

MADRID BUSINESS INTELLIGENCE SCHOOL

PROYECTO FIN DE MÁSTER

MÁSTER EN DATA SCIENCE PARA PROFESIONALES

MODELO DE PREDICCIÓN DE PRECIOS

Autores:

Edson Rodrigues

Héctor Fernández Juárez

Keyla Flores

Tutores:

Jacinto Arias

Madrid, 04 de noviembre de 2022

TÍTULO: **Modelo de predicción de precios.**

AUTORES: **D. Edson Rodrigues**
 D. Héctor Fernández
 D. Keyla Flores

TUTORES: **D. Jacinto Arias**

Resumen del PFM

El presente proyecto tiene por objeto la creación de un modelo de predicción de precios de Amazon.

Desde el punto de vista de negocio, la idea de este trabajo es ayudar al cliente a tomar una decisión en la compra de un producto.

Dada una visión global de lo que va a consistir el trabajo, diferenciaremos dos partes principales de las que consta el proyecto.

La parte visual, que abarca la creación de la plataforma, más en concreto del microsite, es la interfaz con la que interacciona el usuario (No tiene cabida en el proyecto, sólo desde un punto de vista de negocio).

Por otro lado, tenemos el proceso ETL, del cual una parte importante consta de la extracción de información mediante técnicas de web scraping. Optamos por otras fuentes de datos para completar la anterior, en concreto, del plugin denominado “Keppa”. La extracción de datos la realizaremos en el website de amazon.es

Una vez obtenida la información, aplicaremos diferentes algoritmos para predecir la subida o no del precio de un producto.

La principal necesidad a cubrir desde el punto de vista del usuario, es ayudarlo a tomar la mejor decisión en cuanto a su compra, lo que se puede traducir en un ahorro significativo. Es decir, mediante técnicas de machine learning, predecir el precio de determinados productos del website de amazon y de esta forma dar la posibilidad al usuario a decidir cuándo es el mejor momento para realizar su compra.

Palabras clave

Web scraping, machine learning, ETL.

Abstract

The purpose of this project is to create a price prediction model for Amazon.

From a business point of view, the idea of this work is to help the customer make a decision to purchase a product.

Given a global vision of what the work is going to consist of, we will differentiate two main parts that the project consists of.

The visual part, which covers the creation of the platform, more specifically the microsite, is the interface with which the user interacts (it has no place in the project, only from a business point of view).

On the other hand, we have the ETL process, of which an important part consists of extracting information using web scraping techniques. We opted for other data sources to complete the previous one, specifically, from the plugin called “Keppa”. The data extraction will be carried out on the amazon.es website

Once the information is obtained, we will apply different algorithms to predict the rise or not of the price of a product.

The main need to cover from the user's point of view is to help them make the best decision regarding their purchase, which can translate into significant savings. That is, using machine learning techniques, predict the price of certain products on the Amazon website and thus give the user the opportunity to decide when is the best time to make their purchase.

Keywords

Web scraping, machine learning, ETL.

ÍNDICE.

Contenido

1	Introducción.....	1
1.1	Motivación y objetivos.	1
1.1.1	Motivación.	1
1.1.2	Objetivos.	1
2	Análisis de negocio.....	3
2.1	Evolución del comercio electrónico.	3
2.2	Estudio de mercado.	3
2.2.1	Análisis de competencia.....	4
2.2.2	Análisis DAFO.....	7
2.3	Modelo de negocio.	8
2.3.1	Propuesta de negocio.....	8
2.3.2	Elemento diferenciador.	9
3	Adquisición y exploración de datos.	10
3.1	Fuente de información “Keepa”.	10
3.2	Proceso ETL	12
3.3	Extracción de los datos.....	13
3.4	Información disponible en el conjunto de datos.....	14
3.5	Análisis de los datos.	15
3.6	Set de datos final.....	19
4	Aplicación de Modelos Machine Learning.	21
4.1	Métricas.	21
4.2	Modelos Machine Learning aplicados.....	23
4.2.1	KNN.	23

4.2.2	DecisionTreeClassifier.....	24
4.2.3	Naive Bayes.....	25
4.2.4	SGD.....	26
4.2.5	Regresión logística.....	27
4.2.6	XGB.....	28
4.2.7	Máquinas de vector de soporte (SVM)	29
4.2.8	AdaBoost.....	30
4.2.9	Bagging.....	31
4.2.10	GradientBoostig.....	32
4.2.11	ExtraTree.....	33
4.3	Resultados tras aplicación modelos.....	34
4.4	Posibles soluciones al Balanceo de datos.....	36
4.4.1	Penalización para compensar el desbalanceo.....	37
4.4.2	Subsampling.....	38
4.4.3	Oversampling de la clase minoritaria.....	41
4.4.4	Resampling con Smote-Tomek.....	43
4.4.5	Ensamble de Modelos con Balanceo.....	46
4.4.6	Conclusión tras aplicación técnicas balanceo de datos.....	47
5	Implantación, monetización y retorno del Proyecto.....	49
5.1	Implantación.....	49
5.2	Datos fiables.....	49
5.3	Plataforma integrada.....	49
5.4	Buscar siempre la simplicidad.....	50
5.5	Monetización.....	50
5.6	Retorno del proyecto.....	52
6	Conclusiones y líneas futuras.....	54
7	Bibliografía.....	58

1

Introducción.

1.1 Motivación y objetivos.

1.1.1 Motivación.

Debido a la creciente demanda del comercio electrónico, aún más acentuada con la situación sanitaria COVID-19, y un cambio en los hábitos de consumo, nos vemos en la tesitura de poner en marcha el presente trabajo.

Vemos la necesidad de brindar un servicio al consumidor final, ahorrándose cuanto pueda en una compra que haga con cierta frecuencia.

Creemos que la aplicación de algoritmos y machine learning, en general, aplicado al mundo del retail son buena combinación para desarrollar este proyecto.

Existen páginas del tipo de ‘chollometro’ que son una comunidad en el que se publican ofertas de todo tipo. Nosotros queremos ir un paso más allá y poder predecir el precio de un producto o conjunto de productos y determinar cuál es el mejor momento para realizar la compra.

También queremos dar respuesta a una de las preguntas que se hace todo consumidor, ¿he realizado buena compra? Hacer una predicción del posible precio y compararlo con el actual nos ayudaría tomar una decisión respecto a realizar o no la compra.

Queremos descubrir patrones de comportamiento dentro de los grandes retailers y comprender en qué se basan para fijar el precio final.

1.1.2 Objetivos.

Desde un punto de vista técnico el objetivo es el desarrollo de un modelo de clasificación capaz de predecir la subida, bajada o igualdad de precio a una semana

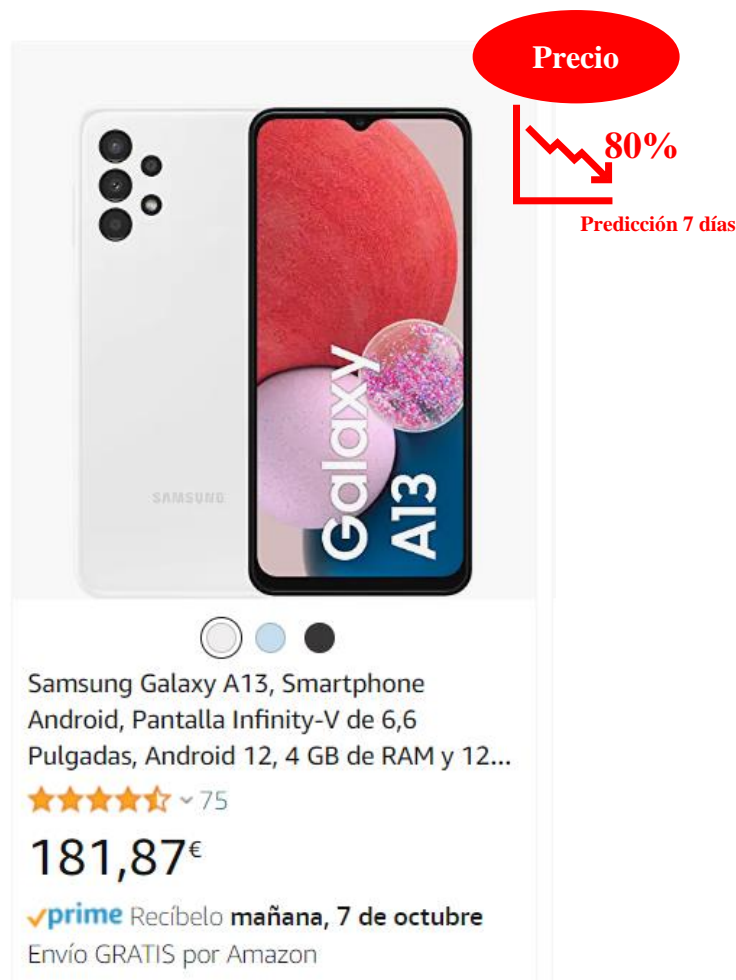
vista del precio de un producto de la categoría electrónica de amazon con todo el procesamiento previo que implica.

Desde el punto de vista del cliente, queremos poder ayudar en la toma de decisiones a la hora de comprar un producto.

Desde el punto de vista del negocio la intención es el desarrollo de una página web con la cual el consumidor pueda interactuar y ver el precio del producto actual y el posible precio del producto a futuro.

Monetizar esta idea de negocio con un modelo de dropshipping del tipo B2C. Obteniendo una comisión por venta redirigiendo el tráfico hacia la web del proveedor, en este caso Amazon.

La diferencia respecto a un e-commerce habitual sería la posibilidad de incorporar un indicador con la probabilidad de bajada o subida del precio del producto.



The image shows a product listing for the Samsung Galaxy A13 on Amazon. The product is a white smartphone with a 6.6-inch Infinity-V display. The listing includes the product name, specifications (Android 12, 4 GB RAM, 12 GB storage), a 4.5-star rating from 75 reviews, and a price of 181,87€. A Prime delivery promise is shown for tomorrow, October 7th. A red overlay in the top right corner, labeled 'Precio', features a line graph showing a downward trend with a label '80%' and the text 'Predicción 7 días'.

Precio

80%

Predicción 7 días

Samsung Galaxy A13, Smartphone
Android, Pantalla Infinity-V de 6,6
Pulgadas, Android 12, 4 GB de RAM y 12...

★★★★☆ ~ 75

181,87€

✓prime Recíbelo mañana, 7 de octubre
Envío GRATIS por Amazon

2

Análisis de negocio.

2.1 Evolución del comercio electrónico.

Es evidente el aumento del comercio electrónico en España debido a un cambio en los hábitos de consumo de la sociedad, cada vez más todo lo gestionamos vía web/app. Este tipo de hábitos se ha acentuado más a raíz de la crisis sanitaria que hemos vivido. Varios medios de comunicación respaldan esta idea:

“El comercio electrónico superó en España los 12.000 millones de euros en el segundo trimestre de 2020”. Fuente CNMC.

“VII Estudio anual de eCommerce en España 2020”. Fuente Marketing4Ecommerce

“El salto del comercio electrónico”. Fuente INE.

La plataforma que queremos desarrollar se enfocará directamente en la evaluación de determinadas categorías de productos de los cuales se realizará la predicción relacionada a la fluctuación de este.

2.2 Estudio de mercado.

Requiere de una importancia capital conocer cómo está el mercado, y, en particular, quiénes pueden ser nuestros potenciales clientes y qué es lo que éstos pueden demandar.

En línea con lo anteriormente expuesto, la necesidad de utilizar la obtención y el análisis de datos como estrategia para crear ventajas competitivas está cada vez más presente en el mercado. Siendo esta última característica, el principal potencial a explotar de nuestro proyecto. La gran ventaja competitiva es el conjunto de técnicas y procedimientos en torno al dato, explotados de una forma óptima.

2.2.1 Análisis de competencia.

Como hemos mencionado, el aumento del comercio electrónico hace que se cree un ecosistema de ideas de negocio en torno a él. Los clientes cada vez más tienen multitud de sitios webs, o incluso dentro de la misma web, tienen distintos precios y no son capaces de valorar todos a la vez, pues no cuentan con suficientes herramientas que permitan gestionarlos todos.

Actualmente, existen algunas plataformas que ayudan a decidir al cliente de forma similar a la idea que queremos desarrollar en este proyecto. Algunas de estas plataformas:

2.2.1.1 *Keepa.*

Es una extensión disponible para Chrome y Firefox. Cuando la descargas, con ella puedes ver, dentro de Amazon, en la ficha de un producto el historial de sus precios, y de esta manera, ver que hay una oferta de algo que te interesa. Puedes valorar hasta qué punto su precio es bajo comparado con cómo ha estado en el pasado. Esta herramienta te permite:

- Visualizar gráficos completos del historial de precios.
- Generar alertas de caídas de precios y ver la disponibilidad del producto.
- Comparar los precios internacionales de Amazon.
- Soporte de configuración regional de Amazon [.com | .co.uk | .de | .co.jp | .fr | .ca | .it | .es | .in | .com.mx]
- Registro opcional.
- Importación de lista de deseos.
- Ofertas, una descripción general de las recientes bajadas de precios.

2.2.1.2 *El portal Watch4Price.*

Watch4Price.com es un portal dedicado a seguir y comparar los cambios de precios de Amazon.it, Amazon.de, Amazon.fr y Amazon.es. y eBay.

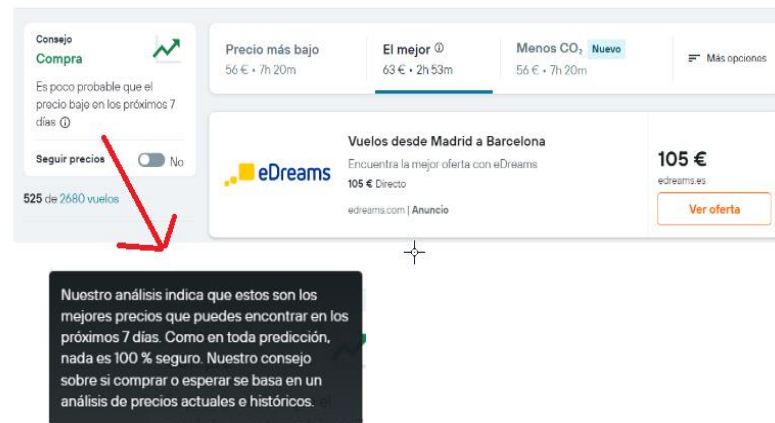


Tiene menos funcionalidades que el anterior, pero puede considerarse como una segunda opción.

2.2.1.3 Kayak.

Es un metabuscador de viajes que procesa miles de búsquedas de viajes en sus plataformas ayudando a millones de viajeros a tomar las mejores decisiones de viajes.

Se centra en la contratación de viajes, no es competencia directa ya que se centra en un tipo de producto, pero es interesante porque incorpora una herramienta de predicción en el precio de los vuelos y te ofrece consejos. Según explican en su web:



“Nuestros científicos han desarrollado estas predicciones de tendencias de precios de vuelos utilizando algoritmos y modelos matemáticos. Las predicciones basadas en historiales de búsqueda no son perfectas, así que no podemos garantizar que sean siempre correctas. Por eso siempre te indicamos el grado de fiabilidad de nuestro análisis”.

2.2.1.4 Amazon Forecast.

Puede que no sea competencia directa, pero resulta interesante destacar este servicio que proporciona Amazon. Según explican en su web, se trata de un servicio completamente administrado que utiliza el aprendizaje automático para crear previsiones con un alto nivel de exactitud. Recalca que no es necesario experiencia previa en aprendizaje automático para utilizar esta tecnología:

“Amazon Forecast, basándose en la misma tecnología que se utiliza en Amazon.com, usa el aprendizaje automático para combinar datos de serie temporal con variables adicionales a los fines de crear previsiones. Amazon Forecast no requiere experiencia previa en aprendizaje automático para empezar a usarlo. Sólo necesita proporcionar datos históricos, además de cualquier información adicional que considere que pueda incidir en sus previsiones. Por ejemplo, la demanda de un color particular de camisa



puede cambiar con las estaciones y la ubicación de las tiendas. Esta relación compleja es difícil de determinar por sí sola, pero el aprendizaje automático es ideal para reconocerla. Una vez hecho esto, Amazon Forecast examinará los datos automáticamente, identificará lo que es significativo y producirá un modelo de previsión capaz de hacer predicciones que son hasta un 50 % más precisas que si solo mira los datos de serie temporal.

Amazon Forecast es un servicio completamente administrado, por lo que no hay servidores que aprovisionar ni modelos de aprendizaje automático que crear, entrenar ni implementar. Solo pagará por lo que utiliza: no se requieren pagos mínimos ni compromisos iniciales.”

Afirma que solo se paga el servicio mientras se utiliza.

2.2.1.5 CamelCamelCamel.

Es una web que permite comparar los diferentes precios de los productos de Amazon. Es algo similar a lo que tenemos en mente, esta web permite ahorrar dinero ya que muestra un histórico de los precios y permite tomar una decisión a la hora de comprar a un mejor/peor precio para el consumidor. Da la opción de generar alertas ante la llegada del precio que esperamos.

Te ayudamos a ahorrar dinero.



Avisos de bajada de precios Amazon

Crea seguimientos de precio Amazon y recibe avisos por email y Twitter cuando bajen los precios.



Gráficas de historial de precios de Amazon

Ver el historial de precios de casi 18 millones de productos Amazon



Extensiones DE Navegador

Añade nuestras gráficas de historial de precio a tu navegador con El Camelizer, nuestra extensión para Mozilla Firefox y Google Chrome.

2.2.2 Análisis DAFO.

Valorada rápidamente la competencia, ahora pasaremos a analizar y hacer un resumen de las fortalezas, debilidades y oportunidades y amenazas de este mercado o lo que es lo mismo vamos a realizar un análisis DAFO que nos ayudará a clarificar las dudas que surgen sobre el mercado al que vamos a entrar.

Debilidades	Amenazas
Credibilidad.	Desconfianza en las nuevas tecnologías.
Complicado plan de marketing.	Herramientas de aprendizaje automático (Amazon forecast) que pueden usarlas sin un perfil técnico.
Actualización de los algoritmos y efectividad de estos.	Análisis de un solo website sin realizar comparaciones.
Dificultades técnicas en el desarrollo.	Margen de beneficio bajo.
Alta inversión en SEO.	
Dependencia del proveedor.	
Complicado desarrollo en tiempo real.	
Fortalezas	Oportunidades
Incorporación de técnicas innovadoras (machine learning) es este tipo de productos.	Escalable a otras webs.
Flexibilidad.	Extrapolable a otros servicios.
Optimización de procesos (web scraping de imágenes...)	Pivotar hacia otra idea sería sencillo.
Baja inversión económica inicial.	
No almacenamiento físico de productos.	
Trabajar desde cualquier punto.	

Tabla 1. Análisis DAFO.

2.3 Modelo de negocio.

Trabajaremos con un modelo de negocio muy popular, actualmente conocido como Dropshipping. que viene siendo una variante del Ecommerce en donde una tienda no mantiene los productos que vende en stock; en su lugar, cuando se vende un producto, se compra la mercancía de una tercera persona y luego se envía directamente al cliente. Esto significa que el comerciante nunca ve o manipula el producto.

En otras palabras, la logística tradicional del comercio electrónico cambia por completo, ya que, con el Dropshipping, la tienda online solamente se encarga de gestionar las órdenes de los clientes, la facturación y la generación de una base de datos. Es la empresa mayorista la que se encarga de almacenar, empaquetar y enviar los productos a los clientes en nombre de la compañía online.

La diferencia entre el Dropshipping y el modelo de venta tradicional es que el comerciante que vende no cuenta con un inventario. En lugar de esto, el comerciante compra inventario de un tercero a medida que es necesario cumplir con los pedidos. Este tercero es generalmente un mayorista o incluso el propio fabricante del producto.

Cabe destacar que también nos enfocaremos en un modelo de negocio B2C ya que se orientan los servicios y productos al cliente final, además una de las principales ventajas que presenta el B2C para el consumidor es que le permitirá conseguir unos precios más económicos.

2.3.1 Propuesta de negocio.

Crear una plataforma que podremos utilizar en cualquier navegador de forma totalmente gratuita enfocada a la compra/venta de productos online, además permitiéndonos tener un mayor control sobre los precios de los productos que se encuentren dentro de las categorías y subcategorías predefinidas en la plataforma. Nos mostrará una gráfica del precio de un producto, con su historial, para saber en todo momento cuándo ha bajado o qué tendencia lleva.

También permitirá establecer alertas para que nos indique si un producto específico a bajado o subido de precio si necesidad de estar activamente en la plataforma.

2.3.2 Elemento diferenciador.

Nuestro elemento diferenciador del resto de las plataformas que ofrecen un servicio semejante es que utilizaremos Machine Learning donde el algoritmo informático encuentra un patrón en los datos y predice los resultados probables.

Cuando decimos predice los resultados probables nos referíamos a que a diferencia del resto de plataformas que solo te ofrecen servicios simplemente para ver el historial de ventas de determinado producto, nosotros además de ese servicio como bien se menciona anteriormente utilizaremos Machine Learning para realizar predicciones referentes a los cambios en el precio de un producto específico, de manera que el usuario podrá identificar fácilmente si el precio del producto va a subir o bajar en el tiempo. Estos patrones de aprendizaje automático son muy adaptables en la forma en que se actualizan constantemente a medida que se introducen nuevos datos. Esto hace que sea cada vez más precisos en sus predicciones cuanto más tiempo opera.

El problema que le resolveremos al usuario sería el siguiente:

“Quiero ahorrar en un producto y puedo esperar si el ahorro es significativo”

3

Adquisición y exploración de datos.

Esta parte del TFM es una de las partes más importantes del proyecto, en este capítulo realizamos una exposición de los datos, así como un análisis y preprocesado de los mismos, con el fin de poder tenerlos preparados para la próxima fase.

3.1 Fuente de información “Keepa”.

Como indicamos anteriormente, Keepa, es nuestra principal fuente de información. Nos hemos decantado por esta herramienta porque muestra de una forma clara el historial de precios de los productos de Amazon. Además, existe una extensa información de uso.

Hemos realizado una serie de comprobaciones de determinados productos comparando el precio actual con el de días pasados y podemos afirmar que Keepa es una fuente de información fiable.

Para la adquisición de los datos hemos visto que se puede hacer de dos formas: mediante una API o a través de su web.

- **Adquisición de datos mediante la API.**

Existe un módulo de Python que permite interactuar con la API en Keepa para consultar información e historial de productos de Amazon. También contiene un módulo de trazado del producto. Podemos acceder a la documentación de Keepa en el siguiente enlace <https://keepaapi.readthedocs.io/en/latest/> .

En la documentación, podemos encontrar la información acerca de cómo conectar la API, así como información sobre cómo podemos realizar consultas en el historial de productos.

La API nos permite crear una conexión directa con el servidor y extraer la información.

Para utilizar la API debemos acogernos a un plan de pago. Existen varios planes en función del número de tokens que necesitemos. Por ejemplo: si contratamos el plan de 20 tokens por minuto, este genera 20 tokens cada minuto y cada token es válido durante 60 minutos, por lo que la capacidad del cubo de tokens es de 1200 tokens. Los tokens se generan las 24 horas del día, tanto si se realizan solicitudes como si no. Cada producto solicitado consume un token. Si utilizara todos los tokens generados a lo largo de los 31 días del plan, podría solicitar un total de 892800 productos. El coste de este plan en concreto son 39€/ mes.

- **Adquisición de datos mediante la web de KEEPA¹.**



La web de Keepa muestra los últimos cambios de precio de los productos que se venden en Amazon, permite descargar una extensión para el navegador del usuario que da una visualización del histórico de precios, se puede hacer seguimiento de los precios de los productos que más nos interesen, etc. pero sobre todo te da la opción de descargar sus datos en formato CSV y Excel.

Desde su web, puedes descargar la información de los productos en función de la categoría del producto, reseñas, valoraciones de cada producto. En definitiva, muestra gran cantidad de información de todo tipo suficiente para llevar a cabo el presente proyecto.

A diferencia de la API, la descarga vía web tiene un plan de pago de 15€ mensuales con acceso a consultas ilimitadas con un número máximo de productos de 5 mil por búsqueda.

Dado al coste de las opciones disponibles para poder obtener los datos, nos hemos decantado por esta opción debido a que es una cuota más económica y que podemos aplicarlo al problema que queremos solventar. Aunque de cara al posible desarrollo del proyecto, no descartamos el uso de API.

¹ <https://keepa.com/#!>

Cabe resaltar que el hecho de extraer la información de esta forma surge tras valorar otras ideas como, por ejemplo, hacer **Web Scraping**. La idea inicial era utilizar esta técnica de extracción con Python para obtener información del website de Amazon. Pero dado que cada producto cuenta con unas características dinámicas, nos resultó imposible poder generar un código que generalice.

El repositorio de datos del que disponemos, lo hemos generado nosotros, descargando diariamente, a lo largo de un año, la información disponible en Keepa.

3.2 Proceso ETL

Es la parte de un proyecto de Data Science donde se procede a la extracción, transformación y carga de datos.

Es la capa “oculta” de un sistema Business Intelligence o Big Data. Es una capa tan trabajosa y ardua como imprescindible para el éxito de un proyecto BI, Big Data o Data Science.

Se aplica en diferentes ámbitos: desde proyectos Data Science a proyectos de integración de datos complejos: Data Warehouse, Data Lake...

En este tipo de procesos los datos deben estar actualizados, deben ser confiables (calidad), centralizados (repositorio único) y fácilmente consultables.

- **E: Extracción.** Procesos que establecen la conexión a las diferentes fuentes origen que realizan las extracciones de datos.
- **T: Transformación.** Procesos de validación, homogeneización, transformación y enriquecimiento del dato.
- **L: Load/Carga.** Carga o entrega de información en los repositorios destino de información.

3.3 Extracción de los datos.

De cara a nuestro proyecto, nosotros hemos seguido la siguiente dinámica para la recopilación de datos:



1. **Extracción de datos a partir de la aplicación de Keepa.** Como hemos comentado anteriormente, nuestro método para la obtención de los datos es una descarga que se hace a través de la web de Keepa. Para conseguir generar nuestra BBDD hemos descargado diariamente información en formato CSV. El nombre diario que contienen estos ficheros es “Best_Sellers_List.XXXX_XX_XX.products.csv”. Como hay mucha información dentro de Keepa para poder acotar decidimos descargar solo datos de la categoría “Electrónica” y de los 5.000 productos más vendidos diariamente.
2. **Automatización del proceso de descarga diaria.** Una vez identificada la información que queríamos recopilar, automatizamos el proceso a través de un script, que se ejecuta diariamente mediante la librería de Python denominada **SELENIUM**. El script lo que hace exactamente es mediante código Python conectarse a la web de Keepa, seleccionar la información (variables) que nos interesan, descargarla y guardarla en formato CSV. Este script lo hemos alojado

en nuestro pc y se ejecutaba gracias al programador de tareas de Windows, que estaba configurado todos los días a las 22:00 horas.

3. **Almacenamiento de la información.** Una vez descargada la información en local, la hemos trasladado a Google Drive, de forma manual y es allí donde hemos almacenado por días y semanas los archivos.
4. **Gestión de los datos.** Para poder tratar los datos obtenidos y almacenados en Google Drive hemos usado un notebook Jupyter, ya que nos permite gestionar tanto la importación de información como la gestión de los datos: limpieza, transformaciones, modelado de datos, etc.

3.4 Información disponible en el conjunto de datos.

Nuestro set de datos está formado inicialmente por:

- **Filas.** Cada fila corresponde a un producto vendido en la web/app de Amazon en el día en curso, teniendo en cuenta los 5.000 más vendidos.
- **Columnas.** Tenemos 134 columnas, variables. Dado a que son excesivas variables, dejamos un enlace donde se podrá consultar cada una de ellas: [variables.xlsx](#)

Al descargar la información de nuestra fuente de datos hemos decidido coger todas las variables disponibles, de esta forma, posteriormente, eliminaremos a la hora de limpiar/analizar los datos. Las variables que a priori valoramos como importantes de cara a nuestros modelos son las que a continuación detallamos, pero mantendremos todas hasta no avanzar con el análisis de todas.

VARIABLE	DESCRIPCIÓN	TIPO
Título	Título descriptivo del producto	OBJECT
Rango de ventas: Actual	El ranking indica la posición de ese producto en ventas de acuerdo a su categoría. Mientras menor es el número, más ventas y más rotación tiene el artículo.	FLOAT64
Rango de ventas: Subcategory Sales Ranks	Rango de ventas dentro de la categoría del producto	OBJECT
Reviews: Valoraciones	Puntuación del producto	FLOAT64
Reviews: Recuento de Revisiones	Nº comentarios que tiene el producto	FLOAT64
Review Count - Format Specific	Indica el número de comentarios que tiene un producto.	FLOAT64
Last Price Change	Última fecha en la que cambió pvp el producto	OBJECT

Last Update	Última fecha en la que se actualizó el producto	OBJECT
Last Offer Update	Última fecha en la que se actualizó una oferta del producto	OBJECT
Amazon: Actual	Precio más bajo por el cual Amazon está vendiendo el producto en "estado nuevo" y con stock	OBJECT
Nuevo: Actual	Precio más bajo por el cual, un tercer vendedor (FBA, FBM) o Amazon, está vendiendo un producto en "estado nuevo"	OBJECT
Usado: Actual	Precio del producto "usado" más bajo que el que vende Amazon Warehouse o un tercer vendedor. No incluye el coste de envío, el vendedor FBA debe agregarlo al precio.	OBJECT
Oferta flash: Actual	Precio del producto en oferta	OBJECT
Oferta flash: Upcoming Deal	Variable categórica que indica si el producto tendrá o no una próxima oferta	OBJECT
Oferta de almacén: Actual	Precio del producto "oferta almacén"	OBJECT
Tracking since	Fecha desde la que se le hace seguimiento al producto	OBJECT
Listed since	Fecha de creación del producto	OBJECT
ASIN	Identificador del producto	OBJECT
Product Group	Grupo dentro del cual podemos encasillar al producto	OBJECT

Considerando ya solo estas variables, vemos que tendremos que hacer transformaciones de fechas, ya que vienen en un formato "Object", cambios en las variables que son pvp, etc. pero lo iremos detallando en el próximo punto.

3.5 Análisis de los datos.

Para nuestro análisis hemos importado primero la información de las primeras 20 semanas del conjunto de datos y sobre estas hemos hecho la analítica, de esta forma partimos de un dataset de 134 variables y 694.882 registros. Nuestros pasos dados han sido:

- A. **Borrar columnas que tuvieran valores ausentes y valores únicos.** Las variables con valores únicos directamente las eliminamos como, por ejemplo: **Prime Eligible (Amazon offer)** o **Locale o Restricción MAP**, ya que la información que dan no nos aporta valor. Sin embargo, en las variables con valores ausentes hemos decidido primero ver el valor que pueden aportarnos y

decidir si procede directamente eliminarlas porque no tienen cabida en este estudio o en su caso valorar que tipo de imputación podríamos hacer para mantenerlas.

B. Por contexto de negocio, hemos borrado todas estas:

- **Manufacturer:** tiene muchos ausentes y es la misma columna que Brand.
- **Author, Edición, Number of Pages, Languages, Edicion, Publication Date:** estas hacen referencia a productos como libros y no tiene mucho sentido disponer de esta información en artículos de la categoría electrónica.
- **Number of Items:** hace referencia al número de artículos que contiene el producto, tiene muchos valores ausentes y consideramos que no nos es útil de cara al análisis.
- **Release Date:** indica la fecha de lanzamiento del producto. Consideramos que la fecha de lanzamiento no es un punto relevante para tener en cuenta en el análisis, principalmente porque no viene informada en la mayoría de los casos y es una variable que no podemos generar, extrapolar, etc. en función de los datos que tenemos.
- **Subscribe and Save Coupon: Percentage:** la variable no tiene datos fiables. Deriva de la variable Subscribe and Save así que nos quedamos con ésta última al poseer más información, aunque sea agregada.
- **Freq. Bought Together:** Indica productos complementarios, pero tiene muchos ausentes por eso consideramos la variable irrelevante.
- **Color:** consideramos que no aportan el suficiente valor.
- **Model:** consideramos que no aportan el suficiente valor.
- **Variation attributes:** consideramos que no aportan el suficiente valor.
- **Rango de ventas: Reference:** hace referencia a la categoría, la eliminamos ya que todos los productos corresponden a la categoría electrónica.

- **Binding:** igual a la variable product group.
- **Tamaño:** no aporta valor en nuestro conjunto de datos.
- **Item: Dimension, Length, Width, Height, Weight:** keepa es una plataforma destinada a la reventa para nuestro caso no nos interesan.
- **Listed since:** solo nos interesa desde cuando está rastreado el producto ya que obtenemos el precio y demás características.
- **Reacondicionado: Actual:** esta variable indica si el producto es reacondicionado, posee demasiados ausentes lo que hace difícil analizarla, además entendemos que posee una gran dependencia respecto al precio actual. Incluye todas las variables que derivan de ésta.
- **Trade-In:** indica la posibilidad de intercambiar el producto. Todos sus valores están ausentes. No influye en los objetivos perseguidos. Incluye todas las variables que derivan de ésta.
- **URL: Amazon y URL: Keepa:** son las urls de los productos, tanto en la plataforma Keepa como en Amazon. Las eliminamos porque no aportan nada de valor a nuestro modelo.
- **Product Code:** Eliminamos las variables relacionadas con EAN o códigos de productos similares ya que hemos conservado el identificador único ASIN ('Product Codes: EAN' 'Product Codes: UPC' 'Product Codes: PartNumber' 'Parent ASIN' 'Variation ASINs')

Eliminando todas estas variables, conseguimos reducir el dataset a 80 variables. Lo cual está muy bien para haber hecho solo un análisis de las variables dándole sentido de negocio.

C. **Antes de seguir eliminando, observamos los tipos de datos disponibles**, vemos que algunas variables aparecen como string siendo variables numéricas debido a que tienen la unidad monetaria (€) o están en porcentaje. Vamos a transformarlas en numéricas y a su vez en las filas que contienen el carácter '-' lo eliminaremos. Realizamos las siguientes

transformaciones:

- Definimos una función para la transformación de variables, en concreto string del tipo x.xx,xx € los pasamos a float.
- Reemplazar el símbolo del '-'
- Reemplazar el símbolo del '?'

D. **Creamos una nueva variable que llamaremos CATEGORY** a partir de la variable **RANGO DE VENTAS: SUBCATEGORY SALES RANKS**: con el objetivo de diferenciar los distintos productos que tenemos en la BBDD, ejemplo: móviles, tablets, cámaras, etc.

E. Pasamos a formato fecha las variables correspondientes: ya que en nuestro dataset tenemos mucha información sobre fechas para los diferentes cambios de precios de los productos.

F. **Reincidimos sobre el tema de los valores ausentes**: seguimos viendo que tenemos variables con muchos ausentes y por eso, generamos una función que nos cuente los ausentes en cada variable y así poder ir trabajando sobre estas. Por ejemplo:

a. **Para algunas variables imputamos los valores nulos por la media y/o mediana**. Analizamos una serie de variables en grupo, porque estaban correlacionadas y decidimos imputar los nulos por la media de la categoría de los productos. Como seguimos teniendo nulos ya que hay productos sin categoría, hemos calculado la mediana de la variable y sustituido por ese dato. Algunas variables con esas casuísticas han sido: **Rango de ventas: Actual, Rango de ventas: 30 days avg., Rango de ventas: 90 days avg., Rango de ventas: 180 days avg., etc.**

b. **Creamos dummies**. Por ejemplo: **Usado: Actual u Oferta flash** las hemos transformado de categóricas a numéricas

c. **Para otras hemos hecho cálculos que nos ayuden a solventar los nulos**. Por ejemplo: La variable Amazon: Actual, que es el precio al que

vende Amazon y distribuye, consideramos que es importante no eliminarla, pero tiene demasiados valores ausentes por esto se nos ocurre imputar en función de la variable Nuevo: Actual_x. Restamos a la primera variable la segunda obteniendo la diferencia, aplicamos la media a esos valores y sumamos esa media al valor de la variable Nuevo: Actual_x.

G. **Creamos nuevas variables.** Generamos la variable **fecha_y** y **fecha_x**, que son las fechas de los precios de los productos en las distintas semanas del análisis y como consecuencia generamos la variable '**diferencia**', que es el resultado de restar el precio actual respecto al de la semana pasada. Derivada de ésta creamos '**variación**', que es lo mismo que la variable diferencia, pero en porcentaje. La idea es que a partir de la variable 'diferencia' podamos crear la variable '**target**'.

3.6 Set de datos final

Finalmente, una vez analizados todos los datos y tras haber hecho las transformaciones que hemos considerado necesarias, obtenemos un set de datos mucho más reducido al que teníamos inicialmente. Ahora tenemos un total de 34 características y 3 etiquetas.

Estas 3 etiquetas serían:

- **Etiqueta -1:** indica que el precio del producto **cae**.
- **Etiqueta 0:** indica que el precio del producto **se mantiene**.
- **Etiqueta 1:** indica que el precio del producto **aumenta**.

Estas etiquetas nos ayudarán a identificar las subidas/bajadas de precios o su estancamiento, para así poder indicar al consumidor si es buen momento o no para proceder a la compra.

Estos datos los vamos a tratar con **modelos de clasificación**. Consideramos que por la naturaleza de los datos con estos conseguiremos una predicción menos volátil.

Descartamos la aplicación de **modelos de forecasting**, ya que no vemos una continuidad de los productos de la BBDD que tenemos acumulada, un año. Lo que hemos extraído diariamente, es información de los 5000 productos más vendidos de Amazon de la categoría de electrónica, al día, esto implica que un producto puede estar

un cierto tiempo dentro de los 5000 productos más vendidos, pero durante otro periodo no estarlo.

En definitiva, no tenemos una serie temporal estable y reiteramos, por esta cuestión, no consideramos aplicar forecasting pero si clasificación, lo cual también nos permite introducir otras variables al modelo y ver la importancia de estas en relación con el target.

También, otra idea que nos impulsa a usar modelos de clasificación es que no queremos predecir el precio exacto del producto, nuestra finalidad es pronosticar la tendencia del precio en los diferentes productos.

Consideramos que categorizando el target y planteando un problema de clasificación, evitamos que la magnitud del error sea mayor que en la predicción de una variable continua.

4

Aplicación de Modelos Machine Learning.

Una vez terminado el análisis y la limpieza de los datos y ahora vamos a probar una serie de modelos de clasificación que nos ayudarán a encontrar la solución óptima a nuestro problema. Para esto vamos a utilizar principalmente la librería SciKit-Learn.

4.1 Métricas.

Antes de probar los modelos, queremos indicar previamente, las métricas que vamos a valorar para considerar si un modelo es bueno o no. Serán:

- **La matriz de confusión:** describe los tipos de errores que se cometen en un problema de clasificación. Nos muestra también las predicciones correctas e incorrectas, es decir, los datos que hemos clasificado correctamente y los que no. Concretamente, con esta podemos valorar el rendimiento del modelo. La estructura de la matriz es la siguiente:

		OBSERVACIONES	
		Clase 1	Clase 2
PREDICCIONES	Clase 1	VP	FP
	Clase 2	FN	VN

donde:

- ❖ Verdadero positivo (VP): el dato observado es positivo y se predice positivo.
 - ❖ Falso negativo (FN): el dato observado es positivo y se predice negativo.
 - ❖ Verdadero negativo (VN): el dato observado es negativo y se predice negativo.
 - ❖ Falso positivo (FP): el dato observado es negativo y se predice positivo.
- **Accuracy:** la tasa de clasificación o de precisión en los resultados y también depende de los cuatro términos definidos anteriormente: VP, FN, FP y VN. La

fórmula que utiliza el algoritmo internamente para su cálculo es la siguiente:

$$\text{accuracy} = \text{VP} + \text{VN} / \text{VP} + \text{FN} + \text{FP} + \text{VN}$$

- **Precisión:** indica el valor de dividir el número total de datos positivos correctamente clasificados entre el número de datos positivos predichos. Nos da la calidad de la predicción: ¿qué porcentaje de los que hemos dicho que son la clase positiva, en realidad lo son? La fórmula, por tanto, es la siguiente:

$$\text{precision} = \text{VP} / \text{VP} + \text{FP}$$

- **Recall:** indica el valor de dividir el número total de datos positivos correctamente clasificados entre el número total de los datos positivos, bien y mal clasificados. Un recall alto muestra que la clase es correctamente reconocida (esto es porque el valor de los datos positivos mal clasificados (FN) es pequeño). La fórmula es la siguiente:

$$\text{recall} = \text{VP} / \text{VP} + \text{FN}$$

- **F1:** combina Precisión y Recall en una sola medida. Si tenemos:
 - ❖ **Recall alto y precision baja:** los datos positivos se reconocen correctamente, pero, tenemos muchos falsos positivos, es decir, datos que no son positivos se predicen como positivos.
 - ❖ **Recall bajo y alta precision:** los datos que se predicen como positivos son realmente positivos, pero existen otros muchos datos positivos que no están bien clasificados.

En resumen, lo que necesitamos es obtener un alto recall y una alta precision, es decir, que los falsos negativos y positivos sean pocos.

4.2 Modelos Machine Learning aplicados.

Como indicamos en la sección anterior, hemos considerado tratar nuestros datos como un problema de clasificación, el principal motivo es que no tenemos un histórico estable del precio de cada producto.

En primer lugar, tras seleccionar los modelos con los que íbamos a trabajar, lo primero que hemos hecho es probarlos todos de forma natural, ya que la idea es tras ver los valores de las primeras ejecuciones que podemos hacer para conseguir mejores resultados.

A priori, dividimos el dataset en training y test tomando un 70% de las muestras para el primero y el 30% restante para test.

A continuación, comentaremos brevemente, los modelos que hemos empleado.

4.2.1 KNN.

Se utiliza para clasificar un conjunto de datos etiquetados. Como la idea es clasificar cuando sube, baja o se mantiene el precio de un producto, consideramos que es adecuado para nuestros datos.

Al aplicar este modelo sin ningún tipo de parámetro adicional obtenemos los siguientes resultados:

Etiqueta	Precision	Recall	F1-score
-1	0,22	0,39	0,28
0	0,61	0,52	0,56
1	0,28	0,23	0,26
accuracy			0,42

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	2.233	2.135	1.315
True 0	5.189	9.403	3.633
True 1	2.697	3.791	1.969

Claramente, los resultados obtenidos son muy deficientes, ni el accuracy, ni la precisión, ni el recall nos indican que la predicción con este modelo es buena, no acierta en muchos de los supuestos.

4.2.2 DecisionTreeClassifier.

Aplicamos este modelo porque no requiere de datos complejos para dar unos buenos resultados, es fácil de entender y de graficar. A priori, aplicamos el modelo sin ningún tipo de parametrización y obtenemos los siguientes resultados:

Etiqueta	Precision	Recall	F1-score
-1	1,00	1,00	1,00
0	1,00	1,00	1,00
1	1,00	1,00	1,00
accuracy			1,00

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1

True -1	5.683	0	0
True 0	0	18.225	0
True 1	0	0	8.457

Los resultados obtenidos nos indican un modelo perfecto lo que nos hace pensar que está sucediendo algo anómalo. Podría estarse produciendo un sobreajuste en los datos (overfitting).

4.2.3 Naive Bayes.

Probamos este modelo porque es un algoritmo rápido y sencillo para predecir clases de un conjunto de datos y porque sabemos que funciona mejor que otros algoritmos cuando hay más de una etiqueta y en este caso tenemos 3. Los resultados obtenidos son los siguientes:

Etiqueta	Precision	Recall	F1-score
-1	0,50	0,14	0,22
0	0,64	0,93	0,76
1	0,53	0,28	0,36
accuracy			0,62

MATRIZ DE CONFUSIÓN

	Predicted -1	Predicted 0	Predicted 1
True -1	792	3.830	1.061
True 0	362	16.885	978
True 1	445	5.674	2.338

Los resultados obtenidos para este modelo son mejores que el modelo de k-vecinos, pero están lejos al objetivo marcado que sería cercano al 0,9 en accuracy, recall y precisión, por lo que seguiremos probando modelos.

4.2.4 SGD.

Este estimador implementa modelos lineales regularizados con aprendizaje de descenso de gradiente estocástico (SGD). Al probar este modelo, los resultados obtenidos son los siguientes:

Etiqueta	Precision	Recall	F1-score
-1	0,61	0,31	0,41
0	0,04	0,68	0,07
1	0,78	0,33	0,46
accuracy			0,33

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1

True -1	3.495	83	2.105
True 0	6.163	649	11.413
True 1	1.590	229	6.638

Fijándonos en las métricas habituales los resultados obtenidos no son buenos, la tasa de acierto mostrada en la matriz de confusión es demasiado baja. Este modelo arroja peores resultados que KNN probablemente uno de los motivos pueda ser porque no hemos escalado los datos.

Se observa que la precisión aumenta en las clases minoritarias, un dato a tener en cuenta para la predicción de la etiqueta.

4.2.5 Regresión logística.

Utilizado para predecir el resultado de una variable categórica. En este caso la variable categórica (subida, bajada, mantiene) la hemos pasado a numérica (1, -1, 0). Probamos el modelo y obtenemos los siguientes resultados:

Etiqueta	Precision	Recall	F1-score
-1	0,14	0,31	0,20
0	0,91	0,59	0,72
1	0,07	0,36	0,11
accuracy			0,55

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	816	4.576	291
True 0	918	16.564	743
True 1	931	6.955	541

Como en el caso de otros de los modelos anteriores, este no nos arroja buenos resultados.

4.2.6 XGB.

Se trata de un algoritmo paralelizable. Esto nos permite usar de manera óptima toda la potencia de procesamiento de la que disponemos. Para este caso los resultados obtenidos son los siguientes:

Etiqueta	Precision	Recall	F1-score
-1	1,00	1,00	1,00
0	1,00	1,00	1,00
1	1,00	1,00	1,00
accuracy			1,00

MATRIZ DE CONFUSIÓN

	Predicted -1	Predicted 0	Predicted 1
True -1	5.683	0	0
True 0	0	18.225	0
True 1	0	0	8.457

Sucede algo parecido al caso del árbol de decisión. Las métricas toman el valor de 1 que sería un modelo ideal por lo que sospechamos que debe estar ocurriendo algo parecido al modelo de árbol de decisión, overfitting.

4.2.7 Máquinas de vector de soporte (SVM)

El uso de este algoritmo se debe a que está enfocado a resolver problemas de clasificación como es el caso.

Los parámetros más importantes utilizados por defecto son: el parámetro de regularización que utilizamos es uno y la penalización es l2 al cuadrado. El tipo de kernel utilizado en el algoritmo es 'rbf' y el grado de la función es 3. La tolerancia para el criterio de parada es 1e-3. No establecemos un límite de iteraciones. Los resultados obtenidos son los siguientes:

Etiqueta	Precision	Recall	F1-score
-1	0,34	0,03	0,06
0	0,57	0,99	0,72
1	0,35	0,00	0,01
accuracy			0,56

La matriz de confusión:

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	174	5.503	6
True 0	166	18.021	38
True 1	167	8.266	24

A nivel de predicción, los resultados son malos, pues tenemos muchos falsos positivos y negativos. El modelo no nos arroja buenos resultados.

4.2.8 AdaBoost

Consiste en crear varios predictores sencillos en secuencia. Se realizan ajustes de forma iterativa.

Comienza ajustando un clasificador en el conjunto de datos original y luego ajusta copias adicionales del clasificador en el mismo conjunto de datos, pero donde los pesos de las instancias clasificadas incorrectamente se ajustan de modo que los clasificadores posteriores se centren más en casos difíciles. No utilizamos un objeto estimador base. El número de estimadores es de 50 y la tasa de aprendizaje es de 1. El algoritmo de impulso es 'Sammer.R' generalmente converge más rápido que 'Samme'. Los resultados obtenidos son:

Etiqueta	Precision	Recall	F1-score
-1	1,00	1,00	1,00
0	1,00	1,00	1,00

1	1,00	1,00	1,00
accuracy			1,00

La matriz de confusión:

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	5.683	0	0
True 0	0	18.225	0
True 1	0	0	8.457

Obtenemos los mismos resultados que el árbol de decisión y XGB. Sospechamos que los datos de train y test son muy parecidos por eso generaliza tan bien en los datos de test.

4.2.9 Bagging.

Este tipo de algoritmos pretende reducir la varianza. Utiliza algoritmos simples utilizados en paralelo. Intenta aprovecharse de la independencia que hay entre los algoritmos simples, ya que el error se puede reducir bastante al promediar las salidas de los modelos simples.

El estimador base (base_estimator) utilizado es el un árbol de decisión, el número de estimadores (n_estimators) base en el conjunto es de 10 y el número máximo de características (max_features) que se extraerán para entrenar cada estimador será uno.

Los resultados obtenidos:

Etiqueta	Precision	Recall	F1-score
----------	-----------	--------	----------

-1	1,00	1,00	1,00
0	1,00	1,00	1,00
1	1,00	1,00	1,00
accuracy			1,00

La matriz de confusión:

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	5.683	0	0
True 0	0	18.225	0
True 1	0	0	8.457

Los resultados obtenidos son los mismos que el caso anterior por el motivo anteriormente explicado. Los datos de train y test son similares.

4.2.10 GradientBoostig.

La ventaja de este modelo es que tiene una gran facilidad de uso que puede manejar valores faltantes, valores atípicos y valores categóricos de alta cardinalidad en sus funciones sin ningún tratamiento especial.

La función de pérdida que utilizaremos es '*log_loss*' se refiere a la desviación binomial y multinomial, la misma que se usa en la regresión logística. Es una buena opción para la clasificación con salidas probabilísticas. La tasa de aprendizaje reduce la contribución de cada árbol en este caso toma el valor de 0,1.

Número de estimadores: El aumento de gradiente es bastante resistente al sobreajuste, por lo que un gran número generalmente da como resultado un mejor rendimiento. Toma como valor 100. Los resultados obtenidos son:

Etiqueta	Precision	Recall	F1-score
-1	1,00	1,00	1,00
0	1,00	1,00	1,00
1	1,00	1,00	1,00
accuracy			1,00

La matriz de confusión es:

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	5.683	0	0
True 0	0	18.225	0
True 1	0	0	8.457

4.2.11 ExtraTree.

Más aleatorio que Random Forest. Además de considerar un subconjunto de las características predictivas para cada uno de los árboles a crear, a la hora de escoger una característica y un valor de corte (threshold) para dividir cada nodo, en lugar de escoger el threshold que mejor divida cada característica (y escoger la característica y valor de corte que minimice el criterio de impureza), se genera un valor de corte aleatorio para cada característica planteada, escogiéndose como regla de división el mejor de ellos.

Algunos de los parámetros más importantes que hemos utilizado es la función para medir la calidad de la división (criterion) que para este caso es la impureza de Gini (gini) y la estrategia para elegir la división en cada nodo (splitter) es aleatoria.

Etiqueta	Precision	Recall	F1-score
-1	0,65	0,48	0,55
0	0,71	0,81	0,76
1	0,65	0,64	0,64
accuracy			0,68

La matriz de confusión obtenida es la siguiente:

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	3.684	1.264	735
True 0	2.854	12.964	2.407
True 1	1.110	1.835	5.512

Como en otros modelos las métricas de precision, recall y f1 score se obtiene de la clase mayoritaria (etiqueta 0: el precio se mantiene).

4.3 Resultados tras aplicación modelos.

Haciendo una comparativa de los modelos probados, tenemos los siguientes resultados en cada uno de ellos:

Model	Score train	Score test
KNN	0,857	0,420

ÁRBOL	1,00	1,00
RF	1,00	0,999
NB	0,523	0,618
SGD	0,340	0,206
XGB	1,00	1,00
LR	0,461	0,554
ADA	1,00	1,00
BAGGING	1,00	1,00
GRADIENT	1,00	1,00
EXTRA	1,00	0,685

Observamos que los modelos en train tienen un score aceptable mientras que, en test, el score, en ocasiones, cae notablemente. Esto se debe a que el modelo se ajusta demasiado a los datos de entrenamiento.

Otra de las conclusiones a las que hemos llegado es que la proporción de etiquetas tanto para train como para test son parecidas en la clase mayoritaria o al menos tanto en test como train hay un mayor número de muestras de la etiqueta 0:

Etiqueta	Train	Test
-1	31.732	5.683
0	45.989	18.225
1	25.995	8.457

En porcentaje:

Etiqueta	Train	Test
-1	31%	18%
0	44%	56%
1	25%	26%

Concluyendo, vemos que los modelos no están funcionando correctamente y consideramos que eso se debe fundamentalmente a que tenemos un desbalance en las etiquetas, esto resulta lógico ya que por lo general el precio de un producto no suele

variar drásticamente y en poco tiempo.

Realizando un conteo de las etiquetas, obtenemos los siguientes resultados:

- 45.989 productos no varían su precio (etiqueta 0)
- 31.732 productos bajan de precio (etiqueta -1)
- 25.995 productos suben de precio (etiqueta 1)

Con los resultados obtenidos y observando solo el accuracy, entendemos que no nos podemos fiar solo de esta métrica, pues puede ser una métrica engañosa debido a la inequidad de los datos. Si solo observamos esta medida, podríamos pensar que el modelo es mejor de lo que en realidad es, como es el caso del Árbol de Decisión o del XGBOST.

Apoyándonos en las otras medidas, precisión y recall, nos extraña conseguir predicciones tan perfectas sin apenas tocar los modelos, con lo que consideramos que tampoco nos podemos fiar al cien por cien de estas, lo cual nos lleva a pensar que estamos teniendo un sobreajuste en los datos.

Como primera medida se nos ocurre, normalizar los datos para ver si conseguimos una mejora en las predicciones, pero pese a hacerlo, no vemos mejoras. Por lo que decidimos aplicar una serie de técnicas de balanceo de datos sobre el modelo de Regresión Logística, que inicialmente tiene un accuracy muy bajo.

4.4 Posibles soluciones al Balanceo de datos.

Como indicamos, vamos a aplicar técnicas para solucionar el problema de equidad en los datos, sobre el modelo de Regresión Logística. Estas técnicas son:

- Penalización para compensar / Métricas
- Subsampling
- Oversamplig
- Combinación
- Balanced Ensemble

4.4.1 Penalización para compensar el desbalanceo.

La idea con esta medida es conseguir un mejor resultado ajustando algún parámetro del modelo.

‘Tuneamos’ el modelo inicial y añadimos el parámetro **weight = “balanced”** y con esto el algoritmo se encargará de equilibrar a la clase minoritaria durante el entrenamiento.

El algoritmo para resolver el problema de optimización que utilizaremos será ‘newton-cg’ ya que se trata de un problema multiclase. Utilizaremos regularización ‘l2’ la única que es soportada por el algoritmo ‘newton-cg’.

Los resultados obtenidos son los siguientes:

Etiqueta	Precision	Recall	F1-score
-1	1,00	0,84	0,91
0	0,91	1,00	0,95
1	0,99	0,88	0,93
accuracy			0,94

MATRIZ DE CONFUSIÓN			
Etiqueta	Predicted -1	Predicted 0	Predicted 1
True -1	4.792	890	1
True 0	4	18.167	54
True 1	0	1.008	7.449

Comparamos los modelos, con penalización y el modelo en bruto:

Etiqueta	Reg. Log. Con penalización			Reg. Log.		
	Precision	Recall	F1-score	Precision	Recall	F1-score

-1	1,00	0,84	0,91	0,14	0,31	0,20
0	0,91	1,00	0,95	0,91	0,59	0,72
1	0,99	0,88	0,93	0,07	0,36	0,11
accuracy	0,94			0,55		

Observamos que mejoramos notablemente los resultados anteriores. Mejora tanto el accuracy como la precisión y el recall, son cercanos al 0,9 e incluso en algunos casos lo superan.

Ahora comparamos la matriz de confusión:

	MATRIZ DE CONFUSIÓN					
	Reg. Log. Con penalización			Reg. Log.		
Etiqueta	Predicted -1	Predicted 0	Predicted 1	Predicted -1	Predicted 0	Predicted 1
True -1	4.792	890	1	816	4.576	291
True 0	4	18.167	54	918	16.564	743
True 1	0	1.008	7.449	931	6.955	541

Se aprecia un aumento notable de verdaderos positivos y verdaderos negativos. El método de 'newton-cg' equilibra las clases minoritarias mediante la sucesión iterativa en la búsqueda de raíces.

4.4.2 Subsampling.

La idea es usar un algoritmo que nos ayude a reducir la clase mayoritaria. También podríamos reducir de forma manual esta clase y conseguir equiparar las muestras, pero tras haberlo probado consideramos que es mejor hacerlo con un algoritmo.

Optamos por usar el algoritmo: **NearMiss** que puede ayudar a equilibrar un conjunto de datos desequilibrado. Este algoritmo lo que hace es observar la distribución de las clases y eliminar aleatoriamente muestras de la clase más grande. Cuando dos puntos que pertenecen a diferentes clases están muy cerca uno del otro en la distribución,

elimina el punto de datos de la clase más grande, tratando así de equilibrar la distribución.

Los pasos que sigue este algoritmo son:

- Calcula la distancia entre todos los puntos de la clase más grande con los puntos de la clase más pequeña. Para facilitar el proceso de submuestreo.
- Selecciona las instancias de la clase más grande que tienen la distancia más corta con la clase más pequeña. Estas “N” clases deben almacenarse para su eliminación.
- Si hay “M” instancias de la clase más pequeña, el algoritmo devolverá $M \times N$ instancias de la clase más grande.

Dentro de este podemos probar como parámetro, 3 versiones:

- Versión 1: en la primera versión, los datos se equilibran calculando la distancia mínima promedio entre la distribución más grande y las tres distribuciones más pequeñas más cercanas.
- Versión 2: aquí, los datos se equilibran calculando la distancia mínima promedio entre la distribución más grande y las tres distribuciones más pequeñas más alejadas.
- Versión 3: aquí, se consideran las instancias de clase más pequeñas y se almacenan “M” vecinos. Luego se toma la distancia entre esta y la distribución más grande y se elimina la distancia más grande.

Volviendo a nuestros datos, la idea es conseguir una distribución más equitativa de la clase mayoritaria:

Etiqueta	Distribución antes del submuestreo	Distribución después del submuestreo
-1	31.732	31.732
0	45.989	25.995
1	25.995	25.995

Es decir, lo que queremos conseguir es reducir la clase 0. Esto lo hemos conseguido

“Tuneando” el algoritmo modificando los siguientes parámetros:

- **Sampling_strategy:** con este indicamos que queremos reducir la clase mayoritaria. Nosotros usaremos la opción: **majority**, porque lo que queremos conseguir es reducir la clase 0, la mayoritaria. Cabe resaltar que tenemos otras opciones:
 - **not minority:** remuestrea todas las clases excepto la clase minoritaria
 - **not majority:** volver a muestrear todas las clases excepto la clase mayoritaria
 - **all:** remuestrear todas las clases
 - **auto:** equivalente a 'not minority'
- **N_neighbors:** indicamos el número de vecinos para poder hacer los cálculos. Tras probar consideramos que el óptimo es 11
- **Versión:** explicada previamente, usaremos la opción 2.

Una vez que la clase mayoritaria se ha equiparado a la minoritaria, probamos nuevamente sobre estos datos el modelo de Regresión Logística y obtenemos los siguientes resultados:

Etiqueta	Precision	Recall	F1-score
-1	0,99	0,88	0,93
0	0,93	0,99	0,96
1	0,99	0,91	0,95
accuracy			0,95

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	4.990	692	1
True 0	54	18.123	48
True 1	1	776	7.680

Comparamos ambos modelos, con la técnica de balanceo de clases y sin el:

	Reg. Log. con subsampling			Reg. Log.		
Etiqueta	Precision	Recall	F1-score	Precision	Recall	F1-score
-1	0,99	0,88	0,93	0,14	0,31	0,20
0	0,93	0,99	0,96	0,91	0,59	0,72
1	0,99	0,91	0,95	0,07	0,36	0,11
accuracy	0,95			0,55		

Observamos también la matriz de confusión:

	MATRIZ DE CONFUSIÓN					
	Reg. Log. con subsampling			Reg. Log.		
	Predicted -1	Predicted 0	Predicted 1	Predicted -1	Predicted 0	Predicted 1
True 1	4.990	692	1	816	4.576	291
True 0	54	18.123	48	918	16.564	743
True 1	1	776	7.680	931	6.955	541

Apreciamos que tanto a nivel de accuracy, recall y precision los resultados son bastante mejores.

4.4.3 Oversampling de la clase minoritaria.

Con esta técnica queremos generar nuevas muestras sintéticas en la clase minoritaria. Para eso usaremos el algoritmo: **RandomOverSampler**. Al igual que en el caso del subsampling, no es necesario usar un algoritmo para aplicar este remuestreo, podríamos limitar el dataset de forma aleatoria y manual.

Este algoritmo, es una práctica mejor que el submuestreo/subsampling ya que no

perdemos datos, sino que se generan nuevos datos, lo que puede resultar bueno para el modelo.

Como en subsampling vamos a indicar al algoritmo con el parámetro: **Sampling_strategy** que ajuste las clases, pero esta vez necesitamos que aumente la clase minoritaria, de forma que usaremos la opción **minority**:

Etiquetas	Distribución antes sobremuestreo	Distribución después sobremuestreo
-1	31.732	31.732
0	45.989	45.989
1	25.995	45.989

Con las etiquetas algo más equiparadas, aplicamos el modelo de Regresión Logística y obtenemos los siguientes resultados:

Etiqueta	Precision	Recall	F1-score
-1	1,00	0,83	0,91
0	0,91	1,00	0,95
1	1,00	0,89	0,94
accuracy			0,94

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	4.704	979	0
True 0	7	18.181	37
True 1	0	922	7.535

Comparamos ambos modelos, con la técnica de balanceo y sin el:

	Reg. Log. Oversampling			Reg. Log.		
Etiqueta	Precision	Recall	F1-score	Precision	Recall	F1-score
-1	1,00	0,83	0,91	0,14	0,31	0,20
0	0,91	1,00	0,95	0,91	0,59	0,72
1	1,00	0,89	0,94	0,07	0,36	0,11
accuracy	0,94			0,55		

	MATRIZ DE CONFUSIÓN					
	Reg. Log. Oversampling			Reg. Log.		
	Predicted -1	Predicted 0	Predicted 1	Predicted -1	Predicted 0	Predicted 1
True 1	4.704	979	0	816	4.576	291
True 0	7	18.181	37	918	16.564	743
True 1	0	922	7.535	931	6.955	541

Al igual que en subsampling, vemos una gran mejora al aplicar oversampling. Mejoran todas las métricas, lo cual nos indica que vamos por buen camino.

4.4.4 Resampling con Smote-Tomek.

Ahora probaremos una técnica que consiste en aplicar simultáneamente un algoritmo de subsampling y otro de oversampling a la vez al dataset.

Usaremos **SMOTE** para oversampling, que busca los puntos vecinos cercanos y agrega puntos “en línea recta” entre ellos. Y usaremos **TOMEK** para undersampling, que quita los de distinta clase que sean vecinos cercanos y deja ver la mejor decisión.

SMOTE-Tomek es un método híbrido que mezcla los dos métodos anteriores. Su proceso es:

- 1) **Inicio de SMOTE:** primero elige datos aleatorios de la clase minoritaria.
- 2) Calcula la distancia entre los datos aleatorios y sus k-vecinos más cercanos.
- 3) Multiplica la diferencia con un número aleatorio entre 0 y 1, luego agrega el resultado a la clase minoritaria como una muestra sintética.
- 4) **Fin de SMOTE.** Repite los pasos 2 y 3 hasta que se alcance la proporción deseada de clase minoritaria.
- 5) **Inicio de TOMEK.** Elige datos aleatorios de la clase mayoritaria.
- 6) Si el vecino más cercano de los datos aleatorios son los datos de la clase minoritaria (es decir, cree el enlace Tomek), elimine el enlace Tomek.

Al aplicar este algoritmo nuestra distribución de las etiquetas varía de la siguiente forma:

Etiquetas	Distribución antes Smote-Tomek	Distribución después Smote-Tomek
-1	31.732	44.093
0	45.989	44.024
1	25.995	43.924

En el algoritmo, esta vez hemos considerado usar **Sampling_strategy: 'auto'** que quiere decir, que va a muestrear todas las clases excepto la clase mayoritaria. Hecho esto, los resultados obtenidos son:

Etiqueta	Precision	Recall	F1-score
-1	1,00	0,85	0,92
0	0,91	1,00	0,95
1	1,00	0,89	0,94
accuracy			0,95

MATRIZ DE CONFUSIÓN			
	Predicted -1	Predicted 0	Predicted 1
True -1	4.857	825	1
True 0	4	18.197	24
True 1	0	917	7.540

Comparamos ambos modelos, con la técnica de balanceo y sin esta, y vemos que efectivamente también conseguimos mejorar el modelo de Regresión Logística inicial, en todas las métricas.

	Reg. Log. Smote-Tomek			Reg. Log.		
Etiqueta	Precision	Recall	F1-score	Precision	Recall	F1-score
-1	1,00	0,85	0,92	0,14	0,31	0,20
0	0,91	1,00	0,95	0,91	0,59	0,72
1	1,00	0,89	0,94	0,07	0,36	0,11
accuracy	0,95			0,55		

MATRIZ DE CONFUSIÓN						
	Reg. Log. Smote-Tomek			Reg. Log.		
	Predicted -1	Predicted 0	Predicted 1	Predicted -1	Predicted 0	Predicted 1
True 1	4.857	825	1	816	4.576	291
True 0	4	18.197	24	918	16.564	743

True 1	0	917	7.540	931	6.955	541
--------	---	-----	-------	-----	-------	-----

4.4.5 Ensamble de Modelos con Balanceo.

Para esta estrategia usaremos un Clasificador de Ensamble que utiliza Bagging y el modelo será un DecisionTree. La idea de combinar surge con el fin de conseguir una mayor estabilidad y predicción.

Al aplicar este algoritmo conseguimos los siguientes resultados:

Etiqueta	Precision	Recall	F1-score
-1	1,00	1,00	1,00
0	1,00	1,00	1,00
1	1,00	1,00	1,00
accuracy			1,00

MATRIZ DE CONFUSIÓN			
Etiqueta	Predicted -1	Predicted 0	Predicted 1
True -1	5.683	0	0
True 0	0	18.225	0
True 1	0	917	8.457

Al igual que con el resto de las soluciones propuestas, con este algoritmo conseguimos nuevamente mejorar todas las métricas.

	Ensamble			Regresión Logística		
Etiqueta	Precision	Recall	F1-score	Precision	Recall	F1-score

-1	1,00	1,00	1,00	0,14	0,31	0,20
0	1,00	1,00	1,00	0,91	0,59	0,72
1	1,00	1,00	1,00	0,07	0,36	0,11
accuracy	1,00			0,55		

	Ensamble			Regresión Logística		
	Predicted -1	Predicted 0	Predicted 1	Predicted -1	Predicted 0	Predicted 1
True 1	5.683	0	0	816	4.576	291
True 0	0	18.225	0	918	16.564	743
True 1	0	917	8.457	931	6.955	541

4.4.6 Conclusión tras aplicación técnicas balanceo de datos.

Tras haber probado las técnicas para solventar los problemas de desbalanceo de datos, los resultados obtenidos son muy favorables.

MODEL	SCORE_TRAIN	SCORE_TEST
REGRESIÓN LOGÍSTICA	0.460671	0.554766
PENALIZADO	0.948889	0.939533

OVERSAMPLING	0.947923	0.951429
SUBSAMPLING	0.943424	0.939904
SMOTEK	0.954188	0.945280
ENSAMBLE	1.000000	1.000000

Finalmente vamos a optar por quedarnos con el modelo de SMOTEK para el balanceo de datos posteriormente aplicaremos regresión logística porque consideramos que a nivel de métricas es el que mejor resultado nos aporta. El ensamble nos da unos buenos resultados, pero sospechamos que existe algún tipo de anomalía ya que los resultados del modelo son totalmente ideales.

Como conclusión de este apartado queda pendiente la puesta en producción del modelo con datos que no haya ‘visto’ nunca.

5

Implantación, monetización y retorno del Proyecto.

5.1 Implantación

Para la implantación del proyecto nos inclinamos por 3 puntos que consideramos los más importantes.

5.2 Datos fiables.

La calidad de los datos es esencial para que nuestra herramienta pueda desarrollar su trabajo con eficiencia. Además, que estamos optando por una modalidad de aprendizaje supervisado, estos datos de origen los etiquetamos para que el algoritmo aprenda a predecir la etiqueta de salida correcta. De igual manera tenemos que contar con datos 100% fiables que como bien mencionamos en apartados anteriores son obtenidos diariamente de una de las herramientas de extracción como lo es Keepa.



5.3 Plataforma integrada.

Lo vemos como la inversión más rentable en un primer proyecto de machine learning para llevarlo a cabo. Ya que tiene que generar alta confiabilidad dichas herramientas además de que estén totalmente integradas, para nuestro caso hemos hecho uso de Google Cloud como plataforma de almacenamiento de datos ya que manejamos volumen de 1 año para más de 5000 productos, dicha plataforma es donde contenemos grueso del proyecto acoplado, además de ello no descartamos de ninguna manera un cambio a otros servicios como el bien conocido AWS, Azure, IBM Cloud, o cualquier

otro que nos brinden dicho servicio para almacenamiento masivo de datos además de la facilidad y practicidad a la hora de darle el uso necesario o requerido.

5.4 Buscar siempre la simplicidad.

Nos hemos enfocado en que sea siempre lo más simple posible, hemos enfatizado en buscar la simplicidad en cualquier ámbito del proyecto y evitar construir complejas y costosas redes neuronales que incluso generan al usuario una mala experiencia, resumidas cuentas, mientras más simple y amigable sea, pues mejor será la experiencia.

5.5 Monetización

Sabemos que las fuentes son el principal pilar para una buena monetización de los datos, porque es el origen de toda la información. Esta primera parte es crucial y conlleva tres pasos importantes:

I. Acceso abierto

Permitir el acceso gratuito a un cierto conjunto de información/datos y así poder publicitar nuestra Herramienta e invitar a futuros clientes a inscribirse posteriormente al modelo de pago que se ofrecerá.

II. Data as a Service

En este modelo, puede haber una producción regular de contenidos o bien permitimos un acceso a búsqueda por semana o mes, vía app, para que los usuarios puedan interactuar con la información más directa y rápidamente.

III. Pago por uso

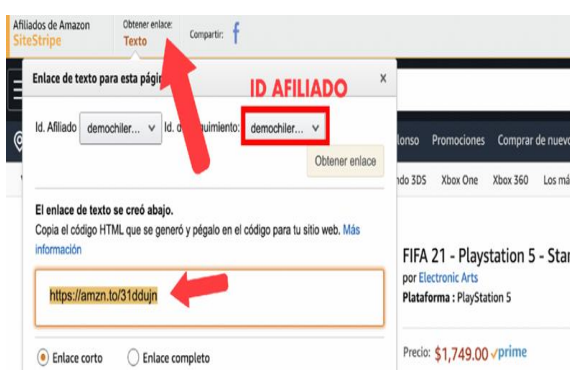
Este último sistema es más habitual en muchas herramientas no solo de este ámbito o tipo, se usa en muchos sectores. Se plantean suscripciones a la información tiempo, por interacción o por la información que está accesible, este

es el pilar en el cual queremos enfatizar en nuestro proyecto además de otras opciones.

Como bien se menciona el pago por uso, se proporcionará un acceso abierto donde el usuario podrá hacer una consulta cada determinado tiempo, aunque tendrá deshabilitados ciertas funciones como seguir, colocar alarma, incluir a favoritos, entre otras. Dichas funciones van totalmente relacionadas a los resultados que arrojan nuestras predicciones de manera que le permite al usuario poder ejecutar la función que desee según el resultado que se le muestro, como ejemplo podemos decir que un usuario tiene planteado el comprar un cámara de XXX modelo, al realizar la búsqueda en nuestra herramienta donde los resultados que se le muestran de la predicción es que existe una alta probabilidad de que dicha cámara baje de precio en cierto tiempo específico a manera de que el usuario agrega dicha búsqueda (la cámara de XXX modelo) en favoritos y además incluye una alarma en los rangos de precio que el defina para que se le avise cuando esté en los mismo.

En este caso que se presenta en el ejemplo en donde entramos ya directamente en los casos de pagos con uso donde aplicaremos 3 tipos de membresías.

- **Membresía Bronce:** en donde tendrás un límite de búsqueda diaria de 5 artículos y ajuste ilimitados a dichos artículos.
- **Membresía Plata:** en donde tendrás un límite de búsqueda diaria de 10 artículos y ajustes ilimitados a dichos artículos.
- **Membresía Oro:** en donde tendrás un límite de búsqueda diaria ilimitada de artículos y ajustes ilimitados a dichos artículos.



Al tener cualquiera de las membresías podrás usar la herramienta sin ningún tipo de ADS, excepto links de Amazon Afiliados*

Cada artículo dentro de las búsquedas que se realicen en la herramienta contará con un link de enlace directo para su compra, dicho

link se incluirá mediante el servicio de Amazon Afiliados, ya sea como enlace corto o enlace completo.

Para los accesos abiertos solo se podrán realizar 2 búsquedas al día, donde se mostrarán entre cada búsqueda distintos tipos de ADS ya sean de tipo búsqueda, gráficos, vídeos o aplicaciones. Preferiblemente luego de cada búsqueda se incluirá un anuncio de tipo video con tiempo determinado de 30 a 60 segundos (según el anuncio) y en el resto de la herramienta si estará presente otros tipos de ADS como los de gráficos, entre otros.

5.6 Retorno del proyecto

A pesar de que la medición del retorno de inversión en proyectos de data es difícil de cuantificar, creemos que para calcular el retorno se ha de comenzar haciéndonos unas preguntas específicas y concretas, para recopilar el dato adecuado, desarrollar una metodología y realizar un modelo para el análisis para finalmente evaluar los resultados.

Para ello podemos evaluarlo de varias formas una de ellas podría ser, analizar el retorno de inversión siguiendo dos enfoques diferentes. Uno es calcular el coste del proyecto como tal y contrastarlos con otras alternativas. Por ejemplo, pongamos que necesitamos expandir el data warehouse de la empresa para dar cabida a más y diversos tipos de nuevos datos. Podríamos emplear más tecnología (OLTP) e ir añadiendo servidores y software de bases de datos a unos costes de miles de euros, o podríamos emplear el framework Hadoop y conseguir el mismo resultado por menos euros. En cualquier caso, conseguimos extraer valor de dato residente en el

data warehouse, pero el modelo de infraestructura basado en Hadoop nos desahoga de la necesidad de añadir mayor capacidad de procesamiento.

Algunos de los retos que nos encontramos a la hora de analizar el retorno de inversión son:

1. Entender cómo obtener valor real de los datos.
2. Definir apropiadamente el alcance de la investigación de los datos.
3. Encontrar los recursos adecuados y las habilidades necesarias para extraer los descubrimientos de los resultados.

El reto está en definir las métricas que nos permitan medir el retorno de inversión de la manera adecuada, y más importante aún, comprender cuándo esperar los resultados del retorno de la data analizada. Aunque no hay una metodología única dada la diversidad de casuísticas, según un reciente estudio de IDG Research los proyectos de big data suelen llevar más tiempo del esperado, y se obtiene menor retorno de inversión del esperado ya que se requiere de ayuda externa para extraer significado práctico de los descubrimientos.

La mejor estrategia para incrementar el ROI y reducir el time-to-insight es una vez más comenzar con proyectos más pequeños. Usar el big data como tecnología habilitadora más que como una solución en sí misma. Limitar el alcance de las queries para desvelar respuestas a preguntas que permitan obtener retornos inmediatos.

Nos afianzamos muchísimo en un ejemplo real de una empresa que ha obtenido un retorno de inversión muy positivo. Utilizó big data para comprender más sobre el comportamiento de su cliente y conseguir convertir compradores únicos en clientes con pedidos repetidos. El alcance del proyecto estaba muy focalizado y consiguió un retorno de inversión del 120% de incremento en ventas. El tiempo para llegar a los ingresos esperados se acortó y la prueba de concepto fue suficientemente prometedora para la empresa como para comprometerse a realizar nuevos proyectos con mayor presupuesto y tecnologías para análisis de resultados.

6

Conclusiones y líneas futuras.

Tras exponer los métodos y materiales utilizados y mostrar y discutir los resultados obtenidos, en este capítulo se presentan las conclusiones extraídas y las posibles líneas futuras por las que podría continuar este trabajo.

Como bien se menciona desde el inicio del proyecto, se enfatizó en trabajar con los 5000 productos electrónicos más vendidos dentro de Amazon España, teniendo como primer check point el automatizar todo el proceso tanto de obtención de datos, carga de los datos y su procesado, dicho esto como siguiente paso a dar sería el ir incorporando más departamentos dentro del propio Amazon España como lo son deportes, calzado, ropa, entre los muchos otros que disponen y a su vez ir maximizando el número de productos hasta manejar la totalidad que dispongan ellos en web y no solo limitarnos a los 5000 productos más vendidos de cada categoría, con esto sería un paso grande incluso para proceder ya no solo con Amazon España, sino además ir incluyendo otras tan importantes a nivel nacional como lo son El Corte Inglés, Booking, AliExpress, eBay, entre muchas otras.

Llegado este punto ya tendríamos a disposición no solo lo enfatizado en dicho proyecto con la predicción de precios de productos de venta con entrega física, sino que también lo llevaríamos a nivel de servicios o productos no físicos como una entrada a un evento, compra de cupón, compra de boleto para viajes, entre muchas otras.

De esta fase tan grande que abarca un montón de procesos en los que indiscutiblemente se tendrán que ir mejorando y verificando viabilidad, ejecución, resultados y muchos otros puntos determinantes para el éxito del mismo, no porque se pretenda parar allí sino porque el paso más grande ya sería llevarlo a un nivel internacional e ir incluyendo las distintas web de Amazon según los países en los que esté disponible, Alibaba según país que esté disponible, marcas reconocidas con ventas online mundialmente como Adidas, Lacoste, Nike, entre muchas otras, con esto englobaríamos una cantidad bastante grande de productos a manera que nuestros usuarios independientemente de donde se encuentren geográficamente situados

cuenten con la disponibilidad de poder obtener el servicio que se ofrece, claro está sujeto a la economía de cada país según la web.

Nuestra visión a futuro es muy grande y complicada, pero para nada imposible de hecho hoy por hoy lo más semejante a lo que mencionamos anteriormente que tenemos como plan a futuro es la recientemente web de Google flights que puedes realizar búsqueda de vuelos desde cualquier parte del mundo y dando como resultados todas y cada una de las aerolíneas que desees, esto aplicado con nuestro modelo predictivo podría ser muy pero muy interesante.

Además de lo mencionado anteriormente, que serían líneas futuras enmarcadas en el análisis de los datos utilizados, existen otras líneas derivadas del trabajo de análisis de datos genérico realizado, como son:

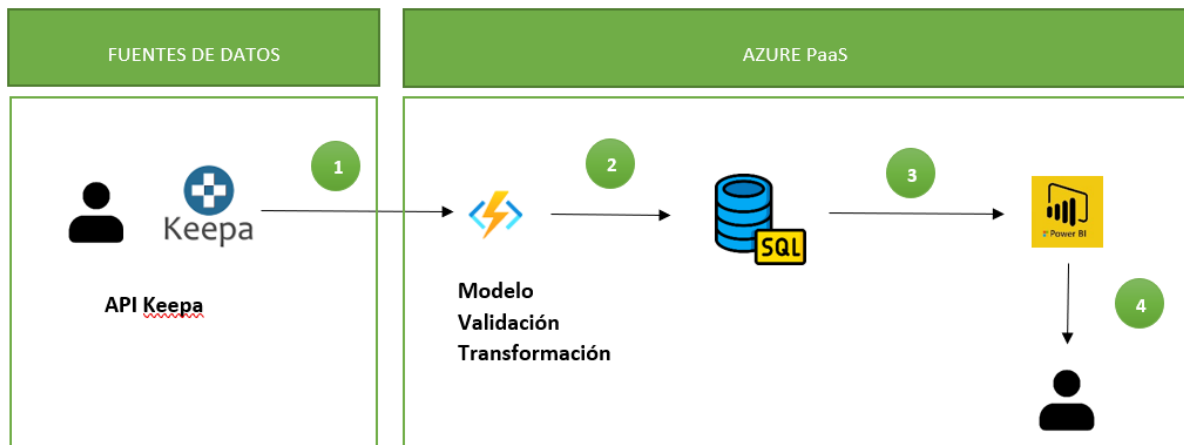
- Generalizar el código desarrollado, de modo que permita realizar un análisis rápido similar al realizado en este trabajo cambiando únicamente la fuente de datos y unas pocas opciones de configuración.
- Relacionado con el punto anterior, desarrollar una interfaz de usuario amigable y de alto nivel, que permita realizar esta labor sin necesidad de modificar el código.

Por otro lado, como conclusiones más directas y específicas podemos denotar los siguientes puntos

- La calidad del dato y su fuente origen son parte fundamental para la calidad del modelo, además de evitar ciertos tratamientos de este que pueden resultar complicados.
- Descartamos la opción de obtención mediante web scraping al encontrar una plataforma especializada en trackear los precios de Amazon España, que además a futuro nos brinde las mismas posibilidades no solo para Amazon España sino

también para muchas otras no solo a nivel nacional sino también a nivel internacional.

- Se creó un procedimiento para la descarga automática de los CSV a través de Python mediante un Script automatizado tanto para la obtención de los datos como también para la carga en la nube de estos.
- La inferencia de datos no implica un mejor resultado en las métricas del modelo.
- En el modelo de árbol de decisión, no suele estar recomendado utilizar un One-Hot Encoder dado que induce una gran dispersión de los datos, además de que, para un árbol de decisión, las variables son independientes, incluidas las creadas a partir de la codificación.
- Como efecto del presente trabajo de investigación, se presentó el diseño de un método predictivo para costes de productos electrónicos de Amazon.es, en el que se pusieron a prueba distintos modelos a manera de verificar resultados con cada uno de ellos, no solo ello, sino que además se implementaron a tiempos específicos para dichas predicciones a manera que el usuario pueda tomar la mejor decisión con base en los mismos.
- Se investigó la fuente de obtención de los datos día a día siempre bajo las mismas especificaciones para mantener su calidad, los cuales fueron procesados, analizados y clasificados para el modelo propuesto, además se investigó todo lo concerniente a las tecnologías asociadas a Machine Learning para luego optar por la tecnología más adecuada o acorde a los resultados requeridos.
- Con el fin de automatizar el proceso de extracción de la información, así como la creación del modelo y su ejecución cada cierto periodo de tiempo. Proponemos, la que creemos que es, una posible solución:



Una de las soluciones óptimas consta de un modelo de datos propio del caso, la lógica de carga y el modelo en Azure Paas.

1. **Ingesta** de datos a través de la API de Keepa.
2. **Aplicación** del **modelo** y base de datos del proyecto.
3. Extracción para **visualización**.
4. Visualización mediante **power BI**.

7

Bibliografía.

- [1] <https://www.cnmc.es/CNMC/ecommerce-1T2020-20201002>.
 - [2] <https://marketing4ecommerce.net/vii-estudio-anual-de-ecommerce-en-espana-2020-el-ano-en-el-que-el-coronavirus-cambio-para-siempre-la-forma-de-comprar/>
 - [3] https://www.ine.es/ss/Satellite?L=es_ES&c=INECifrasINE_C&cid=1259952923622&p=1254735116567&pagename=ProductosYServicios%2FINECifrasINE_C%2FPYSDetalleCifrasINE#ancla_1259952923566
 - [4] <https://www.xataka.com/basics/como-saber-cuanto-baja-de-precio-un-articulo-en-amazon-viendo-su-historial-de-precios-con-keepa>
 - [5] <https://www.watch4price.es/>
 - [6] <https://www.kayak.es/price-trend-explanation>
 - [7] <https://aws.amazon.com/es/forecast/>
 - [8] https://es.wikipedia.org/wiki/M%C3%A9todo_de_Newton
-