TRABAJO FIN DE GRADO

GRADO EN CIENCIA, GESTIÓN E INGENIERÍA DE SERVICIOS

CURSO ACADÉMICO 2023/2024

CONVOCATORIA JUNIO 2024

INTEGRACIÓN DE POWER BI CON MODELOS DE APRENDIZAJE AUTOMÁTICO PARA LA OPTIMIZACIÓN DE DECISIONES EN LA GESTIÓN HOTELERA: UN ENFOQUE PRÁCTICO

AUTORA: Robles Angelina, Lucía

DNI: 03942076Z

TUTOR: Granada Mejía, Juan David

En Madrid, a 7 de junio de 2024

TABLA DE CONTENIDOS

TABLA DE ILUSTRACIONES

RESUMEN

Este trabajo se enfoca en explorar y demostrar la integración efectiva de Power BI, una herramienta de análisis y visualización de datos, con modelos de aprendizaje automático para mejorar la toma de decisiones en el sector hotelero.

El estudio se centra en cómo estas tecnologías pueden ser implementadas de manera práctica para optimizar procesos de gestión y operaciones en hoteles. Se analizará un conjunto de datos representativo de un hotel y se diseñarán modelos de aprendizaje automático adecuados para abordar desafíos específicos, como la predicción de la demanda, la optimización de precios y la gestión de inventarios.

A través de la integración con Power BI, se demostrará cómo estos modelos pueden ser visualizados de manera intuitiva y cómo los *insights* derivados de ellos pueden ser utilizados para tomar decisiones más informadas y estratégicas en la gestión diaria de hoteles, contribuyendo así a mejorar su competitividad y eficiencia operativa.

Palabras clave: aprendizaje automático, industria hotelera, optimización de precios, gestión de la ocupación, visualización de datos, eficiencia operativa, modelos predictivos, conocimiento, toma de decisiones.

1. INTRODUCCIÓN

En este apartado, se introduce el tema, el motivo de llevar a cabo el estudio y los objetivos:

1.1. Contextualización del tema

La industria hotelera, al igual que otras industrias, está experimentando un cambio significativo debido a los avances en tecnologías de la información, aunque este cambio está siendo relativamente lento en el sector. Muchos investigadores han mostrado un interés creciente en explorar e implementar las nuevas tecnologías de inteligencia artificial y aprendizaje automático en la industria hotelera.

En la era actual, la capacidad de recopilar, analizar y actuar ante vastas cantidades de datos se ha convertido en una ventaja competitiva crucial. Power BI, una herramienta líder en el análisis y la visualización de datos, junto con los avances en el aprendizaje automático, ofrecen oportunidades significativas para transformar la manera en que los hoteles operan y toman decisiones. Este trabajo se sitúa en la intersección de estas tecnologías emergentes y su aplicación práctica en la gestión hotelera, donde la eficiencia operativa y la toma de decisiones estratégica pueden ser mejoradas sustancialmente a través de la utilización efectiva de los datos.

1.2. Justificación de la relevancia e importancia del estudio

La justificación de este estudio radica en la creciente complejidad del entorno competitivo en el que operan los hoteles, caracterizado por la fluctuación de la demanda, la variabilidad de precios y la necesidad de optimizar la gestión del inventario. En este contexto, las herramientas tradicionales de gestión y análisis de datos se quedan cortas ante la necesidad de respuestas rápidas y predicciones basadas en datos.

Power BI proporciona una plataforma intuitiva para analizar los datos hoteleros, mientras que los modelos de aprendizaje automático ofrecen la capacidad de predecir tendencias y optimizar decisiones. La integración de estas tecnologías promete mejorar la competitividad de los hoteles y su capacidad para adaptarse a un mercado en constante cambio.

1.3. Objetivos del trabajo

El objetivo principal de este trabajo es explorar y demostrar cómo la integración de Power BI y modelos de aprendizaje automático puede mejorar la toma de decisiones en el sector hotelero. Específicamente, este estudio busca:

- Analizar conjuntos de datos relevantes para la industria hotelera y diseñar modelos de aprendizaje automático que aborden desafíos específicos como la optimización de precios y la gestión de inventarios.

- Demostrar la implementación práctica de estos modelos en Power BI, facilitando así la visualización y el análisis.

- Evaluar cómo estos *insights* pueden ser empleados para tomar decisiones más informadas y estratégicas en la gestión diaria de hoteles, mejorando su competitividad y eficiencia operativa.

2. MARCO TEÓRICO

En este apartado, se desarrollará el marco teórico y las definiciones imprescindibles para comprender los conceptos abordados en el trabajo:

2.1. Fundamentos de Power BI

Power BI es una herramienta de análisis de datos desarrollada por Microsoft, diseñada para proporcionar informes o cuadros de mando (*dashboard* en inglés) interactivos y en tiempo real de los datos. Es una solución potente y flexible que permite a los usuarios conectar, modelar y visualizar sus datos de manera eficiente, facilitando la toma de decisiones basada en datos.

A continuación, exploraremos los fundamentos de Power BI, incluyendo su arquitectura, características principales, y cómo se utiliza para transformar datos brutos en información valiosa.

2.1.1. Arquitectura de Power BI

La arquitectura de Power BI se compone de tres componentes principales (ilustración 1), los cuales se describen a continuación:

- Power BI Desktop: una aplicación de escritorio que permite a los usuarios crear y diseñar informes y cuadros de mando. Es aquí donde se realiza la mayor parte del trabajo de importación de datos, transformación, modelado y visualización.

- Power BI Service: un servicio en línea (SaaS) que permite a los usuarios publicar, compartir y colaborar en los informes. Facilita la distribución de los informes a través de la organización y el acceso a los datos desde cualquier lugar.

- Power BI Mobile: aplicaciones móviles para iOS, Android y Windows que permiten acceder a los informes y cuadros de mando publicados en el Power BI Service desde dispositivos móviles.

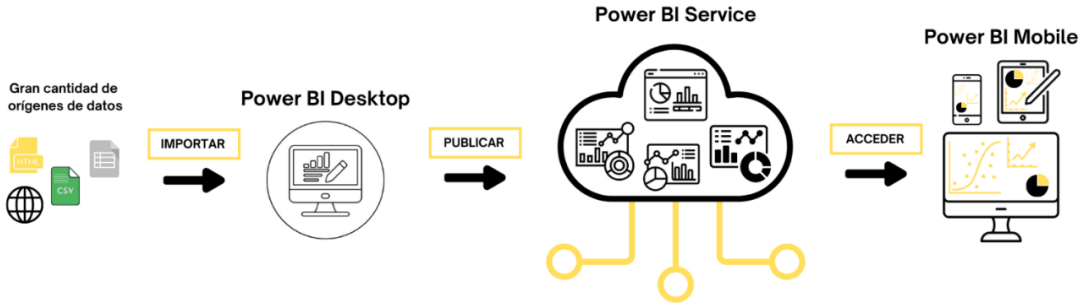


Ilustración 1 - Arquitectura Power BI. Elaboración propia.

2.1.2. Características principales

- Conexión a diversas fuentes de datos: Power BI permite conectar con una amplia gama de fuentes de datos, incluidas bases de datos locales, servicios en la nube, archivos Excel, o incluso obtener datos pasándole un enlace a una página web, entre otros.

- Modelado de datos: ofrece una herramienta para transformar y modelar datos (llamada Power Query), facilitando la creación de relaciones entre diferentes conjuntos de datos, la creación de nuevas medidas y dimensiones, y la preparación de los datos para análisis.

- Visualización de datos: proporciona una biblioteca de visualizaciones interactivas, desde gráficos básicos hasta mapas y visualizaciones personalizadas. Los usuarios pueden crear informes y *dashboards* interactivos que facilitan la exploración de los datos.

- Interacción con los datos: Power BI facilita la interacción con los datos al permitir a los usuarios manipular y explorar la información de manera detallada. Esta capacidad interactiva apoya el análisis en profundidad, mejorando el entendimiento y las decisiones basadas en los datos.

- Publicación y facilidades para compartir: los informes y *dashboards* pueden ser fácilmente publicados en Power BI Service y compartidos con otros usuarios dentro de la organización, permitiendo una colaboración efectiva y toma de decisiones informada.

- Actualización en tiempo real: la herramienta soporta la actualización en tiempo real de los datos, lo que permite a los usuarios tener siempre acceso a la información más reciente.

Power BI transforma la manera en que las organizaciones acceden, interpretan y utilizan sus datos. Al convertir datos brutos en visualizaciones interactivas y comprensibles, facilita la identificación de tendencias, patrones y anomalías.

2.2. Fundamentos del Aprendizaje Automático

El aprendizaje automático (o *Machine Learning* en inglés (ML)) es una rama de la inteligencia artificial que se centra en la creación de sistemas capaces de aprender y mejorar a partir de la experiencia sin estar explícitamente programados para ello. Esta tecnología permite a las máquinas identificar patrones en los datos y tomar decisiones basadas en ellos [7].

"El Machine Learning es un campo de estudio que da a los ordenadores la capacidad de aprender sin ser programados de forma explícita (Arthur Samuel, 1959)."

En este apartado, se explorarán los conceptos básicos del aprendizaje automático, incluyendo sus tipos y modelos fundamentales.

2.2.1. Tipos de Aprendizaje Automático

El aprendizaje automático se puede clasificar en tres categorías principales, basadas en la naturaleza de la señal de aprendizaje o la retroalimentación disponible para el sistema:

- Aprendizaje Supervisado: en este enfoque, el algoritmo se entrena usando un conjunto de datos etiquetado, donde cada *input* (X) ofrece un *output* (Y). El objetivo es aprender una función que mapee las entradas a las salidas para poder hacer predicciones en datos que son nuevos para la máquina. Algunos ejemplos incluyen el modelo de regresión lineal (predice un número) o el modelo de clasificación (predice una categoría).

- Aprendizaje No Supervisado: aquí, el algoritmo se entrena usando un conjunto de datos sin etiquetar, y se utiliza para encontrar patrones ocultos o agrupaciones intrínsecas en los datos. Para ello, se utiliza una técnica denominada *clustering*, que consiste en agrupar un conjunto de objetos de tal manera que los objetos en el mismo grupo (o *cluster*) sean más similares entre sí que con los de otros grupos [5].

- Aprendizaje por Refuerzo: en el aprendizaje por refuerzo, un agente[1] toma decisiones o realiza acciones dentro de un entorno para alcanzar un objetivo. A través de ensayo y error, y basándose en la retroalimentación de recompensas o castigos, el agente aprende la mejor estrategia para lograr su objetivo [15].

2.2.2. Modelos fundamentales

En este punto, diferenciaremos entre los principales modelos basados en aprendizaje supervisado, y aquellos basados en aprendizaje no supervisado:

- Modelos supervisados:

  - Regresión lineal: la regresión lineal se utiliza para predicciones continuas. Un ejemplo de regresión lineal usado típicamente en la literatura de *Machine Learning* es la predicción del precio de una casa en función de una o más variables como pueden ser los metros cuadrados de esta. El resultado ha de ser una recta que minimice el error lo máximo posible, definiendo el error como la diferencia de altura entre cada punto y la recta.

[1] Agente: entidad capaz de percibir su entorno, procesar tales percepciones y responder o actuar en dicho entorno de manera racional.

- Regresión logística: se utiliza para modelar la probabilidad de una variable dependiente binaria en función de una o más variables independientes (predictores). Es especialmente útil para tareas de clasificación binaria, como decidir entre dos categorías mutuamente excluyentes, por ejemplo, en el contexto de este trabajo, para predecir la decisión de reservar un hotel con dos valores ("usuario reserva" = 1 o "usuario no reserva" = 0) [16].

En el siguiente gráfico (ilustración 2), se muestra la diferencia entre la regresión lineal y la regresión logística con sus respectivas formulas:

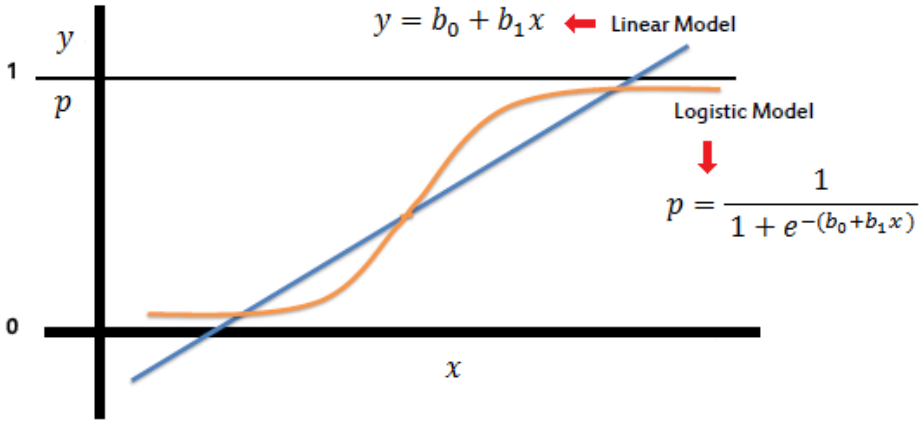


Ilustración 2 - Regresión lineal y regresión logística. Fuente: TheMachineLearners.

- Árboles de decisión y *Random Forest*: métodos que utilizan estructuras de árbol para tomar decisiones, siendo eficaces tanto para clasificación como para regresión. Los bosques aleatorios combinan múltiples árboles de decisión para mejorar la precisión [13].

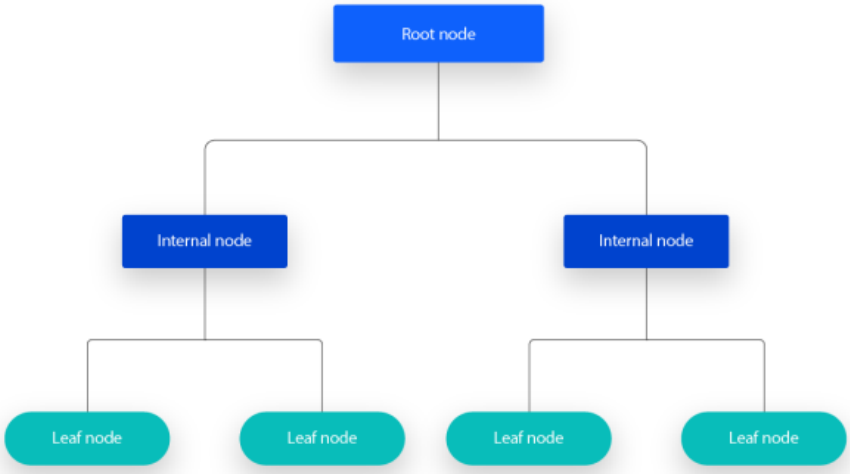


Ilustración 3 - Árbol de decisión. Fuente: IBM.

- Redes neuronales y *Deep Learning*: modelos inspirados en el funcionamiento del cerebro humano, eficaces en tareas complejas como el reconocimiento de imágenes o procesamiento del lenguaje natural.

- Modelos no supervisados:

  - K-Means: es un algoritmo de *clustering* que se utiliza para dividir un conjunto de datos en *K* grupos (o *clusters*) distintos. El objetivo principal del algoritmo es minimizar la suma de las distancias entre los puntos de datos y el centroide[2] de su *cluster* asignado.

  - DBSCAN[3]: destaca por su capacidad para identificar *clusters* de forma arbitraria y manejar puntos de ruido en el conjunto de datos. Funciona con base en la densidad de los puntos, lo que significa que busca áreas del espacio de datos donde los puntos están densamente agrupados.

En la siguiente imagen (ilustración 4) se muestra una comparativa entre los efectos de aplicar el algoritmo K-MEANS y usar DBSCAN:

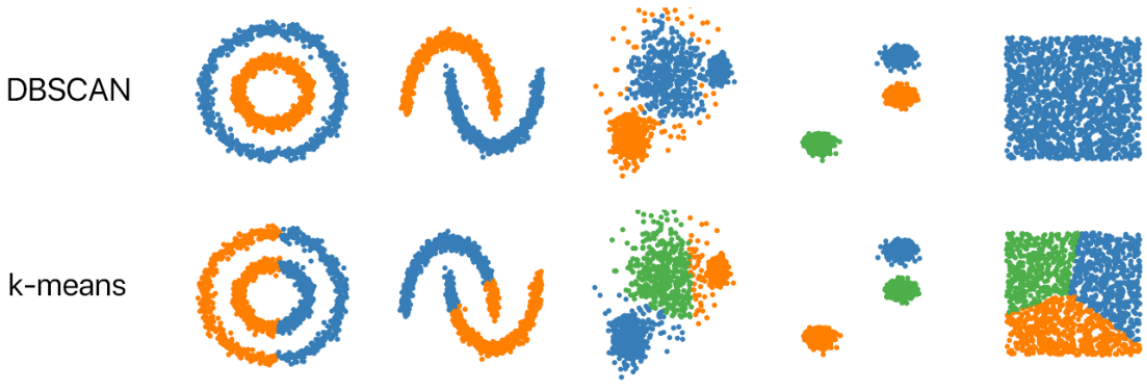


Ilustración 4 - Comparativa entre K-Means y DBSCAN. Fuente: TowardsDataScience.

2.3. Aplicaciones de Power BI en la industria hotelera

Power BI tiene un amplio rango de aplicaciones en la industria hotelera, ayudando a los establecimientos a optimizar sus operaciones, mejorar la experiencia del cliente y aumentar su rentabilidad. A continuación, se detallan algunas de las aplicaciones más relevantes de Power BI en este sector:

[2] Centroide: punto donde se produce la intersección de las medianas que forman parte de un triángulo [17].
[3] DBSCAN: Density-Based Spatial Clustering of Applications with Noise.

a) Análisis del rendimiento operativo

Power BI permite a los hoteles recopilar y analizar datos operativos en tiempo real para evaluar el rendimiento de diferentes áreas, como servicios de habitaciones, limpieza, y mantenimiento. Mediante la creación de *dashboards*, los gerentes pueden identificar rápidamente áreas de ineficiencia o sobresalientes, permitiendo la implementación de mejoras operativas basadas en datos.

b) Gestión de ingresos y tarifas

La gestión de ingresos es crítica en la industria hotelera, donde la demanda puede fluctuar significativamente. Power BI ayuda a analizar tendencias de reservas, ocupación, tarifas promedio diarias (ADR) e ingresos por habitación disponible (RevPAR) para optimizar las estrategias de precios y maximizar los beneficios. Los modelos predictivos integrados en Power BI pueden prever la demanda futura, permitiendo a los hoteles ajustar sus tarifas dinámicamente.

c) Análisis de satisfacción del cliente

Mediante la recopilación y análisis de encuestas de satisfacción del cliente, reseñas en línea y puntos de contacto digitales, Power BI puede ayudar a los hoteles a obtener una comprensión profunda de la experiencia del cliente. Estos *insights* permiten a los hoteles identificar áreas de mejora y personalizar los servicios para satisfacer mejor las necesidades y preferencias de los clientes.

d) Optimización de la gestión de inventarios

La gestión eficiente del inventario de habitaciones, alimentos y bebidas, y suministros es fundamental para la rentabilidad hotelera. Power BI puede analizar patrones de consumo y prever necesidades de inventario, ayudando a reducir el desperdicio y asegurar que los recursos estén disponibles donde y cuando se necesiten.

e) Benchmarking y análisis competitivo

Power BI facilita la comparación de indicadores clave de rendimiento (KPIs) con los de otros hoteles en la región o categoría, permitiendo a los hoteles entender su posición en el mercado. Esto es crucial para identificar ventajas competitivas y áreas de mejora.

f) Análisis de tendencias de mercado y demanda

La capacidad de anticipar cambios en la demanda turística y preferencias de los consumidores es muy importante. Utilizando Power BI, los hoteles pueden analizar datos de mercado, tendencias turísticas y factores externos como eventos locales o temporadas para ajustar sus estrategias de marketing y operaciones en consecuencia.

g) Gestión de Relaciones con Clientes (CRM)

Integrando Power BI con sistemas de CRM, los hoteles pueden segmentar a los clientes basándose en su comportamiento, preferencias y valor económico. Esto permite la creación de ofertas personalizadas, mejorando la lealtad del cliente y aumentando las oportunidades de ingresos repetidos.

h) Eficiencia energética y sostenibilidad

El análisis de datos relacionados con el consumo de energía y recursos puede ayudar a los hoteles a identificar oportunidades para mejorar la eficiencia energética y promover prácticas sostenibles. Power BI puede visualizar patrones de consumo y ayudar a implementar medidas que reduzcan el impacto ambiental y los costes operativos.

En resumen, Power BI ofrece a la industria hotelera una plataforma poderosa para el análisis de datos, capaz de transformar grandes volúmenes de datos mejorando la toma de decisiones y aumentando significativamente los beneficios de la empresa.

2.4. Aplicaciones del Aprendizaje Automático en la industria hotelera

El aprendizaje automático se ha aplicado en la industria hotelera para la previsión de la demanda, la predicción de cancelaciones de reservas, la eficiencia operativa y financiera, y la mejora de la eficiencia laboral. Los algoritmos de aprendizaje automático han demostrado ser superiores en precisión de pronóstico en comparación con los modelos estadísticos.

En términos de liderazgo global en la implementación de estas tecnologías, China y Estados Unidos se destacan como los países más avanzados (ilustración 5). Estas naciones han invertido significativamente en la adopción de tecnologías de inteligencia artificial y aprendizaje automático en la industria hotelera, lo que ha permitido a sus hoteles mejorar considerablemente su eficiencia operativa y competitividad. La capacidad de estos países para liderar en este ámbito también refleja su inversión en infraestructura tecnológica y en la formación de profesionales capacitados en ciencia de datos y *machine learning*.

En España, estamos en cuarta posición en el uso de esta tecnología dentro de la industria hotelera [4]. Esta posición refleja un avance significativo y un compromiso con la innovación tecnológica en el sector. Sin embargo, también indica que hay espacio para un mayor desarrollo y adopción de estas tecnologías. España, con su fuerte industria turística, puede beneficiarse enormemente al ampliar el uso de algoritmos de aprendizaje automático para optimizar operaciones, mejorar la experiencia del cliente y aumentar la rentabilidad.

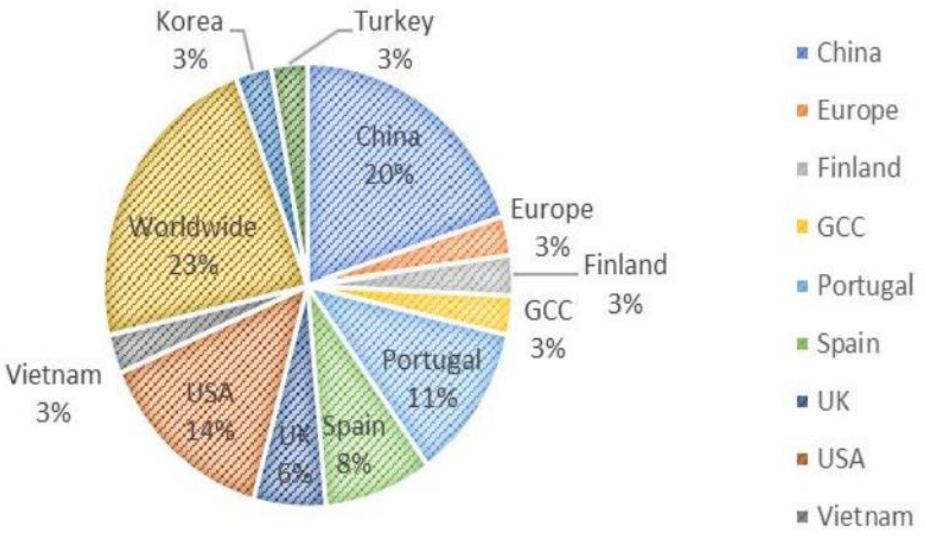


Ilustración 5 - Implementación de ML en la industria hotelera en diferentes partes del mundo (Alotaibi, 2020).

3. CASO PRÁCTICO

A continuación, se aborda el caso práctico con el que se pretenden cumplir los objetivos mencionados en la introducción. Este caso representa un ejemplo a pequeña escala de todas las ventajas que se pueden obtener de la combinación del aprendizaje automático y Power BI en la industria hotelera.

3.1. Metodología

En este punto, se describirá el enfoque y las herramientas utilizadas para recopilar y procesar los datos, así como los pasos seguidos para evitar fallos a la hora de introducir código en Power BI. Este enfoque metodológico asegura una base sólida para el correcto desarrollo del caso:

3.1.1. Descripción de la metodología de investigación

- Recopilación de los datos:

Se utilizará una base de datos de un hotel ficticio obtenida de Kaggle[4], una plataforma perteneciente a Google que reúne a científicos de datos, profesionales del *Machine Learning* y aficionados de los datos. En ella ciertas empresas organizan competiciones en las que se ofrece un premio a la solución más eficiente. También se pueden encontrar cursos, modelos de aprendizaje automático creados por la comunidad y bases de datos en abierto de muchos ámbitos distintos. La base de datos utilizada incluirá información detallada sobre reservas de hotel, incluyendo fechas, precios, cancelaciones, tipos de habitación, y otros factores relevantes para el análisis.

- Procesamiento de los datos:

Para el procesamiento de los datos se utilizará Jupyter Notebooks[5], una aplicación web de código abierto que permite crear y compartir documentos interactivos que contienen código en vivo, ecuaciones, visualizaciones y texto narrativo. Es una herramienta poderosa y ampliamente utilizada en el campo de la ciencia de datos, la investigación académica y la educación [2].

Los Jupyter Notebooks permiten a los usuarios combinar texto explicativo con fragmentos de código ejecutable en diferentes lenguajes de programación, incluyendo Python y R, entre otros. Estos fragmentos de código se ejecutan en tiempo real dentro del *notebook*, lo que facilita la exploración de datos, la experimentación con algoritmos y la creación de visualizaciones. Además de ejecutar código, los cuadernos admiten la creación de gráficos

[4] Enlace a la web de Kaggle: https://www.kaggle.com/
[5] Enlace a la web de Jupyter Notebooks: https://jupyter.org/

interactivos, tablas, ecuaciones matemáticas y otros elementos multimedia. Esto los hace ideales para la presentación de resultados de análisis de datos y para compartir conocimientos en forma de documentos autocontenidos y reproducibles.

Como lenguaje de programación se hará uso de Python [9]. Este entorno permitirá aplicar técnicas de limpieza, transformación y preparación de los datos para su posterior análisis. Para ello, se recurrirá a las siguientes librerías incluidas (ilustración 6) en este lenguaje de programación [2]:

- Pandas: proporciona estructuras de datos y herramientas de análisis de datos de alto rendimiento y fáciles de usar. Su objeto principal, el *dataframe*, permite almacenar y manipular datos tabulares en filas de observaciones y columnas de variables. Pandas es ideal para diversas tareas de ciencia de datos como la manipulación de datos, la limpieza, la exploración y el análisis complejo.

- NumPy: es una biblioteca fundamental para la computación científica en Python. Proporciona un objeto de matriz multidimensional, diversos objetos derivados (como matrices), y una variedad de rutinas para operaciones rápidas en arrays[6]. NumPy es muy valorado por su capacidad de proporcionar estructuras de datos de alto rendimiento y operaciones matemáticas eficientes.

- Matplotlib: es una biblioteca de visualización de datos en Python que permite la creación de gráficos en una variedad de formatos. Es conocida por su flexibilidad y capacidad para crear una amplia gama de visualizaciones, desde gráficos de líneas y barras hasta histogramas y diagramas de dispersión, con solo unas pocas líneas de código.

- Seaborn: está construida sobre Matplotlib, y proporciona una interfaz de alto nivel para crear gráficos estadísticos atractivos y con estilo. Facilita la creación de visualizaciones complejas como gráficos de distribución y diagramas de dispersión con ajustes automáticos y paletas de colores predefinidas, simplificando el proceso de generación de gráficos informativos y visualmente agradables.

- Scikit-Learn: es una biblioteca que ofrece herramientas simples y eficientes para el análisis predictivo de datos y la minería de datos. Ofrece algoritmos comunes de *machine learning* como *clustering*, regresión, clasificación, y reducción de dimensionalidad, además de herramientas para la selección de modelos, la evaluación de modelos, y la preparación de datos.

[6] Arrays: son objetos similares a una lista cuyo prototipo proporciona métodos para efectuar operaciones de recorrido y de mutación [18].

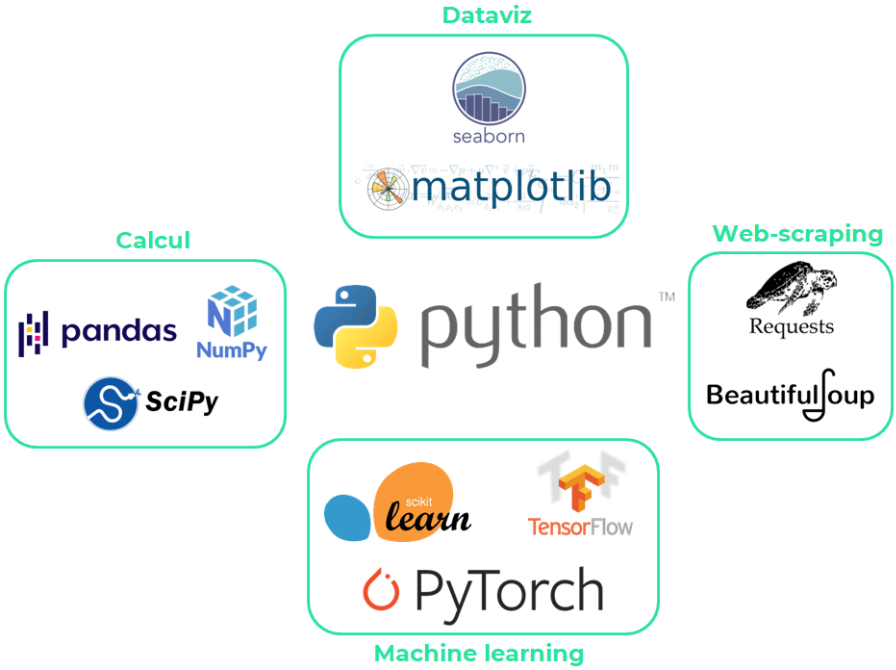


Ilustración 6 - Principales librerías de Python para *Data Science*. Fuente: data-bird.co.

3.1.2. Configuración del entorno de desarrollo

Para garantizar un ambiente adecuado para el tratamiento de los datos con Python y sus bibliotecas utilizando Power BI, es recomendable configurar un entorno virtual en Anaconda[7]. Anaconda es una distribución gratuita y de código abierto de los lenguajes de programación Python y R, especialmente diseñada para la ciencia de datos y el aprendizaje automático. Provee de una gestión conveniente de paquetes y entornos que facilitan la instalación, la actualización y la gestión de paquetes y dependencias de software. Esto permite gestionar las dependencias y evitar conflictos entre paquetes de manera eficiente.

Los pasos seguidos para la creación de este entorno son los siguientes:

1. Iniciar Anaconda Prompt, una interfaz de línea de comandos (consola) que viene con Anaconda y ejecutar el siguiente comando:

```
conda create -n powerbitfg python=3.8
```

Este comando configura un entorno aislado con su propia instalación de Python. Seleccionar una versión específica de Python asegura compatibilidad y estabilidad para las bibliotecas que se instalarán en ella.

[7] Enlace a la web de Anaconda: https://www.anaconda.com/

2. Activar el entorno con el comando **`conda activate powerbitfg`** e instalar las bibliotecas necesarias con **`pip install`** y el nombre de las bibliotecas de las que se hará uso (las de visualización no son necesarias porque estas serán generadas por el propio Power BI):

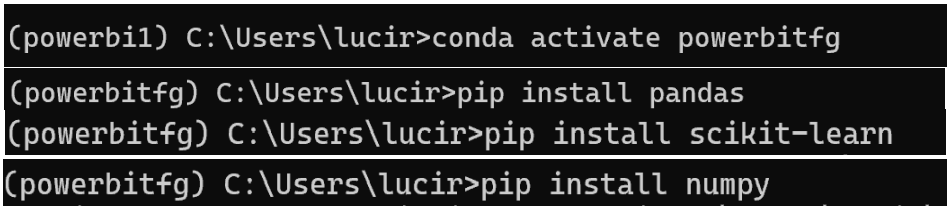


Ilustración 7 - Instalación de las bibliotecas en el entorno de Anaconda.

3. Configurar Power BI: ir a "Opciones y configuración" (ilustración 8), dentro de las opciones, seleccionar "Opciones de script de Python" (ilustración 9). En esta sección, se puede especificar el directorio donde Python está instalado y configurar Power BI para utilizar el entorno Python que se ha generado.

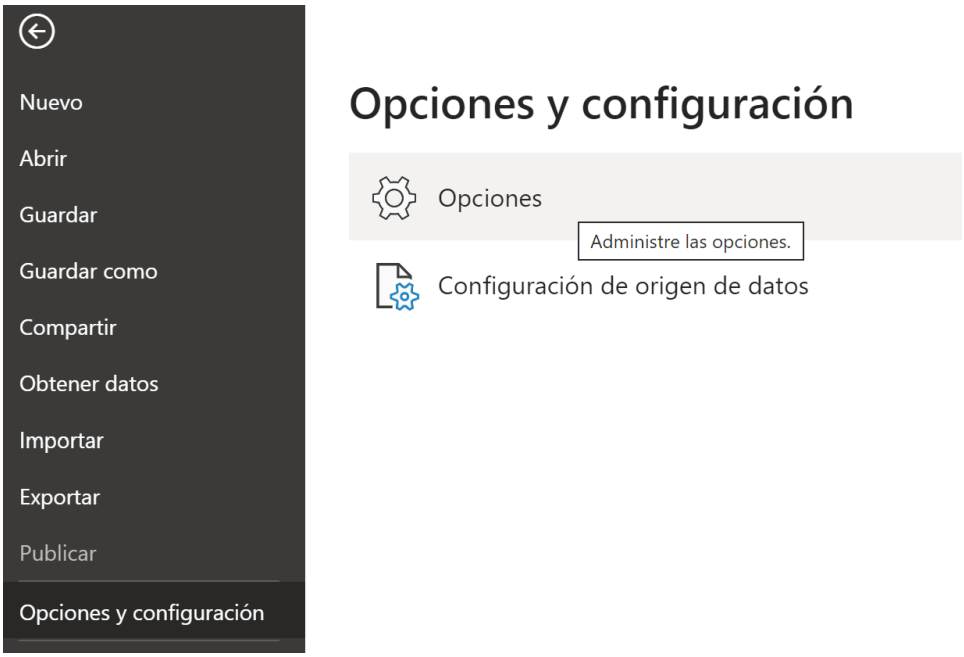


Ilustración 8 - Menú opciones y configuración Power BI.

Ilustración 9 - Selección del directorio raíz para acceder al entorno creado con Anaconda.

3.2. Preparación de los datos en Jupyter Notebooks

En este punto, se abordará la preparación necesaria antes de pasar los datos a Power BI. Para ello, se llevará a cabo un proceso de extracción, transformación y carga de los datos.[8]

3.2.1. Recopilación de los datos

Como se ha mencionado en el apartado anterior, se hará uso de Kaggle para la obtención de los datos. Como se observa en las siguientes imágenes (ilustraciones 10 y 11) la base de datos escogida para este análisis se llama “Hotel Booking Demand” [8].

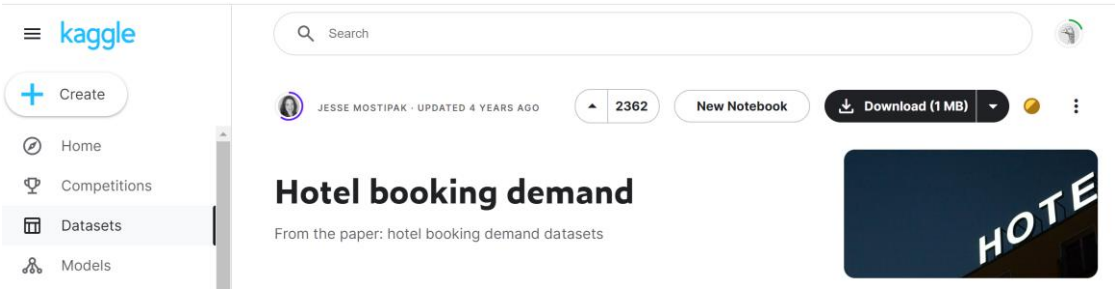


Ilustración 10 - Kaggle. Base de datos utilizada.

[8] ETL (Extract, Transform and Load).

hotel_bookings.csv (16.86 MB)

Detail Compact Column 10 of 32 columns

hotel Hotel (H1 = Resort Hotel or H2 = City Hotel)	is_canceled Value indicating if the booking was canceled (1) or not (0)	lead_time Number of days that elapsed between the entering date of the booking into the PMS and the arrival date	arrival_date_year Year of arrival date	arrival_date_month Month of arrival date	arrival_date_wee... Week number of year for arrival date	arrival_date_day_... Day of arrival date
City Hotel 66% Resort Hotel 34%	0 1	0 737	2015 2017	August 12% July 11% Other (92852) 78%	1 53	1 31
Resort Hotel	0	342	2015	July	27	1
Resort Hotel	0	737	2015	July	27	1
Resort Hotel	0	7	2015	July	27	1
Resort Hotel	0	13	2015	July	27	1
Resort Hotel	0	14	2015	July	27	1
Resort Hotel	0	14	2015	July	27	1
Resort Hotel	0	0	2015	July	27	1

Ilustración 11 - Extracto de los datos.

En primer lugar, se cargarán los datos en el cuaderno Jupyter (ilustración 12). Se ha decidido llamar al *dataframe* **`df`** para mayor comodidad a la hora de escribir el código:

```
In [2]:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

In [6]:
#Importar los datos

df = pd.read_csv('./hotel_bookings.csv')

In [7]:
df.head()
```

Out[7]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	st
0	Resort Hotel	0	342	2015	July	27	1	0	
1	Resort Hotel	0	737	2015	July	27	1	0	
2	Resort Hotel	0	7	2015	July	27	1	0	
3	Resort Hotel	0	13	2015	July	27	1	0	
4	Resort Hotel	0	14	2015	July	27	1	0	

5 rows × 32 columns

Ilustración 12 - Carga de datos en Jupyter Notebooks.

A continuación, se explorarán los metadatos para extraer las columnas con las que cuenta nuestro *dataframe*. Usando el comando **`info()`** de Pandas, se obtiene el nombre de cada columna y el tipo de dato de esta.

```
## Metadatos

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 32 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   hotel                           119390 non-null  object
 1   is_canceled                     119390 non-null  int64
 2   lead_time                       119390 non-null  int64
 3   arrival_date_year               119390 non-null  int64
 4   arrival_date_month              119390 non-null  object
 5   arrival_date_week_number        119390 non-null  int64
 6   arrival_date_day_of_month       119390 non-null  int64
 7   stays_in_weekend_nights         119390 non-null  int64
 8   stays_in_week_nights            119390 non-null  int64
 9   adults                          119390 non-null  int64
 10  children                        119386 non-null  float64
 11  babies                          119390 non-null  int64
 12  meal                            119390 non-null  object
 13  country                         118902 non-null  object
 14  market_segment                  119390 non-null  object
 15  distribution_channel            119390 non-null  object
 16  is_repeated_guest               119390 non-null  int64
 17  previous_cancellations          119390 non-null  int64
 18  previous_bookings_not_canceled  119390 non-null  int64
 19  reserved_room_type              119390 non-null  object
 20  assigned_room_type              119390 non-null  object
 21  booking_changes                 119390 non-null  int64
 22  deposit_type                    119390 non-null  object
 23  agent                           103050 non-null  float64
 24  company                         6797 non-null    float64
 25  days_in_waiting_list            119390 non-null  int64
 26  customer_type                   119390 non-null  object
 27  adr                             119390 non-null  float64
 28  required_car_parking_spaces     119390 non-null  int64
 29  total_of_special_requests       119390 non-null  int64
 30  reservation_status              119390 non-null  object
 31  reservation_status_date         119390 non-null  object
dtypes: float64(4), int64(16), object(12)
memory usage: 29.1+ MB
```

Ilustración 13 - Metadatos.

Se trata de un *dataset* compuesto de 119.390 entradas y 32 columnas. Cada columna representa una variable específica relacionada con las reservas del hotel, incluyendo tanto características de las reservas como detalles del huésped y del alojamiento. A continuación, se describen las columnas y sus tipos de datos:

1. **`hotel`**: tipo objeto, indica el nombre del hotel.

2. **`is_canceled`**: tipo entero (int64), indica si la reserva fue cancelada (1) o no (0).

3. **`lead_time`**: tipo entero (int64), es el número de días que transcurrieron entre la fecha de entrada de la reserva en el PMS y la fecha de llegada.

4. **`arrival_date_year`**: tipo entero (int64), año de llegada.

5. **`arrival_date_month`**: tipo objeto, mes de llegada.

6. **`arrival_date_week_number`**: tipo entero (int64), número de la semana de llegada.

7. **`arrival_date_day_of_month`**: tipo entero (int64), día del mes de llegada.

8. **`stays_in_weekend_nights`**: tipo entero (int64), noches de estancia durante el fin de semana.

9. **`stays_in_week_nights`**: tipo entero (int64), noches de estancia entre semana.

10. **`adults`**: tipo entero (int64), número de adultos.

11. **`children`**: tipo flotante (float64), número de niños.

12. **`babies`**: tipo entero (int64), número de bebés.

13. **`meal`**: tipo objeto, régimen de comidas incluidas en la reserva.
    - SC: sin comidas incluidas.
    - BB: desayuno incluido (*Bed & Breakfast*).
    - HB: media pensión.
    - FB: pensión completa.

14. **`country`**: tipo objeto, país de origen del huésped.

15. **`market_segment`**: tipo objeto, segmento de mercado.

16. **`distribution_channel`**: tipo objeto, canal de distribución.

17. **`is_repeated_guest`**: tipo entero (int64), indica si el huésped ha hecho reservas previas.

18. **`previous_cancellations`**: tipo entero (int64), número de cancelaciones previas.

19. **`previous_bookings_not_canceled`**: tipo entero (int64), reservas previas no canceladas.

20. **`reserved_room_type`**: tipo objeto, tipo de habitación reservada.

21. **`assigned_room_type`**: tipo objeto, tipo de habitación asignada.

22. **`booking_changes`**: tipo entero (int64), número de cambios en la reserva.

23. **`deposit_type`**: tipo objeto. Indica si el cliente hizo un depósito de una parte del precio de su estancia o no a la hora de reservar y, en caso de haberlo hecho, si este era reembolsable o no.

24. **`agent`**: tipo flotante (float64), identificador de la agencia de viajes. El valor NULL indica que la reserva pertenece a un particular o que no se tiene registro de la agencia.

25. **`company`**: tipo flotante (float64), identificador de la empresa/entidad que pagará la reserva (NULL en caso de particular o empresa no registrada).

26. **`days_in_waiting_list`**: tipo entero (int64), días que la reserva estuvo en lista de espera antes de ser confirmada al cliente.

27. **`customer_type`**: tipo objeto, tipo de cliente. *Transient* son los particulares, *Transient-Party* son los grupos que reservan sin intermediarios, *Contract* representa las reservas realizadas por emplesas y *Group* los grupos reservados por medio de intermediarios.

28. **`adr`**: tipo flotante (float64), tarifa diaria promedio.

29. **`required_car_parking_spaces`**: tipo entero (int64), espacios de estacionamiento requeridos.

30. **`total_of_special_requests`**: Tipo entero (int64), total de solicitudes especiales.

31. **`reservation_status`**: tipo objeto, estado de la reserva: *Check-out* (el cliente ha completado su estancia en el hotel y ya se ha ido), *Cancelled* (reserva cancelada) o *No-Show* (el cliente no ha cancelado, pero no se ha presentado).

32. **`reservation_status_date`**: tipo objeto, fecha del estado de la reserva.

3.2.2. Transformación de los datos

a) Gestión de tipos de datos

Lo primero que se puede observar, es que el año, semana y día están en formato de número entero. Ya que no tiene sentido hacer cálculos numéricos con estos datos, se procede a

convertirlos a formato de texto (*string* u *object*) **usando `astype()`**. En la siguiente imagen (ilustración 14), se muestran las columnas transformadas a formato de texto:

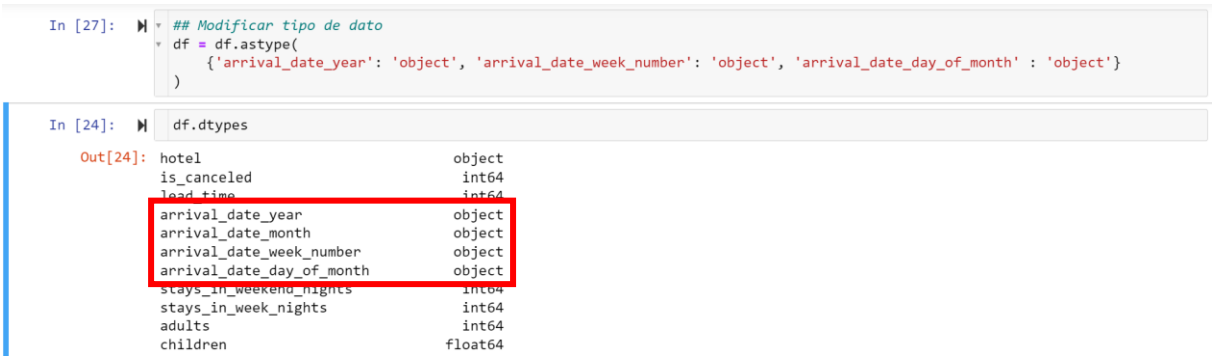


Ilustración 14 - Modificación tipo de dato para fechas.

También se creará una nueva columna que contenga el dato de la fecha completa [11], la cual recibirá el nombre '**`arrival_date`**' ya que nos podrá ser útil a la hora de implementar los datos en Power BI. Para ello, en primer lugar (ilustración 15), se transformará el nombre de cada mes a formato numérico (por ejemplo **`'January = 1', 'February' =2`**…) y a continuación se concatenarán los valores de día, mes y año en formato DD/MM/YYYY. Por último, se transformará esta columna a tipo fecha (*datetime*) como se observa en la ilustración 16.

Ilustración 15 - Creación de la columna arrival_date (fecha de llegada concatenada).

```
In [11]:  ## Convertir la columna 'arrival_date' a tipo datetime
          df['arrival_date'] = pd.to_datetime(df['arrival_date'], format='%d/%m/%Y')

          print(df[['arrival_date']].head())
```

...

```
arrival_date                    datetime64[ns]
```

Ilustración 16 - Conversión de la columna arrival_date a formato datetime.

Se puede observar que las columnas de tipo booleano como '**`is_cancelled`**' están marcadas como tipo entero. Los modelos de *Machine Learning* solo reconocen datos en forma de números enteros, por lo que se ha decidido dejarlo así [9].

b) Gestión de valores nulos

Usando el comando **`df.isnull().sum()`** se obtiene una lista con los valores nulos que existen en cada columna.

```
In [17]:  ## Buscar valores nulos
          df.isnull().sum()

Out[17]: hotel                           0
         is_canceled                     0
         lead_time                       0
         arrival_date_year               0
         arrival_date_month              0
         arrival_date_week_number        0
         arrival_date_day_of_month       0
         stays_in_weekend_nights         0
         stays_in_week_nights            0
         adults                          0
         children                        4
         babies                          0
         meal                            0
         country                       478
         market_segment                  0
         distribution_channel            0
         is_repeated_guest               0
         previous_cancellations          0
         previous_bookings_not_canceled  0
         reserved_room_type              0
         assigned_room_type              0
         booking_changes                 0
         deposit_type                    0
         agent                       16280
         company                     112442
         days_in_waiting_list            0
         customer_type                   0
         adr                             0
         required_car_parking_spaces     0
         total_of_special_requests       0
         reservation_status              0
         reservation_status_date         0
         month_number                    0
         arrival_date                    0
         dtype: int64
```

Ilustración 17 - Búsqueda de valores nulos

Como se puede observar, existen valores nulos en las columnas: **`children`**, **`country`**, **`agent`** y **`company`**:

- **`Children:`** se reemplazarán los valores nulos en esta columna con la mediana de los valores existentes. La mediana es menos sensible a los valores atípicos en comparación con el promedio, lo que la hace más adecuada para datos que pueden no estar uniformemente distribuidos. La opción **`inplace=True`** modifica el *DataFrame* original sin necesidad de asignarlo a una nueva variable.

- **`Country:`** los valores nulos se reemplazan con la moda de los valores existentes, es decir, el país que aparece con más frecuencia en el *dataset*. Esto asume que es más probable que las reservas sin país especificado pertenezcan al país más común entre todas las reservas.

- **`Agent:`** se trata del ID de la agencia que llevó a cabo la reserva. Los valores nulos representan aquellas reservas que no se hicieron a través de una agencia (reservas directas) o cuya agencia no esté registrada en la base de datos del hotel. Por tanto, se reemplazarán los valores nulos por un ID = 0 en cada una de las entradas.

- **`Company:`** se ha decidido eliminar la columna del *dataframe* debido a la gran cantidad de valores nulos. Se considera que esta columna no aporta una gran cantidad de información para el análisis que se desea hacer ya que la mayoría de las reservas no están asignadas a alguna empresa.

Además, se ha decidido buscar aquellas filas en las que los valores '**`adults`**', '**`childrens`**' y '**`babies`**' sea 0, ya que se entiende que carecen de sentido. Una buena práctica consiste en almacenar estas filas en otra variable en lugar de eliminarlas definitivamente (ilustración 18), por si en algún momento se necesita hacer uso de estos datos. Para ello, se crean las variables '**`nulls`**' y '**`country_column`**'. El comando **`df = df[~nulls]`** elimina del *dataframe* original los valores nulos.

```
df['children'].fillna(df['children'].median(), inplace=True)  ###Se reemplazan valores por mediana
df['country'].fillna(df['country'].mode()[0], inplace=True) ###Se reemplazan valores por moda
df['agent'].fillna(0, inplace=True) ###Se reemplazan valores por 0

country_column = df.pop('company') ###Se borra la columna country y se almacena en otra variable

nulls = (df.children == 0) & (df.adults == 0) & (df.babies == 0)
```

```
## Almacenar valores nulos en otro dataframe

df[nulls]
df = df[~nulls]
```

Ilustración 18 – Limpieza de valores nulos.

3.3. Diseño y evaluación de modelos de Aprendizaje Automático

En este apartado se procederá a la implementación de modelos de aprendizaje automático, explorando en primer lugar el conjunto de datos, seleccionando las variables más relevantes y desarrollando modelos predictivos que ayuden a entender y anticipar diferentes comportamientos y resultados dentro del contexto de las reservas de hotel.

3.3.1. Exploración del conjunto de datos y elección de los modelos de ML

Antes de comenzar a diseñar el modelo de *Machine Learning*, es necesario explorar el *dataframe* para ver qué información relevante extraer de él.

Uno de los objetivos será **predecir si una reserva de hotel será probablemente cancelada** (`is_canceled`), ya que este es un objetivo común y útil para la gestión de hoteles, permitiendo optimizar la ocupación y los ingresos. Para este fin, se utilizará un **modelo de regresión logística** debido a su adecuación para problemas de clasificación binaria, como es el caso de determinar si una reserva será cancelada o no. Este modelo es particularmente valioso por su simplicidad y efectividad. Además, permite manejar tanto variables numéricas como categóricas, una característica útil dado que los datos de las reservas del hotel incluyen una mezcla de ambos tipos. Para implementarlo, se hará uso de la biblioteca *scikit-learn* de Python.

El segundo objetivo consistirá en **generar un precio estimado a partir de unos valores de entrada**. Esto típicamente es el trabajo de un *Revenue Manager*, la persona encargada de fijar las tarifas de un hotel. La falta de variables en el *dataset* como información sobre la competencia, datos sobre la economía local, o eventos especiales limitará la capacidad modelo para capturar todas las dinámicas del mercado que influyen en los precios. Sin embargo, contamos con la variable `adr` indicativa de la tarifa media diaria. Este valor puede ser influenciado por varias de las variables disponibles como el tipo de habitación reservada, el mes de llegada, y el número de noches de estancia. Para este objetivo, se ha decidido usar un **modelo de regresión lineal** para predecir la tarifa diaria promedio (ADR). La regresión lineal

ofrece una forma sencilla de identificar cómo otras variables contribuyen a la tarifa de la habitación, facilitando la interpretación y aplicación de los resultados para tomar decisiones informadas sobre la gestión de precios en el hotel. Además, al ser un modelo inicial, establece una línea base para comparar y posiblemente mejorar con técnicas más complejas si las relaciones lineales resultan insuficientes.

3.3.2. Implementación de los modelos de ML

a) Modelo para predecir la posible cancelación de una reserva

Para el desarrollo del modelo predictivo enfocado en determinar si una reserva de hotel será cancelada, se ha implementado un modelo de regresión logística. A continuación, se describe cada paso llevado a cabo:

- Creación de la instancia del modelo: se inició instanciando un modelo de regresión logística utilizando la biblioteca scikit-learn.
- Definición de la variable objetivo: la variable **`is_canceled`** fue identificada como la variable objetivo para el modelo, representando si una reserva ha sido cancelada o no.
- Selección de características: se seleccionaron varias características del conjunto de datos basadas en su relevancia potencial para predecir el resultado de la reserva. Las características incluidas fueron:
    - **`lead_time`**: tiempo en días entre la reserva y la llegada.
    - **`total_of_special_requests`**: número total de solicitudes especiales hechas por el cliente.
    - **`previous_cancellations`**: número de cancelaciones previas hechas por el cliente.
    - **`adults, children, babies`**: número de adultos, niños y bebés dentro de la reserva.
    - **`booking_changes:`** número de cambios hechos a la reserva.

Estas características fueron elegidas basadas en la hipótesis de que influirían significativamente en la probabilidad de cancelación de una reserva.

- División de datos en conjuntos de entrenamiento y prueba: para evaluar la eficacia del modelo de manera objetiva, el conjunto de datos fue dividido en dos partes: un conjunto de entrenamiento y un conjunto de prueba (ilustración 19), con el **80% de los datos destinados para entrenamiento** y el **20% para prueba**. Esta división fue realizada usando la función **`train_test_split`** de scikit-learn, con un **`random_state`** de 42 para garantizar la reproducibilidad de los resultados.

```
# Crear una instancia del modelo de regresión logística
model = LogisticRegression()

#'is_canceled' es la variable a predecir
target = df['is_canceled']

# Selección de algunas características consideradas relevantes
features = df[['lead_time', 'total_of_special_requests', 'previous_cancellations', 'adults', 'children', 'babies', 'booking_cha

# Dividir los datos en conjunto de entrenamiento y de prueba
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)
```

Ilustración 19 - Generación del modelo de regresión logística.

El paso de entrenar el modelo es fundamental en el proceso de aprendizaje automático. Utilizando la instancia del modelo de regresión logística creada previamente, se procedió a entrenarlo con el conjunto de datos de entrenamiento usando el método **`fit()`.**

```
In [15]: ## Entrenar el modelo
         model.fit(X_train, y_train)

Out[15]: LogisticRegression()
```

Ilustración 20 - Entrenamiento del modelo.

- Evaluación de la efectividad del modelo: **`accuracy_score`** y **`classification_report`** son importados desde la biblioteca de Scikit-Learn. Estas funciones son utilizadas para medir la precisión global del modelo y proporcionar un informe detallado de las métricas de clasificación, respectivamente. El método **`predict()`** se utiliza para realizar predicciones sobre el conjunto de prueba **`X_test`**. El método devuelve **`y_pred`**, que contiene las etiquetas para cada entrada en el conjunto de prueba. Estas etiquetas pre marcadas representan si el modelo cree que cada reserva será cancelada (1) o no (0). El resultado se observa en la siguiente imagen:

```
In [16]: from sklearn.metrics import accuracy_score, classification_report

         # Predecir las etiquetas para el conjunto de prueba
         y_pred = model.predict(X_test)

         # Calcular la precisión y generar un informe
         print("Accuracy:", accuracy_score(y_test, y_pred))
         print(classification_report(y_test, y_pred))

Accuracy: 0.7140760003355423
              precision    recall  f1-score   support

           0       0.72      0.89      0.80     14958
           1       0.70      0.41      0.52      8884

    accuracy                           0.71     23842
   macro avg       0.71      0.65      0.66     23842
weighted avg       0.71      0.71      0.69     23842
```

Ilustración 21 – Evaluación de la efectividad del modelo.

Los resultados obtenidos muestran una precisión general (*accuracy*) del 71,4%, lo cual es un buen punto de partida. A continuación, se detalla cada métrica del reporte:

- *Accuracy* (precisión general): 71% indica que aproximadamente 71 de cada 100 predicciones fueron correctas. Este es un resultado decente, pero la efectividad del modelo puede variar dependiendo de la complejidad del problema y la calidad de los datos.
- Precisión: para la clase 0 (no cancelada), la precisión es 0,72, lo que significa que de todas las reservas que el modelo predijo como no canceladas, el 72% efectivamente no fueron canceladas. Para la clase 1 (cancelada), la precisión es 0,70, indicando que el 70% de las predicciones de cancelaciones fueron correctas.
- *Recall* (sensibilidad): para la clase 0, la sensibilidad es 0,89, lo que significa que el modelo fue capaz de identificar el 89% de todas las reservas no canceladas actualmente. Para la clase 1, el *recall* es mucho más bajo, 0,41, lo que indica que el modelo solo identificó correctamente el 41% de todas las reservas canceladas realmente.
- F1-Score: se trata de una medida que combina la precisión y el *recall* en un solo número. Para la clase 0, es 0,80, lo que es relativamente alto. Para la clase 1, es 0,52, lo que es significativamente más bajo y sugiere que el modelo no es tan bueno para identificar reservas canceladas como para identificar reservas no canceladas.

- Matriz de confusión: la matriz de confusión es una tabla que resume cómo de exitoso es el modelo de clasificación al predecir ejemplos que pertenecen a varias clases. Un eje de la matriz de confusión es la etiqueta que el modelo predijo, y el otro eje es la etiqueta real [1]. En el modelo, se ha obtenido la siguiente matriz:

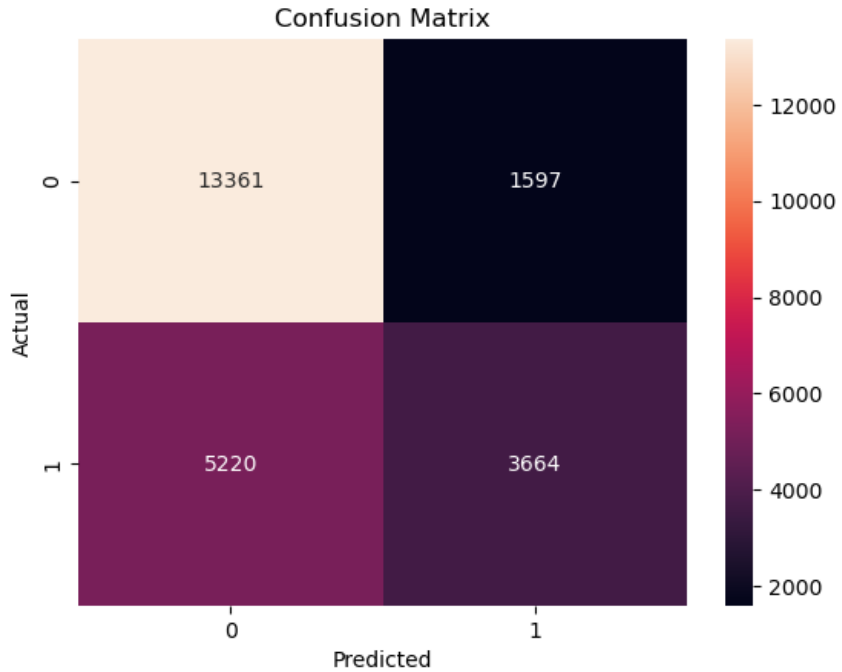


Ilustración 22 - Matriz de confusión.

- Celda superior izquierda (verdaderos negativos): 13.361 reservas que el modelo predijo correctamente como no canceladas.
- Celda superior derecha (falsos positivos): 1.597 reservas que el modelo incorrectamente predijo como canceladas, pero que en realidad no lo fueron.
- Celda inferior izquierda (falsos negativos): 5.220 reservas que el modelo incorrectamente predijo como no canceladas, pero que en realidad fueron canceladas.
- Celda inferior derecha (verdaderos positivos): 3.664 reservas que el modelo predijo correctamente como canceladas.

- Mejora del modelo: un alto *recall* en la predicción de cancelaciones es importante porque significa que el modelo es capaz de identificar la mayoría de las cancelaciones reales. Esto es importante para los hoteles porque les permite reaccionar a tiempo, por ejemplo, poniendo las habitaciones nuevamente disponibles y evitando pérdidas de ingresos. Para ello, se aplicó un umbral de decisión ajustado de 0,3 para la clasificación, es decir, las reservas con una probabilidad de cancelación superior al 30% se clasificaron como canceladas. Esta modificación del umbral está diseñada para ser más inclusiva de posibles cancelaciones, potencialmente aumentando el *recall* del modelo a expensas de disminuir la precisión.

```
# Aplicar un nuevo umbral
y_pred_umbral = np.where(probabilidades > 0.3, 1, 0)  # Umbral ajustado a 0,3

# Evaluar el nuevo umbral
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred_umbral))
```

```
Accuracy: 0.7140760003355423
              precision    recall  f1-score   support

           0       0.83      0.55      0.66     14958
           1       0.52      0.80      0.63      8884

    accuracy                           0.65     23842
   macro avg       0.67      0.68      0.65     23842
weighted avg       0.71      0.65      0.65     23842
```

Ilustración 23 - Efectividad del modelo tras ajuste del umbral.

La nueva matriz de confusión es la siguiente:

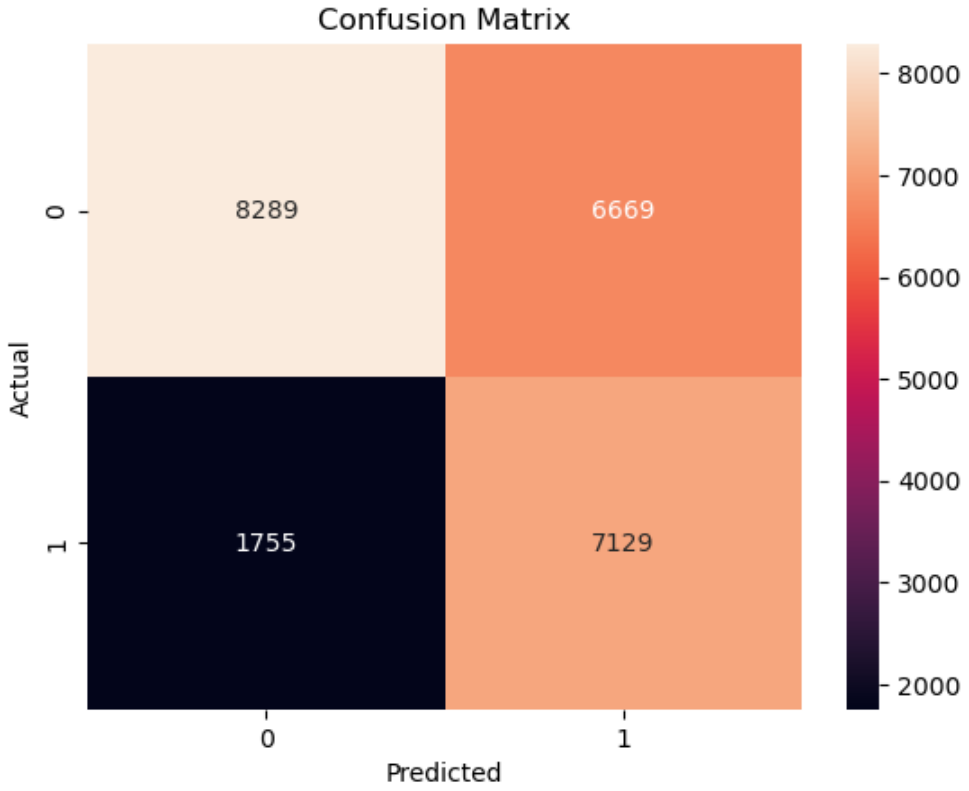


Ilustración 24 - Matriz de confusión tras ajuste del umbral.

La nueva matriz muestra un *recall* significativamente más alto (80% vs. 41%), lo cual es crucial en el contexto de la predicción de cancelaciones de hoteles, como se explicó anteriormente. Aunque la precisión es menor (51% vs. 69%), el *recall* más alto significa que el modelo es más eficaz para capturar cancelaciones, lo cual puede ser más valioso para un hotel que quiere maximizar la capacidad de reacción ante cancelaciones potenciales.

b) Modelo para generar una tarifa estimada

Para poder utilizar los datos categóricos en un modelo de regresión lineal, se ha implementado una técnica conocida como "*one-hot encoding*" para transformar variables en un formato que pueda ser interpretado por modelos estadísticos y de aprendizaje automático. Las columnas específicas que se han codificado son '**`meal`**', '**`arrival_date_month`**', '**`hotel`**', y '**`reserved_room_type`**' (ilustración 25). Cada una de estas columnas contiene datos categóricos que representan diferentes categorías como tipos de comidas, meses, nombres de hoteles y tipos de habitaciones reservadas.

El proceso de *one-hot encoding* convierte cada categoría única dentro de una columna en una nueva columna binaria (0 o 1). Esto significa que, por ejemplo, como la columna '**`meal`**' contiene cuatro opciones distintas para el régimen de comidas, se crearán cuatro nuevas

columnas, cada una representando uno de estos tipos, donde el valor será 1 si la observación corresponde a ese tipo de comida y 0 si no.

```
# Lista de columnas para one-hot encoding
columns_to_encode = ['meal', 'arrival_date_month', 'hotel', 'reserved_room_type']

# Aplicar one-hot encoding
df = pd.get_dummies(df, columns=columns_to_encode)

# Imprimir las primeras filas para verificar los cambios
print(df.head())
```

Ilustración 25 - Aplicación de one-hot encoding a variables categóricas.

A continuación, se describen los pasos seguidos para la generación del modelo de regresión lineal:

- Creación de la instancia del modelo: se crea una instancia del modelo de regresión lineal utilizando la clase **`LinearRegression()`**, obtenida también de la biblioteca Scikit-Learn.
- Selección de las características: se procede a la selección de características que serán utilizadas para entrenar el modelo. Esto incluye las características numéricas como '**`adults`**', '**`children`**', y '**`babies`**'. Además, se añaden todas las columnas generadas por el proceso de *one-hot encoding*.
- Definición de la variable objetivo: con las características definidas, se crea el conjunto de datos X que contiene las variables independientes e Y que contiene la variable dependiente, en este caso, '**`adr`**'. Se procede a dividir estos datos en conjuntos de entrenamiento y prueba utilizando la función **`train_test_split()`**, asignando el 20% de los datos al conjunto de prueba y el resto al conjunto de entrenamiento.
- Entrenamiento del modelo: el modelo de regresión lineal es entrenado usando el conjunto de datos de entrenamiento (**`X_train`** e **`y_train`**). Este proceso ajusta el modelo para minimizar el error entre las predicciones y los valores reales, utilizando los métodos matemáticos subyacentes de la regresión lineal para encontrar la mejor línea de ajuste. El entrenamiento del modelo (ilustración 26) busca comprender cómo las características de las reservas del hotel influyen en su tarifa diaria promedio, con el objetivo de utilizar este modelo para hacer predicciones precisas sobre nuevos datos en el futuro.

```
# Crear una instancia del modelo de regresión lineal
model = LinearRegression()

# Lista todas las columnas
features = ['adults', 'children', 'babies'] + [col for col in df.columns if col.startswith(('meal_', 'arrival_date_mont

# Selección de características
X = df[features]
y = df['adr']

# Dividir los datos en conjunto de entrenamiento y de prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Entrenar el modelo
model.fit(X_train, y_train)
```

Ilustración 26 - Generación del modelo de regresión lineal.

El modelo de regresión lineal entrenado se utiliza para realizar predicciones sobre el conjunto de prueba. La función **`model.predict(X_test)`** se emplea para obtener las predicciones de las tarifas diarias promedio (‘**`adr`**’) basadas en las características del conjunto de prueba **`X_test`**. Estas predicciones son almacenadas en la variable **`y_pred`**.

Una vez obtenidas las predicciones, se calcula el error cuadrático medio (MSE) usando la función **`mean_squared_error(y_test, y_pred)`**. El MSE es una métrica común que mide el promedio de los cuadrados de los errores, es decir, la diferencia cuadrática entre los valores observados reales y los valores predichos por el modelo. Posteriormente y como se observa en la ilustración 27, se extrae la raíz cuadrada del MSE para obtener el error cuadrático medio raíz (RMSE), que proporciona una medida en las mismas unidades que la variable de respuesta y ofrece una interpretación más directa del promedio de error en las predicciones del modelo.

```
from sklearn.metrics import mean_squared_error

# Predecir las tarifas para el conjunto de prueba
y_pred = model.predict(X_test)

# Calcular RMSE
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print("RMSE:", rmse)
```

```
RMSE: 35.26137795550806
```

Ilustración 27 - Cálculo del RMSE.

Como se observa, el RMSE obtenido es de 35,26, es decir, en promedio, las predicciones del modelo se desvían en 35 unidades de los valores reales. La interpretación de si este valor de RMSE es alto o bajo depende del rango y la escala de las tarifas que se están modelando. En este caso, las tarifas varían aproximadamente entre 30 y 300, por lo que un RMSE de 35 puede ser razonablemente bueno. El gráfico resultante del modelo es el siguiente:

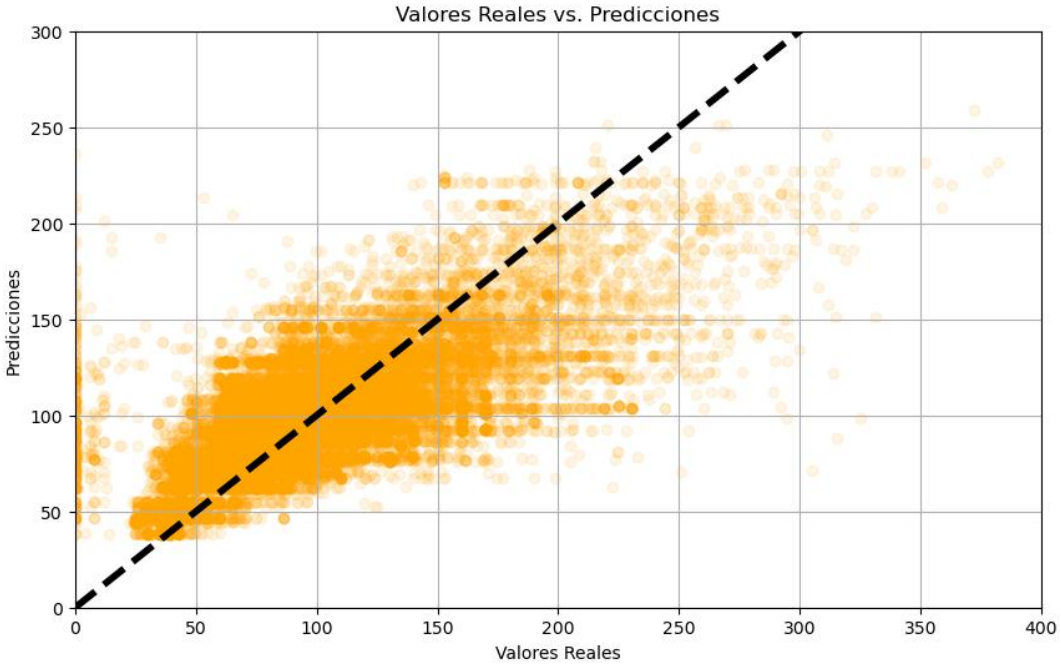


Ilustración 28 - Resultado del modelo de regresión lineal.

3.4.Integración en Power BI

Para implementar los modelos de aprendizaje automático desarrollados con Python en Power BI, el primer paso es exportar los datos que hemos modificado en Jupyter Notebooks. Para ello, se escribe la siguiente línea de comando:

```
df.to_csv ('./hotel_bookings_modificado.csv',index=False)
```

Una vez exportados los datos, el siguiente paso es abrir un nuevo informe en Power BI, importar los datos en formato CSV [14] y cargarlos:

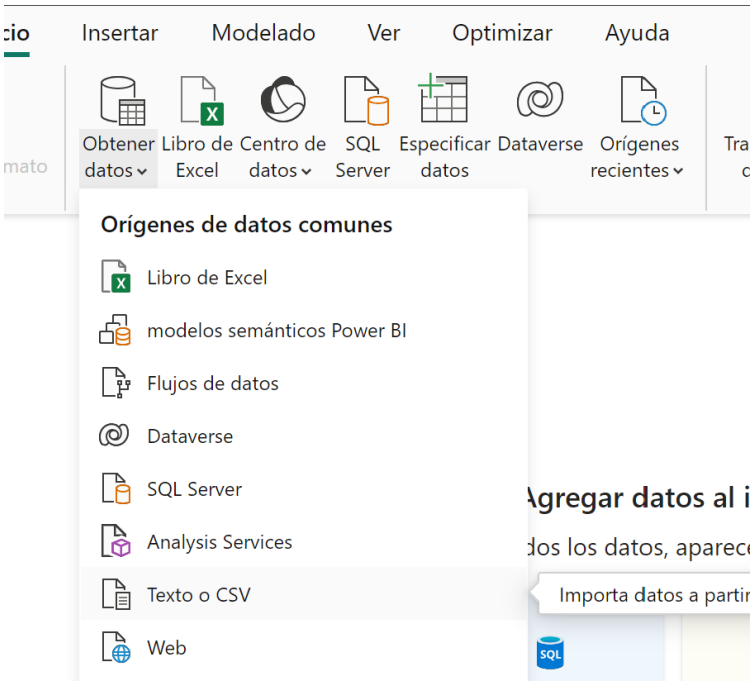


Ilustración 29 – Importación datos en Power BI.

A continuación, se procede a transformar los datos (ilustración 30) para ajustar detalles específicos, como el formato de los números. En el *dataset* original, los datos están en formato americano (o inglés), donde se utiliza el punto (.) en lugar de la coma (,) para los decimales. Este formato puede causar problemas al importar los datos a Power BI, ya que la herramienta no detecta correctamente los valores numéricos y, si se convierten directamente a números enteros, no se reconocerán las comas. Por lo tanto, es necesario convertir primero el punto (.) en coma (,) y, posteriormente, cambiar el formato a *float*. De manera similar, las fechas en el *dataset* no son detectadas correctamente por Power BI, lo que requiere una especificación adicional para su correcta interpretación.

Una vez realizados estos ajustes en el formato de los datos, se hace clic en el botón "Ejecutar script de Python" (ilustración 31). Este paso permite introducir el código de los modelos de *machine learning* que se han desarrollado previamente en Jupyter Notebooks (ilustración 32), integrando así el análisis y las predicciones en la visualización de datos dentro de Power BI. Este procedimiento asegura que los datos sean interpretados y procesados correctamente, facilitando un análisis más preciso y eficaz.

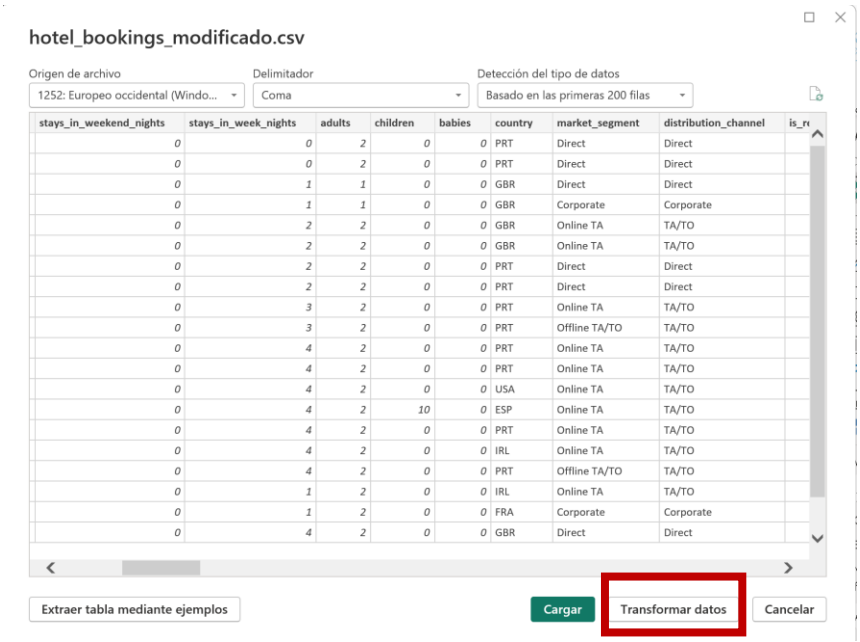


Ilustración 30 - Transformar datos.

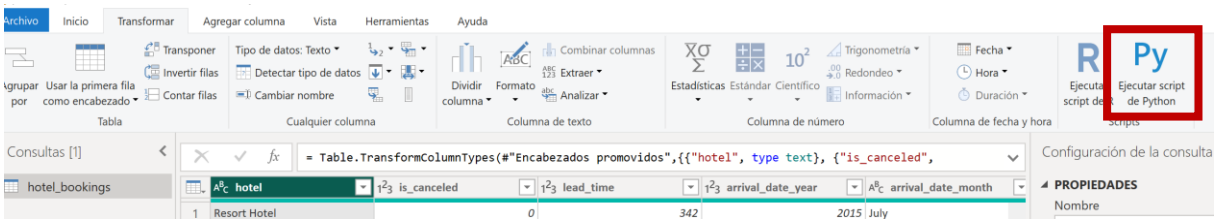


Ilustración 31 - Ejecutar script de Python.

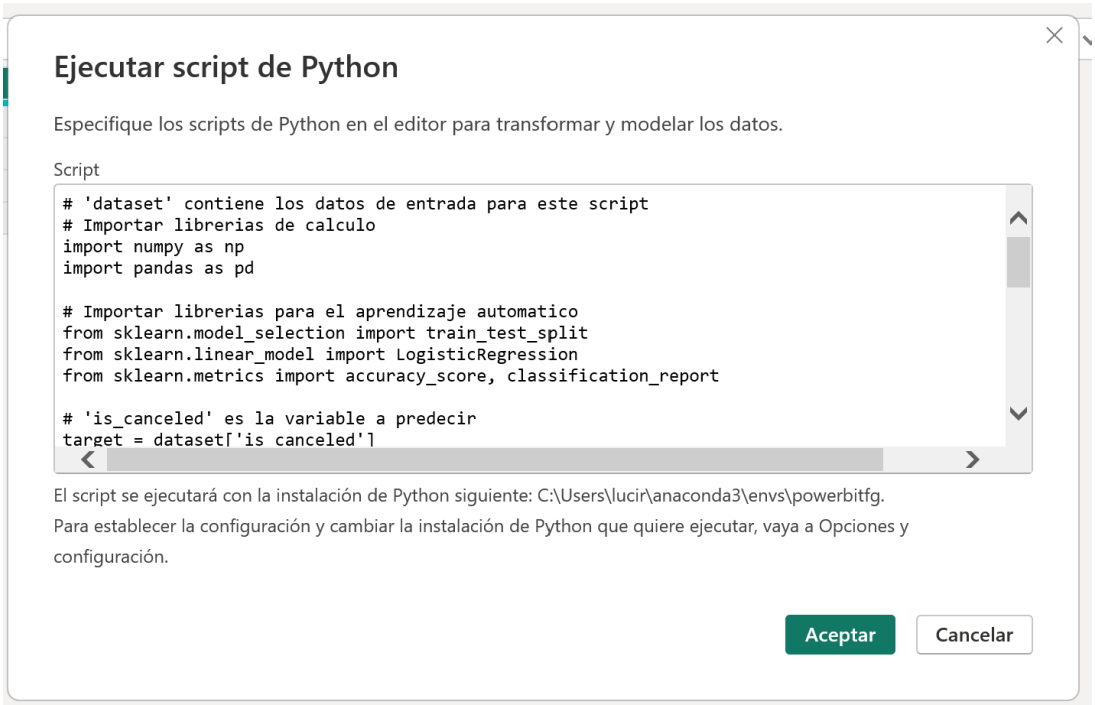


Ilustración 32 - Insertar script de los modelos de aprendizaje automático.

Es importante tener en cuenta que Power BI no detecta la nomenclatura *df* a la hora de insertar los datos, por lo que hay que modificar esto por *dataset* para asegurar el buen funcionamiento del *script*. El *script* introducido solo contiene las librerías necesarias para el cálculo y la generación de modelos automáticos, asi como la generación, entrenamiento y predicciones de los modelos. Como se mencionó previamente, las librerías de visualización no son necesarias aquí puesto que Power BI ya genera sus propias visualizaciones. El fragmento de código introducido es el siguiente:

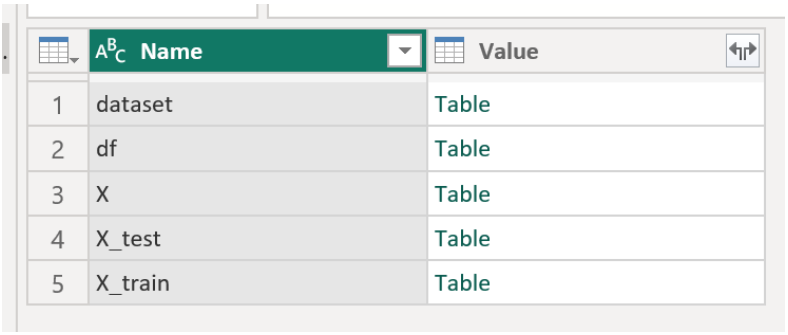
```
# 'dataset' contiene los datos de entrada para este script

# Importar librerías de cálculo
import numpy as np
import pandas as pd

# Importar librerías de visualización
import matplotlib.pyplot as plt
import seaborn as sns

# Importar librerías para el aprendizaje automático
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

df = dataset

# Crear una instancia del modelo de regresión lineal
model = LinearRegression()

# Lista todas las columnas
features = ['adults', 'children', 'babies'] + [col for col in df.columns if col.
 ↪startswith(('meal_', 'arrival_date_month_', 'hotel_',␣
 ↪'reserved_room_type_'))]

# Selección de características
X = df[features]
y = df['adr']

# Dividir los datos en conjunto de entrenamiento y de prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Entrenar el modelo
model.fit(X_train, y_train)

# Predecir las tarifas para el conjunto de prueba
y_pred = model.predict(X_test)
```

```
df['prediccion_adr'] = np.nan

# Asignar las predicciones a sus respectivas filas utilizando los índices de␣
 ↪X_test
df.loc[X_test.index, 'prediccion_adr'] = y_pred

# Aplicar el modelo a todo el dataset para predecir ADR
prediccion_adr_completa = model.predict(X)

# Guardar las predicciones para todo el dataset
df['prediccion_adr_completa'] = prediccion_adr_completa

# Crear una instancia del modelo de regresión logística
model = LogisticRegression()

# 'is_canceled' es la variable a predecir
target = df['is_canceled']

# Selección de algunas características consideradas relevantes
features = df[['lead_time', 'total_of_special_requests',␣
 ↪'previous_cancellations', 'adults', 'children', 'babies', 'booking_changes']]

# Dividir los datos en conjunto de entrenamiento y de prueba
X_train, X_test, y_train, y_test = train_test_split(features, target,␣
 ↪test_size=0.2, random_state=42)

# Entrenar el modelo
model.fit(X_train, y_train)

# Predecir las etiquetas para el conjunto de prueba
y_pred = model.predict(X_test)

# Aplicar un nuevo umbral
probabilidades = model.predict_proba(X_test)[:, 1]
y_pred_umbral = np.where(probabilidades > 0.3, 1, 0)  # Umbral ajustado a 0.3

# Guardar la predicción resultante en una nueva columna en el dataset
df.loc[X_test.index, 'prediccion_cancelacion'] = y_pred_umbral

# Aplicar el modelo a todo el dataset para predecir las cancelaciones
probabilidades_completas = model.predict_proba(features)[:, 1]
predicciones_completas = np.where(probabilidades_completas > 0.3, 1, 0)  # Usar␣
 ↪el mismo umbral

# Guardar las predicciones para todo el dataset
dataset['prediccion_cancelacion_completa'] = predicciones_completas
```

Ilustración 33 - Script introducido en Power BI.

La idea detrás de la implementación en Power BI es que los datos se actualicen automáticamente a medida que ingresan nuevas reservas. Esto asegura que las predicciones se generen de manera continua y en tiempo real, gracias a la integración del script de aprendizaje automático dentro de Power BI. Al tener el script implementado en esta plataforma, cada vez que se actualizan los datos de reservas, las predicciones se recalculan automáticamente sin necesidad de intervención manual.

El resultado de ejecutar este *script* es la siguiente tabla, de la cual se seleccionará *dataset* para obtener los datos actualizados:



	Name	Value
1	dataset	Table
2	df	Table
3	X	Table
4	X_test	Table
5	X_train	Table

Ilustración 34 - Resultado de ejecutar el script.

3.5. Análisis y resultados

Se ha elaborado un informe que ejemplifica la utilidad de la integración de los modelos de aprendizaje automático en Power BI. Una página del informe está dedicada a la detección de posibles cancelaciones y la otra a la estimación del ADR (tarifa promedio diaria).

3.5.1. Detección de cancelaciones

La siguiente página del informe ofrece una visión de las predicciones de cancelaciones utilizando el modelo de regresión logística:

Ilustración 35 - Página 1 del informe de Power BI para la predicción de cancelaciones.

En el centro del informe, destacan dos tarjetas de resumen importantes. La primera tarjeta indica el número de reservas detectadas como cancelaciones por el modelo, que es de 69.338. La segunda tarjeta muestra el número de reservas canceladas reales, que es de 44.199. La diferencia entre estas dos cifras indica que el modelo predictivo está detectando más cancelaciones de las que realmente ocurren, esto es debido a la elección que se tomó de aumentar el *recall* en el proceso de mejora del modelo. Este resultado, aunque no ideal, no es completamente negativo. Permite a los gestores del hotel identificar reservas potencialmente problemáticas y tomar medidas preventivas, contactar a los clientes para confirmar sus reservas, ofrecer incentivos para que mantengan la reserva o revisar las políticas de cancelación.

A la derecha, se muestra una tabla con las últimas reservas entrantes. La columna “Posible cancelación”, representa si una reserva es propensa a ser cancelada (1) o no (0). Esta tabla facilita el análisis de las reservas más recientes y sus probabilidades de cancelación según el modelo.

En la parte inferior del informe, se encuentran dos tablas adicionales que comparan las características de las reservas canceladas y no canceladas. La segunda tabla, titulada "Características de la predicción", compara las mismas características que la primera, pero para las predicciones del modelo. Estas tablas permiten observar las diferencias y similitudes entre las reservas canceladas y no canceladas, tanto en los datos reales como en las predicciones del modelo. Se observa que en lo que concierne a las características que determinan la probabilidad de que una reserva se cancele, el modelo generado se ajusta bastante bien.

3.5.2. Predicción del ADR

En esta segunda página, se muestra el resultado de las predicciones del ADR, es decir, de la tarifa promedio diaria del hotel:

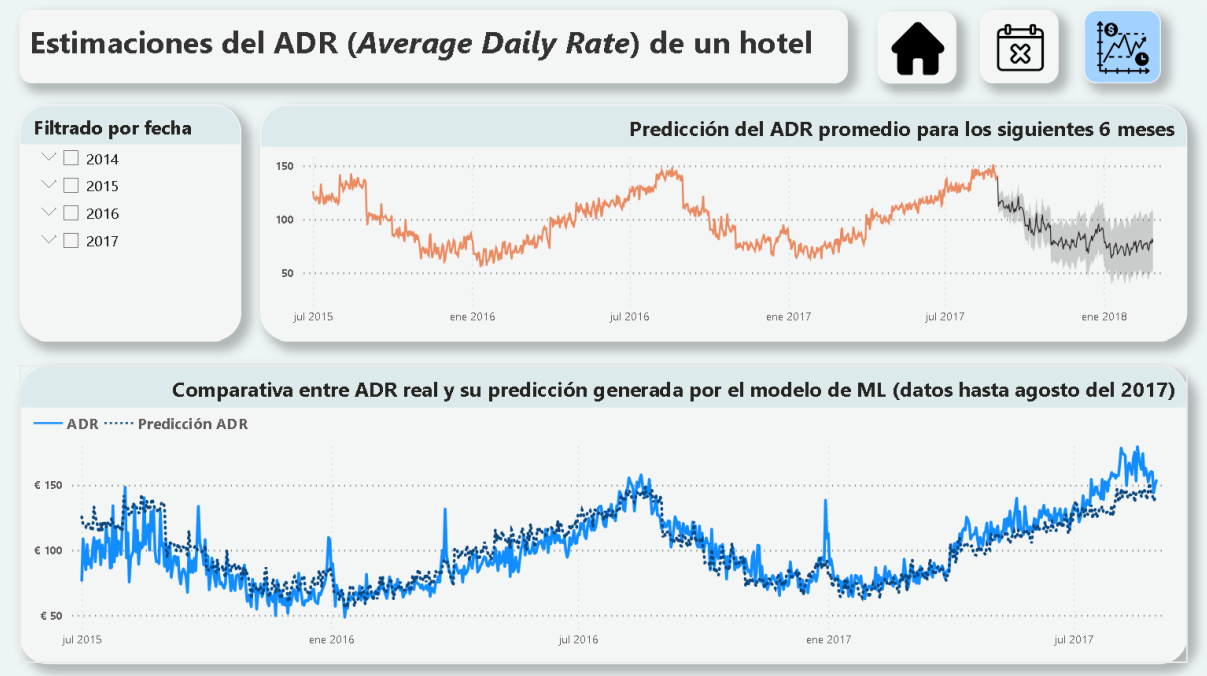


Ilustración 36 - Página 2 del informe de Power BI para la predicción del ADR.

En el gráfico de líneas de arriba a la derecha, se ha utilizado una herramienta especifica de Power BI para predicciones. En el panel de "*Analytics*", se añadió una nueva predicción seleccionando la opción “Predicción” [19]. Se configuraron los siguientes parámetros para ajustar la predicción:

- Las unidades de tiempo se establecieron en días, y se configuró la duración de la predicción para 180 días, equivalente a aproximadamente 6 meses. Además, se dejó en 0 la opción de omitir puntos de datos del final de la serie, asegurando que se utilizara toda la información disponible para la predicción.

- Para capturar patrones anuales en los datos, se configuró la estacionalidad en 365 puntos, asumiendo que hay una estacionalidad anual en las tarifas diarias promedio. Finalmente, se estableció un intervalo de confianza del 95%, lo que proporciona una banda alrededor de la predicción que indica el rango dentro del cual se espera que caigan los valores reales con un 95% de probabilidad.

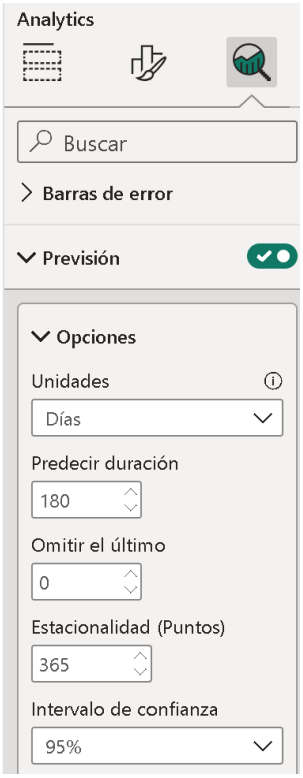


Ilustración 37 - Ajustes de la previsión en Power BI.

La predicción del ADR para los siguientes seis meses permite a los gestores anticipar ingresos y ajustar sus estrategias de precios y marketing en consecuencia. Por ejemplo, identificar períodos con tarifas más bajas puede permitir la implementación de promociones o descuentos para atraer más clientes, optimizando así la ocupación y los ingresos del hotel. Además, la comparación entre las predicciones del modelo de aprendizaje automático y los datos reales permite evaluar la precisión y efectividad del modelo. Esta evaluación permite identificar posibles áreas de mejora. Al realizar estos ajustes, se puede mejorar la exactitud de las predicciones futuras, lo que a su vez mejora la toma de decisiones basada en datos.

El informe completo de Power BI se puede visualizar escaneando el siguiente código QR o haciendo click en el enlace:

https://bit.ly/tfgluciarobles

4. CONCLUSIONES

Por último, se resumen las principales conclusiones obtenidas de este estudio, las limitaciones encontradas y direcciones futuras:

4.1. Implicaciones prácticas y teóricas de la investigación

Las principales implicaciones prácticas de este trabajo se centran en la mejora de la toma de decisiones operativas y estratégicas en la gestión hotelera. Una de las áreas más beneficiadas por estos avances es el *Revenue Management*. Esta práctica se enfoca en maximizar los ingresos del hotel mediante la gestión estratégica de los precios y la ocupación. Con las predicciones de ADR generadas por los modelos de aprendizaje automático, los gestores pueden ajustar dinámicamente los precios de las habitaciones en función de la demanda prevista, las tendencias de mercado y el comportamiento histórico de los clientes. Este enfoque basado en datos permite a los hoteles establecer tarifas más competitivas y atractivas, ajustando los precios en tiempo real para maximizar los ingresos durante períodos de alta demanda y minimizar las pérdidas en períodos de baja ocupación. Además, la capacidad de predecir cancelaciones de reservas permite a los gestores implementar políticas más efectivas para reducir el impacto de las cancelaciones de última hora. Al identificar reservas con alta probabilidad de ser canceladas, los hoteles pueden tomar medidas preventivas, como ofrecer incentivos para asegurar la reserva o ajustar sus políticas de cancelación para reducir la incidencia de cancelaciones.

Es importante destacar que, aunque estos modelos de aprendizaje automático son herramientas poderosas que pueden mejorar significativamente la eficiencia y precisión en la toma de decisiones, no son sustitutos de la supervisión humana. Los modelos pueden proporcionar información valiosa y predicciones basadas en patrones de datos históricos, pero los gestores deben interpretar y validar estos resultados, considerando contextos específicos y variables que el modelo podría no haber tenido en cuenta. La supervisión humana es necesaria para garantizar que las decisiones tomadas a partir de estos modelos sean apropiadas y alineadas con los objetivos estratégicos del hotel.

Desde una perspectiva teórica, esta investigación contribuye al campo de la ciencia de datos al demostrar cómo los modelos de aprendizaje automático pueden ser aplicados efectivamente en la industria hotelera. La investigación destaca la importancia de seleccionar y preprocesar adecuadamente los datos, y cómo el ajuste de parámetros del modelo puede influir significativamente en la precisión de las predicciones.

Hasta ahora, muchas empresas de alojamientos turísticos han dependido en gran medida de Excel para gestionar sus datos operativos. Aunque Excel es una herramienta muy buena para la manipulación y el análisis básico de datos, su uso tradicionalmente no ha permitido una profundización real en los datos para extraer *insights* más complejos. Los análisis suelen ser estáticos y limitados a operaciones manuales, lo que dificulta la detección de patrones y tendencias más sutiles que podrían influir en la toma de decisiones estratégicas. La integración

de estos modelos con herramientas de visualización como Power BI subraya la relevancia de combinar análisis avanzado de datos con visualizaciones intuitivas para facilitar la toma de decisiones basadas en datos. Esta investigación ofrece una base teórica para futuras investigaciones en la aplicación de modelos de aprendizaje automático y visualización de datos en la industria.

4.2. Limitaciones del estudio y direcciones futuras

A pesar de los hallazgos positivos y las aplicaciones prácticas de este trabajo, existen varias limitaciones que deben ser reconocidas:

En primer lugar, algunos datos relevantes, como la información sobre la competencia, eventos locales o condiciones económicas no están disponibles en el conjunto de datos utilizado. La ausencia de estas variables puede limitar la capacidad de los modelos para capturar todos los factores que influyen en las reservas y tarifas. El uso de fuentes de datos externas y técnicas avanzadas de enriquecimiento de datos puede proporcionar un contexto más completo para el análisis.

Por otro lado, el enfoque en técnicas específicas de aprendizaje automático, como la regresión logística y la regresión lineal, también representa una limitación. Si bien estos modelos son útiles y relativamente fáciles de interpretar, existen otros métodos más avanzados que podrían mejorar la precisión de las predicciones, como los modelos de árboles de decisión o las redes neuronales.

4.3. Reflexión personal final

Este trabajo representa para mí un primer acercamiento práctico a la ciencia de datos, con resultados satisfactorios a pesar de que los modelos generados son relativamente básicos. Para ser un primer intento, los resultados obtenidos se consideran adecuados. Este proyecto me ha permitido aplicar conceptos teóricos aprendidos durante el grado en un contexto real, proporcionando una comprensión más profunda de cómo la ciencia de datos puede resolver problemas prácticos en la industria hotelera.

El proceso de recopilación de información y gestión de los errores derivados de la programación en Python ha supuesto un verdadero reto. La limpieza y preparación de datos, la implementación de modelos de aprendizaje automático y su integración en Power BI requirieron una capacidad de resolución de problemas constante. Enfrentar y superar estos desafíos ha sido una experiencia enriquecedora que me ha ayudado a desarrollar habilidades en el manejo de datos y en la programación.

En resumen, este proyecto ha sido una experiencia formativa significativa, proporcionando una base sólida en ciencia de datos y preparándome para abordar proyectos más complejos en el futuro. La motivación para seguir aprendiendo y mejorando mis habilidades en este campo se ha incrementado, así como el entusiasmo por las oportunidades que esta área ofrece para la innovación y la mejora continua en diversas industrias.

5. BIBLIOGRAFÍA

[1] Burkov A. (2019) *The Hundred-Page Machine Learning Book*.

[2] Grus, J. (2015) *Data Science from Scratch: First Principles With Python*. 2nd Edition. O'Reilly Media.

[3] Matthes, E. (2016) *Python Crash Course*.

[4] Alotaibi, Eid. (2020). *Application of Machine Learning in the Hotel Industry: A Critical Review*. Journal of Association of Arab Universities for Tourism and Hospitality. Recuperado el 4 de marzo de 2024 de https://jaauth.journals.ekb.eg/article_108732.html

[5] Van Leeuwen, R., & Koole, G. (2022). *Data-driven market segmentation in hospitality using unsupervised machine learning*. Machine Learning with Applications, 10, 100414.

[6] OpenAI. (2024). *ChatGPT-4*. [Modelo de lenguaje de gran tamaño]. https://chat.openai.com/chat

[7] Google Cloud. *¿Qué es el aprendizaje automático?* Recuperado el 23 de febrero de 2024 de https://cloud.google.com/learn/what-is-machine-learning?hl=es-419#:~:text=Descargar%20la%20gu%C3%ADa-,Definici%C3%B3n%20de%20aprendizaje%20autom%C3%A1tico,de%20grandes%20cantidades%20de%20datos

[8] Kaggle. *Hotel booking demand*. (2020). Recuperado el 16 de marzo de 2024 de https://www.kaggle.com/datasets/jessemostipak/hotel-booking-demand

[9] Data Science Stack Exchange. (s.f.) *Do I need to convert booleans to ints to enter them in a machine learning algorithm?* Recuperado el 23 de marzo de 2024 de https://datascience.stackexchange.com/questions/42465/do-i-need-to-convert-booleans-to-ints-to-enter-them-in-a-machine-learning-algori

[10] W3Schools. (s.f.) *Python Tutorial*. https://www.w3schools.com/python/default.asp

[11] CódigoFacilito. (s.f.) *Fechas con Python*. Recuperado el 23 de marzo de 2024 de https://codigofacilito.com/articulos/fechas-python

[12] Niteshyadav. (2021). *Hotel Booking Prediction (99.5% acc).* Kaggle. Recuperado el 24 de marzo de 2024 de https://www.kaggle.com/code/niteshyadav3103/hotel-booking-prediction-99-5-acc

[13] IBM. (s. f.) *¿Qué es un árbol de decisión?* Recuperado el 29 de marzo de 2024 de https://www.ibm.com/es-es/topics/decision-trees

[14] Davidiseminger. (2024). *Orígenes de datos en Power BI Desktop - Power BI*. Microsoft Learn. Recuperado el 9 de mayo de 2024 de https://learn.microsoft.com/es-es/power-bi/connect-data/desktop-data-sources

[15] Wikipedia. (2023). *Agente inteligente (inteligencia artificial)*. Wikipedia, la Enciclopedia Libre. Recuperado el 16 de mayo de 2024 de https://es.wikipedia.org/wiki/Agente_inteligente_(inteligencia_artificial)

[16] Torres, L. (2021). *¿En qué consiste la regresión logística? ¿Qué es la regularización?* The Machine Learners. Recuperado el 16 de mayo de 2024 de https://www.themachinelearners.com/regresion-logistica-regularizacion/

[17] Porto, J. P., & Merino, M. (2022*). Centroide. Qué es, características, en la física y en la economía*. Definición.de. Recuperado el 16 de mayo de 2024 de https://definicion.de/centroide/

[18] MDN Web Docs. (s.f.) *Array - JavaScript*. Recuperado el 24 de mayo de 2024 de https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Array

[19] Dataseito. (2020). *Power BI et prévision / forecast en 5 min* [Vídeo]. YouTube. Recuperado el 26 de mayo de 2024 de https://www.youtube.com/watch?v=gVV2pL0rkkQ