AUTOMATED DATA DISCOVERY AND PROFILING IN SOCIAL NETWORK ANALYSIS

Trabajo Fin de Máster

Máster Universitario en Big Data Science

Curso académico 2023-2024

Autor: Mario Lamas Herrera

Autor: Pablo Legerén Somolinos

Tutor Académico: Horacio Grass Boada

Tutor de empresa: Pilar Caruso

Tutor de empresa: Alfonso Sopelana Arrarte

Madrid, 2024

RESUMEN

Cada vez es más común que los viajeros opten por apartamentos particulares en lugar de hoteles debido a la flexibilidad, comodidad e independencia que ofrecen. Sin embargo, seleccionar el apartamento óptimo entre la gran cantidad de opciones disponibles puede ser un desafío. Al mismo tiempo, los propietarios necesitan mantenerse informados sobre las tendencias del mercado y las necesidades de los consumidores para satisfacer eficazmente estas demandas. Este proyecto aborda estos problemas proporcionando recomendaciones personalizadas a los usuarios según sus perfiles y ofreciendo un análisis detallado del mercado y de los competidores a los propietarios.

El sistema de recomendación utiliza datos obtenidos directamente desde la propia web, que son limpiados y procesados para predecir las preferencias de los usuarios y así poder emparejarlos con alojamientos óptimos. Además, el sistema analiza variables relevantes del mercado y obtiene información de las reseñas de los arrendadores para ayudar a los propietarios a entender y adaptarse a las dinámicas y necesidades del mercado. En este proyecto se propone como posible solución un sistema de recomendación que mediante una aplicación web haga más atractiva la interacción entre los usuarios, los modelos de recomendación y de análisis de alojamientos.

Palabras clave: Apartamento, usuario, comentario, *rating*, clustering, CatBoost, Embeddings, aplicación web.

ÍNDICE

1. INTRODUCCIÓN.

1.1. MOTIVACIÓN.

A lo largo de la historia, se ha observado una tendencia generalizada hacia la recopilación de datos. Ya sea para gestionar un inventario de productos en un almacén, realizar censos de la población, verificar el adecuado funcionamiento de un proceso industrial o documentar el historial de devolución de préstamos, el acto de recopilar información ha sido una práctica común.

Aunque pueda parecer que la única dificultad frente a esta situación radica en el análisis de datos, cuando se añaden atributos como un volumen considerable de información, variedad en el contenido, velocidad de almacenamiento y falta de veracidad, surge lo que conocemos como entorno Big Data. Además, cada una de estas cuatro características aportan la siguiente información:

1. Volumen: Hace referencia a la gran cantidad de datos.
2. Velocidad: Representa la rapidez con la que se cargan, eliminan o actualizan los datos.
3. Variedad: Al tener distintos orígenes, los datos no tienen porque estar ni estructurados ni tener un formato común.
4. Veracidad: Por la incertidumbre de no saber si están completos o si son erróneos.

Aunque conforme pasa el tiempo siguen apareciendo “V” -valor, visualización, vulnerabilidad- estas cuatro son las que en un principio designaban un entorno de Big Data. (1)

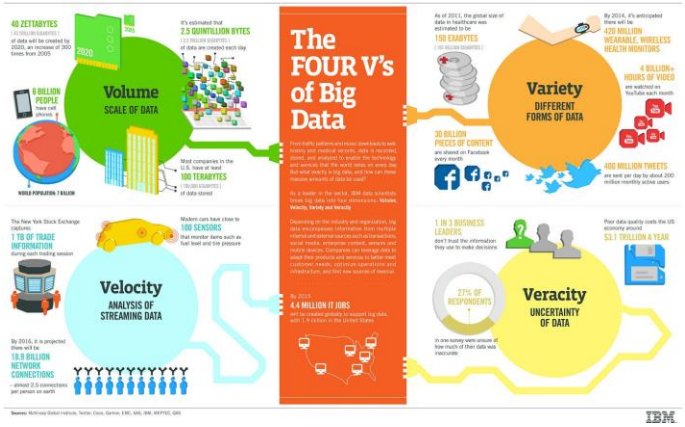


Ilustración 1: Las 4 V's del Big Data según IBM

En la era digital en la que vivimos, la cantidad de productos, servicios y contenido en línea experimenta un crecimiento exponencial constante. Ante esta abundancia, los usuarios se ven abrumados a la hora de encontrar y elegir el servicio/producto/contenido que necesitan. Debido a esta problemática, surgieron

los sistemas de recomendación con el objetivo de simplificar y personalizar la experiencia del usuario.

En un mundo donde la atención del usuario es un bien escaso, la personalización se ha convertido en un factor vital para retener y atraer audiencia. Los sistemas de recomendación ofrecen una gran adaptabilidad en torno al usuario proporcionando sugerencias adaptadas a los intereses y preferencias de los usuarios mejorando así la fidelidad a la plataforma o servicio ofrecido.

Los sistemas de recomendación son el puente entre los usuarios y el contenido, personalizando los resultados y sugiriendo información que el sistema cree que les puede interesar.

Esta capacidad de personalización acompañada del crecimiento sin precedentes en el uso de algoritmos de inteligencia artificial ha impulsado la investigación, el desarrollo y la sofisticación de los sistemas de recomendación, gracias a la capacidad de procesamiento y análisis de grandes volúmenes de datos proporcionando recomendaciones aún más precisas. El auge de estos sistemas en la industria no solo beneficia a los usuarios, la economía digital y los modelos de negocio en línea se ven significativamente impactados con la implementación de esta tecnología desde el crecimiento en la ventas y retención de clientes como en el aumento de la satisfacción del usuario y en consecuencia de la lealtad a la marca.

El mercado turístico, uno de los sectores más saturados con oferta de servicios, se sitúa entre los mayores beneficiados por los sistemas de recomendación facilitando a los huéspedes encontrar apartamentos que se adapten a sus gustos y necesidades, debido a esto, el presente proyecto propone facilitar a los usuarios la búsqueda de apartamentos en España mediante los datos de la plataforma Airbnb, ofreciendo unas sugerencias personalizadas y memorables mediante un preciso sistema de recomendación.

Airbnb es una empresa de alquiler de apartamentos a particulares y turísticos fundada en 2008 en San Francisco por tres compañeros de piso que ante la subida del precio tuvieron que alquilar un colchón hinchable (en inglés, *airbed*) a algún asistente de un congreso de diseñadores al que asistieron para poder costearse parte del alquiler. (2)

En la actualidad, el número de reservas anuales de Airbnb sigue en pleno crecimiento, después de superar el bache del año pandémico. (3) Entre las principales ventajas que puede ofrecer esta plataforma frente a métodos tradicionales de pernoctación son la facilidad de gestión, la comunicación directa con el huésped, la existencia de valoraciones previas de los alojamientos y el precio, en general, más barato que un hotel.

Ilustración 2:Número de reservas en Airbnb desde 2015

Es por este crecimiento anual continuado por el que pensamos que este trabajo llega en un momento idóneo en el que puede tener una mayor importancia, en clave de ayudar a seleccionar el alojamiento que mejor se adecúa a las necesidades del viajero.

1.2. OBJETIVOS Y ALCANCE.

El proyecto consiste en desarrollar e implementar un recomendador de viajes en función de las características y/o necesidades del usuario, proporcionando además un análisis de competencias para permitir a los arrendadores maximizar el potencial de sus apartamentos. Según los datos que proporcione el usuario, el modelo será el encargado de recomendar las mejores opciones teniendo en cuenta las valoraciones de perfiles similares en la página web Airbnb.

Para poder perfilar a los usuarios, se obtiene una estimación de la edad y del sexo con la foto de perfil del usuario mediante algoritmos de aprendizaje profundo que acompañados con el perfilado según gustos extraído a partir de las evaluaciones y los comentarios de los apartamentos, nos permite entender las preferencias de los usuarios y sugerir recomendaciones personalizadas.

El sistema pretende dar servicio a usuarios en búsqueda de apartamentos y estudio de competencias (caso de arrendadores) en España y más concretamente centrados en 3 tipos diferentes de apartamentos: “A pie de playa”, “En el campo” y “Cabañas”.
A continuación, se exponen los objetivos planteados para el adecuado desarrollo del proyecto:

- Estudio del estado del arte de los algoritmos de recomendación en el sector turístico.

- Obtención de un conjunto de datos tanto de apartamentos como de comentarios mediante Web Scrapping.
- Exploración, análisis y visualización de los atributos obtenidos previamente.
- Limpieza y preprocesamiento del conjunto de datos, este objetivo se descompone en 3 fases:
    - Limpieza de las bases de datos de apartamentos y comentarios
    - Extracción de edad y sexo de cada usuario mediante algoritmos de aprendizaje profundo preentrenados.
    - Extracción del grado de satisfacción del usuario y su compañía (en familia, en pareja, en solitario…) mediante algoritmos de NLP (Procesado de lenguaje natural).
- Diseño y desarrollo de la arquitectura del sistema de recomendación.
- Diseño, desarrollo y validación del algoritmo de recomendación.
- Diseño y desarrollo de una prueba de concepto con el fin de validar la viabilidad de la solución propuesta.

1.3. PLANIFICACIÓN Y PRESUPUESTO.

De cara a la planificación se ha planteado un marco de trabajo ágil siguiendo una metodología Scrum donde el trabajo a realizar se dividirá en diferentes sprints. Estos sprints equivalen a cada uno de los objetivos del trabajo.

Haberlo dividido en sprints permite abordar el problema por pequeñas tareas más sencillas y así poder llevar un seguimiento del estado continuo del proyecto en cualquier momento y poder estimar más eficientemente tanto el tiempo como los recursos a emplear.

El primer sprint fue el de *Planificación y Diseño*, donde se trataron temas como el alcance del proyecto, la arquitectura del sistema, una estimación de recursos y tiempo, la identificación de requisitos y el diseño del interfaz final.

La segunda fase fue la de *Obtener el dataset*, aquí se obtuvo tanto la información de cada alojamiento como los comentarios de cada una de ellas y se construyó un bucle para realizar esto sobre todos los alojamientos que nos interesaban para el estudio.

La tercera parte del proyecto ha sido la más larga por el formato de los datos, la etapa de *Preprocesamiento*. En esta parte se ha construido el dataset final con el que se trabajará, como ya se ha comentado se extraerá la información que nos

permite clasificar a los usuarios y obtener información adicional sobre los alojamientos gracias a técnicas de aprendizaje profundo, *Deep Learning*, sobre los comentarios.

Una vez obtenido el dataset, pasaremos a construir el modelo de recomendación final en la etapa de *Implementación, testeo y refinamiento* de este.

Finalmente se realizará la etapa de *Despliegue* donde se construirá la interfaz visual que será con la que el usuario interactuará para las recomendaciones finales.

Ilustración 3:Planificación en semanas del desarrollo del proyecto.

El presupuesto estimado para el desarrollo de trabajo será el necesario para contratar los servicios de dos Data Scientists Junior, el de un Data Scientist Senior y los servicios de Azure que se utilizarán.

El sueldo de un Data Scientist Junior en España ronda los 27.750€ anuales, lo cual se traduce en un precio por hora de 14’7€, como el trabajo consta de 18 créditos ECTS que equivalen a 25h cada uno, el coste total sería de 12.600€ brutos. El sueldo de un Data Scientist Senior ronda los 50.000€ anuales, que equivale a 20€ por hora, supone un coste de alrededor de 1.080€ suponiendo una dedicación semanal de 3 horas al proyecto. (4)

Respecto a los servicios cloud contratados, en este caso Azure, el coste de Azure SQL Database es del 0’25$ al mes siendo el almacenamiento asignado automáticamente, por lo que el precio estimado inicial es de 30$ anuales, 27’95€. (5)

El otro servicio necesario es Azure Cosmos DB (6), la base de datos NoSQL donde se almacenarán las interacciones usuario-apartamento. En este caso el coste se dividiría en una parte destinada al almacenamiento y en otra destinada a disponibilizar los datos para realizar operaciones tanto de lectura como de escritura. El coste de almacenaje sería de 3’48€ (10*12*0’029€), y el operacional dependerá del número de operaciones de lectura realizadas, siendo el precio para 10000 transacciones 0’005€.

Cabría mencionar que hay un último servicio, Azure AD B2C (7), que es el que ofrece el servicio de autenticación, en este caso no acarrea ningún coste hasta que el número de usuarios supera los 50000; por tanto, suponemos que el coste es nulo.

2. ESTADO DEL ARTE.

2.1. ALGORITMOS DE RECOMENDACIÓN.

Los sistemas de recomendación se están consolidando como una herramienta de creciente importancia en las empresas que ofrecen múltiples servicios y/o productos, permitiendo al usuario final tomar decisiones basadas en sus preferencias personales. Son numerosas las empresas que han optado por integrar sistemas de recomendación en la gestión de su contenido, como es el caso de Amazon, Netflix, YouTube y Airbnb entre otros.

Existen múltiples enfoques a la hora de diseñar un sistema de recomendación cuyas limitaciones provienen principalmente de los datos disponibles. Estos datos proceden de diversas fuentes directas e indirectas y se clasifican principalmente en función de si proporcionan información sobre el usuario y sus preferencias o sobre las características de los servicios/productos ofrecidos. Podemos agrupar estos sistemas en cuatro categorías:

- Sistemas basados en Filtrado colaborativo
- Sistemas basados en Contenidos
- Sistemas Híbridos
- Otros sistemas

A continuación, se detallan las características de cada uno, así como sus ventajas e inconvenientes que limitan su aplicabilidad.

2.1.1. BASADOS EN FILTRADO COLABORATIVO.

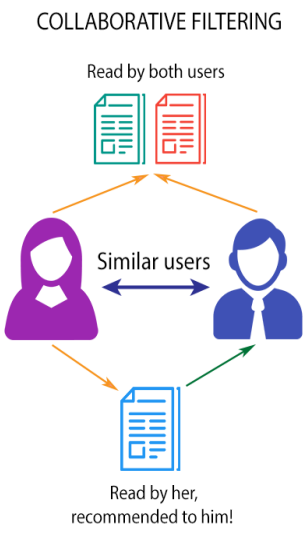
El valor diferencial de este tipo de sistemas de recomendación reside en que no requieren de descripciones detalladas de servicios/productos ni complejos perfiles de usuarios ya que basan su funcionamiento en las interacciones de los usuarios entre ellos y con los servicios/productos ofrecidos, permitiendo en consecuencia acceder a aspectos más complejos especialmente útiles en productos/servicios de difícil caracterización como puede ser el caso de las películas y/o la música.

La hipótesis sobre la cual se fundamentan estos sistemas es que dos usuarios cuyos intereses hayan sido similares en el pasado, probablemente mantengan intereses similares en el futuro. El filtrado colaborativo emplea una matriz de preferencias de ítems por usuarios explotando así las correlaciones existentes entre usuarios. Principalmente existen 3 métodos para implementar el filtrado colaborativo:

- *Matriz de Co-ocurrencia (Reglas de asociación):*

Las reglas de asociación permiten descubrir qué elementos y cómo están relacionados entre sí. Por ejemplo, qué conjunto de productos compran a la vez los consumidores con mayor frecuencia sirviendo como información para la implementación de estrategias como: la venta cruzada de productos, la personalización de recomendaciones, la redistribución de los productos/servicios … Algunas métricas para evaluar las reglas de asociación son el soporte que evalúa la frecuencia de la regla $A \rightarrow B$, la confianza que evalúa la probabilidad de que ocurra B habiendo ocurrido A y el Lift que evalúa la independencia entre A y B siendo un valor mayor que 1 una asociación positiva. (8)

- *Filtrado colaborativo basado en memoria:*



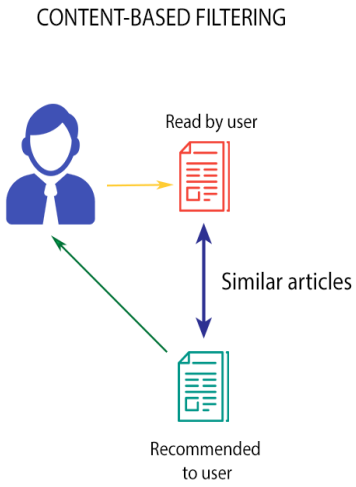
Este método calcula las recomendaciones para un usuario a partir de la información de usuarios con patrones de búsqueda y valoraciones similares al usuario en cuestión, obteniendo dicha similitud mediante métodos de correlación (Coeficiente de Pearson) o distancias (Distancia Coseno).
A partir de la valoración obtenida para un ítem por un usuario, esta se extrapola con ponderación mediante diferentes técnicas sobre los usuarios con perfiles similares. Dentro de este enfoque se distinguen dos tipos de filtrado: basados en usuarios, donde tratamos de encontrar similitudes entre preferencias de los usuarios y basados en ítems donde tratamos de aprovechar las correlaciones entre productos/servicios.

Los métodos basados en ítems son útiles cuando contamos con más información sobre los ítems que sobre los usuarios. Tienen un comportamiento más estable, sin embargo, los métodos basados en usuarios tienden a tener un mayor rendimiento debido a los patrones ocultos de preferencias que se descubren entre usuarios.

- *Filtrado colaborativo basado en modelos:*

Emplea técnicas de aprendizaje automático (modelos de regresión, redes neuronales, factorización de matrices) para descubrir patrones latentes en las interacciones usuario-ítem sin necesidad de comparar directamente con otros usuarios o ítems. Este método muestra un mayor rendimiento en grandes conjuntos de datos debido a su capacidad para manejar la elevada dimensionalidad durante el cálculo, además, reduce los problemas de dispersión de los datos y permite incorporar información adicional sobre los ítems y los usuarios para incrementar la precisión de las recomendaciones. Sin embargo, la calidad del dato y la capacidad de recursos computacionales pueden ser un factor limitante para este tipo de sistemas de recomendación.

2.1.2. BASADOS EN CONTENIDOS.



Los sistemas de recomendación basados en contenidos sugieren elementos similares a aquellos sobre los que se ha mostrado preferencia en el pasado. Con la finalidad de implementar estos sistemas se requiere de dos bloques de información:

- *Descripción de los ítems*

Conjunto de características descriptivas que caracterizan a un ítem. La naturaleza de esta información varía según el contexto, por ejemplo, en el ámbito del mercado de alquileres vacacionales, un ítem (un apartamento) incluiría características como su capacidad, la puntuación recibida, el precio, su ubicación, entre otros. Generalmente, se prefiere utilizar características técnicas en lugar de subjetivas, ya que estas últimas podrían introducir un sesgo en los datos y generar confusión en el modelo.

- *Perfiles de usuario*

Su objetivo es reflejar las preferencias del usuario basándose en sus elecciones o valoraciones anteriores. Por ejemplo, en el contexto del mercado de alquileres vacacionales, un usuario podría mostrar un interés particular por un apartamento en función de la estación del año, el público habitual de la zona donde esté ubicado el apartamento, etc.

Empleando las características de los ítems y el perfilado del usuario, se calcula la similitud entre las preferencias del usuario y el resto de ítems y se recomiendan aquellos elemento con una mayor puntuación de similitud (Coseno, Jaccard o correlación) o mediante otras técnicas de ranking. Este método ofrece recomendaciones más personalizadas debido a la similitud con el histórico de preferencias del usuario, sin embargo, puede dar lugar a una reducida variabilidad en las recomendaciones debido a la elevada similitud. En casos donde la información disponible sobre los usuarios sea reducida o nos enfrentamos a un problema de arranque en frío (ocurre al inicio de las recomendaciones donde se dispone de una información muy reducida, por ejemplo, la inclusión de un nuevo apartamento que aún nadie ha visitado) se ve acentuada la utilidad de este método debido a que no se requiere información sobre otros usuarios. (9)

2.1.3. SISTEMAS HÍBRIDOS.

Los sistemas de recomendación híbridos resultan como su propio nombre indica de la combinación de múltiples enfoques de recomendación, aprovechando las fortalezas de cada uno y reduciendo el impacto de sus debilidades. Estos sistemas, suponen un considerable salto tanto cualitativo como de complejidad de implementación y mantenimiento.

En el caso de Airbnb, ha pasado de ofrecer únicamente alojamientos a ofrecer experiencias y/o restaurantes. Para posicionar cada uno de estos servcios, emplea diferentes métodos: para los alojamientos emplea técnicas como NLP (Natural Language Processing), GBDT (Gradient Boosted Decision Trees), lambda rank y modelos de deep learning. (10)

3. GENERACIÓN DE LAS BASES DE DATOS.

El web scraping es una técnica de extracción de información de sitios web mediante software, de esta forma podemos extraer información de forma masiva y automatizada (11). El funcionamiento es el siguiente:

I. *Acceso al sitio Web:*
Se realiza una petición mediante los protocolo HTTP y HTTPS (Hypertext Transfer Protocol Secure) a la URL del sitio objetivo y se recibe como respuesta el contenido de la página en formato HTML.

II. *Análisis y Extracción de contenido*
Se analiza y extrae la información presente en el archivo HTML devuelto por la página, este puede incluir texto, imágenes, enlaces…

III. *Estructuración y almacenamiento*
Tras haber obtenido la información, se estructura en el formato deseado (JSON, CSV …) para facilitar su posterior análisis y almacenamiento.

Para obtener las bases de datos de Apartamentos y comentario optamos por emplear web scraping sobre la página de Airbnb, siguiendo un proceso modular que se muestra a continuación:

Ilustración 7:Procedimiento de web scrapping de los datos.

La plataforma de AirBnb divide los alojamientos en diferentes tipos como pueden ser: Cabañas, Iconos, Surf, Casas Rurales … El software para generar la base de datos diseñado permite escalabilidad al poder decidir sobre qué tipos de apartamentos extraer información y la cantidad de apartamentos de cada uno, sin embargo, se han seleccionado los alojamientos de la web que se corresponden con las categorías: “A pie de playa”, “En el campo”, y “Cabañas” para el desarrollo del presente proyecto.

Para diseñar este software, se emplearon herramientas como *Beautifulsoup* (12) para el análisis y extracción de información del HTML y *Undetected_chromedriver* (13), una librería de Python basada en *Selenium* (14) que permite evitar ser detectado por las herramientas anti-bot de los sitios web mediante la modificación de encabezados y el bypass de detección automatizada. Durante el proceso de extracción de información se generaron una serie de identificadores únicos para los apartamentos por un lado y para los usuarios por el otro.

Finalmente almacenamos las bases de datos de apartamentos y comentarios en formato CSV (Comma Separated Values) para facilitar la posterior integración con librerías de análisis y lectura de datos como *Pandas* (15). En total se obtuvieron alrededor de 800 opciones vacacionales, en torno a 250 de cada uno de los tres tipos elegidos y más de 38.500 comentarios de 34.000 usuarios diferentes.

4. ANÁLISIS EXPLORATORIO DE LAS BASES DE DATOS.

Al haber obtenido los datos directamente de la web, el formato que presentan no es el más cómodo para trabajar con ellos, por tanto, se ha de realizar un tratamiento de limpieza preliminar durante el análisis para permitir la adecuada definición del proceso final de limpieza anterior a la selección de variables, el estudio de segmentación y el modelado del sistema. Este análisis nos va a permitir proporcionar información clave para los arrendadores, cuyo objetivo sea maximizar la rentabilidad del apartamento.

4.1. BASE DE DATOS DE APARTAMENTOS.

4.1.1 DESCRIPCIÓN GENERAL.

Este conjunto de datos contiene la información descriptiva de cada apartamento teniendo en cuenta características como la ubicación, la capacidad, el precio … Comenzamos el estudio de la base de datos, realizando una limpieza previa donde extraemos información de texto a múltiples atributos como se observa en la figura, extraemos además la latitud y longitud a partir de la ubicación en formato de texto para estudiar la varianza entre apartamentos y su relación con su localización.

Las variables disponibles para el estudio son las siguientes: ID, Titulo, Descripción Simple, Información, Evaluaciones, Tipo, Precio, URL, Limpieza, Veracidad, Llegada, Comunicación, Ubicación, Calidad, Servicios, Localización, Capacidad, Camas, Baños, Dormitorios, Baño Compartido, Wifi, Mascotas, Piscina y Parking.

Contamos con un total de 788 apartamentos divididos entre España (69%), Portugal (4%) y Francia (27%), para comprender qué información aporta cada una de las variables, realizamos un análisis de distribuciones sobre las variables numéricas tal y como se observa en la figura 48 del Anexo. La variable evaluaciones contiene el número de reseñas que recibe cada apartamento, estos reciben una media de 90 evaluaciones. Observamos además que tanto España como Francia cuentan con apartamentos con especial popularidad.

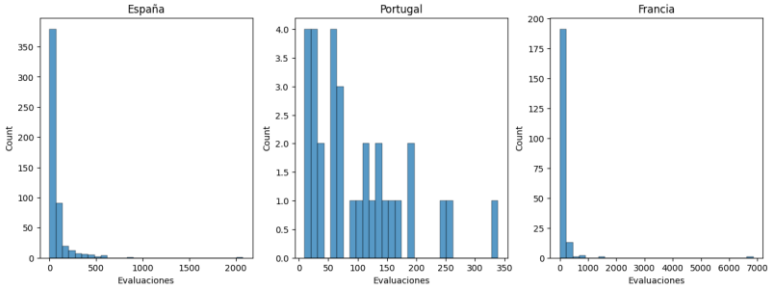


Ilustración 9: Distribución del número de evaluaciones para cada país.

Para corregir el comportamiento sesgado de la distribución, comenzamos comprobando el efecto de eliminar los valores atípicos y posteriormente el efecto de aplicar una transformación logarítmica sobre esta última distribución, tal y como se observa en la figura 10, aplicando esa metodología, es posible normalizar la distribución.

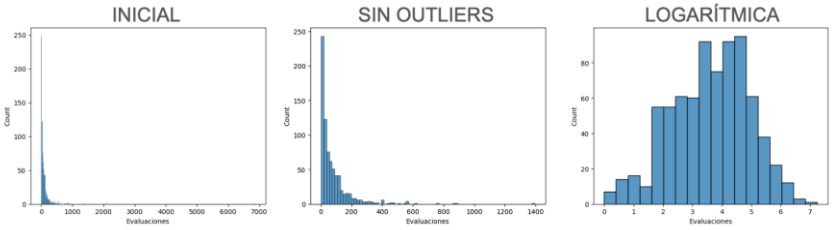


Ilustración 10: Normalización de la distribución de evaluaciones

En la página web de Airbnb los hosts (arrendadores) pueden comentar los servicios adicionales que ofrecen en el apartamento (piscina, parking, wifi…). Se observa que un 80% de los apartamentos cuentan con red wifi lo cual no es de extrañar en pleno siglo 21, sorprendentemente, solo un 50% de los apartamentos permiten mascotas, la gran mayoría de los apartamentos cuentan con parking propio, un detalle especialmente atractivo para viajes de media distancia donde los usuarios tienden a preferir transportarse en su propio vehículo.

En cuanto a las valoraciones de los usuarios, la plataforma de Airbnb permite valorar aspectos como: Limpieza, Veracidad, Llegada, Comunicación, Ubicación y Calidad. Un aspecto sorprendente en cuanto a las evaluaciones es que la puntuación media recibida por los apartamentos en estos aspectos es de entorno a 4.8 sobre 5 lo cual es un valor considerablemente elevado, concluyendo que por lo general la experiencia de los usuarios en Airbnb es muy positiva, un punto que explica la creciente popularidad de dicha plataforma.

4.1.2 ESTUDIO DE PRECIOS Y CAPACIDAD.

La capacidad es un aspecto crucial en la selección de apartamentos por parte del usuario, la capacidad de huéspedes más común es de 4 usuarios, como hipótesis podemos concluir que el público objetivo por lo general son las familias. Podemos agrupar el público objetivo teniendo en cuenta los siguientes grupos: Individual (1), Parejas (2), Familias (3-4) y Grupos numerosos (+5).

Debido a que junto con la capacidad existen variables como el número de baños, de habitaciones… Características a priori con elevada relación entre sí, decidimos mostrar una matriz de correlaciones calculadas mediante el coeficiente de Pearson cuyo resultado se muestra en la figura del anexo. Observamos que existe una correlación elevada de entorno 0.86 entre las variables Capacidad, Camas y Dormitorios, sin embargo, la selección final de variables se expone en el correspondiente apartado.

Para añadir valor a este análisis exploratorio, enfocamos nuestro estudio en el comportamiento del precio en función de diversos aspectos como la localización, el tipo de apartamento, la capacidad, etc. Mediante un histograma y un diagrama de cajas, observamos que la distribución de precios está desplazada hacia la izquierda teniendo su mediana en 105€ por noche.

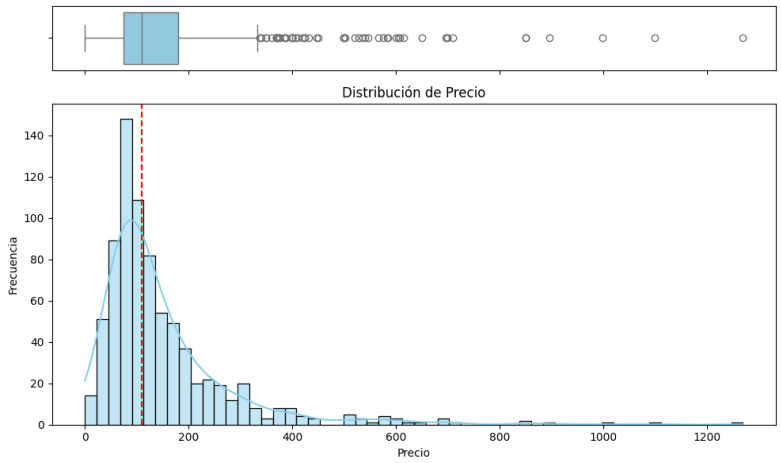


Ilustración 11: Distribución y diagrama de cajas y bigotes de la variable Precio.

Para corregir esta distribución y evitar la disminución del rendimiento en algoritmos que asumen una distribución normal, podemos aplicar una transformación logarítmica que corrige considerablemente el problema. Sobre el mapa observamos que existen principalmente 3 zonas que acumulan los apartamentos más caros que son: Zona norte de Madrid, Alicante y San Sebastián.

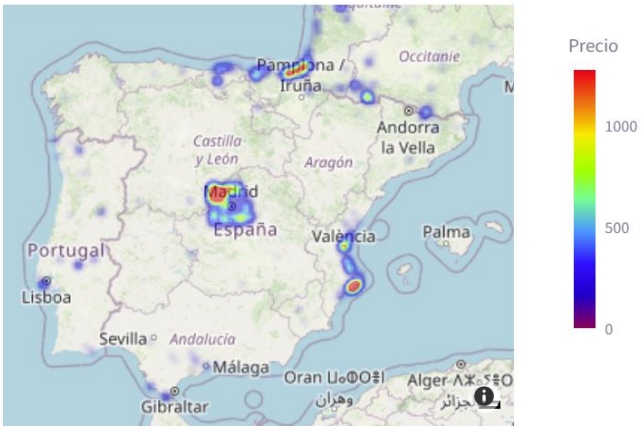
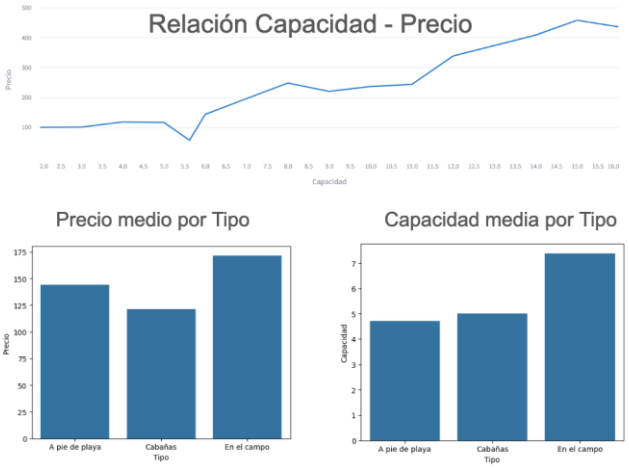


Ilustración 12: Mapa de calor por precio con la ubicación de los alojamientos.

Sobre el gráfico, entendemos que no existe una distinción clara entre el precio para apartamentos en la costa y en el campo, para confirmar esta hipótesis mostramos un gráfico de barras con el precio medio por tipo de apartamento.



El tipo “En el campo” es el más caro, con un precio medio de 180'19€, 75€ superior a la media general. Sin embargo, estudiando la capacidad en función del tipo de apartamento observamos que los apartamentos ‘En el campo’, cuentan con una capacidad media superior a sus competidores. Si estudiamos el precio medio por individuo por noche en función del tipo de apartamento observamos que los apartamentos ‘A pie de playa’, tienen un coste de un 20% superior a los demás lo cual afirma nuestra hipótesis inicial de que por lo general los apartamentos en la playa son más caros.

4.1.3 IDENTIFICACIÓN DE PALABRAS CLAVE.

Finalmente, nos dispusimos a analizar mediante técnicas de procesado de lenguaje natural los títulos y descripciones de los apartamentos, comenzamos preparando una columna con el texto de cada apartamento separado en palabras y observamos que las palabras más repetidas eran: Alojamiento, entero, España y casa lo cual no nos aportaba mucha información, posteriormente, decidimos analizar los bigramas y trigramas más comunes.

Los n-gramas son secuencias contiguas de n elementos (generalmente palabras) de un texto y se utilizan para capturar relaciones y patrones en el texto, como la co-ocurrencia de palabras. En este caso, el análisis de n-gramas sobre secuencias de 2 y 3 elementos no nos aportó información útil, por ello decidimos para concluir el análisis estudiar el precio medio por aparición de palabra, es decir el precio medio de todos los apartamentos donde aparece una palabra en concreto obteniendo los siguientes resultados:

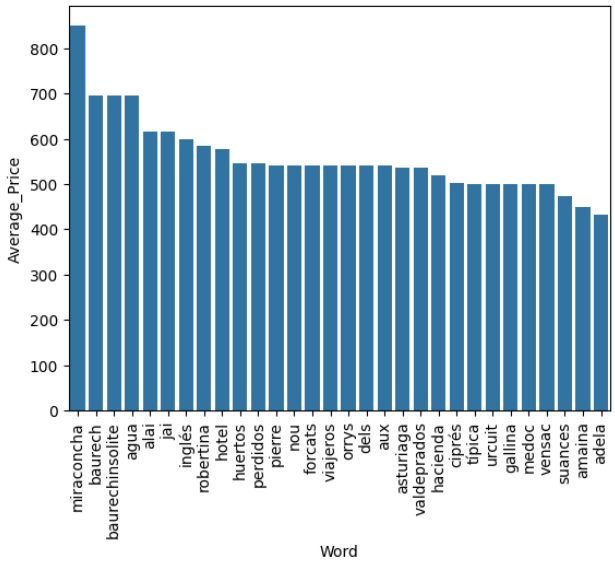


Ilustración 14:Relación del precio con la aparición de algunas palabras.

Los apartamentos más caros incluyen la palabra miraconcha que resulta ser un barrio de lujo de San Sebastián, de ahí que sobre el mapa San Sebastián apareciera como una de las ubicaciones de mayor coste. Seguido aparece baurech, una población francesa con diversas atracciones turísticas donde además los apartamentos por lo general son de gran tamaño. Con el fin de estudiar las palabras clave que emplean los arrendadores de los apartamentos más populares repetimos el gráfico empleando la media de evaluaciones como medida de popularidad.

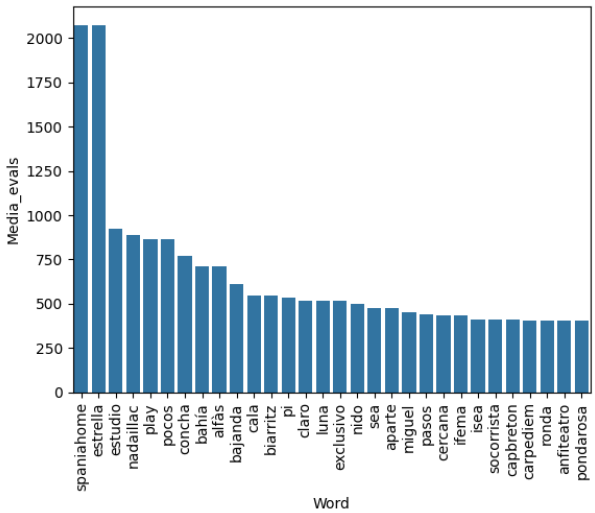


Ilustración 15: Relación de ciertas palabras con el número de evaluaciones, midiendo así la popularidad.

Observamos que palabras como estrella, exclusivo, estudio están entre las más destacadas. Con toda la información recopilada durante el análisis, somos capaces de ayudar a nuevos arrendadores a conocer su competencia, los títulos y descripciones óptimas para atraer clientes y su público objetivo en función de la ubicación y las características del apartamento.

4.2. BASE DE DATOS DE COMENTARIOS.

4.2.1 DESCRIPCIÓN GENERAL Y DATOS NO TEXTUALES.

Esta base de datos contiene información relacionada con características de los usuarios y el comentario dejado por estos tras su estancia en un apartamento. Este conjunto de datos consta de 38.813 observaciones con 7 variables iniciales sobre las que se extrae la información necesaria para caracterizar al usuario. Estas variables son: nombre, imagen del perfil, user_id, valoración (rating), comentario, apart_id y antigüedad. Con el fin de proporcionar a los host con información acerca de la edad y el género de los visitantes del apartamento, empleamos un modelo de la librería *gender_guesser* (16) centrado en la identificación del género a partir de características como el nombre y su lugar de orígen, para extraer la información de la edad empleamos un modelo de Deep Learning de la librería *deepface* (17) que estima la edad a partir de una imagen del usuario, lamentablemente los resultados no cumplieron las expectativas debido a 2 causas principales:

- Un gran número de usuarios no muestra su cara en su perfil.
- Gran parte de las imágenes son en compañía de más personas penalizando el rendimiento del modelo.

Como regla general decidimos eliminar aquellas observaciones sobre las cuales no se dispusiera de comentario ya que como veremos más adelante, es la principal fuente de información de esta base de datos. A continuación, realizamos una corrección de tipos para extraer el valor numérico de la valoración y empleamos la información del apartamento al que hacía referencia el usuario para estimar la compañía (Pareja, familia, grupos grandes…) del usuario mediante la capacidad del apartamento.

4.2.2 ANÁLISIS DE LOS COMENTARIOS.

Textblob (18) es una librería de Python centrada en el procesamiento de lenguaje natural e incorpora múltiples funcionalidades de gran utilidad, para extraer el sentimiento empleamos el valor de polaridad que esta proporciona durante el análisis de un texto siendo -1 una polaridad muy negativa y +1 una polaridad muy positiva. Inicialmente, decidimos estudiar la relación entre dicha polaridad y la evaluación proporcionada por el usuario.

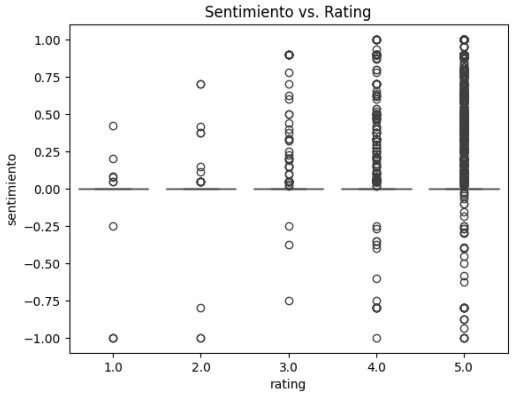


Ilustración 16: Relación entre rating y polaridad de las reseñas.

Observamos una tendencia creciente que indica que a mayor es el rating mayor es la polaridad del comentario, sorprendentemente, para puntuación muy bajas, no está presente una polaridad negativa. Únicamente un 0,3% de los comentarios incluyen preguntas lo cual indica que los hosts ponen especial atención en concretar al detalle las características del apartamento. Con el fin de estudiar la motivación de los comentarios explicada por los apartamentos, decidimos combinar ambas bases de datos para realizar un análisis con mayor detalle.

Comenzamos calculando las correlaciones entre el sentimiento extraído del comentario del usuario con las valoraciones del apartamento en términos de limpieza, veracidad, llegada, comunicación, ubicación y calidad.

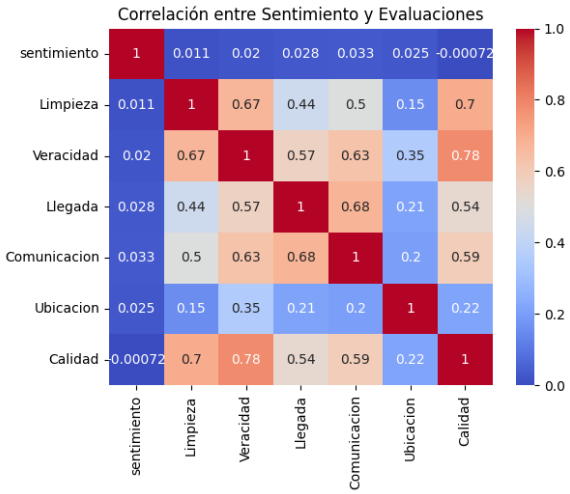


Ilustración 17:Mapa de calor con los coeficientes de correlación entre sentimiento y las evaluaciones.

Vemos que el valor de las correlaciones es muy reducido con lo que concluimos que no están elevadamente correlacionados al menos de forma lineal. Para estudiar la influencia del precio sobre el sentimiento percibido, decidimos graficar el precio medio por agrupaciones de sentimiento.

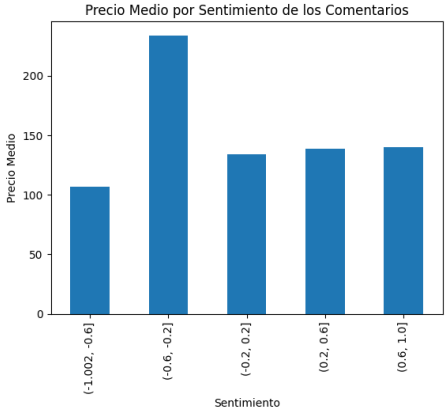


Ilustración 18:Relación del precio medio con el sentimiento de los comentarios.

Observamos que el descontento se centra principalmente en 2 grupos, los apartamentos más caros donde entendemos que el descontento es debido a no responder a las expectativas del cliente, por otro lado, los apartamentos de menor coste medio también muestran un sentimiento negativo posiblemente debido a las condiciones del mismo apartamento tanto de ubicación como de limpieza y otras características valoradas por los usuarios. Seguidamente, analizamos el efecto de la cantidad de adjetivos empleados y la longitud promedio de las palabras sobre variables como el precio o el tipo de apartamento, buscando caracterizar y agrupar a los usuarios.

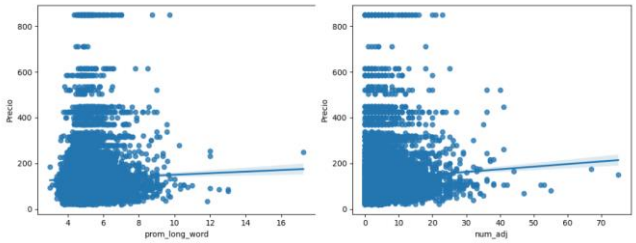


Ilustración 19: Relación entre la longitud y el número de adjetivos de una reseña con el precio.

En ambos gráficos comprobamos que la tendencia es inexistente, sin embargo, consideramos que estas variables pueden ser de interés para el modelado del sistema. Finalmente, sobre los comentarios clasificados como sentimiento negativo, estudiamos los principales problemas presentes de entre una lista preliminar que incluye algunos como: ubicación, ruido, limpieza…

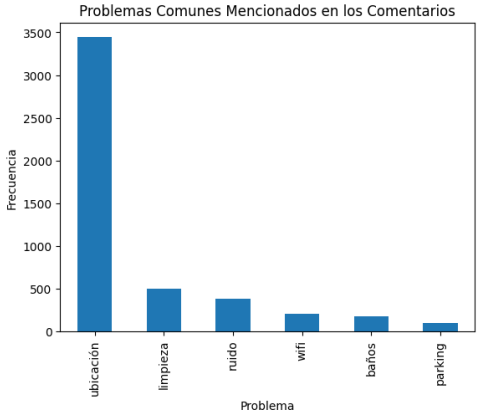


Ilustración 20:Problemas extraídos de los comentarios de los apartamentos con peores valoraciones.

El principal problema que corregir por parte de los arrendadores es la ubicación, las escasas facilidades de transporte hacia zonas con mayor afluencia parecen estar causando un descontento en los clientes, seguidamente la limpieza y el ruido son aspectos fundamentales para el confort de los usuarios.

5. LIMPIEZA DE LAS BASES DE DATOS.

La causa principal del reducido rendimiento de un modelo por lo general reside en la calidad de los datos, es por ello que procedemos a realizar una preparación y limpieza de los datos basada en las conclusiones extraídas durante la exploración de los conjuntos de datos, ambos serán procesados mediante flujos diferentes debido a la naturaleza de su información.

5.1. PROCESADO DE LOS APARTAMENTOS.

Inicialmente, extraemos la información de capacidad a partir de la variable información mediante expresiones regulares, empleando sustitución de cadenas de texto realizamos una corrección de tipos en las variables referentes a las valoraciones (Limpieza, calidad…), concluimos eliminando la columna información pues ya no es necesaria. Mediante la librería *geopy* (19), siguiendo el enfoque para la visualización de los precios sobre el mapa extraemos la longitud y la latitud a partir de la ubicación, este método es el más eficiente para tratar con ubicaciones en formato numérico. Finalizando con la ubicación, extraemos el país al que pertenece el apartamento (España, Francia, Portugal) para posteriormente deshacernos de la variable localización.

En cuanto a los servicios, extraemos la presencia o no de algunos en la descripción de servicios del apartamento con un formato binario (cuenta con el servicio o no). Con anterioridad a los procesos de *feature engineering* y transformaciones, realizamos una imputación para los valores faltantes empleando la mediana para características numéricas continuas y el valor más frecuente para las categóricas.

Decidimos aplicar la transformación logarítmica sobre las variables continuas evaluaciones y precio para prevenir problemas con modelos sensibles a distribuciones no normalizadas y diferencia de escalas entre variables.

Para enriquecer la calidad de nuestros datos generamos 2 variables:

- ***Precio por persona:*** se calcula como el precio entre la capacidad del apartamento y se corresponde con el precio por persona por noche.
- ***Rating:*** se calcula como el promedio entre las variables de evaluación.

Una vez finalizado el proceso, disponemos de un conjunto de datos limpios con los que realizar la selección de variables, la segmentación de apartamentos y el posterior modelaje.

5.2. PROCESADO DE LOS COMENTARIOS.

El proceso de limpieza de la base de datos de comentario es similar al realizado durante el estudio de esta, comenzamos extrayendo el género biológico a partir del nombre de la persona y realizando una corrección de tipos sobre las variables valoración y antigüedad para convertir estas a numéricas.

Seguidamente, extraemos el sentimiento del comentario empleando el criterio de la librería *textblob* donde -1 es una polaridad muy negativa y 1 una polaridad muy positiva. Sobre el mismo comentario, extraemos si el comentario contiene una pregunta o no, la longitud promedio de las palabras y el número de adjetivos presentes en el comentario. Los *stopwords* han sido previamente eliminados para evitar aportar datos no relevantes al modelo, a continuación, se extraen los principales problemas descritos en el comentario, por ejemplo, el ruido o la mala ubicación del apartamento. El número total de tokens nos permite introducir nueva información al modelo para aumentar el rendimiento y la calidad de nuestros datos.

Finalmente, siguiendo las reglas del EDA, decidimos eliminar aquellas observaciones que no contuviera un comentario debido a que más del 50% de nuestras variables finales depende de esta, para imputar los nulos numéricos decidimos emplear la mediana como metodología para mitigar el efecto de los valores atípicos.

6. SELECCIÓN DE VARIABLES.

Durante el análisis exploratorio comprobamos que había múltiples variables con una elevada correlación que parecían estar aportando información redundante, especialmente en las variables referentes a la capacidad y las puntuaciones de los apartamentos. Consecuentemente, decidimos realizar un análisis de correlaciones e importancia de variables para estudiar el efecto de estas sobre el valor a predecir (rating).

6.1 SELECCIÓN PARA APARTAMENTOS.

El coeficiente de Pearson nos permite estimar la correlación lineal entre 2 variables, debemos tener en cuenta que la correlación no implica causalidad. Si observamos el gráfico de correlaciones para los datos de apartamentos, se aprecian dos bloques con elevada correlación, por un lado, el rating global con el desglose de evaluación y por otro lado, las variables relacionadas con la capacidad como con camas, dormitorios y capacidad.

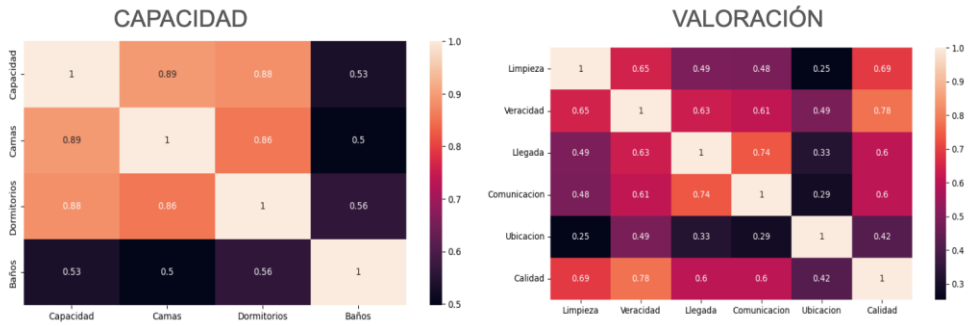


Ilustración 21: Mapas de calor con los coeficientes de correlación de las variables de capacidad y de valoración.

Haciendo un análisis de VIF (Variance Inflation Factor) vemos que tenemos múltiples variables con varianza 0 y por lo tanto un valor para VIF de 0 con lo cual deben ser eliminadas ya que no aportan información. Para concluir con la selección de variables empleamos metodologías de selección intrínseca mediante modelos no paramétricos como por ejemplo Catboost (20), un algoritmo de Gradient Boosting basado en árboles de decisión especialmente eficiente en el tratamiento de datos categóricos. Estos modelos nos retornan la importancia que han considerado para cada variable sobre la variable objetivo. En este caso, nos permitió ratificar las conclusiones de VIF. Finalmente, se decide añadir a la ETL de los apartamentos la eliminación de las siguientes variables: elevada correlación con Capacidad (Camas, Dormitorios) y varianza muy reducida (Wifi, Parking, Piscina, Mascotas, Evaluaciones y Baño compartido).

	Variable	VIF
15	Wifi	0.000000
18	Parking	0.000000
17	Piscina	0.000000
16	Mascotas	0.000000
0	Evaluaciones	1.033110

6.2 SELECCIÓN PARA COMENTARIOS.

Al igual que para la base de datos de apartamentos comenzamos graficando las correlaciones entre las variables numéricas, entre estas no se observa ningún coeficiente a considerar para la selección de variables, únicamente se aprecia una elevada relación entre el número de tokens y el número de adjetivos, lo que significa que gran parte del comentario se corresponde con adjetivos algo razonable tratándose de una valoración.

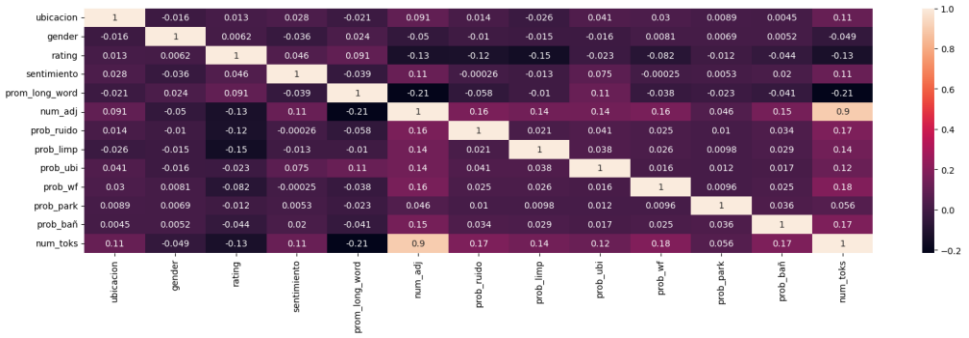


Ilustración 23: Mapa de calor con los coeficientes de correlación de las variables del dataset de comentarios.

Mediante el enfoque de selección intrínseca de variables, apreciamos que el modelo asigna la mayor importancia a una variable creada durante el *feature engineering*, con este ejemplo comprendemos la importancia de este paso en el modelado de una solución de Machine Learning. Adicionalmente, este método nos permite validar la nula importancia de la variable pregunta a la hora de predecir el rating. A excepción de los problemas de limpieza, el resto no parecen ser críticos según Catboost a la hora de predecir el rating, sin embargo, decidimos conservarlos porque aportan algo de información por reducida que sea.

	Importance	Var
4	22.403156	prom_long_word
12	20.494147	num_toks
5	11.975284	num_adj
2	10.527194	sentimiento
0	9.631165	ubicacion
7	8.101818	prob_limp
1	4.908570	gender
6	4.823871	prob_ruido
9	3.834933	prob_wf
8	2.359094	prob_ubi
11	0.586896	prob_bañ
10	0.353871	prob_park
3	0.000000	pregunta

En conclusión, decidimos eliminar la variable pregunta ya que no aporta información debido a su reducida varianza y la variable num_tokens debido a su elevada correlación con la variable num_adj.

7. SEGMENTACIÓN Y REGLAS DE ASOCIACIÓN.

Debido a la naturaleza del problema de recomendación, consideramos relevante realizar un estudio de segmentación de usuarios con el fin de personalizar las recomendaciones en función de los grupos identificados en nuestros datos. Adicionalmente, consideramos estudiar la asociación entre la elección de apartamentos por parte de los usuarios y las segmentaciones existentes entre los apartamentos.

7.1. SEGMENTACIÓN.

El clustering es una técnica de análisis de datos que agrupa un conjunto de objetos de tal manera que los objetos dentro de un mismo grupo (o clúster) son más similares entre sí que a los de otros grupos. Se utiliza para identificar patrones y estructuras en los datos que a priori son desconocidas. En el contexto del problema de recomendación, el clustering es útil porque permite segmentar a los usuarios en grupos basados en sus comportamientos y preferencias y permite segmentar los apartamentos por similitud de características. Estas segmentaciones facilitan la personalización de las recomendaciones, ya que podemos ofrecer sugerencias más relevantes y precisas a cada grupo, mejorando así la experiencia del usuario y la eficacia del sistema de recomendación.

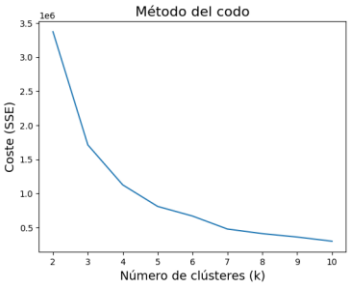
7.1.1. CLUSTERING SOBRE LOS APARTAMENTOS.

Comenzamos estudiando las agrupaciones existentes entre nuestros apartamentos, para ello emplearemos el algoritmo KMeans (21), un algoritmo de clustering basado en particiones, es decir, busca las agrupaciones existentes optimizando una función objetivo, este método trata de maximizar la distancia entre las agrupaciones (clústers) y minimizar la distancia entre elementos del mismo clúster. El principal inconveniente de este algoritmo es que asumen formas hiper esféricas y de tamaño similar lo cual en entornos reales es poco probable. Sin embargo, para validar los resultados realizaremos un clustering basado en densidad con el algoritmo DBSCAN (22) y un clustering jerárquico mediante dendogramas.

K Means clustering

Inicialmente, debemos obtener el número óptimo de clústeres a estudiar, para ello empleamos 2 metodologías, el método del codo una técnica heurística que permite una primera aproximación visual del número óptimo de agrupaciones y que basa su funcionamiento en la observación de que el SSE (Suma de los errores cuadráticos) disminuye rápidamente a medida que se aumenta el número de clústeres al principio, pero después de cierto punto, la disminución se vuelve gradual.
El "codo" representa este punto de inflexión, indicando el número óptimo de clústeres que balancea adecuadamente entre la complejidad del modelo y la

variabilidad explicada. Los modelos de clustering son especialmente sensibles a la aparición de valores atípicos, por ello eliminamos aquellos valores más alejados de 3 desviaciones estándar de la media.



En nuestro caso particular, no se observa un punto de inflexión claro, entendemos que podría encontrarse entre k=3 y k=6, para obtener un resultado fiable recurrimos a 2 métricas que permiten numéricamente determinar el número óptimo como son el coeficiente de silueta y el de Calinski-Harabasz. Silueta evalúa la calidad del clustering de forma individual para cada punto, considerando la cohesión (similaridad dentro del mismo clúster) y la separación (diferencia con el clúster más cercano) y su rango de valores está entre -1 y 1, siendo 1 una buena cohesión y clara separación y -1 un valor asignado erróneamente. Por otro lado, Calinski-Harabasz evalúa la calidad del clustering a nivel global, midiendo la relación entre la dispersión entre clústeres y la dispersión dentro de los clústeres, en este caso a mayor es el valor de Calinski mayor es la calidad del clustering.

Tras calcular ambas métricas obtuvimos que para Silueta el número óptimo de clústers es 3 mientras que para Calinski-Harabasz es 10, al ser tan dispersos los valores decidimos realizar la caracterización de agrupaciones siguiendo ambos criterios y concluyendo cual proporciona mejor información para apoyar la recomendación. Observando la imagen podríamos decir que la opción de tres clústeres ofrece una visión más general y menos detallada de los datos con una mayor dispersión dentro de cada clúster; al contrario que el de 8, que ofrece una visión más detallada con una diferenciación más palpable permitiendo una clasificación más clara y precisa.

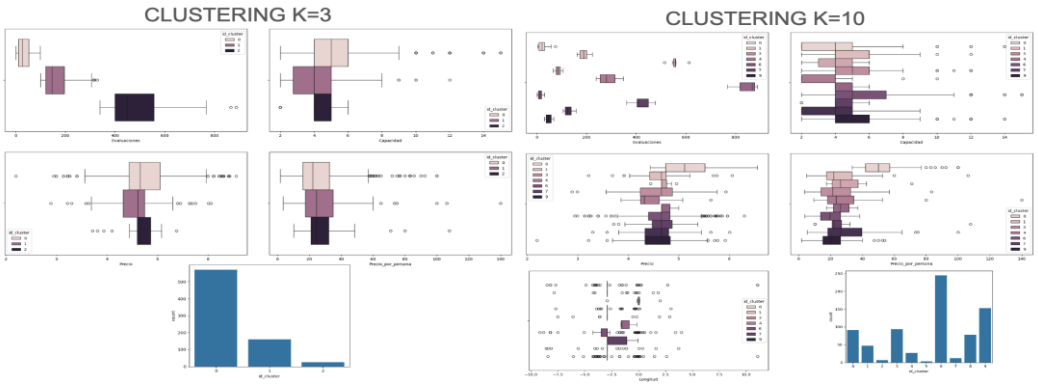


Ilustración 26:: Boxplots con la distribución de las variables para 3 y 10 clústers.

Para el clustering con únicamente 3 clústeres, podemos observar como para las tres variables el rango de valores es muy similar, no se aprecian diferencias notables entre ellos, como ya hemos mencionado anteriormente la visión que nos ofrece es muy general. Como se puede ver en el histograma de la imagen es claro

el desbalance entre ellos, existe un clúster claramente mayoritario que incluye la mayoría de los apartamentos.

En el caso de optar por 10 clústers los resultados podrían considerarse más informativos, en este caso podemos apreciar cierta segregación y diferenciación entre los distintos clústeres. Observamos como el modelo comienza a buscar segregaciones a partir de la ubicación concretamente de las coordenadas de longitud, además, las evaluaciones parecen tener un impacto diferencial a la hora de segregar los apartamentos, lo cual no es nuestro objetivo durante la recomendación ya que estaríamos acentuando el problema de arranque en frío.

DBSCAN y Clustering Jerárquico

El algoritmo DBSCAN es otra manera de clusterizar un conjunto de observaciones. Este algoritmo define los clústeres mediante una estimación de la densidad local, consta de cuatro etapas: para cada observación se mira el número de puntos a una distancia máxima ε, si una observación tiene al menos un cierto número de vecinos se considera una observación central o de alta densidad, todas las observaciones que estén contenidas en la vecindad de radio ε, respecto a una observación central se considera que están en el mismo clúster. Cualquier observación que no sea observación central y que no tenga ninguna observación central en su vecindad se considera una anomalía.

A diferencia del algoritmo K Means, en este no es necesario predefinir el número de clústeres, por tanto, en un principio se consideró que podría ser útil, ya que podría descubrir relaciones de similitud que no son apreciables a simple vista. Sin embargo, al aplicar este algoritmo lo que se obtuvo es que todos los clústeres obtenidos eran ruido, no existen agrupaciones entre los datos. Esto se podría deber a una irregularidad de densidades o a valores erróneos de los parámetros del modelo. Finalmente, implementamos un clustering jerárquico mediante dendrogramas donde se puede observar que para 4 cluster la distinción y el tamaño es ligeramente similar.

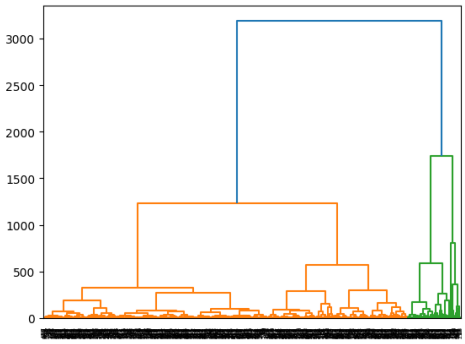


Ilustración 27: Dendograma de apartamentos.

7.1.2. CLUSTERING SOBRE LOS USUARIOS.

Siguiendo la metodología empleada para la segmentación de apartamentos, comenzamos obteniendo el número de clusters óptimos según el algoritmo KMEANS. En este caso, se repiten las discrepancias entre las métricas de silueta y Calinski siendo el número de clusters óptimo de 3 y 4 respectivamente.

Con el conocimiento adquirido en la segmentación anterior decidimos estudiar el clustering de 4 agrupaciones ya que con gran probabilidad permite una mayor personalización. Aun habiendo aplicado la eliminación de outliers y habiendo realizado una adecuada limpieza de los datos observamos la variabilidad en el tamaño de los clústers.

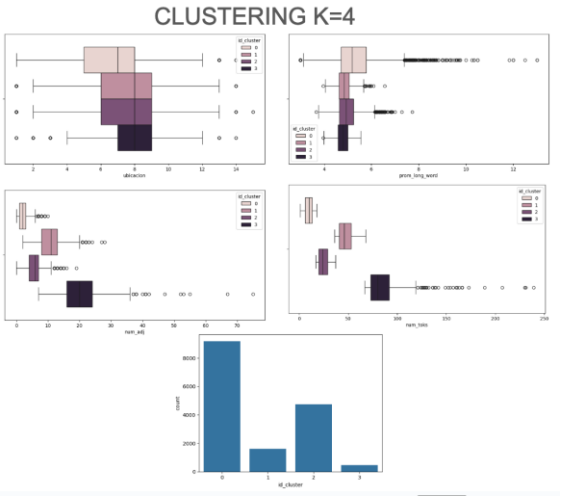


Ilustración 28: : Boxplots con la distribución de las variables para 4 clústers.

Podemos observar cómo no existe diferencia notable entre los ratings medios de los distintos grupos, esto se debe a que en general todos los usuarios concedían valoraciones altas dentro de la escala del 1 al 5, por lo que tampoco permitía demasiada variabilidad. Donde sí que podemos observar claras diferencias es en el número de adjetivos que contiene el comentario, el sentimiento parece no facilitar la segmentación, sino que el algoritmo parece enfocarse más en el tamaño del comentario que en el contenido, vemos como también existe una distinción considerable en la variable tamaño medio de palabra.

El dendograma, muestra resultados similares a el clustering de apartamentos, donde el valor de “k” óptimo parece encontrarse en 4 pero con tamaños heterogéneos.

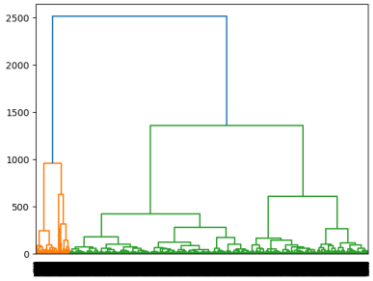


Ilustración 29: Dendograma de usuarios.

En este caso decidimos realizar un estudio de componentes principales mediante el algoritmo de PCA, de esta forma queríamos validar si con esta técnica podíamos facilitar la segmentación a un modelo como K Means. Con 3 componentes principales, decidimos mostrar mediante gráficos de dispersión la relación entre ellos.

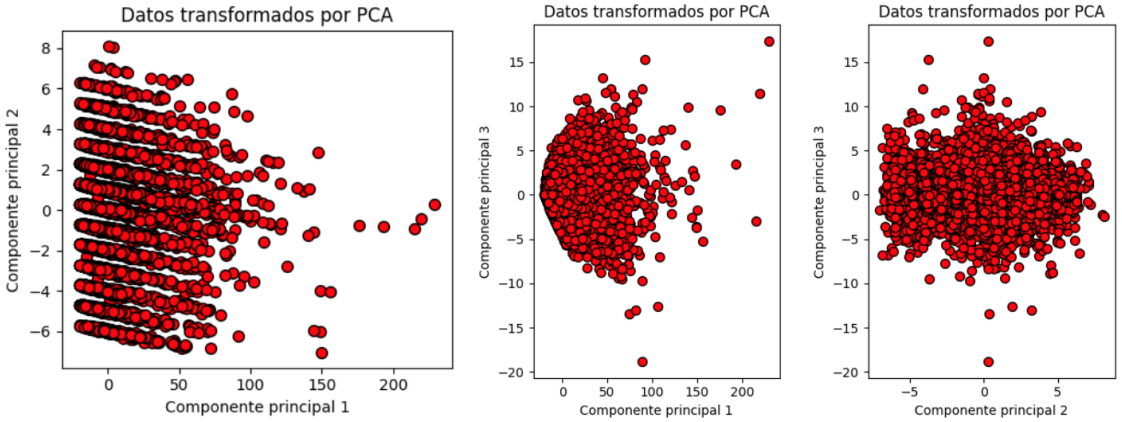


Ilustración 30: Gráficos de dispersión de datos transformados mediante PCA.

En el gráfico de dispersión entre los componentes 1 y 2, se observa una relación lineal parcial, por ello decidimos tratar de segmentar a los usuarios mediante estos componentes principales, sin embargo, los resultados continuaban siendo difusos salvo para el primero de los componentes donde logramos una caracterización significativamente precisa.

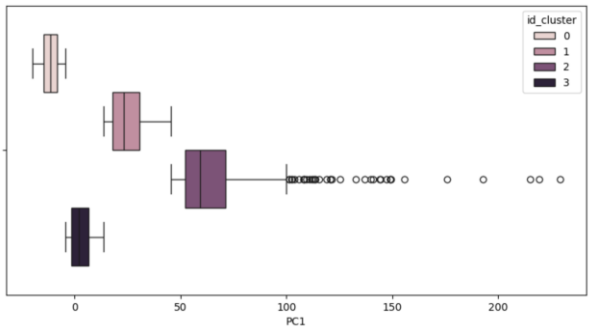


Ilustración 31: Boxplot del Componente Principal 1 para cada clúster.

7.2. REGLAS DE ASOCIACIÓN.

Las reglas de asociación nos permiten descubrir patrones de co-ocurrencia en este caso de usuarios que han visitado múltiples apartamentos, debido a la naturaleza del problema que tratamos de resolver, partíamos de la hipótesis de que no existen patrones de co-ocurrencia entre nuestros datos, para validar dicha hipótesis, decidimos calcular estas reglas de asociación. El paso inicial consiste en crear una matriz de usuarios-apartamentos con valor 1 donde el usuario ha visitado el apartamento y 0 cuando no lo ha visitado. Tras calcular las asociaciones, observamos que no existía ningún conjunto de apartamentos que ocurrieran juntos si no que las reglas nos mostraban que los apartamentos aparecen individualmente.

	support	itemsets
0	0.013320	(0)
169	0.012306	(554)
166	0.011582	(544)
87	0.011582	(277)
25	0.011510	(90)
...	...	...
153	0.001158	(492)
133	0.001086	(429)
29	0.001013	(106)
31	0.001013	(108)
11	0.001013	(36)

Ilustración 32: Support de Itemsets frecuentes.

8. MODELAJE DEL SISTEMA DE RECOMENDACIÓN.

8.1. RECOMENDACIÓN USUARIOS NUEVOS.

En el caso de nuevos usuarios, partimos con varios problemas a resolver: no conocemos las características del usuario ni sus preferencias. Para resolverlo, empleamos el siguiente enfoque:

Ilustración 33:Procedimiento de recomendación de nuevos usuarios.

El nuevo usuario es presentado con un conjunto de selecciones en grupos de 2 apartamento sobre las que elige el que más le gusta, con esta información tenemos un perfilado preliminar del usuario sobre el cual basar nuestras recomendaciones iniciales que se estructuran en 5 bloques principales que son los siguientes:

- *'Los más populares':*
  En este bloque mostramos los apartamentos más populares entre nuestra base de datos de apartamentos considerando tanto el número de evaluaciones como el valor medio de estas evaluaciones, asignando unos pesos definidos por una serie de reglas de negocio donde se trata de prevenir que los nuevos apartamentos (menos evaluaciones) no se vean ocultados por aquellos apartamentos que llevan más tiempo en la plataforma sin perjudicar a estos apartamentos generalmente más elegidos por los usuarios. Actualmente, el 75% del peso de la puntuación de popularidad deriva de la media de los ratings (Calidad, Limpieza, ubicación…) y el 25% restante es debido a la cantidad de evaluaciones que ha recibido el apartamento.

- *'Para ti':*
  Como el propio nombre indica en este bloque se muestran los apartamentos con mayor similitud respecto a la preselección realizada por el usuario, para calcular esta similitud empleamos la media entre 2 métodos de cálculo de similitud: (23)

Método 1 (Distancia Euclidiana):
Distancia Euclidiana, es una medida de distancia especialmente efectiva en variables numéricas continuas (Precio, Capacidad, evaluaciones…) y es robusta ante pequeñas variaciones en los datos. Esta medida se ve afectada por la diferencia de escalas dando mayor peso a las variables de mayor escala, en nuestro caso esto no es un problema ya que en el estudio preliminar observamos que las variables de mayor escala eran las que mejor caracterizan a los apartamentos. Sin embargo, no considera relaciones no lineales ni muestra muy buen rendimiento ante variables binarias o categóricas, en nuestro caso las variables de servicios y tipo de apartamento se ven afectadas por estos problemas, es por ello que decidimos estudiar otro método que trate ambos tipos de variables (numéricas continuas y categóricas) de forma diferente.

$$d(p,q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$$

Método 2 (Distancia Coseno + Similitud Jaccard):

El coeficiente de similitud Jaccard mide la similitud entre conjuntos de datos preferentemente categóricos/binarios ya que compara la presencia/ausencia de elementos en conjuntos y es robusto ante valores ausentes, sin embargo, no considera la intensidad de las relaciones y disminuye su rendimiento ante variables continuas, en nuestro caso nos permitirá medir la distancia entre apartamentos teniendo en cuenta las variables categóricas y binarias cómo los diferentes servicios (Piscina, WiFi…) y el tipo de apartamento.

$$J(A,B) = \frac{A \cap B}{A \cup B}$$

Respecto a las variables continuas, emplearemos la distancia coseno ya que es con estas donde su rendimiento es elevado, además, tiene la capacidad de modelar relaciones lineales y no lineales sin verse afectado por la escala de las variables, gracias a la selección de variables realizada previamente, evitamos sufrir que esta distancia no considera las correlaciones entre las variables.

Al combinar ambas medidas, aprovechamos las ventajas de Jaccard sobre las variables categóricas/binarias y reducimos sus problemas con las variables continuas y viceversa para la distancia Coseno.

$$cos(\propto) = \frac{x \cdot y}{\| x \| \| y \|}$$

- *‘En la playa’, ‘En el campo’ y ‘Cabañas’*
  Para cada uno de estos 3 bloques, recomendamos los 4 apartamentos más similares según el método 2 filtrados por tipo, en el caso de no haber suficiente información, completamos los apartamentos restantes con los más populares del tipo específico.

Este enfoque nos permite resolver la desinformación inicial en usuarios nuevos proporcionando recomendaciones personalizadas durante su primera interacción con el sistema, una vez disponemos de información de las preferencias de los usuarios, recomendamos según el enfoque de usuarios existentes que se expone a continuación.

8.2. RECOMENDACIÓN USUARIOS EXISTENTES.

Conocidos los gustos y preferencias de los usuarios, podemos realizar un perfilado de estos y hacer recomendaciones más complejas empleando sistemas de recomendación híbridos. Nuestro modelo de recomendación híbrido se basa en un filtrado colaborativo basado en usuarios junto con un sistema de filtrado colaborativo basado en modelos, concretamente modelos de regresión.

8.2.1. MODELO DE FILTRADO COLABORATIVO BASADO EN USUARIO.

Como ya se ha explicado anteriormente, un modelo de filtrado colaborativo basado en usuario se basa en recomendar, en este caso, alojamientos a un usuario a partir de la información de usuarios con patrones de búsqueda y valoraciones similares al usuario en cuestión.

En nuestro caso esta similitud se medirá con el coeficiente de correlación de Pearson.

$$\rho_{X,Y} = \frac{Cov(X,Y)}{\sqrt{Var(X) \cdot Var(Y)}}$$

Este índice nos servirá para medir el grado de relación entre los usuarios, un mayor valor indicará una mayor similitud entre usuarios, representando el 1 una correlación perfecta positiva siendo el -1 la perfecta negativa.

La ventaja de usar el coeficiente de correlación de Pearson frente a otras medidas como puede ser la distancia de coseno ya que, al tratarse de valoraciones personales, que suelen tener su base de objetividad pero también una componente de subjetividad muy fuerte, es capaz de tener en cuenta los sesgos que cada uno de los usuarios puede introducir es sus valoraciones.

8.2.2. MODELO DE FILTRADO COLABORATIVO BASADO EN MODELOS.

El objetivo inicial es crear un modelo a partir de la información de los comentarios, las características de los usuarios y las características del apartamento para predecir el posible rating que el usuario potencialmente daría a un nuevo apartamento, para crear este modelo empleamos estudiamos 2 modelos con enfoques diferentes: un modelo centrado en las características del usuario y el apartamento y otro modelo que mediante embeddings procese además de dichas características, la información del comentario.

Catboost Regressor:

CatBoost es un algoritmo de machine learning basado en Gradient Boosting desarrollado por Yandex, diseñado específicamente para manejar eficientemente variables categóricas sin necesidad de un extenso preprocesamiento, como la codificación One-Hot. Este algoritmo se destaca por su capacidad para trabajar con conjuntos de datos mixtos (numéricos y categóricos) y por su robustez frente al *overfitting*, gracias a técnicas avanzadas como la ordenación por permutación y la incorporación de datos por hoja. Se utiliza tanto en tareas de clasificación como de regresión donde es capaz de procesar datos categóricos, ofreciendo alta precisión y menor tiempo de entrenamiento en comparación con otros métodos de boosting, como XGBoost y LightGBM. Combinando las bases de datos de comentarios y apartamentos, disponemos de observación con características del usuario y del apartamento, debido al elevado coste computacional en sets grandes de datos, descartamos la potencial información de los comentarios, sin embargo, como se ha expuesto con anterioridad, en el modelo de embeddings si se tendrá en cuenta.

Tras dividir nuestros datos en una muestra de entrenamiento (80%) y testeo (20%), donde el rating proporcionado por el usuario es la variable a predecir, entrenamos el modelo con la muestra de entrenamiento. Con el fin de evaluar el rendimiento del modelo, calculamos las métricas de error medio absoluto, error medio cuadrado y R^2, obteniendo:

- MAE → 0.27
- MSE → 0.22
- R2 → 0.174

Las métricas obtenidas son positivas, siendo el error del modelo considerablemente reducido, para comprender qué variables tienen mayor efecto sobre la predicción, mostramos un gráfico con la importancia de las variables.

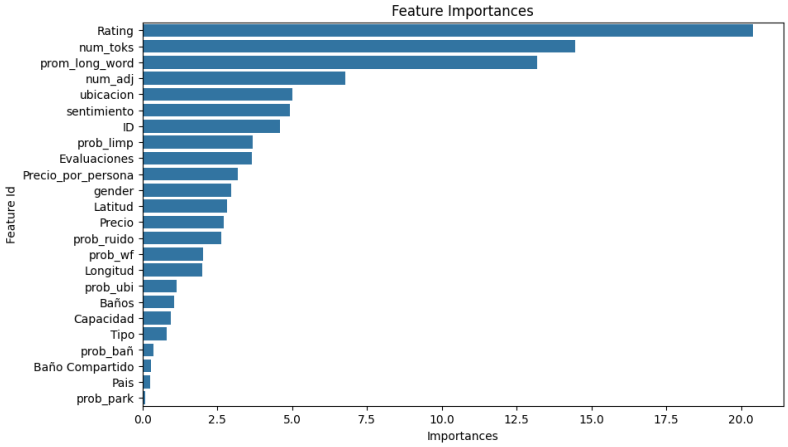


Ilustración 34: Influencia de las variables en el rating final.

Las variables con mayor influencia se deben al rating general del apartamento y la longitud promedio de los comentarios de los usuarios, longitud tanto del comentario como de las palabras empleadas en el comentario. El conjunto de datos que más información aporta es la base de datos de comentarios. Para validar que hemos logrado capturar adecuadamente los patrones presentes en los datos, realizamos un estudio sobre los residuos.

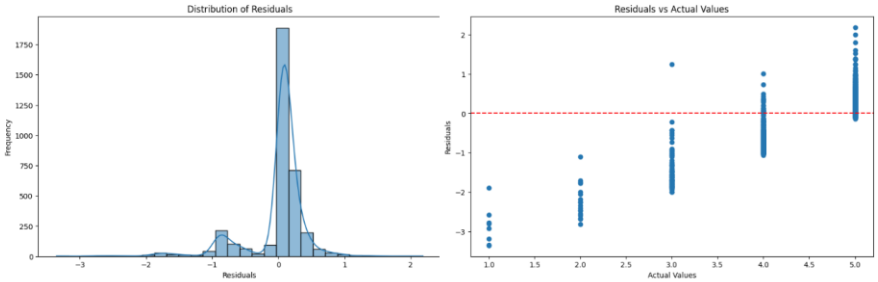


Ilustración 35: Distribución y comparación de residuos frente a los valores reales.

Si bien la distribución de los residuos parece normalizada, al mostrar los residuos frente a los valores reales, se observan tendencias claras con el valor real, esto nos indica que hay presencia de heterocedasticidad. Finalmente, calculamos los *shap values*, estos nos permiten comprender la contribución de cada característica sobre las predicciones del modelo.

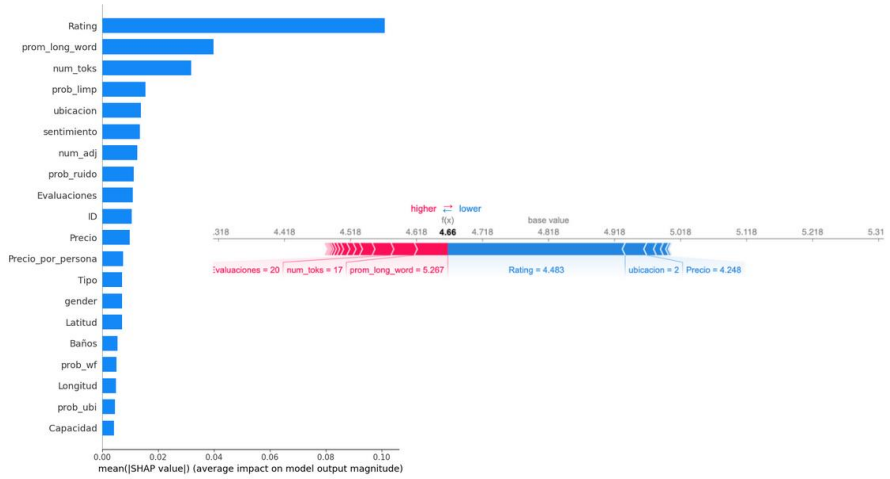


Ilustración 36: Shap values de cada variable para la predicción.

Los resultados reafirman las conclusiones extraídas con las importancias de catboost donde el rating del apartamento y las características medias del tamaño de los comentarios son las variables de mayor influencia. En conclusión, el modelo muestra un rendimiento adecuado, sin embargo, requiere de mayor estudio para optimizar su rendimiento empleando técnicas como el ajuste fino de hiper parámetros con un mapeo de valores como hace por ejemplo GridSearch.

Modelo de deep learning basado en Embeddings

Tensorflow (24) es un framework de aprendizaje automático desarrollado por Google, diseñado para facilitar la construcción, entrenamiento y despliegue de modelos de inteligencia artificial.

Para perfilar al usuario disponemos de información de su identificador, su ubicación, su género y características de su personalidad extraídas a partir de los comentarios realizados sobre los apartamentos, además, contamos con la puntuación que han otorgado a los apartamentos en un rango de 0-5.

La arquitectura del modelo basado en embeddings es la siguiente:

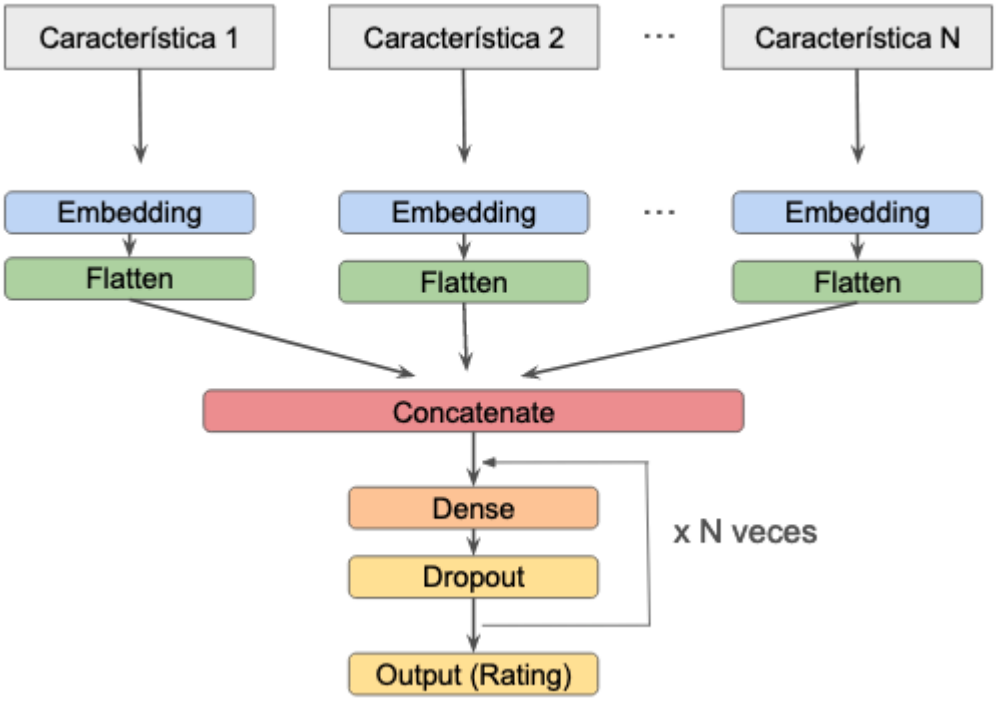


Ilustración 37:Arquitectura de la red neuronal.

Cada una de las características del usuario de forma paralela son pasadas por una capa de embeddings que posteriormente es aplanada. Los embeddings nos permiten capturar relaciones semánticas y similitudes entre usuarios ya que los usuarios similares tienden a tener representaciones en el espacio vectorial cercanas, en adición, transforman datos de elevada dimensionalidad como la representación vectorial del texto en vectores densos de menor dimensionalidad manejando estos datos de forma más eficiente reduciendo el coste computacional del entrenamiento del modelo.

La concatenación de vectores aplanados es posteriormente enviada a una serie de N capas densas con regularización *Dropout*, la regularización intenta resolver el problema del sobreajuste de los modelos que ocurre cuando estos capturan con elevada precisión los patrones existentes en los datos de entrenamiento, pero su rendimiento en datos no vistos durante el entrenamiento disminuye considerablemente. El *Dropout* consiste en 'apagar' aleatoriamente un porcentaje de neuronas de la red en cada paso de entrenamiento, es decir, se desactivan temporalmente y su información no contribuye al aprendizaje, obligando a la red a aprender características más genéricas de los datos previniendo el sobreajuste.

Las funciones de activación determinan la salida de una neurona y generalmente son una combinación lineal de las entradas recibidas por dicha neurona, su propósito es introducir no linealidad en el modelo permitiendo representar relaciones más complejas. Para estas capas densas empleamos *Leaky ReLU*

(Rectified Linear Unit), una modificación de su antecesora ReLU que corrige el problema de las 'neuronas muertas' que ocurre cuando ReLU recibe entradas negativas y las convierte a 0, *Leaky ReLU* introduce una leve pendiente que además de evitar este problema suaviza el comportamiento de la función facilitando en algunos casos la convergencia.

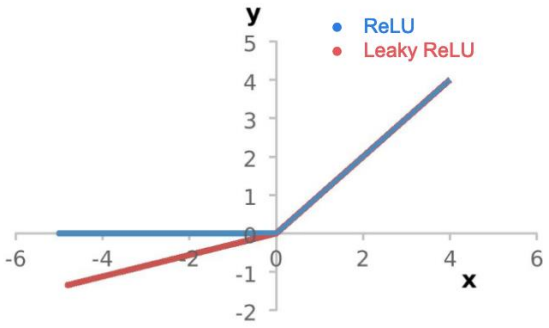


Ilustración 38:: Gráfica de ReLU vs. Leaky ReLU.

Como optimizador de nuestro modelo de red neuronal empleamos *Adam* (Adaptive Moment Estimation), este algoritmo de optimización combina las ventajas del descenso por gradiente con capacidad adaptativa de la tasa de aprendizaje y empleando momentos para ajustar dicha tasa. El extendido uso de este algoritmo reside en su eficiencia para alcanzar la convergencia ayudando además a evitar mínimos locales en el espacio de búsqueda y su comportamiento robusto ante gran variedad de aplicaciones. En nuestro caso una tarea de regresión con el error medio cuadrado como función de pérdida.

Para aumentar el control durante el proceso de entrenamiento, introdujimos un conjunto de callbacks, estos son un conjunto de funciones que se ejecutan en momentos específicos del entrenamiento para realizar funciones como: guardado de modelos durante las etapas del entrenamiento para evitar problemas en caso de ocurrir un error durante el entrenamiento, *earlystopping* que permite detener el entrenamiento en caso de que durante varias épocas del entrenamiento no estemos mejorando el rendimiento del modelo, la visualización de gráficos … En nuestro caso implementamos un *ModelCheckpoint* para ir almacenando el modelo que haya registrado un mayor rendimiento hasta el momento y un *EarlyStopping* para detener el entrenamiento en caso de que el modelo no mejore durante 5 épocas consecutivas.

Entrenamiento del modelo y Resultados

Previo al entrenamiento del modelo, deben definirse 2 parámetros fundamentales: el número de epochs que se corresponde con la cantidad de pasadas de entrenamiento sobre el dataset completo y el tamaño del batch que se corresponde con la cantidad de observaciones con las que entrenar el modelo antes de actualizar los pesos.

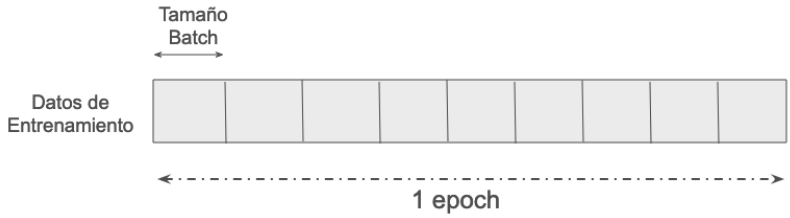


Ilustración 39:Detalle de epoch y batch.

Consideramos 100 épocas de entrenamiento suficientes para alcanzar los criterios de convergencia, sin embargo, para el tamaño de batch decidimos estudiar el comportamiento de modelo para diferentes valores:

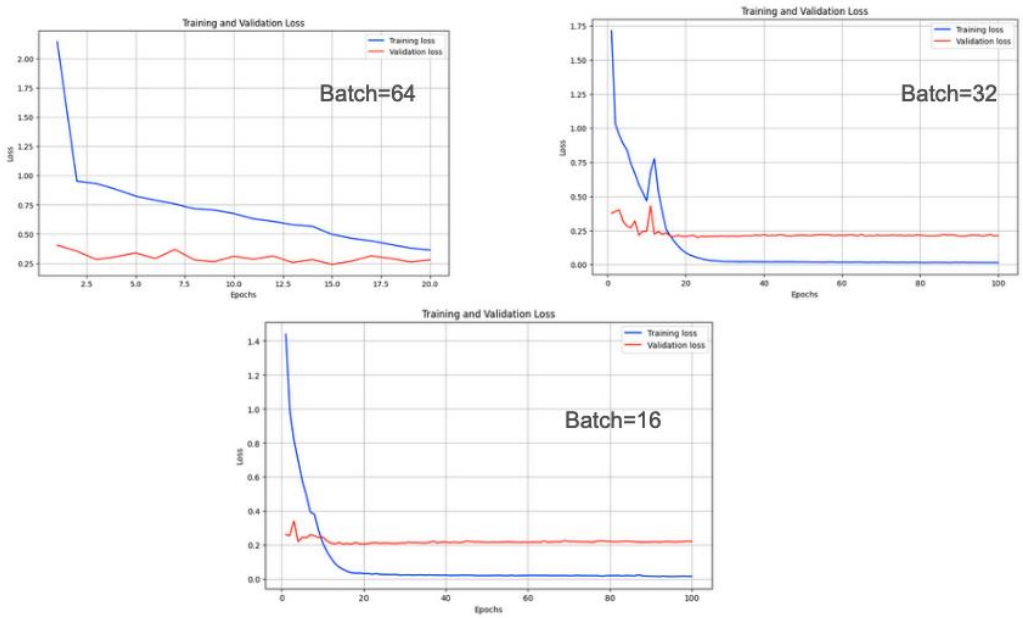


Ilustración 40:Gráficas con la evolución del entrenamiento para distintos valores de batch.

Observamos que a mayor es el tamaño del batch, más tarda el modelo en aprender y se evita el problema de sobreajuste con las consecuencia de que el modelo no aprenda completamente, por el otro lado, con un tamaño reducido de batch el modelo consigue aprender más para los datos de entrenamiento pero no con los de validación, para solventar este problema podemos aumentar la regularización (considerando que el Dropout usado es del 50%), aumentar la cantidad de datos para evitar el sobreajuste y capturar patrones más generales, sin embargo, actualmente no es posible hasta que los usuarios con el tiempo nos proporcionen más información y finalmente podemos tratar de simplificar el modelo este enfoque es el más simplista pero el más alcanzable en el corto plazo.

Los residuos del modelo son las diferencias entre los valores esperados y los valores predichos, estos proporcionan información útil sobre la calidad del modelo, una distribución uniforme alrededor de 0 indica que el modelo tiene un buen ajuste, si estos residuos muestran patrones no capturados como por ejemplo tendencias, quiere decir que existe alguna estructura de datos que no ha sido correctamente

modelada. Si la variabilidad de los residuos cambia a lo largo del rango de valores predichos observamos la presencia de heterocedasticidad. En nuestro caso, observamos que los residuos presentan heterocedasticidad.

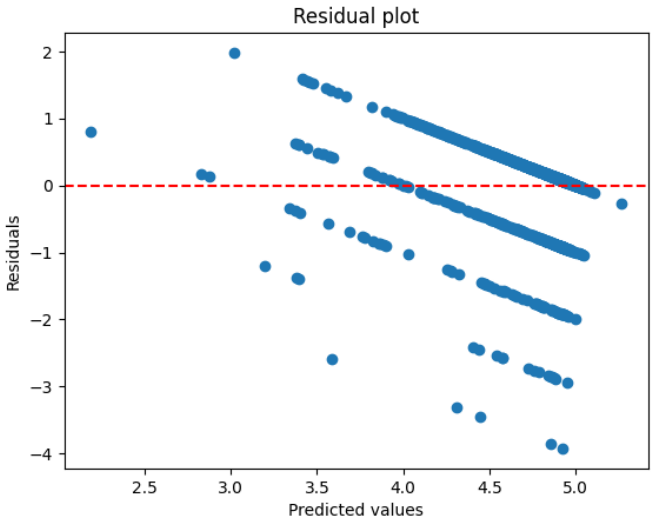


Ilustración 41:Residuos de las predicciones del modelo de Deep Learning.

En conclusión, este modelo de redes neuronales muestra un buen rendimiento con un error medio absoluto de 0.4, sin embargo, aún hay mucho potencial a extraer del mismo, se propone volver a estudiar el modelo incluyendo más información y regularización para evitar el sobreajuste, simplificar la arquitectura (quizás no es necesario incluir los comentarios) y mejorar la calidad de los datos para normalizar los residuos y mejorar el rendimiento general del modelo.

8.2.3. COMBINACIÓN DE MODELOS.

El modelo de filtrado colaborativo basado en modelos a emplear será Catboost ya que ha mostrado un mayor rendimiento en general y una mayor normalidad en los residuos, esto nos permite seguir estudiando nuevos modelos y optimizando el rendimiento principalmente del modelo de embeddings mientras mantenemos el servicio. Para implementar un modelo híbrido aprovechando el modelo de Catboost y el modelo de filtrado colaborativo, vamos a implementar una combinación lineal de los ratings obtenidos por ambos mediante la siguiente fórmula:

$$Rating\ Final\ = alpha\ *\ Rating\ Catboost\ +\ (1-alpha)*\ Rating\ F.Colaborativo$$

De esta forma somos capaces de capturar relaciones complejas y ofrecer contenidos altamente personalizados en función del perfilado del usuario realizado por ambos modelos.

8.3 JUSTIFICACIÓN DE LA SELECCIÓN DE ALGORITMOS

Consideramos emplear un sistema de recomendación basado en contenidos para los nuevos usuarios debido a que estos algoritmos son especialmente útiles cuando disponemos de limitada o nula información del usuario. En el caso de la recomendación para usuarios existentes consideramos emplear un sistema híbrido por los siguientes motivos:

- El sistema de filtrado colaborativo basado en usuarios, unicamente tiene en cuenta la similud de gustos entre usuarios con perfiles similars, sin embargo, no tiene en cuenta directamente la influencia de las características de los apartamentos a la hora de recomendar el rating.

- El sistema de filtrado colaborativo basado en modelos, considera la influencia de las características tanto del apartamento como del usuario sobre el gusto del usuario pero no tiene en cuenta las similitudes de dicho usuario con otros.

Mediante un sistema híbrido, aprovechamos las ventajas combinadas de ambos y cubrimos sus debilidades con la información aportada por el otro respectivamente. En cuanto a la selección entre los modelos de filtrado colaborativo, su justificación se apoya en la siguiente tabla:

	Coste Computacional	Latencia	Rendimiento	Considera comentarios	Residuos
Catboost	✓	✓	✓		✓
Modelo Embeddings			✓	✓	

El modelo con catboost, si bien es cierto que no considera los información presente en el texto del comentario como si hacen los embeddings, el coste computacional y la latencia del modelo durante la predicción lo convierten en la opción óptima actualmente, además, los residuos muestra un comportamiento mas normalizado. Sin embargo, proponemos seguir estudiando modelo a medidas que se incorporen nuevos datos al sistema, siguiendo la metodología CI/CD para integrar y desplegar las modificaciones sobre el modelo en caso de haberlas de forma ágil y veloz.

9. SOLUCIÓN PROPUESTA.

La solución final basada en los resultados obtenidos a lo largo del proyecto, se detalla a continuación mediante un esquema del flujo de datos, una proposición de arquitectura basada en microservicios con Azure y un MVP (Minimum Viable Product) realizado en paralelo para validar la viabilidad de la solución.

9.1. FLUJO DE DATOS.

Inicialmente, cualquier dato nuevo a incorporar en el sistema tras haber sido extraído mediante Web Scraping, es procesado mediante su ETL correspondiente tal y como se muestra en la figura donde los datos inicialmente son transformados a través de los pipelines correspondientes definidos con anterioridad de limpieza y selección de variables.

Ilustración 42: Proceso de ETL del proyecto.

Una vez disponemos de los datos procesados, estos son almacenados en la base de datos correspondiente, para los apartamentos se almacenan en la base de datos de características de apartamentos, para los usuarios se almacenan en la base de datos de usuarios que cuenta con la información descriptiva de cada uno de ellos y finalmente las interacciones que se almacenan en una base de datos NoSQL en formato clave-valor. Observando el flujo de datos con una visión más amplia, estos datos procesado y almacenado son proporcionados mediante diferentes métodos según el usuario y sus necesidades, en el caso de los hosts, cuyo interés principal es entender su competencia y analizar el mercado para maximizar el rendimiento, enviamos los datos limpios sin ningún cálculo a la aplicación que será la encargada de mostrar los datos clave y gráficos requeridos por el usuario.

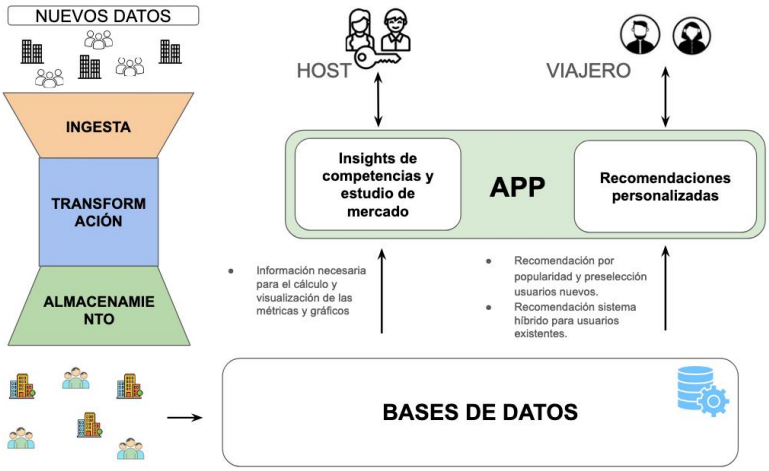


Ilustración 43: Flujo completo de los datos en el proyecto.

Para los viajeros en busca de los apartamentos que más se adaptan a sus gustos, enviamos la información limpia a una función encargada de calcular las recomendaciones en función del usuario (nuevo o existente) y proporcionar a la aplicación dicha información para ser mostrada al usuario.

9.2. ARQUITECTURA DEL SISTEMA.

Presentamos una arquitectura basada en servicios de Azure diseñada para ofrecer un sistema de recomendación ágil, robusto, dinámico y escalable permitiendo además la integración y entrega continua (CI/CD). Esta arquitectura aprovecha las ventajas de la computación en la nube reduciendo los costes de inversión del sistema y facilitando la escalabilidad en caso de un crecimiento espontáneo de usuarios en la web. Empleando el servicio App Service, lanzamos la interfaz de usuario de nuestra aplicación junto con una API desplegada encargada de comunicar las funcionalidades de recomendación codificadas en Python con el frontend.

El almacenamiento de datos seguro y escalable proporcionado por Azure permite con costes reducidos diseñar una estructura de almacenamiento eficiente, por un lado, empleando Azure SQL Database almacenamos de forma estructurada la información de los apartamentos y la información de los usuarios, por otro lado, con un formato NoSQL almacenamos de forma no estructurada las interacciones usuario-apartamento para de esta forma recuperar dicha información con elevada velocidad.

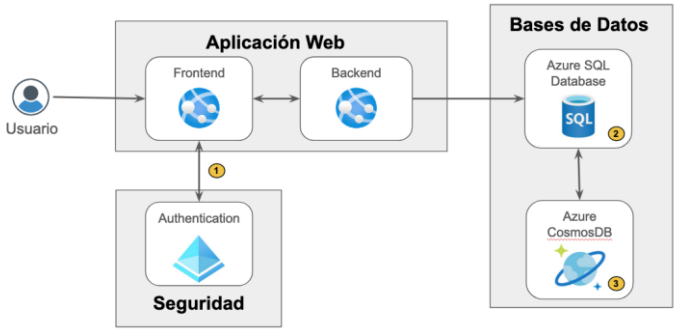


Ilustración 44:Arquitectura propuesta.

Para la gestión de accesos y almacenamiento encriptado de contraseñas empleamos los servicios de autenticación y encriptación disponibles en Azure.

9.3. APLICACIÓN WEB (MVP).

Con el fin de validar la propuesta de valor como el propio sistema recomendador, hemos desarrollado una aplicación web intuitiva y visual, con ayuda de la librería *Streamlit* (25), que permite a los usuarios explorar y recibir recomendaciones personalizadas de apartamentos de manera eficiente y atractiva, ha sido diseñada pensando en la facilidad de uso y en proporcionar una experiencia agradable al usuario durante su búsqueda de alojamiento, así como disponibilizar el estudio de apartamentos y usuarios para ayudar a los arrendadores a maximizar el rendimiento y estudiar su competencia para incrementar su ROI (Retorno sobre la inversión) potencial. Inicialmente el usuario es mostrado con el proceso de preselección expuesto anteriormente para permitir un perfilado preliminar sobre el cual recomendar. Una vez termine la selección llegará a la pestaña final donde estarán las cuatro categorías de apartamentos recomendados mencionadas anteriormente: Los más populares, Para ti, En el campo, En la playa y Cabañas. Clicando en la parte superior izquierda se despliega un menú que permite acceder al resto de características de la aplicación.

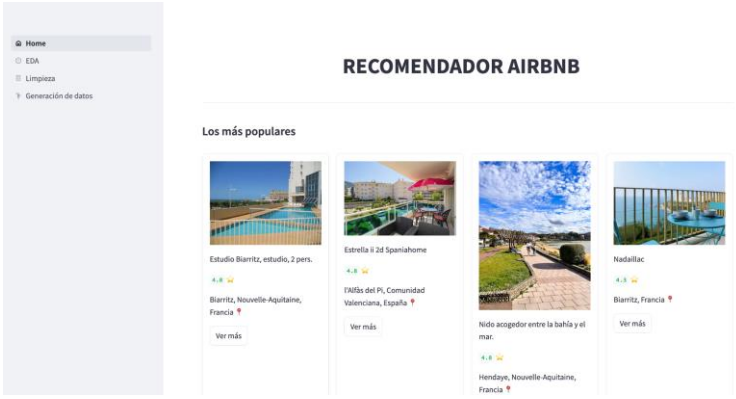


Ilustración 45:Pestaña principal con las distintas recomendaciones.

Estas se dividen en 4, ‘Home’ es la página principal y es donde se encuentran las recomendaciones para el usuario. ’EDA’, permite acceder a los estudios preliminares realizados sobre la base de datos de apartamentos y comentarios, mediante un sistema de botones permitimos mostrar y ocultar diferentes apartados del estudio, a continuación se muestra un ejemplo con el estudio del precio.

Ilustración 46:Pestaña destinada al EDA.

De cara a poder reproducir el estudio y el propio modelo de recomendación así como la inclusión de nuevos aspectos en el análisis, diseñamos un apartado que permite al usuario generar su propia base de datos de apartamentos y comentarios de manera sencilla, con 2 condiciones, el usuario debe estar ejecutando la página en local (este método reduce la facilidad de uso de la aplicación, sin embargo, consideramos que este aspecto solo aplica a usuarios técnicos o de elevado interés) y debe tener instalado Google Chrome y activo como su navegador predeterminado.

Para ejecutar la página en local el usuario debe seguir los siguientes pasos:

1. ***Clonar repositorio:*** git clone https://github.com/pablolegeren/TFM.git
2. ***Instalación de paquetes (requiere python 3.12.4):*** pip install -r requirements.txt
3. ***Ejecución del lanzamiento:***

    3.1. Usuarios linux /MacOS: ./start.sh

3.2. Usuarios Windows: ./start.ps1

Tras arrancar la aplicación en local, el usuario mediante una sencilla interfaz puede definir la cantidad de apartamentos a extraer por cada tipo elegido (‘En la playa’,’Cabañas’...), define el número de versión de sus datos y si desea extraer también la información de los comentarios.

Una vez generados los datos, son mostrados al usuario junto con la posibilidad de descargarlos como un archivo .csv. El tiempo de obtención de la información se calcula mediante la siguiente fórmula:

$$Tiempo\ extracción = 2min + 15s * N°Registros * N°Tipos\ de\ Apartamento$$

Como se observa en la imágen los datos extraídos no están procesados, para procesarlos accediendo a la pestaña ‘Limpieza’ podemos subir el .csv generado y descargar el .csv ya procesado para el análisis.

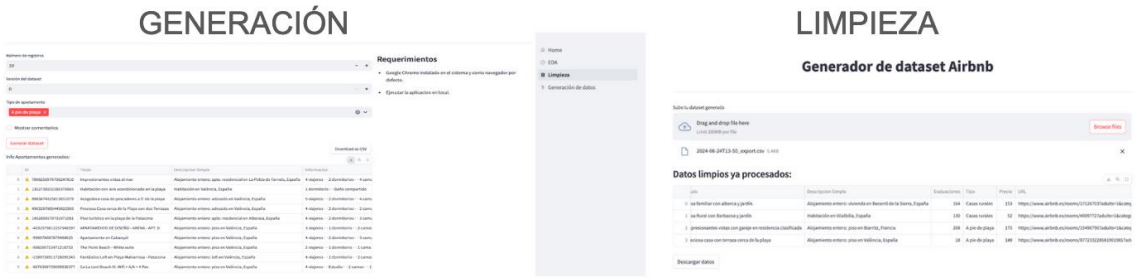


Ilustración 47: Pestaña de replicación de la limpieza y generación de las bases de datos.

Mediante este MVP, podemos validar la utilidad de nuestra solución propuesta para decidir si el proyecto es viable o no, en esta aplicación permitimos a los usuarios que buscan un destino vacacional encontrar apartamentos adaptados a sus preferencias y necesidades. Adicionalmente, proporcionamos un valor diferencial permitiendo a arrendadores de este tipo de apartamentos estudiar a su competencia y entender las tendencias existentes para analizar posibles modificaciones sobre sus propios apartamentos para maximizar el rendimiento.

10. CONCLUSIONES Y TRABAJOS FUTUROS.

Este trabajo tiene principalmente dos objetivos: proporcionar recomendaciones personalizadas a los arrendatarios e información valiosa a los arrendadores.

La mayor diferencia de esta propuesta con la propia plataforma de AirBnb es la personalización de las recomendaciones mediante el filtrado colaborativo basado en usuarios. Airbnb ofrece alojamientos independientemente del perfil del usuario, basándose únicamente en los filtros que el usuario introduzca. Al poder establecer similitudes entre usuarios somos capaces de perfilar a los usuarios ofreciendo así una mayor adaptabilidad a sus necesidades específicas.

Otra de las ventajas o diferencias con la plataforma turística es la capacidad de proporcionar al arrendador insights acerca del mercado, facilitando así que aumente su competitividad y que potencie lo que la gente valora positivamente y solucione lo que se valora negativamente. Esta capacidad de analizar el mercado y detectar posibles tendencias no la ofrece Airbnb.

En conclusión, este proyecto sitúa al propio usuario como uno de los ejes sobre el que se apoya ofreciendo una experiencia en la cual pueden obtener opciones más personalizadas y una ventaja competitiva sobre el resto de usuarios. Habiendo comentado las ventajas que supone, cabría también comentar posibles puntos a mejorar o líneas abiertas de trabajo de cara a un mejor rendimiento.

El primero de todos sería poder lograr unos mejores resultados en el modelo de Deep Learning, para ello se podría realizar una búsqueda de los parámetros e hiper parámetros que mejores resultados ofrecen. Otra medida que podría mejorar los resultados del modelo es el aumento de los datos de entrenamiento, tener un mayor número de datos podría llevar al modelo a encontrar patrones o relaciones que no es capaz de hacer con la cantidad utilizada.

Sabiendo que gran parte de la diferenciación con Airbnb viene de la relación establecida entre usuarios, la segunda posible mejora sería la profundización en la caracterización de los usuarios. Poder obtener un perfil mucho más detallado y completo de cada uno de los usuarios ayudaría a la hora de poder ofrecer recomendaciones más acertadas. Otra de las posibles medidas para obtener recomendaciones más acertadas y ajustadas a las necesidades o intereses de los usuarios, sería poder ofrecer alojamientos ubicados en una zona geográfica que el usuario especifique. Asimismo, la posibilidad de extender la región de estudio más allá de la Península Ibérica aumentaría considerablemente el número de apartamentos ofertados y, por tanto, la posibilidad de encontrar alguno que se adecúe al arrendatario.

Respecto a la aplicación web final, más allá de un posible rediseño para hacerla más atractiva visualmente, una de las mejoras que hemos planteado ha sido la construcción de un ChatBot que pueda ofrecer soporte al usuario y que actúe como asistente personal dentro de la aplicación. Esto aportaría un valor añadido al cliente facilitando su búsqueda de alojamiento.

Acerca del apartado de EDA, quedaría pendiente poder obtener una clusterización de mejor calidad. Usar más variables o incluir más observaciones podría mejorar los resultados, siendo capaz así de encontrar diferencias palpables entre los clústers.

Sobre este mismo apartado, un avance sería la automatización de los gráficos empleados actualmente o de los que reflejen las necesidades e inquietudes del arrendador. Poder visualizar estos datos brindaría al dueño la capacidad de observar posibles tendencias o factores clave del mercado sin necesidad de programación. La parte negativa de esta posible solución es que el usuario tendría que ser capaz de, a partir de estas gráficas, extraer información que le resulte relevante.

ANEXO.

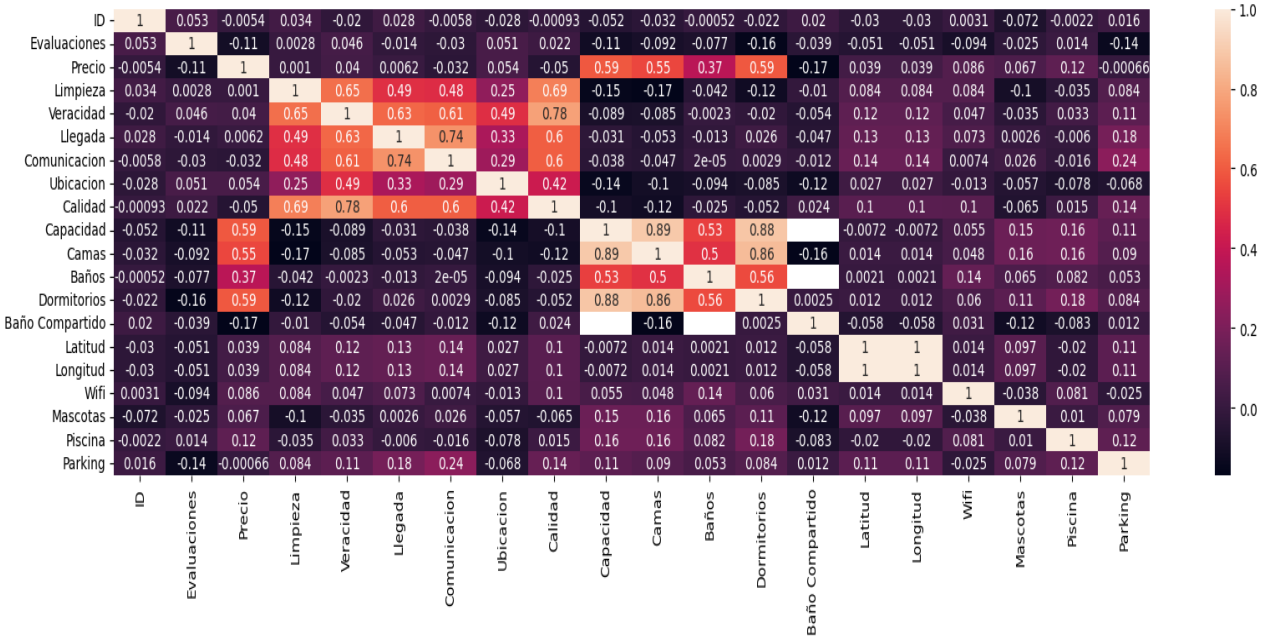


Ilustración 48:Mapa de calor con los coeficiente de correlación de todas las variables de los apartamentos.

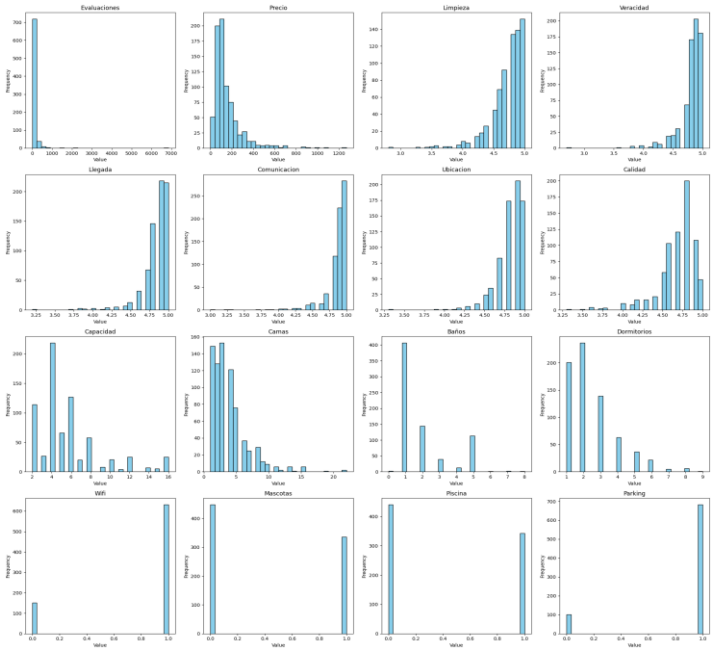


Ilustración 49:Distribución de las variables numéricas del dataset de apartamentos.

BIBLIOGRAFÍA.

(1) Elobosbable. (2014). Las 4 Vs de Big Data. Medium. Retrieved from https://medium.com/@elobosbable/las-4-vs-de-big-data-f7cd441ac31a

(2) Wikipedia contributors. (n.d.). Airbnb. Wikipedia. Retrieved from https://es.wikipedia.org/wiki/Airbnb

(3) Statista. (2022). Número de noches y experiencias reservadas en Airbnb. Retrieved from https://es.statista.com/grafico/26863/numero-de-noches-y-experiencias-reservadas-en-airbnb/

(4) Glassdoor. (2024). Sueldo: Data Scientist. Retrieved from https://www.glassdoor.es/Sueldos/data-scientist-sueldo-SRCH_KO0,14.htm

(5) Microsoft Azure. (2024). Azure SQL Database Pricing. Retrieved from https://azure.microsoft.com/en-us/pricing/details/azure-sql-database/single/

(6) Microsoft Azure. (2024). Azure Cosmos DB Pricing. Retrieved from https://azure.microsoft.com/en-us/pricing/details/cosmos-db/autoscale-provisioned/

(7) Microsoft Azure. (2024). Azure Active Directory B2C Pricing. Retrieved from https://azure.microsoft.com/en-us/pricing/details/active-directory-b2c/

(8) Chyun55555. (2022). Basket Analysis and Association Rules. Medium. Retrieved from https://medium.com/@chyun55555/basket-analysis-and-association-rules-2c000d62e673

(9) Villalonga, R. (2017). Sistemas Recomendadores Basados en Contenido. Medium. Retrieved from https://medium.com/@rvillalongar/sistemas-recomendadores-basados-en-contenido-ece0227e7005

(10) Airbnb Engineering. (2019). Machine Learning Powered Search Ranking of Airbnb Experiences. Medium. Retrieved from https://medium.com/airbnb-engineering/machine-learning-powered-search-ranking-of-airbnb-experiences-110b4b1a0789

(11) Real Python. (n.d.). Python Web Scraping: Practical Introduction. Retrieved from https://realpython.com/python-web-scraping-practical-introduction/

(12) PyPI. (n.d.). Beautiful Soup 4. Retrieved from https://pypi.org/project/beautifulsoup4/

(13) PyPI. (n.d.). Undetected Chromedriver. Retrieved from https://pypi.org/project/undetected-chromedriver/

(14) PyPI. (n.d.). Selenium. Retrieved from https://pypi.org/project/selenium/

(15) PyPI. (n.d.). Pandas. Retrieved from https://pypi.org/project/pandas/

(16) PyPI. (n.d.). Gender Guesser. Retrieved from https://pypi.org/project/gender-guesser/

(17) PyPI. (n.d.). DeepFace. Retrieved from https://pypi.org/project/deepface/

(18) PyPI. (n.d.). TextBlob 0.9.0. Retrieved from https://pypi.org/project/textblob/0.9.0/

(19) PyPI. (n.d.). Geopy. Retrieved from https://pypi.org/project/geopy/
(20) CatBoost. (n.d.). CatBoost. Retrieved from https://catboost.ai/
(21) GeeksforGeeks. (n.d.). K-Means Clustering: Introduction. Retrieved from https://www.geeksforgeeks.org/k-means-clustering-introduction/
(22) GeeksforGeeks. (n.d.). DBSCAN Clustering in ML: Density Based Clustering. Retrieved from https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/?ref=header_search
(23) Graph Everywhere. (n.d.). Algoritmos de Similaridad. Retrieved from https://www.grapheverywhere.com/algoritmos-de-similaridad/
(24) PyPI. (n.d.). TensorFlow. Retrieved from https://pypi.org/project/tensorflow/
(25) Streamlit Documentation. (n.d.). Retrieved from https://docs.streamlit.io/

ÍNDICE DE ILUSTRACIONES.