Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

# Trabajo de Fin de Grado

## Grado en Ingeniería Informática

Sistema de Recomendación de actividades turísticas: mejorando la experiencia del viajero con Inteligencia Artificial

Tourist Activity Recommendation System: Enhancing the Traveler's Experience with Artificial Intelligence

Gabriel Rodríguez Hernández

D. **Julio Antonio Brito Santana**, con N.I.F. 42.812.193-Q profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas, Departamento de la Universidad de La Laguna, como tutor

**C E R T I F I C A**

Que la presente memoria titulada:

*“Sistemas de recomendación de itinerarios y paquetes turísticos”*

ha sido realizada bajo su dirección por D. **Gabriel Rodríguez Hernández**,

con N.I.F. 43486177P.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 4 de septiembre de 2023

# Agradecimientos

En primer lugar quiero agradecer a mi madre
por siempre haberme apoyado en los buenos momentos
y en los malos. Me ha dado libertad y confianza en todo momento
y siempre me ha aconsejado en todos los procesos.
Esta etapa sin ella hubiera sido muy dura,
gracias por tu dedicación y comprensión.

También quiero agradecer al resto de familiares
por siempre darme ánimos en cualquier momento,
ayudarme a elegir este camino, en esta carrera,
y brindarme oportunidades profesionales
para crecer haciendo lo que más me gusta.

Gracias a todos aquellos compañeros y amigos que
han formado parte de este camino, ya que también
han sido un apoyo esencial para
superar ciertas situaciones dadas.

Por último agradecer al profesorado, por haber
compartido conmigo sus conocimientos y ayudarme
a desarrollar las habilidades necesarias
para seguir este camino.

# Licencia

## Resumen

Nuestro objetivo con este proyecto es trabajar en la tarea de recomendar actividades turísticas a los usuarios, de manera que estas, se adapten a sus preferencias. Para ello desarrollamos una solución, que implementa un sistema de recomendación, el cual esta basado en la captura directa de las preferencias de los usuarios asociada a distintas categorías de actividades a realizar en el destino a través de botones interactivos, un clasificador de actividades para categorizarlas eficientemente según su grado de pertenencia a cada una de ellas, y medidas de similitud que seleccionan aquella actividades más cercanas a las

El preprocesamiento de datos implica la extracción de información textual relevante de las descripciones de las actividades, que vendrá en un formato HTML, por ello también gestiona la eliminación de etiquetas y palabras innecesarias. Para la clasificación se utilizará un enfoque de ponderación de palabras. La fase de recomendación se basa en el cálculo con medidas de similitud, entre las preferencias del usuario y las actividades clasificadas, utilizando la distancia euclidiana y la similitud del coseno. Finalmente, ofreceremos al usuario 10 opciones de puntos de interés los cuales serán bastante cercanas a sus gustos. Esto se hace para que el viajero pueda elegir la actividad que quiera en función de la ruta que vaya a hacer, o construir sus itinerarios con ellas. Con este planteamiento, el turista ampliará el conocimiento de la oferta y no limitará en sus viajes a lo conocido.



**Palabras clave:** Sistema de recomendación, clasificador de actividades, procesamiento de lenguaje natural, medidas de similitud, distancia euclidiana, similitud del coseno.

**Abstract**

Our objective with this project is to work on the task of recommending tourist activities to users, so that these are adapted to their preferences. For this purpose, we developed a solution that implements a recommendation system, which is based on the direct capture of user preferences associated with different categories of activities to be performed in the destination through interactive buttons, an activity classifier to efficiently categorize them according to their degree of belonging to each of them, and similarity measures that select those activities that are closest to the user's preferences.

Data pre-processing involves the extraction of relevant textual information from the activity descriptions, which will come in an HTML format, thus also managing the removal of unnecessary tags and words. A word weighting approach will be used for the classification. The recommendation phase is based on the calculation of similarity measures between the user's preferences and the ranked activities, using Euclidean distance and cosine similarity. Finally, we will offer the user 10 choices of points of interest which will be fairly close to his or her tastes. This is done so that the traveler can choose the activity he/she wants depending on the route he/she is going to do, or build his/her itineraries with them. With this approach, the tourist will broaden his knowledge of the offer and will not limit his trips to what is known.

**Keywords:** Recommender system, activity classifier, natural language processing, similarity measures, Euclidean distance, cosine similarity.

# Índice general

# Índice de figuras

# Índice de tablas

# Capítulo 1 Introducción

Con los avances de las tecnologías han surgido diferentes procedimientos para trabajar con la gran inmensidad de datos que se almacenan actualmente. La inteligencia artificial y la ciencia de datos, son áreas de conocimiento que aportan referentes para el tratamiento de datos en diferentes actividades económicas. Con este Trabajo de Fin de grado proveemos de un ejemplo claro de procedimientos de la IA que aportan valor en el uso de datos disponibles en la generación de nuevo conocimiento.

Para diferentes personas viajar tiene significados, características y motivaciones diversas y variadas. Viajar puede ser una experiencia emocionante y enriquecedora que permite ampliar fronteras, tanto física como mentalmente. Un itinerario de viaje o una ruta turística, es un plan detallado de los lugares que se visitarán durante un viaje. Estos incluyen información sobre los destinos a visitar, las atracciones turísticas, los horarios, los medios de transporte, los alojamientos, etc. Si bien esto lo podemos organizar nosotros mismos como viajeros, lo más normal es que, debido a que lo más común es que las personas visiten lugares nuevos como turistas y tengan una gran falta de conocimiento sobre ese destino, se dejen recomendar por las páginas web.[2]

Cada viajero tiene gustos y preferencias únicos, y un recomendador turístico puede proporcionar sugerencias personalizadas de actividades, atracciones y lugares para hospedarse que se alineen con estos intereses específicos. Esto asegura que los viajeros puedan disfrutar al máximo de su viaje, participando en actividades que realmente les interesen.Además, un recomendador turístico facilita el descubrimiento de destinos nuevos. Aunque muchos viajeros se dirigen a destinos turísticos populares, hay lugares menos conocidos que también ofrecen interesantes experiencias. El recomendador puede guiar a los viajeros hacia estos destinos turísticos menos conocidos, con grandes atractivos, diversificando así la actividad turística, promocionando nuevos lugares y actividades, y extendiendo las oportunidades de turismo.

Los sistemas de recomendación turística optimizan el tiempo y los recursos del viajero. Planificar un viaje puede ser abrumador y llevar mucho tiempo, especialmente para aquellos que no están familiarizados con el destino. El recomendador simplifica este proceso proporcionando itinerarios personalizados, sugerencias de actividades y recomendaciones de restaurantes y alojamientos. Esto ayuda a optimizar el tiempo y los recursos del viajero, asegurando una experiencia de viaje más fluida y satisfactoria.

La finalidad de este trabajo será ofrecer una solución tecnológica que permita abordar este tipo de situaciones y problemas que se encuentran los viajeros y los destinos turísticos. Así, en este trabajo desarrollamos, apoyándonos en herramientas y técnicas de inteligencia artificial, una aplicación que sea capaz de aconsejar a los viajeros teniendo en cuenta sus preferencias, sobre qué puntos de interés debería visitar en su estancia.

## 1.1 Objetivos

El desarrollo de un sistema de recomendación de actividades turísticas está orientado al impulso de la mejora de la experiencia del turista y al aumento de la calidad, prestancia y rentabilidad de los servicios en destinos turísticos. En este contexto, el objetivo general de este trabajo el el siguiente:

Mejorar la experiencia del turista cuando viaja o planifica sus viajes mediante sugerencias o propuestas personalizadas de actividades, atracciones o servicios en el destino, que se ajusten a sus preferencias y necesidades, facilitando la toma de decisiones y la exploración de nuevos y desconocidos espacios y vivencias.

Este objetivo general se concreta en los siguientes objetivos específicos que guiarán el desarrollo de este Trabajo de Fin de Grado:

- Establecer una base sólida para comprender tanto las prácticas actuales como las herramientas tecnológicas disponibles en el ámbito de los sistemas de recomendaciones turísticas.

- Realizar un análisis de los datos existentes y evaluar los diversos modelos disponibles para aconsejar servicios turísticos.

- Desarrollar algoritmos clasificadores y diferentes técnicas que permitan al recomendador sugerir distintos tipos de actividades turísticas sin tener un historial de los usuarios, y obtener resultados de diferentes maneras.

- Validar y mejorar los resultados obtenidos a lo largo del proceso de desarrollo.

- Documentar de manera detallada todo el proceso de desarrollo, así como los resultados y conclusiones del proyecto.

## 1.2 Planificación

Para llevar a cabo un proyecto de estas características, y alcanzar los objetivos propuestos en el apartado anterior, se propone la siguiente planificación de tareas:

1. **Fase de Búsqueda de información**

Al enfrentarnos a un problema nuevo, es crucial comprender en qué punto nos encontramos. Por eso, en este apartado nos hemos sumergido en la investigación del estado del arte de los sistemas de recomendación. Esto nos ha permitido obtener una visión completa de su contexto, propósito y evolución a lo largo del tiempo. También, aprovechando esta investigación se aprovechará para redactar el proyecto de TFG además de empezar con el desarrollo del segundo capítulo de este documento.

2. **Fase de análisis del problema**

Una vez que tenemos claro el contexto de nuestro proyecto, es hora de analizar específicamente nuestro problema, que en este caso se centra en el turismo. Esto implica entender las particularidades y desafíos asociados con la recomendación de actividades y experiencias turísticas. Para ello, examinamos detenidamente qué datos tendremos a nuestra disposición.Con esta comprensión en mente, podemos empezar a plantear nuestra solución. Esto implica definir qué tipo de sistema de recomendación vamos a desarrollar , qué algoritmos y técnicas vamos a utilizar, y cómo vamos a evaluar y mejorar el rendimiento de nuestro sistema.

3. **Fase de clasificación de los datos**

Definitivamente, clasificar y agrupar los puntos de interés (actividades) por tipos es una excelente idea para facilitar la búsqueda y mejorar la eficacia de nuestro sistema de recomendación. Esta clasificación nos permitirá organizar los datos de manera más estructurada y proporcionar recomendaciones más relevantes y específicas a los usuarios. Al agrupar los puntos de interés por tipos, podemos crear categorías o etiquetas que reflejen las diferentes características y temas de las actividades turísticas

4. **Fase de desarrollo del algoritmo recomendador**

Con toda la investigación y la preparación que hemos realizado, estamos bien posicionados para poner en práctica nuestros conocimientos y crear nuestro recomendador turístico. Para comenzar, integraremos nuestro clasificador en el sistema de recomendación. Esto implicará adaptar el clasificador para que pueda procesar datos relevantes para el turismo. Además del clasificador, utilizaremos otras técnicas y algoritmos como pueden ser medidas de similitud, etc.

5. **Fase de experimentación, extracción y redacción de conclusiones**

Para finalizar, hemos experimentado con varias técnicas y métodos, eligiendo así el que nos brinda mejores resultados. Además hemos ampliado el rango de posibilidades, no limitando al recomendador a dar una solución única, ahora el algoritmo nos da un abanico de posibilidades igualmente válidas, con el objetivo de que los usuarios tengan más variedad a la hora de seleccionar el resultado. También se ha realizado una documentación de los programas implementados a la hora de realizar esta memoria, aportando finalmente unas conclusiones y líneas futuras con respecto a cómo avanzar en el proyecto.

## 1.3 Estructura de la memoria

En esta memoria del Trabajo de Fin de Grado se desarrollará en detalle todo lo relacionado con los recomendadores turísticos, incluyendo sus tipos, técnicas y utilidades. Además, se expondrá el proceso de realización del proyecto. La memoria se ha estructurado de la siguiente manera:

- **Capítulo 1 Introducción**: Este primer capítulo servirá como introducción, donde se explicará brevemente el problema a tratar, los objetivos planteados desde el inicio del desarrollo y la forma en que se han abordado.

- **Capítulo 2 Estado del Arte**: En este capítulo se contextualizará al lector sobre el estado actual de los recomendadores, con un enfoque especial en los recomendadores turísticos. También se explicarán técnicas utilizadas, como el procesamiento del lenguaje natural y la clasificación de datos.

- **Capítulo 3 Herramientas Utilizadas**: Aquí se analizarán las diferentes herramientas empleadas en el desarrollo del proyecto, destacando sus ventajas y desventajas.

- **Capítulo 4 Desarrollo del Proyecto**: Este capítulo detallará en profundidad el desarrollo del proyecto, describiendo cada una de las fases que lo componen. Se explicará la evolución del proyecto, los primeros resultados obtenidos y cómo estos se han mejorado para ajustarse mejor a la idea principal y a las posibles adaptaciones futuras del código.

- **Capítulo 5 Conclusiones:** El capítulo final presentará las conclusiones del proyecto, explicando los resultados obtenidos y cómo podría abordarse este problema en el futuro.

# Capítulo 2 Antecedentes y estado del arte

Como se habla en **“A Comprehensive Survey on Travel Recommender Systems”[1]**, viajar implica una combinación de elementos como el trayecto, el transporte, el tiempo de viaje, el alojamiento y otros aspectos que la mayoría de las personas experimentarán en algún momento de sus vidas. Para mejorar esta experiencia, es muy común buscar ayuda a la hora de planificar un viaje. Hoy en día, la cantidad de información disponible en Internet sobre temas relacionados con el turismo es inmensa, y encontrar un paquete, producto o servicio de viaje adecuado puede ser muy laborioso.

Un sistema de recomendación puede ayudar con diversas consultas relacionadas con viajes, como los mejores destinos para las vacaciones de verano, las condiciones climáticas ideales para el senderismo, etc. En este capítulo se investigarán todos los tipos de sistemas de recomendación existentes, las técnicas que utilizan, cómo se adaptan las sugerencias y otros aspectos relevantes. Analizaremos también la evolución de estos sistemas, desde los enfoques basados en contenido y filtrado colaborativo hasta las técnicas más avanzadas.

## 2.1 Sistemas de recomendación

Un sistema de recomendación es un sistema inteligente que actúa como guía y sugiere opciones según las preferencias de la persona. Utiliza técnicas de vanguardia como Big Data e Inteligencia Artificial. Estos sistemas se están volviendo muy populares, ya que sirven como guías para las actividades que una persona o un grupo planean realizar de la mejor manera posible, dadas las restricciones impuestas por los usuarios.

En la industria del turismo, un turista tiene muchas opciones y puede perderse fácilmente. Sin embargo, el recomendador actúa como un guía preciso, sugiriendo comodidades necesarias para el viaje. Todos ellos han estado presentes en el mercado desde la década de 1990 como bien se indica en **“Tourism recommendation system: a survey and future research directions” [2]** y se actualizan constantemente con nuevas técnicas de aprendizaje automático y tecnologías avanzadas como dispositivos móviles inteligentes. Estos sistemas no son motores de búsqueda, sino algoritmos que resuelven problemas de sobrecarga de información. Son beneficiosos tanto para los consumidores como para los comerciantes, ya que mejoran la experiencia del usuario al ayudarlo a encontrar productos según sus gustos y necesidades, y aumentando las ventas del negocio.

Su tarea es cuantificar cuánto le gusta a un usuario un producto y luego predecir la lista de ítems que es más probable que el consumidor compre. Los enfoques utilizados en los recomendadores incluyen filtrado basado en contenido, filtrado colaborativo y filtrado híbrido, que es una combinación de varios.

Proporciona sugerencias valiosas y orientación a los turistas para identificar comodidades como transporte, hoteles, atracciones y lugares de interés especial según sus intereses. Para comprender mejor cómo funcionan estos sistemas y sus aplicaciones, es fundamental explorar los diferentes tipos de enfoques utilizados

### 2.1.1 Filtrado colaborativo

El filtrado colaborativo es uno de los enfoques más utilizados en el comercio electrónico y las redes sociales, entre otros campos. Este tipo de sistema recomienda a los clientes artículos que han sido elegidos por otros consumidores con preferencias similares. Esto se logra correlacionando a los usuarios entre sí en función de similitud de sus selecciones o calificaciones pasadas. Por ejemplo, si dos usuarios han dado calificaciones altas a las mismas actividades, películas o productos, es probable que tengan gustos similares y, por lo tanto, las sugerencias hechas a uno de ellos también podrían ser relevantes para el otro. Esta similitud se puede calcular utilizando diversas técnicas estadísticas y matemáticas, como la correlación de Pearson o la similitud del coseno, que cuantifican cuán alineadas están las preferencias de dos usuarios.

Este método es particularmente efectivo en plataformas con una gran base de usuarios y una rica cantidad de datos sobre sus interacciones y preferencias, lo que permite realizar correlaciones precisas y proporcionar recomendaciones altamente personalizadas. Sin embargo, también puede enfrentar desafíos como el problema del arranque en frío, donde es difícil hacer este tipo de sugerencias para nuevos usuarios o ítems con poca información previa.[3]

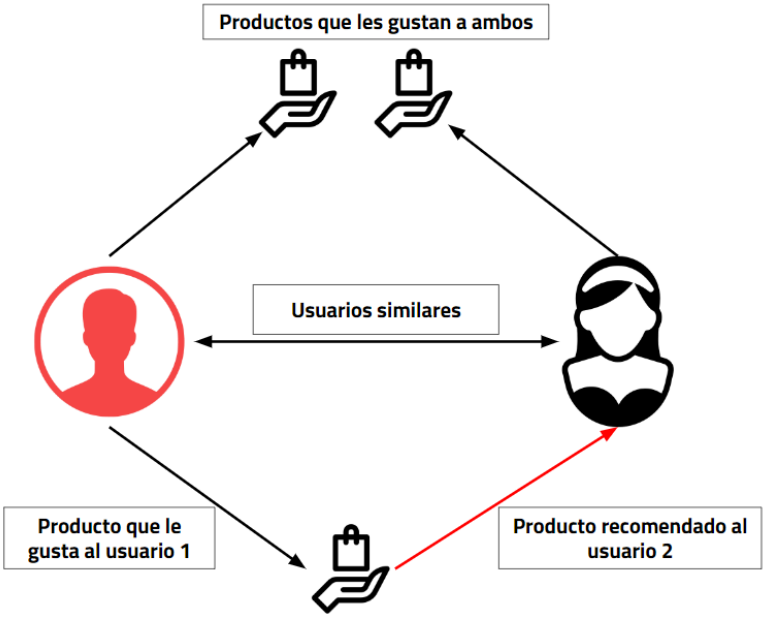


**Imagen 2.1: Funcionamiento de un sistema de filtrado colaborativo**

### 2.1.2 Basado en contenido

El filtrado basado en contenido es un enfoque en los sistemas de recomendación que se centra en los productos con los que el usuarios ha interactuado previamente. Este método utiliza características de artículos que ya sabe que al cliente le gustan.

En particular, el proceso de este sistema implica comparar varios ítems, que pueden ser de interés para el cliente, con los demás que el usuario ha calificado previamente. Los que tienen las características similares a los productos que el consumidor valoró bien en un pasado, son los que se recomiendan.

Por ejemplo, si un usuario ha mostrado preferencia por ciertas actividades turísticas, el sistema buscará otros quehaceres con atributos similares y los sugerirá. Esta similitud se puede determinar utilizando técnicas de procesamiento de texto y minería de datos para analizar descripciones, etiquetas, y otra información que pueda estar asociada con los ítems. Además ofrece ventajas como: independencia de otros usuarios, fáciles de entender y explicar, ya que están basadas en características explícitas de los productos y la adaptación a nuevos servicios, ya que la recomendación se basa en las características y no en la historia de interacciones.

Por otra parte, aunque ofrecen varias ventajas, también presentan algunas desventajas. Una de las principales limitaciones es que estos sistemas pueden llevar a la monotonía al sugerir productos demasiado similares a las preferencias pasadas de un cliente. Esto puede resultar en una falta de diversidad, lo que podría aburrir a los usuarios y reducir su interés en la plataforma.

Además, requieren que los datos de los ítems estén perfectamente detallados y etiquetados. La calidad de las recomendaciones depende en gran medida de lo bien detallada que esté la información disponible sobre cada producto.

Otra desventaja significativa es que no pueden captar los intereses profundos del consumidor. Solo pueden sugerir artículos que comparten características con las preferencias ya identificadas. Esto limita la capacidad del sistema para descubrir y ofrecer servicios nuevos que podrían interesar al usuario pero que no coinciden directamente con sus preferencias anteriores.[3]


**Imagen 2.2: Funcionamiento de un sistema basado en contenido**

### 2.1.3 Basado en conocimiento

El filtrado basado en conocimiento es utilizado, en los sistemas de recomendación, para generar sugerencias, al razonar sobre qué elementos cumplen con los requisitos del usuario. Este método no se basa en las interacciones anteriores del usuario con los ítems, sino en un entendimiento profundo de las necesidades y preferencias del usuario en ese mismo momento.

Por ejemplo, al recomendar un servicio turístico, el sistema considerará si el cliente valora el deporte o la comodidad. Para construir este conocimiento, el sistema puede registrar las preferencias y elecciones del usuario o pedir directamente que proporcionen información sobre la relevancia de las opciones disponibles.

La medida de similitud en estos sistemas estima hasta qué punto las necesidades del se correlacionan con las opciones disponibles. Esta función se utiliza para enseñar la utilidad de cada recomendación, mostrando a los consumidores qué tan bien se ajusta cada opción a sus preferencias.

Es especialmente útil en situaciones donde las preferencias del usuario son específicas, y donde las recomendaciones deben adaptarse a criterios muy concretos. Además, este método puede combinarse con otros para mejorar la precisión y la relevancia de las propuestas.[3]

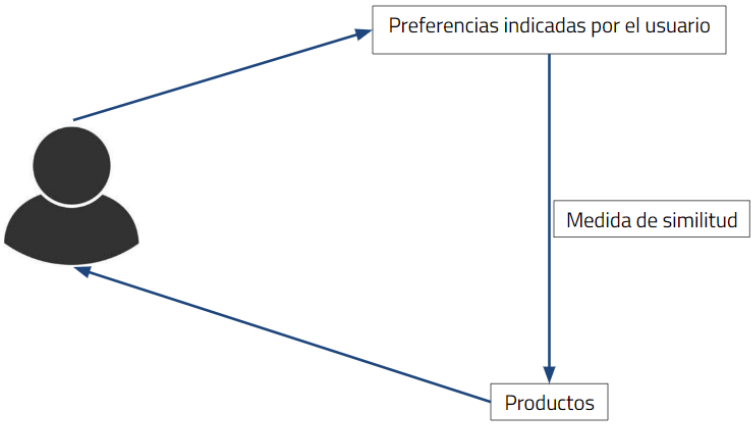


**Imagen 2.3: Funcionamiento de un sistema basado en conocimiento**

### 2.1.4 Sistemas híbridos

Un sistema de recomendación híbrido combina múltiples técnicas para ofrecer sugerencias más precisas y relevantes. Estos son utilizados en una variedad de dominios, desde el comercio electrónico hasta las plataformas de streaming y servicios turísticos. Los enfoques tradicionales, incluyen los tres modelos nombrados anteriormente. Sin embargo, cada uno de estos enfoques tiene sus propias limitaciones.

Para superar las deficiencias de los sistemas individuales y aprovechar sus fortalezas, los híbridos, integran varios algoritmos de manera que se complementen entre sí. Esto permite mejorar mucho la precisión. Existen tres enfoques principales para integrar estas técnicas:

La Selección del Método es el enfoque en el que se incorporan métodos de filtrado demográfico (técnicas que se basan en se basa en las características demográficas de los usuarios, como la edad, el género y otros factores sociodemográficos), basado en contenido y filtrado colaborativo , pero solo aplica uno de ellos según la situación particular de cada usuario. Por ejemplo, cuando un usuario accede por primera vez, se utiliza un método basado en datos demográficos. Más adelante, si se encuentran usuarios similares, se realiza una recomendación colaborativa. En caso contrario, se aplica un procedimiento basado en contenido.

También existen los de uso Secuencial, que aplican diferentes técnicas de recomendación en distintas etapas del proceso. Por ejemplo, un sistema puede usar información contextual para una selección inicial, seguido de técnicas basadas en conocimiento y filtrado colaborativo para refinar las opciones.

**Por último tenemos el uso Integrado**, que acopla múltiples procedimientos simultáneamente durante la generación de las recomendaciones. Esto puede implicar la combinación de puntuaciones obtenidas mediante diferentes métodos para producir una calificación general y seleccionar los mejores ítems.

Los sistemas de recomendación híbridos han demostrado ser efectivos en diversos contextos, proporcionando sugerencias más robustas que los métodos mencionados en los primeros apartados. Su capacidad para integrar técnicas avanzadas permite superar desafíos como el problema del arranque en frío y la diversidad de preferencias de los usuarios.[4]

## 2.2 Procesamiento del lenguaje natural

Tal y como dice James Allen en **“*Natural language understanding*”,** el lenguaje es uno de los aspectos fundamentales del comportamiento humano y es un componente crucial de nuestras vidas. En forma escrita, sirve como un registro a largo plazo del conocimiento de una generación a otra. En forma hablada, sirve como nuestro principal medio para coordinar nuestro comportamiento diario con los demás.

El lenguaje se estudia en varias disciplinas. Cada una define su conjunto de problemas y tiene sus propios métodos para abordarlos. El lingüista, por ejemplo, estudia la estructura del lenguaje, considerando preguntas como por qué ciertas combinaciones de palabras forman oraciones pero otras no, y por qué una oración puede tener algunos significados. El objetivo del lingüista computacional es desarrollar una teoría del lenguaje, utilizando las nociones de algoritmos y estructuras de datos. Por supuesto, para construir un modelo computacional, debes aprovechar lo que se conoce de todas las otras disciplinas.

En el campo del procesamiento del lenguaje natural , se han desarrollado una variedad de enfoques y técnicas para abordar diferentes aspectos del análisis del lenguaje. Uno de estos enfoques es el modelado estadístico del lenguaje, que captura patrones en el uso del lenguaje y predice la probabilidad de secuencias de palabras. Este enfoque permite que estos sistemas aprendan de grandes cantidades de datos y generen resultados más precisos y útiles.[5]

El análisis sintáctico del lenguaje, extrae el significado de los textos. Al comprender la estructura y el significado del habla humana, se pueden realizar tareas como la traducción automática, el resumen de texto y la generación de respuestas en conversaciones..

La comprensión del lenguaje es fundamental para el desarrollo de sistemas que puedan interactuar con los humanos. Esto incluye la creación de asistentes virtuales, chatbots y sistemas de traducción automática que sean capaces de comprender y generar lenguaje humano de manera efectiva. Cada vez son más importantes en una variedad de aplicaciones.

Además tiene aplicaciones en áreas como el análisis de sentimientos, donde se analizan grandes cantidades de texto para identificar emociones y actitudes, y otras muchas más como las mencionadas con anterioridad. Estas aplicaciones se pueden utilizar de muchas maneras, desde la investigación de mercado hasta la vigilancia de redes sociales. Es por ello que las empresas pueden utilizar el análisis de sentimientos para evaluar la percepción del público sobre sus productos o servicios, mientras que los investigadores pueden utilizar la extracción de información para recopilar datos relevantes de fuentes diversas.[6]

## 2.3 Clasificación automática

La idea de clasificación es muy importante en muchos ámbitos del tratamiento de datos. En su conjunto son técnicas que sirven para clasificar todo tipo de servicios. En nuestro caso concreto lo utilizamos para organizar puntos de interés en diferentes clases. Esto lo realizamos mediante la extracción de información a partir de las descripciones de las actividades.

En general, se distingue entre dos tipos de métodos de clasificación principales: la clasificación supervisada y la no supervisada. Estos dos enfoques requieren soluciones diferentes y tienen aplicaciones diversas a la hora de organizar y gestionar la información.

La clasificación supervisada se basa en categorías prediseñadas. En este enfoque, el clasificador debe asignar cada producto a una categoría ya establecida. Este proceso no solo requiere la creación manual de una serie de clases, sino también un entrenamiento previo del programa. Este desarrollo de aprendizaje se realiza utilizando una colección, compuesta por datos ya clasificados manualmente, que sirve para construir patrones de cada categoría y utilizar funciones de similitud para estimar el parecido. La actividad se asignaría al rango cuyo patrón es más similar. Entre los algoritmos utilizados, encontramos el algoritmo probabilístico naive Bayes, el algoritmo de Rocchio, el algoritmo del vecino más próximo (k-nearest neighbors, KNN) y las redes neuronales.

Por otro lado, la clasificación no supervisada, o clustering, no se basa en categorías predefinidas. La información se agrupa según sus propias características, permitiendo que se organicen a su manera. Este método es automático y no requiere intervención manual, lo que lo hace ideal para descubrir estructuras en conjuntos de datos grandes.

Para implementar cualquier metodología de clasificación, es esencial contar con una representación de los puntos de interés. El modelo vectorial es uno de los formalismos más utilizados para este propósito, debido a que cada archivo se representa como un vector de palabras, donde cada una tiene un peso que indica su importancia dentro del documento. Estos pesos se calculan basándose en la frecuencia de los términos tanto en el fichero como en el conjunto de todos ellos.

Independientemente del algoritmo de clasificación utilizado, es importante medir la similitud entre datos. Al representar cada uno como un vector, se pueden aplicar varias funciones de similitud, como el coseno, los coeficientes de Dice y Jaccard, etc. Estas funciones permiten comparar el parecido entre ellos.

La clasificación automática, ha demostrado ser una herramienta eficaz, comparable a la realizada por humanos. Con el avance continuo en técnicas y algoritmos, este campo sigue evolucionando, proporcionando soluciones cada vez más robustas para la gestión de información.[7]

## 2.4 Medidas de similitud

Las medidas de similitud son una pieza fundamental en muchas disciplinas, como son la estadística, el procesamiento del lenguaje natural y otras relacionadas con la ciencia de datos. Permiten cuantificar la semejanza entre diferentes objetos, tales como palabras, documentos, imágenes o usuarios. La capacidad de evaluar lo parecidos que pueden llegar a ser dos o más elementos tiene muchas aplicaciones, que incluyen la recuperación de información, la recomendación de productos, la agrupación de datos, la clasificación de textos, etc.

Ayudan a comparar entidades lingüísticas. Por ejemplo, en los sistemas de recomendación, pueden utilizarse para identificar elementos (Actividades o puntos de interés) que son similares a los que ya han sido preferidos por un usuario, mejorando así la precisión de las sugerencias.

Existen diversas técnicas y métodos para calcular el parecido entre elementos, cada una con sus propias ventajas y limitaciones. La elección de la medida de similitud adecuada depende del tipo de información que se maneje y del contexto específico de la aplicación. A continuación, se exploran algunas de las más comunes y su aplicación en el procesamiento del lenguaje natural y otros campos relacionados.

### 2.4.1 La medida distancia coseno

Es una medida ampliamente utilizada que evalúa la semejanza entre dos vectores calculando el coseno del ángulo entre ellos. Esta medida es especialmente útil en espacios de alta dimensionalidad. También tiene la ventaja de ser invariante a la magnitud de los vectores, lo que significa que se enfoca únicamente en la orientación relativa de los vectores en el espacio. Los resultados que da esta función están entre el rango de -1 (los resultados más alejados) y 1 (los resultados más cercanos). Sin embargo, en la mayoría de sistemas de recomendación y procesamiento del lenguaje natural, los valores típicos oscilan entre 0 y 1, ya que los datos generalmente no incluyen direcciones completamente opuestas (valores negativos).

$$\text{soft\_cosine}_1(a, b) = \frac{\sum_{i,j}^{N} s_{ij} a_i b_j}{\sqrt{\sum_{i,j}^{N} s_{ij} a_i a_j} \sqrt{\sum_{i,j}^{N} s_{ij} b_i b_j}}$$

**Imagen 2.4: Fórmula de la medida de similitud del coseno**

## 2.4.2. La medida distancia euclidiana

Esta medida calcula la longitud del camino más corto entre dos puntos en un espacio n-dimensional. Aunque es fácil de calcular, la distancia euclidiana puede ser menos efectiva en espacios de alta dimensionalidad. Sin embargo, sigue siendo una herramienta útil en muchas aplicaciones donde la interpretación geométrica de los datos es importante.

$$\sqrt{\sum_{i=1}^{p}(Y_{i1}-Y_{i2})^2}$$

**Imagen 2.5: Fórmula de la distancia euclidiana**

## 2.4.3. La medida de distancia de Jaccard

Esta medida es utilizada para comparar la semejanza de conjuntos de datos. Se define como el tamaño de la intersección dividido por la unión de los conjuntos. Es particularmente útil en aplicaciones como la comparación de documentos mediante conjuntos de palabras clave, donde la representación de la información es natural.

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

**Imagen 2.6: Fórmula de la distancia de Jaccard**

# Capítulo 3 Herramientas

Para el desarrollo de este proyecto, se ha utilizado una serie de herramientas que nos han permitido crear y perfeccionar los sistemas y técnicas mencionados en el apartado anterior. En este capítulo se nombran algunas de ellas y proporcionaremos una visión más detallada de las mismas

## 3.1 Python


**Imagen 3.1: Logo Python**

Python es un lenguaje de programación de alto nivel conocido por su simplicidad, versatilidad y legibilidad. Con una sintaxis clara y concisa, se ha convertido en uno de los utensilios preferidos tanto por principiantes como por expertos en programación. Además es utilizado en una amplia variedad de aplicaciones, desde desarrollo web y análisis de datos hasta inteligencia artificial y aprendizaje automático.

En el contexto de la programación de sistemas de recomendación, Python ofrece una serie de ventajas que lo convierten en una opción popular.. Ahora, veamos por qué Python es una elección común para este propósito:

- **Facilidad de Implementación y Experimentación**: Es reconocido por su sintaxis simple y su sencillez, lo que hace muy sencilla su implementación de algoritmos de recomendación. Esta característica es esencial cuando se trabaja en proyectos de investigación y desarrollo iterativos, donde se requiere experimentar con diferentes enfoques para optimizar la relevancia de las recomendaciones.

- **Abundancia de Bibliotecas Especializadas**: Cuenta con una amplia gama de bibliotecas especializadas en análisis de datos y aprendizaje automático, como Pandas, NumPy y scikit-learn. Estas bibliotecas ofrecen algoritmos y herramientas preconstruidas que simplifican significativamente el proceso de desarrollo de este tipo de proyectos, desde la manipulación de conjuntos de datos hasta la implementación de modelos de aprendizaje automático.
- **Flexibilidad para Personalización**: Los sistemas de recomendación suelen requerir adaptaciones y personalizaciones según las necesidades específicas del dominio y los usuarios. Como un lenguaje flexible y dinámico, permite a los desarrolladores implementar fácilmente algoritmos personalizados y ajustar los modelos según las peculiaridades del problema de recomendación en cuestión.
- **Comunidad Activa y Recursos Abundantes**: Python cuenta con una comunidad de desarrolladores activa y vibrante que contribuye constantemente con nuevos recursos, bibliotecas y conocimientos relacionados con el desarrollo de cualquier tipo de sistema. Esto proporciona acceso a una amplia gama de tutoriales, ejemplos de código y casos de estudio que pueden servir como punto de partida para trabajos como este.

## 3.2 Google Colab


**Imagen 3.2: Logo Google Colab**

Google Colab, abreviatura de Google Colaboratory, es una plataforma en línea gratuita que permite escribir y ejecutar código Python en el navegador, sin necesidad de configurar un entorno de desarrollo local. Esta herramienta, está basada en el popular entorno de código abierto Jupyter Notebook y está integrada con Google Drive, lo que facilita el acceso y la colaboración en proyectos de programación. Nos ofrece una serie de ventajas que lo hacen especialmente atractivo:

- **Acceso a Recursos de Computación en la Nube**: Proporciona acceso gratuito a recursos de computación en la nube, incluidas GPU y TPU (Unidad de Procesamiento Tensorial), lo que permite ejecutar y entrenar modelos de aprendizaje automático de manera eficiente, especialmente en conjuntos de datos grandes.
- **Librerías Preinstaladas y Facilidad de Uso**: Google Colab viene preinstalado con muchas de las bibliotecas más populares, como las nombradas en el apartado anterior. Esto facilita la configuración del entorno de desarrollo y la escritura de código sin necesidad de instalar nada adicionalmente.
- **Entorno de Desarrollo Colaborativo**: Al estar integrado con Google Drive, se hace muy fácil compartir el programa a la hora de correcciones o de explicaciones sobre los avances pudiendo cualquiera ejecutar el código a su ritmo.

## 3.3 Nltk


**Imagen 3.3: Logo NLTK**

NLTK (Natural Language Toolkit) es una biblioteca de Python ampliamente utilizada para el procesamiento del lenguaje natural (PLN). Provee herramientas y recursos para trabajar con texto y lenguaje humano de manera efectiva, incluyendo *tokenización*, etiquetado de partes del discurso, análisis sintáctico, lematización y mucho más. NLTK es una librería invaluable para tareas como análisis de sentimientos, extracción de información, clasificación de textos y generación de lenguaje natural. Cuando se trata de desarrollar sistemas de recomendación basados en texto, nos ofrece una serie de ventajas significativas:

- **Preprocesamiento de Texto Avanzado**: Proporciona una amplia gama de instrumentos para el preprocesamiento de texto, lo que permite limpiar y normalizar datos de texto de manera efectiva. Esto incluye técnicas como tokenización, eliminación de stopwords, lematización y stemming, que son fundamentales para mejorar la calidad de las recomendaciones basadas en texto.

- **Extracción de Características Textuales**: Ofrece herramientas para extraer características textuales relevantes de los documentos, como palabras clave, n-gramas y características lingüísticas específicas. Estas características pueden utilizarse como entrada para modelos de recomendación basados en texto, ayudando a identificar patrones y relaciones significativas en los datos.
- **Clasificación de Texto**: NLTK también suministra algoritmos y técnicas para la clasificación de texto, que pueden ser útiles en sistemas de recomendación para categorizar y organizar documentos en función de su contenido. Esto permite construir sistemas de recomendación más sofisticados que pueden adaptarse a las preferencias y necesidades específicas de los usuarios.

## 3.4 BeautifulSoup


**Imagen 3.4: Logo de beautifulSoup**

En el mundo de la web, existe muchísima información. Sin embargo, acceder a esta información de manera estructurada y procesable puede ser un desafío. Aquí es donde entra en juego el concepto de web scraping, una técnica utilizada para recoger datos de páginas de forma automatizada. En este contexto, BeautifulSoup emerge como una herramienta esencial en el arsenal de un desarrollador. Se trata de una biblioteca de Python que simplifica el proceso de análisis y extracción de datos de documentos HTML y XML. Al proporcionar una interfaz fácil de usar para navegar por la estructura de un documento web, BeautifulSoup permite a los desarrolladores recolectar información precisa y relevante de manera eficiente.

Uno de los aspectos fundamentales de BeautifulSoup es su capacidad para interpretar y analizar documentos o ficheros en formato HTML. Al inicializar un objeto BeautifulSoup, se debe especificar un analizador HTML que se encargará de desglosar el código en una estructura comprensible para Python. En nuestro caso, hemos empleado el analizador *"html.parser"*, una opción integrada en este lenguaje que, si bien puede no ser la más rápida en términos de rendimiento, ofrece una solución conveniente y sin necesidad de instalaciones adicionales para la mayoría de los casos de uso. [9]

Otra función esencial de BeautifulSoup es *"find_all()"*, la cual facilita la búsqueda de elementos específicos dentro de un documento. Esta función permite buscar todas las instancias de una etiqueta, clase, atributo o patrón específico dentro del código. En nuestro caso concreto, como se podrá observar en el próximo capìtulo, se ha usado para encontrar la información que se encuentra entre párrafos, definidos con la etiqueta <p>.[10]

## 3.5 OpenPyXL


**Imagen 3.5: Logo de OpenPyXL**

En el ámbito de la manipulación de datos en Python, trabajar con archivos de Excel es una necesidad común en muchas aplicaciones empresariales y de análisis de datos. Sin embargo, procesar estos archivos puede resultar tedioso debido a la complejidad de su formato binario. Es aquí donde OpenPyXL entra en juego como una biblioteca de Python que permite la creación, manipulación y lectura de archivos Excel en formato xlsx. Al proporcionar una interfaz intuitiva para trabajar con hojas de cálculo, simplifica en gran medida las tareas relacionadas con la automatización de procesos de datos.

La librería facilita la manipulación de hojas de cálculo de manera eficiente, permitiendo a los desarrolladores crear nuevas, modificar las ya existentes, agregar o eliminar información y hacer una diferentes operaciones comunes realizadas en entornos de Excel. Esto brinda un control completo sobre el contenido y la estructura de los archivos, permitiendo personalizar flujos de trabajo complejos.

Una ventaja clave es su compatibilidad nativa con el formato de archivo xlsx, utilizado por defecto en las versiones más recientes de Excel. Esto garantiza una perfecta integración con el ecosistema de Microsoft Office y la capacidad de trabajar con ficheros sin problemas en diferentes plataformas y entornos de trabajo.[11]

# Capítulo 4 Desarrollo del proyecto

## 4.1 Problemática

Cuando una persona decide visitar un nuevo país o explorar lugares desconocidos, la falta de conocimiento sobre el destino puede generar incertidumbre. Muchos viajeros optan por descubrir por su cuenta y disfrutar sin un rumbo fijo, pero existe un gran número de personas que prefieren organizar un itinerario detallado. Estos turistas, al no conocer bien el lugar, requieren de recomendaciones que les ayuden a maximizar su experiencia.

Un desafío común es que las sugerencias pueden provenir de amigos o lugareños cuyas preferencias no coinciden con las del turista. Esto puede llevar a que la experiencia no resulte del todo satisfactoria. Por ejemplo, un viajero interesado en la historia y la cultura puede no encontrar útiles las recomendaciones de alguien que prefiere actividades al aire libre y deportes extremos. Es por ello que la solución a este problema es dejarse guiar por aquellos algoritmos que mediante técnicas probabilísticas calculas que destinos se corresponden mejor a nuestros gustos

### 4.1.1 Arranque en frío

El "arranque en frío" es un gran desafío en los sistemas de recomendación, sobre todo si hablamos del filtrado colaborativo. Este problema ocurre cuando hay una falta de datos suficientes para realizar recomendaciones precisas, lo que puede suceder en tres escenarios principales:

Cuando un usuario recién se une a la plataforma y no ha interactuado aún con muchos ítems, el sistema no tiene ningún tipo de información sobre sus preferencias y es difícil generar recomendaciones relevantes.

En el caso de que se añadan nuevos productos a la plataforma, no hay suficientes calificaciones de los usuarios, lo que dificulta recomendarlos, ya que no se sabrían las categorías a las que pertenece el producto, los usuarios a los que les ha gustado, etc .

Al implementar un nuevo sistema en una plataforma, inicialmente puede haber una falta de datos históricos suficientes sobre las interacciones entre clientes y productos.

En nuestro caso concreto, se cumplen dos de esos tres inconvenientes: Sistema nuevo y falta de usuarios. Es por ello, que el filtrado colaborativo se convierte en una posibilidad inviable, al igual que el basado en contenido, ya que ambos necesitan de información previa de los usuarios.

### 4.1.2 Solución planteada

La solución propuesta consiste en instanciar una serie de botones configurables a través de los cuales los usuarios puedan, al ingresar a la página web o aplicación, especificar sus preferencias. Esto permitirá recopilar información de primera mano por el consumidor.

Como resultado, y de acuerdo con las explicaciones proporcionadas en el **capítulo 2**, el sistema que mejor se adapta a esta situación es el basado en conocimiento. Este utiliza la información detallada proporcionada por los usuarios para ofrecer sugerencias que se alinean con los gustos y necesidades individuales de cada cliente.

Este modelo es particularmente adecuado en este caso, porque permite una personalización precisa. Al contar con datos específicos sobre sus propios gustos, el sistema puede ofrecer recomendaciones más útiles.

Además, este método facilita la adaptación del contenido nuevo y cambios en las preferencias de los usuarios, lo que puede contribuir a mantener el interés y la participación activa en la plataforma.

## 4.2 Datos de entrada

Para afrontar el problema descrito en el apartado anterior, se nos ha proporcionado una serie de datos. Este conjunto cuenta con un total de 265 filas. La información de cada una de las actividades se presenta en forma de varias columnas que contienen distintos tipos de datos tales:

- **Código**: Identificador de una actividad, suele ser una abreviatura que representa dicha actividad de manera reconocible.
- **Nombre**: Nombre completo de la actividad.
- **Tipo_Servicio**: Los tipos se diferencian principalmente entre productos iniciales que se ofrecen al cliente para comenzar su experiencia(entradas), o servicios más profundos y duraderos que proporcionan un valor significativo al cliente(cursos).

- **Dirección**: Información que indica la ubicación específica de la actividad.
- **Codigo_Postal**: Código postal de la ciudad en la que se encuentra la actividad.
- **Latitud_Longitud**: Coordenadas geográficas en las que se encuentra el servicio.
- **Aeropuerto_Próximo1**: Aeropuerto más cercano a dicho producto turístico.
- **Aeropuerto_Próximo2**: Segundo Aeropuerto más cercano a dicho producto turístico.
- **Zona_Turística**: Ciudad o conjunto de ciudades, que conforman una región turística, a la que pertenece esta actividad.
- **Título_Meta**: Información compuesta por el Nombre del producto, La ciudad en la que se encuentra el mismo y el nombre de la empresa que lo oferta.
- **Descripción**: Información detallada sobre el servicio en formato HTML, en la que se introduce un breve resumen sobre lo que trata la actividad.
- **Empresa**: Nombre de la compañía que ofrece el producto.
- **Página_Web**: Enlace URL de la página web de la empresa.
- **Dirección_Empresa**: Información que indica la ubicación específica de la compañía.
- **Código_Postal.1**: Código postal de la ciudad en la que se encuentra la entidad.

En la siguiente imagen tienen un ejemplo gráfico del orden de los datos mencionados :

| CODIGO | NOMBRE | IPO_SERVICI | DIRECCION | DIGO_POST | LATITUD_LONGITUD | EROPUERTO_PROXIM | UERTO_PRO | A_TURIS | ITULO_MET | DESCRIPCION | EMPRESA | AGINA_WE | CCION_EMP | DIGO_POSTA | LOCALIDAD |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LP-ACTTCI | Loro Parque | Entrada | Avenida Lor | 38400 | 28.40914849656296, -1 | TFN | TFS | Puerto de la | Loro Parque | <h3 class="encabezado-editor-rico">El mejor zool&<br> Loro Parque es la mejor experiencia para conoce<br>  <br />En la familia Loro Parque contamos<br>  <br />Un lugar lleno de emoción | Loro Parque | www.loropa | Avenida Lor | 38400 | Puerto de La |
| FOR | Forestal Park | Entrada | Ctra. TF-24, | 38290 | 28.414683325623084, -1 | TFN | TFS | Las Lagunet | Forestal Par | Diver<br> Forestal Park es el parque temático de a<br> Un Parque de Aventura para disfrutar de la natu<br>   <br> Ofrecido por:<br /></str<br> T. 630 385 742 | Forestal Par | www.forest | Carretera TF | 38290 | La Esperanz |
| ESC | Experiencia Volcánica | Entrada | | | 28.344607456426075, -1 | TFN | TFS | Icod de los V | Experiencia | Volcánica Isla Baja \| Tenerife, Icod de los Vinos \| Can | El Cardón Vi | www.elcard | Calle la finca | 38480 | Buenavista |
| DEL | Delicatesen Tenerife: | Entrada | | | 28.292571077859147, -1 | TFN | TFS | Cañadas del | Delicatesen | Disfruta de una agradable noche de verano y des<br>   <br> Ofrec<br> El Cardón <br> Calle la finca, s/n <br> Buenavista del Norte <br> T. 922 127 938 | El Cardón Vi | www.elcard | Calle la finca | 38480 | Buenavista |
| KM | Kayak de mar | Entrada | Puerto Depo | 38683 | 28.247208568378355, -1 | TFN | TFS | Los Gigante | Kayak de ma | <h3 class="encabezado-editor-rico">Navega en kaya<br> Es la mejor forma de admirar estas impresionant<br> Además podrás disfrutar de un b | El Cardón Vi | www.elcard | Calle la finca | 38480 | Buenavista |
| RU | Secretos de Punta de | Entrada | Carretera G | 38400 | 28.402969873952788, -1 | TFN | TFS | Garachico | Ruta de las | <h3 class="encabezado-editor-rico">Secretos de Pur<br> Ven cada domingo al secreto mejor guardado de<br>   <br> La ruta comienza en un lugar lleno de encanto: E<br>   <br> Ofrecido por:<br> El Cardón Natu<br> T. 9 | El Cardón Vi | www.elcard | Calle la finca | 38480 | Buenavista |
| LP-ACTTCI | Siam Park | Entrada | Avenida Sia | 38670 | 28.07198980112754, -1 | TFN | TFS | Costa Adeje | Siam Park \| | <h3 class="encabezado-editor-rico">El mejor parque<br> Un lugar lleno de ambientes diferentes en el que<br>  <br />Siam Park no es sólo un pa<br>   <br> Los pequeños podrán disfrutar de<br>  <br />Para los amantes de la adrenalina,<br> <br> | Loro Parque | www.siamp | Avenida Lor | 38400 | Puerto de La |
| | | | | | | | | | | <h3 style="text-align: justify;" class="encabezado-ed<br> Despertar los sentidos desayunando entre | | | | | |

**Imagen 4.1: Conjunto de datos iniciales**

## 4.3 Clasificador

Para el desarrollo del clasificador, tenemos un inconveniente, no hay muchas actividades de las cuales extraer información. Es por ello que para poder realizar pruebas simulando actividades nuevas, solo se ha recogido información de las primeras 100 actividades. El clasificador es una parte muy importante de nuestro sistema, sin embargo no es el propio sistema en su totalidad.

Esta estrategia nos permitirá recopilar información directamente de los usuarios, facilitando las recomendaciones. Sin embargo, para que el recomendador funcione de manera eficiente, es necesario realizar un trabajo previo de clasificación.

Por esta razón, además de desarrollar el recomendador, nos enfocaremos en crear un clasificador de actividades. Este, analiza y categorizará las diferentes actividades turísticas en función de características específicas, permitiendo que las recomendaciones sean más precisas y relevantes para cada usuario.

### 4.3.1 Preprocesamiento

Para comenzar con el desarrollo del clasificador, primero debemos examinar detenidamente los datos mencionados anteriormente. Es importante seleccionar las columnas que proporcionan información valiosa sobre las actividades turísticas.

Finalmente, tras eliminar los campos que carecen de información útil, hemos decidido centrarnos exclusivamente en los apartados "Título_Meta" y "Descripción". Estas dos columnas contienen los datos más significativos para nuestro propósito de clasificación, ya que proporcionan información esencial sobre el contenido y la naturaleza de cada actividad.

Una vez que hemos identificado las columnas "Título_Meta" y "Descripción" como las más relevantes, es momento de iniciar el proceso de preprocesamiento de los datos. Este paso es fundamental para prepararlos de manera que puedan ser utilizados eficazmente por el clasificador de actividades. Para llevar a cabo esta tarea, empezaremos eliminando todas las etiquetas HTML presentes en la descripción. Nos centraremos exclusivamente en conservar el contenido que se encuentra dentro de las etiquetas de párrafos. Estas etiquetas son  para abrir un párrafo y  para cerrarlo.

```
for row in sheet.iter_rows(min_row=2, values_only=True):
    info = row[9].split(' | ')
    desc = row[10]
    full_string = 'La actividad es ' + info[0]
    full_string += ', situada en ' + info[1]
    full_string += ', proporcionada por ' + info[2]
    full_string += '. La descripción es: ' + get_description(desc)
    data.append(full_string)
```

**Imagen 4.2: Recopilación de datos**

```
def get_description(inf):
  if inf is None:
    return ""
  bs = BeautifulSoup(inf, 'html.parser')
  aux = bs.find_all('p')
  paragraphs = ''
  for x in aux:
    paragraphs += str(x)
  descArray = re.split('<p>|</p>|<br/>|\\xa0', paragraphs)
  desc = ''
  for x in descArray:
    desc  += x
  return desc
```

**Imagen 4.3: Función de extracción de información HTML**

En el siguiente paso, emplearemos la librería NLTK para eliminar cualquier etiqueta que pueda haber quedado, así como también eliminaremos cualquier palabra que contenga símbolos o números, y finalmente eliminaremos las *stopwords*. Estas *stopwords* son palabras comunes que no aportan un significado informativo al contenido. Se utilizan principalmente para conectar otros términos o estructurar oraciones.

```
def processActivities(data):
  dictionary = word_tokenize(str(data))
  dictionary = [word for word in dictionary if word.isalpha()]
  dictionary = [word.lower() for word in dictionary]
  stopWords = set(stopwords.words('spanish'))
  dictionary = [word for word in dictionary if word not in stopWords]
  return dictionary
```

**Imagen 4.4: Función de preprocesamiento con Nltk**

### 4.3.2 Vocabularios

Una vez que se han separado y procesado las palabras clave, el siguiente paso será la creación de un archivo llamado "*vocabulario.txt*", donde se almacenarán todas las palabras importantes. Este archivo servirá como referencia para la clasificación, cuando hayamos avanzado más. Posteriormente, se ha elaborado de forma manual, una clasificación de actividades con el propósito de entrenar el algoritmo. Las categorías seleccionadas incluyen:

- **Bienestar**: Esta categoría abarca actividades relacionadas con el cuidado personal, la relajación y el bienestar mental y físico. Incluye opciones como spa, yoga, meditación y terapias alternativas.
- **Cultural y Gastronómico**: Aquí se agrupan actividades que tienen un componente cultural o gastronómico significativo. Esto puede incluir visitas a museos, galerías de arte, así como experiencias gastronómicas como catas de vino y degustaciones de platos locales.
- **Deportes**: Esta categoría comprende actividades relacionadas con el ejercicio físico y la práctica de deportes como por ejemplo senderismo, ciclismo o surf.
- **Ocio**: Aquí se incluyen actividades de ocio y entretenimiento que no se clasifican específicamente en las otras categorías. Esto puede abarcar desde salidas al cine o teatro, hasta festivales de música o parques de atracciones.
- **Tours**: Esta categoría engloba actividades guiadas o recorridos turísticos que permiten explorar y conocer lugares de interés. Puede incluir tours históricos, excursiones a la naturaleza, visitas a monumentos o recorridos temáticos por la ciudad.

Se procede a leer la clasificación realizada. A continuación, se recorren todas las actividades de nuevo, preprocesándolas como se explicó anteriormente, para extraer las palabras clave. En este paso, se llevan a cabo una serie de acciones específicas: Se identifican y cuentan las palabras clave que pertenecen a cada categoría, se contabiliza cuántas veces se repite cada término dentro de su clase correspondiente y se registra cuántas actividades hay en total dentro de cada categoría.

```
#Incrementamos el numero de actividades de ocio que hay
AllCategories[selectedCat].increment()
#recorremos las palabras claves de la descripcion de la actividad
for j in range(len(activity)):
  AllCategories[selectedCat].add_words(activity[j])
```

**Imagen 4.5: Incremento de palabras por categoría.**

```
def add_words(self, word):
  if word in self.words:
    self.words[word] += 1
  else:
    self.words[word] = 1
```

**Imagen 4.6: Método de adición de palabras a una clase.**

Con todos estos datos recopilados, podemos calcular el peso de cada palabra para cada categoría. Esto se realiza utilizando la siguiente fórmula:

$$log\left(\frac{X+1}{\sum diccionario + \sum Categoría}\right)$$

**Imagen 4.7: Fórmula para calcular los pesos de las palabras.**

Donde $X$ es el número de veces que se repite un término en esa categoría, $\Sigma_{diccionario}$ es el número total de palabras que hay en el vocabulario y $\Sigma_{Categoría}$ es el número total de veces que se repiten todas en esa categoría. Como resultado, obtenemos un fichero para cada clase que contiene los pesos de cada palabra específica para esa categoría en concreto.

```
Palabras totales: 4899
abajo
abalada
abandona
abandonaban
abarca
abarcar
abba
abbey
abdominales
abeja
abejas
```

**Imagen 4.8: Resultado del fichero *vocabulario.txt.***

```
Número de documentos (actividades) del corpus: 51
Número de palabras del corpus: 1080
Palabra: abajo Frec: 0 LogProb: -8.696008608880904
Palabra: abalada Frec: 0 LogProb: -8.696008608880904
Palabra: abandona Frec: 0 LogProb: -8.696008608880904
Palabra: abandonaban Frec: 0 LogProb: -8.696008608880904
Palabra: abarca Frec: 0 LogProb: -8.696008608880904
Palabra: abarcar Frec: 0 LogProb: -8.696008608880904
Palabra: abba Frec: 1 LogProb: -8.002861428320958
Palabra: abbey Frec: 0 LogProb: -8.696008608880904
Palabra: abdominales Frec: 0 LogProb: -8.696008608880904
Palabra: abeja Frec: 2 LogProb: -7.597396320212795
```

**Imagen 4.9: Resultado del fichero de la categoría Ocio.**

### 4.3.3 Clasificación

Se deben gestionar estas ponderaciones recorriendo cada actividad de nuevo. Primero, se preprocesa cada actividad y, una vez se obtienen las palabras clave, se comprueba el peso de cada una. Luego, se suman los pesos para sus respectivas clases. Finalmente, la categoría con el mayor número(suma de ponderaciones) será la correspondiente a ese producto. Este procedimiento garantiza que cada servicio se clasifique de manera precisa según las palabras clave en una única categoría.

```
for i in range(len(data)):
  activity = processActivities(data[i])
  fullDesc = data[i]
  for cat in range(len(AllCategories)):
    AllCategories[cat].reset_finalSum()
    for j in range(len(activity)):
      if AllCategories[cat].had_log(activity[j]):
        AllCategories[cat].add_finalSum(AllCategories[cat].get_logarithm(activity[j]))
    AllSums[cat] = AllCategories[cat].get_finalSum()
```

**Imagen 4.10: Sumatorio de pesos para cada categoría.**

classificationInfo.txt

```
actividad loro parque situada tenerife puerto cruz proporcionada canar
OCIO: -602.5457787178441
DEPORTES: -681.1082693777565
TOURS: -701.1363165200597
CULTURAL Y GASTRONÓMICO: -680.0313358843902
BIENESTAR: -700.5651548273881
OCIO

actividad forestal park situada tenerife lagunetas proporcionada canar
OCIO: -510.70366062683786
DEPORTES: -474.9354429396978
TOURS: -530.1855948596298
CULTURAL Y GASTRONÓMICO: -525.5969198706492
BIENESTAR: -545.274729428812
DEPORTES
```

**Imagen 4.11: Ejemplo de clasificación.**

Sin embargo, el objetivo no es que cada producto pertenezca exclusivamente a una única clase. Lo que se busca es que cada actividad pueda tener porcentajes de pertenencia distribuidos entre varias categorías. Como se explicó inicialmente, la intención principal es recabar información de los usuarios a través de su interacción con unos botones que reflejarán sus preferencias, que se representarán como porcentajes asignados a cada opción, y nuestra tarea será asignar esos mismos porcentajes a los diferentes productos.

Por lo tanto, la solución propuesta consiste en establecer un rango entre los pesos de manera que el mayor se asigne un 100% y el menor un 0%. Para lograr esto, se aplica una regla de tres: si el mayor peso representa el 100%, se calcula el valor proporcional para los demás pesos. La fórmula que se aplica sería tal que:

$$\% = \left( \frac{peso\ Actual - peso\ Mínimo}{peso\ Máximo - peso\ Mínimo} \right) \times 100$$

**Imagen 4.12: Fórmula para calcular porcentaje**

Sin embargo, esto deriva en un nuevo problema: las actividades siempre tendrán una categoría con un 100% y otra con un 0%. Esto está muy alejado de la realidad, ya que un usuario puede marcar todas las casillas con valores iguales o con valores dispares sin necesariamente tener un 100% y un 0%.

Para reflejar mejor las preferencias del usuario, necesitamos una metodología que permita distribuir los valores de manera más equilibrada, respetando la posibilidad de tener múltiples categorías con porcentajes similares o variados, sin extremos absolutos. La solución planteada es ampliar este rango. En lugar de ir del menor al mayor, ahora se calculará la desviación típica de los resultados de todas las categorías de cada actividad. Al mayor se le sumará esta desviación y al menor se le restará. Esto permitirá que los valores se distribuyan de manera más realista, evitando extremos absolutos y reflejando más fielmente las preferencias de los usuarios. De manera que la nueva fórmula sería:

$$\% = \left( \frac{peso\ Actual - (peso\ Mínimo - \sigma)}{(peso\ Máximo + \sigma) - (peso\ Mínimo - \sigma)} \right) \times 100$$

**Imagen 4.13: Fórmula final para calcular porcentajes**

```
def calc_standard_deviation(logarithm_array):
  average = 0
  for category, number in logarithm_array.items():
    average = average + abs(number)
  average = average/len(logarithm_array)
  acc = 0
  for category, number in logarithm_array.items():
    acc = acc + ((abs(number) - average) ** 2)
  return acc / average;
```

**Imagen 4.14: Cálculo de la desviación típica**

A continuación, se presentan ejemplos del código y los resultados actualizados:

```python
def convert_percentages(min, max, final_log):
  #max_percentage = abs(abs(min[1]) - abs(max[1]))          #100
  final_percentages = {}
  standard_derivation = calc_standard_deviation(final_log)
  print(standard_derivation)
  max_percentage = abs((abs(min[1]) + standard_derivation) - (abs(max[1]) - standard_derivation))
  for category, number in final_log.items():
    actual_percentage = (abs(abs(number) - (abs(min[1]) + standard_derivation))) / abs((abs(max[1])- standard_derivation) - (abs(min[1]) + standard_derivation))

    final_percentages[category] = int((actual_percentage * 100)) #calculo de porcentaje


  return final_percentages
```

**Imagen 4.15: Cálculo de porcentajes**

```
{'OCIO': 91, 'CULTURA': 26, 'DEPORTES': 25, 'BIENESTAR': 8, 'TOURS': 8}
```

**Imagen 4.16: Ejemplo del resultado para la actividad 1**

## 4.4 Recomendador

Es innegable que, tal como se ha planteado la solución, el clasificador facilitará en gran medida la tarea a realizar. No obstante, a pesar de tener esta herramienta a nuestra disposición, el problema aún no está completamente resuelto. Lo único que falta es recopilar la entrada de datos proporcionada por el usuario, analizarla y compararla con las actividades que ya han sido clasificadas gracias al trabajo realizado hasta el momento.

Esta fase final del proceso implica desarrollar un sistema que permita realizar un análisis de estos datos, utilizando el instrumento previamente desarrollado y entrenado para comparar las preferencias recogidas con las actividades clasificadas.

### 4.4.1 Ordenación

Debido a que la entrada de datos por parte del usuario viene dada en un cierto orden (vendría dado por la compañía), en nuestro caso usaremos el orden alfabético, y el clasificador ordena las clases de cada actividad por porcentajes, de mayor a menor, es necesario reordenar las categorías de cada producto alfabéticamente.

Este reordenamiento garantiza que las comparaciones y cálculos sean precisos, alineando el formato de la entrada de usuario con el formato de los datos clasificados. De esta manera, se simplifica el procesamiento y análisis de la información, asegurando que todas las operaciones se realicen de manera eficiente.

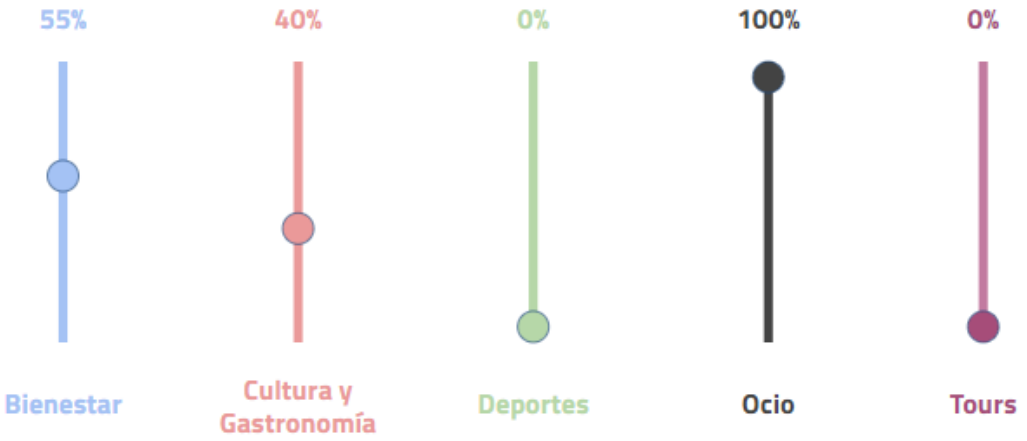


**Imagen 4.17: Ejemplo de selección de usuario**

### 4.4.2 Recomendación euclidiana

Para finalizar, solo falta aplicar una medida de similitud para determinar qué actividades se asemejan más a las preferencias del usuario. La medida utilizada en un principio ha sido la distancia euclidiana, ya que tanto las preferencias como las categorías forman vectores de, en este caso, cinco dimensiones. La distancia euclidiana es ideal porque calcula la distancia entre dos puntos en un espacio n-dimensional. Otras ventajas que tenemos son:

La simplicidad, ya que es una de las formas más simples y directas de medir la similitud entre dos puntos en un espacio multidimensional. Es fácil de calcular e interpretar.

La visualización de distancia, debido a que el resultado nos proporciona, es la longitud del segmento de línea recta que une los dos vectores. Esto nos ayuda a visualizar cómo de "lejos" están las preferencias del usuario de las características de una actividad.

Por último trata todas las dimensiones (en este caso, categorías) de manera uniforme. Cada categoría contribuye de manera igual a la distancia total, lo cual puede ser deseable si creemos que todas las categorías son igualmente importantes.

$$Distancia = \sqrt{\sum \left(Preferencia_i - Categoría_i\right)^2}$$

**Imagen 4.18: Fórmula de la distancia euclidiana**

Pasando al caso práctico, para probar su funcionalidad, creamos de manera artificial un vector de porcentajes que represente a un usuario. Este vector se lo pasamos al programa, que recorrerá todas las actividades comprobando la distancia euclidiana de cada una y seleccionará la menor. Ese será el punto de interés a recomendar. Así mismo se podrá simular diferentes escenarios de usuario y evaluar la precisión. Al utilizar un vector de porcentajes artificial para realizar pruebas asegurará que el algoritmo funciona correctamente y selecciona la actividad más adecuada en función de las preferencias del usuario.

```
# Simulamos la entrada de un usuario
UserPreferences = [9.0, 90.0, 20.0, 36.0, 13.0]

bestActivity = None
bestDistance = float('inf')  # Inicializar con un valor muy grande

for activity, percentage in activities.items():
    distance = Eculidean_distance(UserPreferences, percentage)
    if distance < bestDistance:
        bestActivity = activity
        bestDistance = distance

print("Actividad recomendada:", bestActivity)
```

```
Actividad recomendada: 9
```

**Imagen 4.19: Código recomendador**

### 4.4.3 Recomendación con Coseno

Aunque los resultados con la medida anterior han sido positivos, esta es muy simple y no siempre proporciona resultados profundos o elaborados. Además, como se explicó en el Capítulo 2, hay otras medidas que son mejores opciones en espacios de alta dimensionalidad. Por ello, se busca mejorar la fiabilidad del recomendador aplicando una nueva más robusta, como es la similitud del coseno. Permite evaluar la similitud entre dos vectores en un espacio multidimensional, proporcionando resultados más precisos y elaborados al tener en cuenta el ángulo entre los vectores, en lugar de sólo su magnitud.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

**Imagen 4.20: Fórmula de la similitud del coseno**

Como se puede observar en la fórmula, primero debemos calcular el producto escalar de los dos vectores. Luego, se calculará por separado la magnitud de cada vector para multiplicarlas. Por último, dividiremos el producto escalar entre el producto recién calculado. Este será el resultado de la similitud entre las preferencias del usuario (vector A) y la actividad en concreto (vector B). Si esto lo visualizamos en el código quedaría de la siguiente manera:

```
def cosineSimilarity(userPreferences, activity):
    scalarProduct = 0
    for i in range(0, len(UserPreferences)):
      scalarProduct += userPreferences[i] * activity[i]
    magnUser = math.sqrt(sum(u ** 2 for u in userPreferences))
    magnActivity = math.sqrt(sum(a ** 2 for a in activity))

    return scalarProduct / (magnUser * magnActivity)
```

**Imagen 4.21: Función de cálculo de similitud del coseno**

El resultado devuelto por la función, varía entre los valores 0 (más alejado) y 1 (más cercano). Es por ello, que nos interesa quedarnos con los resultados que estén más cercanos al 1, debido a que estos serán los más similares.

Para concluir, se ha implementado una funcionalidad adicional en el recomendador con el objetivo de proporcionar una variedad de opciones al usuario. Para ello se han implementado las siguientes líneas. Lo que hacen es elegir los valores, y por ende las actividades más similares a las preferencias del usuario.

```
similarities = {}
UserPreferences = [9.0, 20.0, 93.0, 34.0, 11.0]

for activity, percentage in activities.items():
    similarity = cosineSimilarity(UserPreferences, percentage)
    similarities[activity] = similarity

maxMinOrder = sorted(similarities.items(), key=lambda item: item[1], reverse=True)

# Obtener los primeros 10 pares clave-valor del diccionario ordenado
bestIndexs = maxMinOrder[:10]

wb = openpyxl.load_workbook('activities.xlsx')
sheet = wb.active
data = []
i= 0
for row in sheet.iter_rows(min_row=2, values_only=True):
    activityName = row[1]
    data.append(activityName)

# Imprimir los 10 primeros pares clave-valor
for key, value in bestIndexs:
    print("La actividad '" + data[key - 1] + "': " + str(value))
```

```
La actividad 'Barranco de Masca': 0.9995690081172794
La actividad 'Forestal Park': 0.996055685939852
La actividad 'Kayak de mar': 0.9885719510767947
La actividad 'Forestal Park': 0.9814841641112686
La actividad 'Unlimited Golf': 0.9797012605148627
La actividad 'Senderos de la Biosfera: Tegueste – Punta del Hidalgo (por Bejía - El Batán - Chinamada)': 0.9745467222429773
La actividad 'El Teide, entre volcanes y estrellas': 0.9701697136965962
La actividad 'Pico Teide, el reino de las nubes': 0.970156544312039
La actividad 'Kayak Los Gigantes + Avistamiento Cetáceos': 0.9688377481964305
La actividad 'Bautismo de buceo': 0.9640388078537605
```

**Imagen 4.22: Selección final de los mejores resultados**

### 4.4.4 Resultados Obtenidos

A continuación se mostrarán los resultados obtenidos con una serie de gráficos. Para ello simularemos la entrada de datos de un cliente ficticio, asignando valores concretos a los porcentajes de las categorías. Para el ejemplo usaremos:

| Bienestar | Cultura y gastronomía | Deporte | Ocio | Tours |
|---|---|---|---|---|
| 9% | 20% | 95% | 31% | 11% |

**Tabla 4.1: Preferencias de usuario para el ejemplo**

El resultado obtenido sería:

| Actividades | Similitud de Coseno |
|---|---|
| Barranco de Masca | 0.9995 |
| Forestal Park1 | 0.9960 |
| Kayak de mar | 0.9885 |
| Forestal Park2 | 0.9814 |
| Unlimited Golf | 0.9797 |
| Senderos de la biosfera(Tegueste) | 0.9745 |
| El teide entre volcanes y estrellas | 0.9701 |
| Pico del teide, el reino de las nubes | 0.9701 |
| Kayak los gigantes | 0.9688 |
| Bautismo de buceo | 0.9640 |

**Tabla 4.2: Resultados del ejemplo**

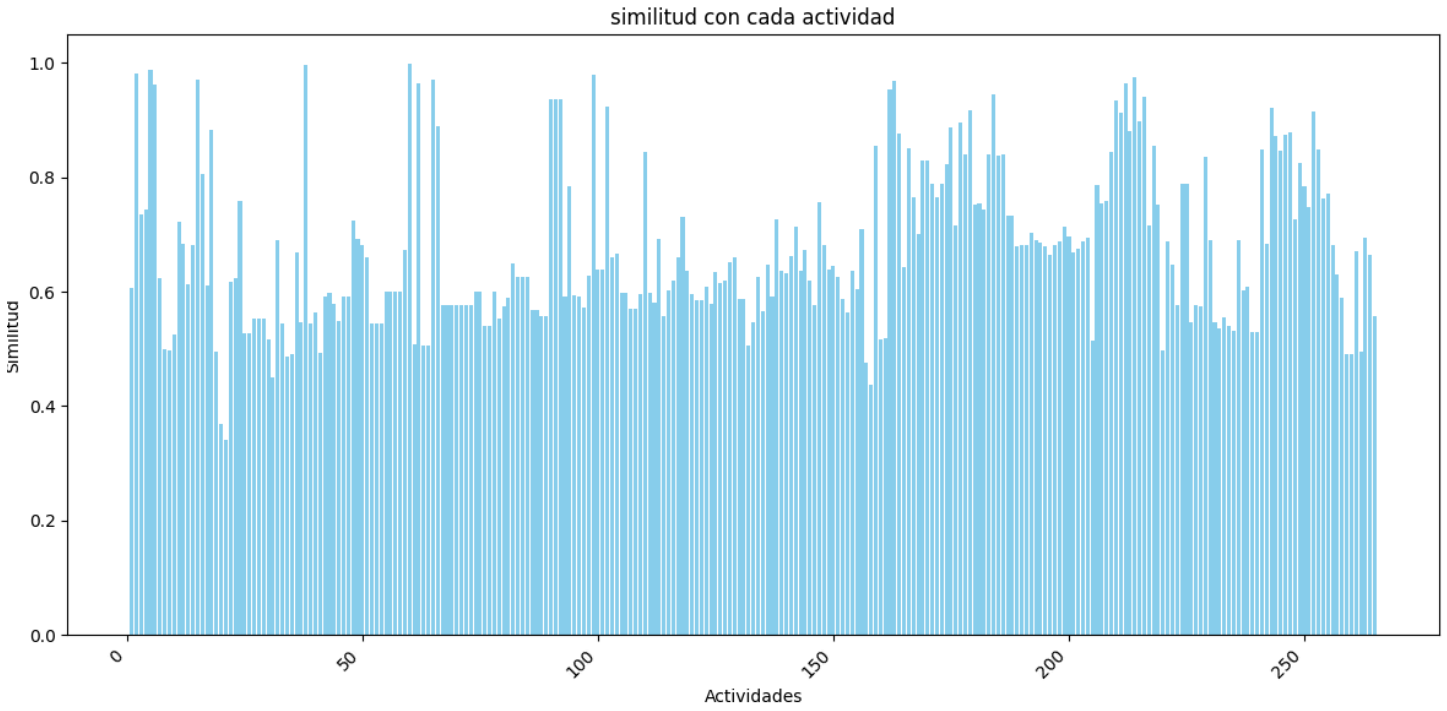


**Imagen 4.23: Comparativa de similitud con todas las actividades**

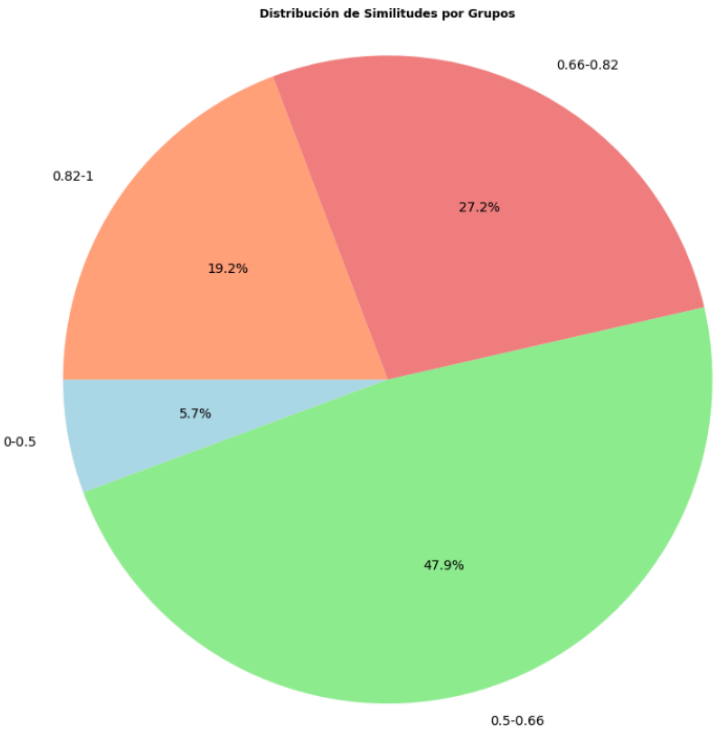


**Imagen 4.24: Distribución de similitudes por grupos**

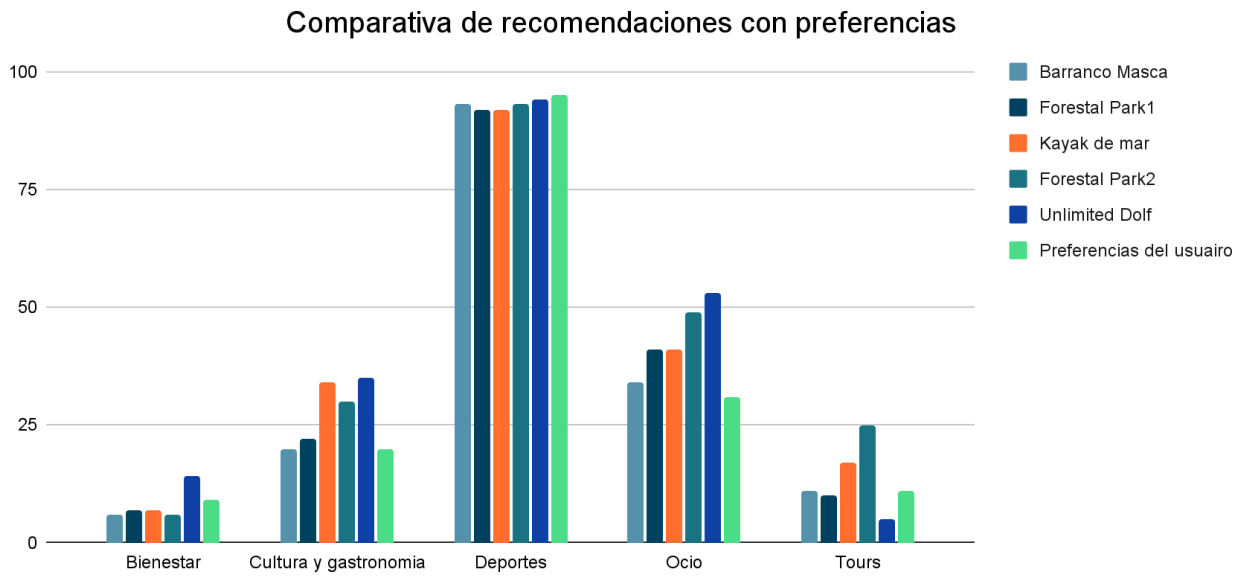


**Imagen 4.24: Comparación de las recomendaciones con las preferencias del usuario**

# Capítulo 5 Conclusiones y líneas futuras

En este proyecto, nos hemos propuesto abordar el desafío de predecir qué actividades podrían resultar más adecuadas para los usuarios que desean emprender un viaje. Hemos empleado diversas técnicas de inteligencia artificial y ciencia de datos, tal como se ha detallado a lo largo de este documento.

Los resultados obtenidos parecen prometedores a pesar de la limitada cantidad de información disponible para entrenar al clasificador, hacer pruebas, así como las limitaciones de tiempo. Además, hemos utilizado diversas medidas en la implementación del recomendador.

Este proyecto de desarrollo ha sido de gran interés, como ell desafío de combinar diferentes técnicas y enfoques, nuevos para mi, para resolver un problema de recomendación de actividades turísticas. Así, el problema abordado, me ha permitido aprender nuevos conceptos y adquirir habilidades nuevas. Además, he tomado conciencia del impacto que tienen este tipo de sistemas en la experiencia de viaje de los usuarios y como una buena implementación puede mejorar la satisfacción y ayudar a descubrir nuevas experiencias que muchas veces están ocultas. Es importante que estos sistemas no solo recomienden los lugares conocidos, sino que permitan al viajero conocer nuevas actividades y tener experiencias originales.

Como líneas futuras de trabajo y con el fin de profundizar en el estudio de este problema, se podrían considerar varias estrategias adicionales. En primer lugar, explorar la implementación de un sistema híbrido, en función de la disponibilidad de información. Por ejemplo, un sistema híbrido de selección de método, considero que sería el más acertado en nuestro caso. Cuando un usuario se registra en la página web y no contamos con información previa sobre él, se le solicitarían sus preferencias y se le recomendarían actividades utilizando el sistema existente. Sin embargo, si disponemos de información previa sobre ese cliente, podríamos optar por utilizar otro método que se adapte mejor a su perfil y preferencias históricas, ya sea filtrado o basado en contenido.

Además, sería recomendable llevar a cabo un test de la clasificación con un grupo de personas, no necesariamente consumidores, para que clasifiquen las actividades turísticas. Esto ayudaría a obtener una clasificación más objetiva.

# Capítulo 6 Summary and Conclusions

In this project, we set out to address the challenge of predicting which activities might be most suitable for users wishing to undertake a journey. We have employed a variety of techniques, as detailed throughout this paper.

The results obtained seem promising despite the limited amount of information available to train the classifier and run different tests. In addition, we have tested several techniques in the implementation of the recommender.

Personally, I found this project extremely interesting for several reasons. Firstly, the challenge of combining different techniques and approaches, new to me, to solve a problem such as the recommendation of tourist activities has allowed me to learn a variety of new concepts and skills. In addition, I have realized the impact that this type of system has on the travel experience of users, as it is a problem that I had never considered before and I have seen that with a good implementation it can improve their satisfaction and help them to discover new experiences that are often hidden. It is important that these systems not only recommend typical places, but also allow the traveler to carry out original activities.

As future lines of work and in order to further study this problem, several additional strategies could be considered. First, we could explore the implementation of a hybrid system, depending on the availability of information. For example, a hybrid method selection system, I believe, would be the most successful in our case. When a user registers on the website and we have no prior information about him, we would ask him for his preferences and recommend activities using the existing system. However, if we have prior information about that customer, we could choose to use another method that better suits their profile and historical preferences, either filtered or content-based.

In addition, it would be advisable to conduct ranking surveys with a group of people, not necessarily consumers, to rank tourism activities. This would help to obtain a more objective ranking, as the current program suggests based on my perceptions.

# Capítulo 7 Presupuesto

Este capítulo se usará para representar el valor económico que tendría este proyecto. Para establecer el valor económico del mismo, se emplearán las horas invertidas de manera que el precio viene dado por €/h

| Actividad | Horas | Coste |
|---|---|---|
| Estudio previo sobre sistemas | 60 | 120€ |
| Documentación | 100 | 200€ |
| Obtención de datos | 0 | 0€ |
| Preprocesamiento de datos | 20 | 40€ |
| Clasificación | 15 | 15€ |
| Pruebas y mejoras del Clasificador | 50 | 100€ |
| Desarrollo del recomendador | 25 | 50€ |
| Desarrollo de presentaciones y seminarios | 30 | 60€ |

**Tabla 7.1: Coste de desarrollo**

# Capítulo 8 Código desarrollado

El código de este proyecto se encuentra en el siguiente repositorio:
https://github.com/GabriRH21/Recomendador-Turistico.git

# Bibliografía


[1] Kinjal Chaudhari, Ankit Thakkar. (2020). A Comprehensive Survey on Travel Recommender Systems. Springer

[2] Joy Lal Sarkar, ·Abhishek Majumder, Chhabi Rani Panigrahi, Sudipta Roy, Bibudhendu Pati. (2023). Tourism recommendation system: a survey and future research directions. Springer

[3] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, Grammati Pantziou. (2014). Journal of Network and Computer Applications. Elsevier Ltd

[4] Joan Borràs, Antonio Moreno, Aida Valls. (2014). Expert Systems with Applications. Elsevier Ltd

[5] James Allen. (2002). Natural language understanding, Pearson india

[6] Jacob Eisenstein. (2019). Introduction to Natural Language Processing. The MIT Press

[7] Carlos G. Figuerola, José L. Alonso Berrocal, Angel F. Zazo Rodríguez, Emilio Rodríguez. (2002). Algunas Técnicas de Clasificación Automática de Documentos. Grupo Reina

[8] Daniel Jurafsky, James H. Martin. (2000). Speech and Language Processing. https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf

[9] “BeautifulSoup Documentation” https://www.crummy.com/software/BeautifulSoup/bs4/doc/#for-html-documents

[10] “Aprende con Python - beautifulSoup” https://aprendepython.es/pypi/scraping/beautifulsoup/

[11] “openpyxl - A Python library to read/write Excel 2010 xlsx/xlsm files” https://openpyxl.readthedocs.io/en/stable/