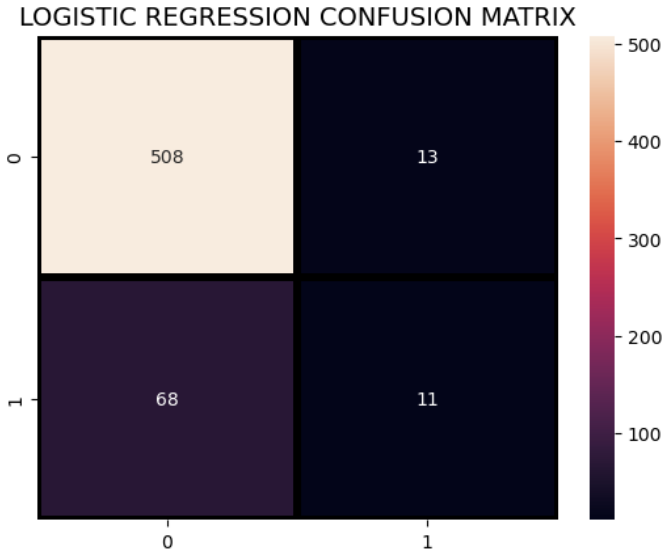
```
Logistic Regression accuracy is : 0.865
              precision    recall  f1-score   support

           0       0.88      0.98      0.93       521
           1       0.46      0.14      0.21        79

    accuracy                           0.86       600
   macro avg       0.67      0.56      0.57       600
weighted avg       0.83      0.86      0.83       600
```

```
# Calcula el AUROC
auroc_log = roc_auc_score(y_test, logreg_pred)

# Imprime el valor del AUROC
print("AUROC:", auroc_log)
```

```
AUROC: 0.5571442454870138
```

```
# Calculo los valores falsos positivos y verdaderos positivos
fpr_log, tpr_log, _ = roc_curve(y_test, logreg_pred)
```

```
plt.figure()
sns.heatmap(confusion_matrix(y_test, logreg_pred),
                annot=True,fmt = "d",linecolor="k",linewidths=3)

plt.title("LOGISTIC REGRESSION CONFUSION MATRIX",fontsize=14)
plt.show()
```



7.12. Modelo árbol de decisión

A continuación, se realiza el algoritmo del árbol de decisión y se muestran sus respectivos resultados.

Decision Tree

```
decision_tree = DecisionTreeClassifier()
decision_tree.fit(x_train,y_train)
predict_y = decision_tree.predict(x_test)
accuracy_dt = decision_tree.score(x_test,y_test)
print("Decision Tree accuracy is :",accuracy_dt)
```

```
Decision Tree accuracy is : 0.8683333333333333
```

```
print(classification_report(y_test, predict_y))
```

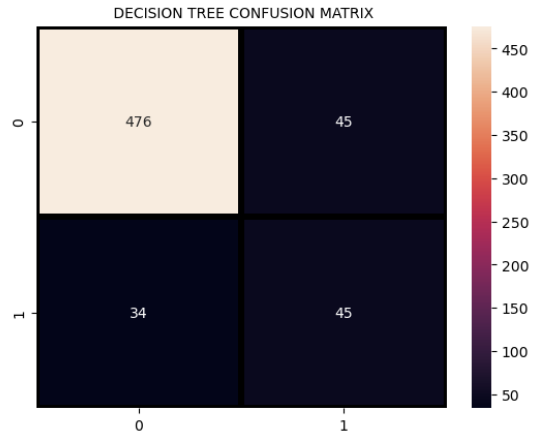
```
              precision    recall  f1-score   support

           0       0.93      0.91      0.92       521
           1       0.50      0.57      0.53        79

    accuracy                           0.87       600
   macro avg       0.72      0.74      0.73       600
weighted avg       0.88      0.87      0.87       600
```

```
# Calcula el AUROC
auroc_dt = roc_auc_score(y_test, predict_y)

# Imprime el valor del AUROC
print("AUROC:", auroc_dt)
```

```
AUROC: 0.7416239461600136
```

```
# Calculo los valores falsos positivos y verdaderos positivos
fpr_dt, tpr_dt, _ = roc_curve(y_test, predict_y)
```

```
plt.figure()
sns.heatmap(confusion_matrix(y_test, predict_y),annot=True,fmt = "d",linecolor="k",linewidths=3)

plt.title(" DECISION TREE CONFUSION MATRIX",fontsize=10)
plt.show()
```

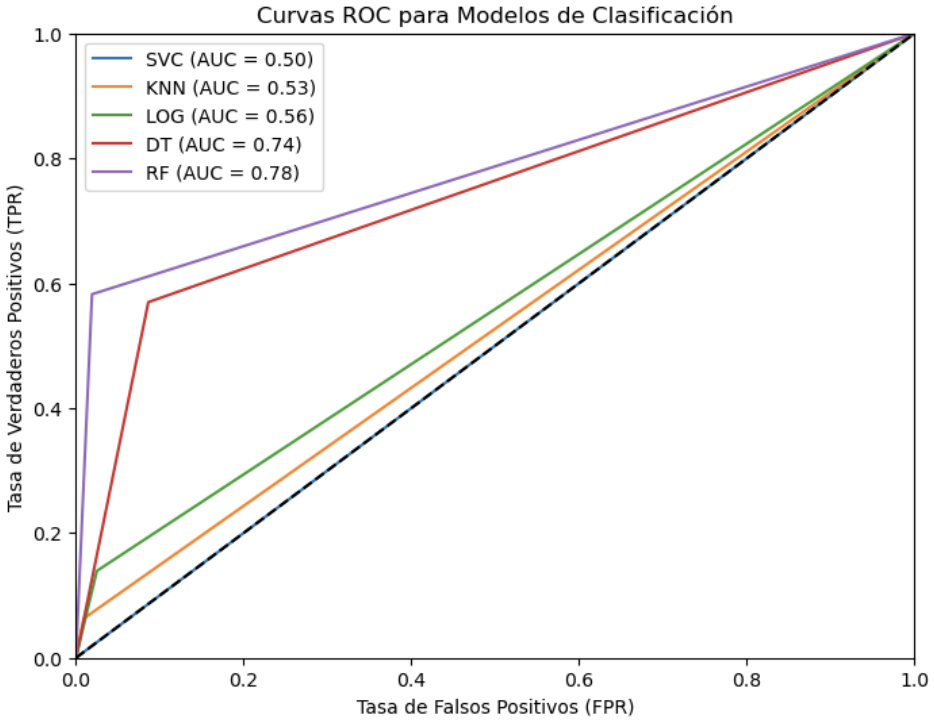


7.13. Gráfico AUC-ROC

Este apartado muestra el gráfico agregado de todos los modelos empleados.

Gráfico con todas las curvas roc y areas de cada modelo

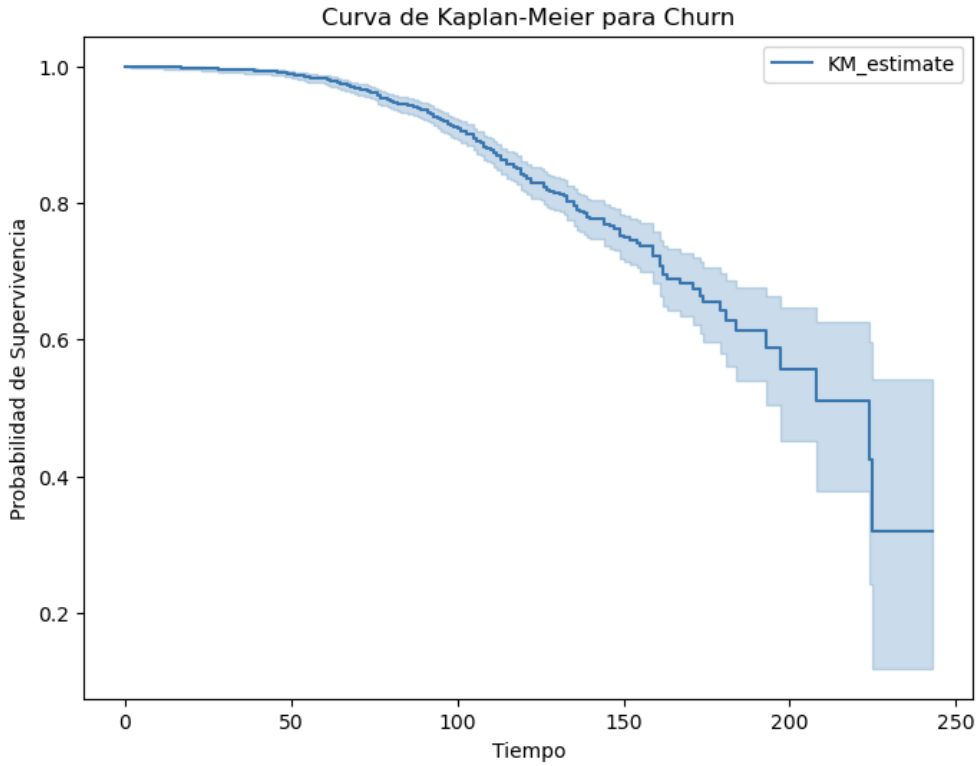
```
# Crea una figura y un eje
fig, ax = plt.subplots(figsize=(8, 6))

# Trazar las curvas ROC para cada modelo con etiquetas específicas
ax.plot(fpr_svc, tpr_svc, label='SVC (AUC = {:.2f})'.format(auroc_svc))
ax.plot(fpr_knn, tpr_knn, label='KNN (AUC = {:.2f})'.format(auroc_knn))
ax.plot(fpr_log, tpr_log, label='LOG (AUC = {:.2f})'.format(auroc_log))
ax.plot(fpr_dt, tpr_dt, label='DT (AUC = {:.2f})'.format(auroc_dt))
ax.plot(fpr_rf, tpr_rf, label='RF (AUC = {:.2f})'.format(auroc_rf))



# Configurar etiquetas y título
ax.set_xlabel('Tasa de Falsos Positivos (FPR)')
ax.set_ylabel('Tasa de Verdaderos Positivos (TPR)')
ax.set_title('Curvas ROC para Modelos de Clasificación')

# Dibujar la línea de línea de referencia
ax.plot([0, 1], [0, 1], 'k--')

# Establecer límites del eje x e y
ax.set_xlim([0.0, 1.0])
ax.set_ylim([0.0, 1.0])

# Agregar una leyenda con los valores AUC
ax.legend()

# Mostrar el gráfico
plt.show()
```



7.14. Análisis de supervivencia

Supervivencia

```
from lifelines import KaplanMeierFitter

# Creo un objeto KaplanMeierFitter
kmf = KaplanMeierFitter()

# Ajusto el modelo de Kaplan-Meier a los datos
tiempo_hasta_churn = datos['no_of_days_subscribed']
churn = datos['churn']
kmf.fit(tiempo_hasta_churn, event_observed=churn)

# Calculo y grafica la curva de Kaplan-Meier
plt.figure(figsize=(8, 6))
kmf.plot(ci_show=True)
plt.title('Curva de Kaplan-Meier para Churn')
plt.xlabel('Tiempo')
plt.ylabel('Probabilidad de Supervivencia')
plt.show()

mediana = kmf.median_survival_time_
print(f"Mediana de Supervivencia: {mediana} días")
```

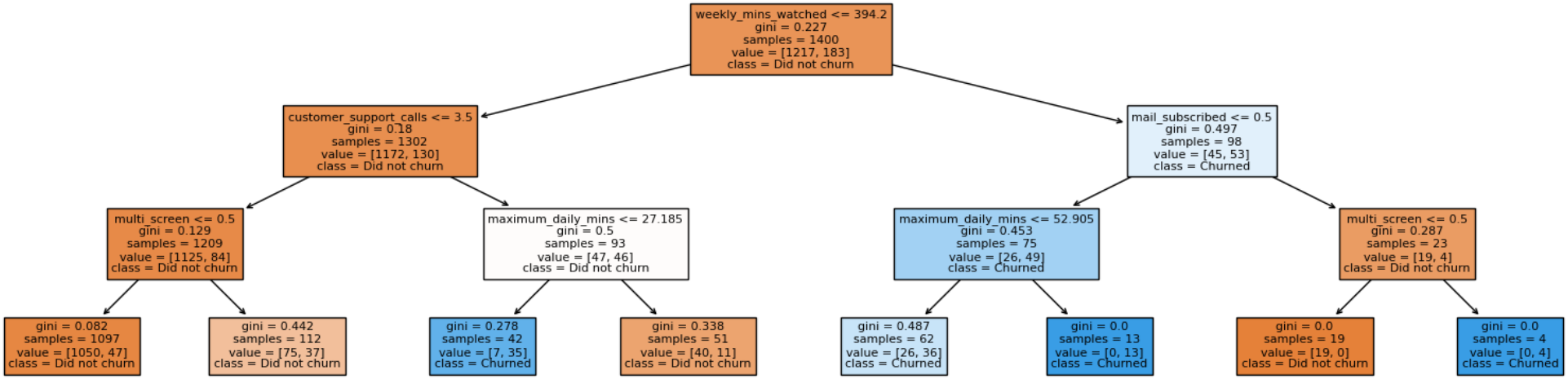



```
Mediana de Supervivencia: 224.0 días
```

7.15. Gráfico árbol de decisión

Se presenta el código utilizado para obtener el árbol de decisión, que se muestra en la página siguiente con el fin de obtener una mejor visualización.

```
from sklearn.tree import plot_tree


# Crear un clasificador de árbol de decisión
clf = DecisionTreeClassifier(max_depth=3)

# Entrenar el clasificador en el conjunto de entrenamiento
clf.fit(x_train, y_train)

# Realizar predicciones en el conjunto de prueba
y_pred = clf.predict(x_test)

feature_names_list=x.columns.tolist()

# Visualizar el árbol de decisión
plt.figure(figsize=(20, 5))
plot_tree(clf, filled=True, feature_names=feature_names_list, class_names=['Did not churn', 'Churned'])
plt.show()
```


weekly_mins_watched <= 394.2
gini = 0.227
samples = 1400
value = [1217, 183]
class = Did not churn
customer_support_calls <= 3.5
gini = 0.18
samples = 1302
value = [1172, 130]
class = Did not churn
mail_subscribed <= 0.5
gini = 0.497
samples = 98
value = [45, 53]
class = Churned
multi_screen <= 0.5
gini = 0.129
samples = 1209
value = [1125, 84]
class = Did not churn
maximum_daily_mins <= 27.185
gini = 0.5
samples = 93
value = [47, 46]
class = Did not churn
maximum_daily_mins <= 52.905
gini = 0.453
samples = 75
value = [26, 49]
class = Churned
multi_screen <= 0.5
gini = 0.287
samples = 23
value = [19, 4]
class = Did not churn
gini = 0.082
samples = 1097
value = [1050, 47]
class = Did not churn
gini = 0.442
samples = 112
value = [75, 37]
class = Did not churn
gini = 0.278
samples = 42
value = [7, 35]
class = Churned
gini = 0.338
samples = 51
value = [40, 11]
class = Did not churn
gini = 0.487
samples = 62
value = [26, 36]
class = Churned
gini = 0.0
samples = 13
value = [0, 13]
class = Churned
gini = 0.0
samples = 19
value = [19, 0]
class = Did not churn
gini = 0.0
samples = 4
value = [0, 4]
class = Churned

7.16. Contraste de hipótesis

```
from sklearn.model_selection import cross_val_score, StratifiedKFold
from scipy.stats import ttest_1samp
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score

# modelos aplicados
modelos = [KNeighborsClassifier(), RandomForestClassifier(), DecisionTreeClassifier(), SVC(), LogisticRegression()]

# número de folds para la validación cruzada
n_folds = 10

# proporción regla de la mayoría
proporcion_mayoria = 0.6

# tamaño de la muestra de acuerdo a la proporción de la regla de la mayoría
tamaño_muestra = len(y_test)
clientes_se_van_esperados = tamaño_muestra * proporcion_mayoria

# lista para almacenar AUCROC de cada modelo
auc_roc_scores = []

# validación cruzada para cada modelo y cálculo de los AUCROC
for modelo in modelos:
    # Utiliza StratifiedKFold para preservar la proporción de clases en cada fold
    skf = StratifiedKFold(n_splits=n_folds, shuffle=True, random_state=42)

    # Calcula el AUCROC de cada fold
    auc_rocs = cross_val_score(modelo, x_train, y_train, cv=skf, scoring='roc_auc')

    # Calcula el AUCROC medio de todos los folds
    auc_roc_media = np.mean(auc_rocs)

    # Almacena el AUCROC medio
    auc_roc_scores.append(auc_roc_media)

for i, modelo in enumerate(modelos):
    print(f"AUC-ROC medio del modelo {modelo.__class__.__name__}: {auc_roc_scores[i]*100:.2f}%")

# calcula AUCROC esperado bajo la regla de la mayoría
auc_roc_esperado_mayoria = 0.5  # aleatorio
```

```
# contraste de proporciones utilizando un test t
t_statistic, p_value = ttest_1samp(auc_roc_scores, auc_roc_esperado_mayoria)

if p_value < 0.05:
    print(f"El contraste de proporciones muestra que al menos uno de los modelos supera significativamente la regla
else:
    print(f"No hay evidencia suficiente para concluir que alguno de los modelos supera significativamente la regla d
```

```
AUC-ROC medio del modelo KNeighborsClassifier: 64.27%
AUC-ROC medio del modelo RandomForestClassifier: 85.23%
AUC-ROC medio del modelo DecisionTreeClassifier: 72.21%
AUC-ROC medio del modelo SVC: 68.28%
AUC-ROC medio del modelo LogisticRegression: 73.79%
El contraste de proporciones muestra que al menos uno de los modelos supera significativamente la regla de la mayor
ía.
```