GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

Análisis de los efectos producidos en la combinación de diferentes fuentes de información en la recomendación de puntos de interés

Autor: Beatriz Sicilia Gómez

Director: Pablo Sánchez Pérez

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Análisis de los efectos producidos en la combinación de diferentes fuentes de información en la recomendación de puntos de interés

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2022/23 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

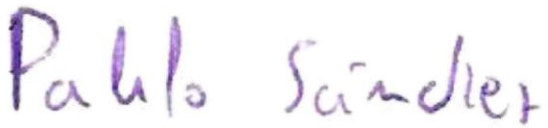
El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Beatriz Sicilia Gómez Fecha: 04/07/2023

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO



Fdo.: Pablo Sánchez Pérez Fecha: 04/07/2023

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

Análisis de los efectos producidos en la combinación de diferentes fuentes de información en la recomendación de puntos de interés

Autor: Beatriz Sicilia Gómez

Director: Pablo Sánchez Pérez

Madrid

Agradecimientos

A mi familia por haberme acompañado durante toda la carrera, a mis amigas de Boston por haberme apoyado durante este año y a mi tutor, Pablo, por su disponibilidad, guía y consejo.

ANÁLISIS DE LOS EFECTOS PRODUCIDOS EN LA COMBINACIÓN DE DIFERENTES FUENTES DE INFORMACIÓN EN LA RECOMENDACIÓN DE PUNTOS DE INTERÉS

Autor: Sicilia Gómez, Beatriz.
Director: Sánchez Pérez, Pablo.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas.

RESUMEN DEL PROYECTO

En este trabajo de fin de grado se ha llevado a cabo un estudio del estado del arte de los sistemas de recomendación, centrándonos en la recomendación de puntos de interés. Se ha desarrollado un pipeline de recomendación que, empleando datos de redes sociales basadas en localización como Foursquare o Gowalla, proporcionaba recomendaciones sobre puntos de interés. Además, se han empleado técnicas de dominio cruzado para analizar el impacto que tiene en las recomendaciones producidas la inclusión de nuevas fuentes de datos.

Palabras clave: Sistemas de recomendación, recomendación de puntos de interés, redes sociales basadas en localización (LBSN), dominio cruzado.

1. Introducción

Los sistemas de recomendación son herramientas software que, utilizando distintas técnicas y algoritmos, permiten sugerir a un usuario específico artículos de su interés. Están presentes en la mayoría de las plataformas en línea ya que proporcionan una experiencia de usuario más personalizada y satisfactoria, mitigando problemas como el de la sobrecarga de información a la que están expuestos los usuarios.

Estos sistemas son aplicados en diversos dominios como la recomendación de películas, libros, o artículos en sitios de comercio electrónico. Además de estos dominios de recomendación clásicos, han surgido otros dominios como el de la recomendación de puntos de interés. En este caso, se sugieren a los usuarios lugares que visitar al llegar a una nueva ciudad o región. Para generar recomendaciones en este dominio, se suelen emplear algunos datos adicionales como la ubicación geográfica de los usuarios combinado con datos temporales o secuenciales.

El mayor desafío de los sistemas de recomendación es la falta de conocimiento sobre los gustos de los usuarios en relación con todos los productos o elementos disponibles. Esto es lo conocido como sparsity o dispersión de los datos, lo cual dificulta la generación de recomendaciones relevantes. Por esa razón, es común combinar los datos de diferentes fuentes de información para paliar parcialmente esta escasez de datos.

2. Definición del proyecto

En este trabajo de fin de grado se ha desarrollado un pipeline de recomendación que trabaja con datos procedentes de la red social basada en localización (LBSN) Foursquare para efectuar recomendaciones y posteriormente evaluarlas utilizando diversas métricas. Además, se ha analizado el efecto que tiene la combinación de diferentes conjuntos de datos (otras redes sociales) en la calidad de las recomendaciones, utilizando para ello técnicas de cross-domain o dominio cruzado.

3. Descripción del modelo

En este proyecto se ha desarrollado un pipeline completo de recomendación el cual consta de tres fases fundamentales: preprocesado de datos, generación de recomendaciones y evaluación.

Para generar las recomendaciones se utilizaron datos proporcionados por redes sociales basadas en localización (LBSN). Se escogió Foursquare como fuente de datos principal ya que era el dataset con mayor número de check-ins. En total se procesaron 33 millones de datos (interacciones entre usuarios y puntos de interés), los cuales fueron combinados con otras fuentes de información como Gowalla o Brightkite que también contenían varios millones de registros. Este proceso de integración permitió enriquecer los datos y aumentar la cantidad de información disponible para generar recomendaciones.

La primera fase consiste en el preprocesado de los datos y su separación en bade a datos temporales en los conjuntos de entrenamiento validación y test para aplicarlos al modelo.

También, en esta fase se llevará a cabo la búsqueda de equivalencias entre la fuente de datos principal seleccionada (Foursquare) y las otras fuentes de datos adicionales.

En la segunda etapa del proceso, se generaron recomendaciones utilizando diferentes algoritmos como el basado en popularidad, vecinos próximos o híbrido. Para los algoritmos basados en vecinos próximos, se escogieron como parámetro el número de vecinos que optimizaran la precisión de las recomendaciones.

Por último, en la etapa de evaluación se midió la calidad de las recomendaciones proporcionadas en base a tres propiedades: la relevancia (acierto en las recomendaciones), novedad (recomendación de artículos poco conocidos) y diversidad (recomendación de artículos variados). A su vez, las métricas se dividieron en métricas calculadas por cada usuario y por sistema (considerando todas las recomendaciones en su conjunto).

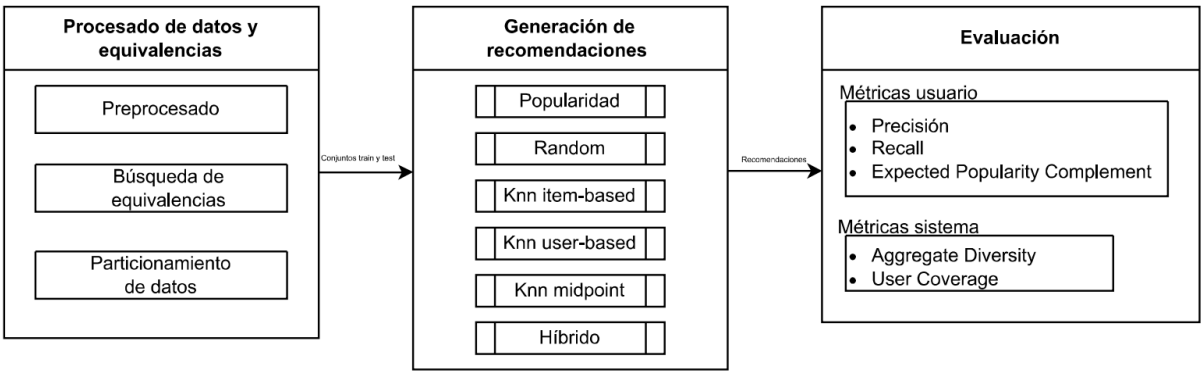


Ilustración 1-1 Esquema del framework de recomendación

4. Resultados

- En general las métricas obtenidas para todos los algoritmos tuvieron valores bajos, algo común en este dominio de recomendación, debido a la alta dispersión de los conjuntos de datos empleados. No obstante, se observó que los algoritmos desarrollados eran efectivos ya que demostraban un rendimiento mayor que el recomendador aleatorio.

- Se observó un sesgo de popularidad en las recomendaciones, siendo este el algoritmo con mejores resultados, a pesar de no ser pesonalizado. Por otro lado, el algoritmo híbrido, el cual combinaba las recomendaciones proporcionadas por los algoritmos de popularidad, knn y knn midpoint, presentó un buen balance entre las métricas de relevancia y las de novedad y diversidad, haciendo que este recomendador sea el más adecuado en términos generales.

- Al incluir conjuntos de datos adicionales se observó un decremento en la precisión media de las recomendaciones de hasta un 18%. Sin embargo, se observaron mejoras en las métricas de diversidad y novedad. Específicamente, el aggregate diversity aumentó hasta un 14% para algunos algoritmos.

5. Conclusiones

El objetivo principal del trabajo era demostrar que, al utilizar dominio cruzado, es decir, combinar datos de diferentes fuentes, se podría aumentar la efectividad de los sistemas de recomendación al contar con más información. En términos de acierto vimos un efecto contrario al esperado, es decir, la precisión de las recomendaciones disminuyó al incluir nuevos datos, esto puede atribuirse a la falta de comparabilidad entre los conjuntos de datos utilizados. Esto puede indicar también que los resultados reportados en los artículos analizados del dominio de la recomendación de puntos de interés no sean tan comparables entre ellos como se podría suponer en un primer momento.

Sin embargo, al incorporar nuevos datos métricas como el Aggregate Diversity o el Expected Popularity Complement aumentaron en todos los casos, con lo cual, aunque se haya perdido acierto se les recomendaron a los usuarios POIs más novedos y variados. Estos resultados prometedores pueden favorecer la aparición de nuevos algoritmos que sean capaces de proporcionar recomendaciones diferentes a los usuarios.

Como trabajo futuro, se plantea explorar nuevas fuentes de información como el tiempo atmosférico, la cantidad de visitantes que hay en tiempo real en el punto de interés o el horario de cada punto de interés. Estos datos se incorporarán con el objetivo de brindar una experiencia más completa y satisfactoria para los usuarios, ya que el clima, la afluencia de personas y el horario de los puntos de interés son factores que influyen en las preferencias y decisiones de los usuarios.

ANALYSIS OF THE EFFECTS PRODUCED BY COMBINING DIFFERENT SOURCES OF INFORMATION IN THE RECOMMENDATION OF POINTS OF INTEREST.

Author: Sicilia Gómez, Beatriz.
Supervisor: Sánchez Pérez, Pablo.
Collaborating Entity: ICAI – Universidad Pontificia Comillas.

ABSTRACT

In this final thesis we have carried out a study of the state of the art of recommender systems, focusing on the recommendation of points of interest. A recommendation pipeline has been developed using data from location-based social networks such as Foursquare or Gowalla, providing recommendations on points of interest. In addition, cross-domain techniques have been used to analyze the impact of the inclusion of new data sources on the recommendations produced.

Keywords: recommender systems, point-of-interest recommendation, location-based social networks (LBSN), cross-domain.

1. Introduction

Recommender systems are software tools that, using different techniques and algorithms, allow suggesting items of interest to a specific user. They are present in most online platforms as they provide a more personalized and satisfactory user experience, mitigating problems such as information overload to which users are exposed.

These systems are applied in various domains such as the recommendation of movies, books, or articles on e-commerce sites. In addition to these classic recommendation domains, other domains have emerged, such as the recommendation of points of interest. In this case, users are suggested places to visit when arriving in a new city or region. To generate recommendations in this domain, some additional data such as the geographic location of the users combined with temporal or sequential data is usually employed.

The biggest challenge of recommender systems is the lack of knowledge about users' tastes in relation to all available products or items. This is known as sparsity or data dispersion, which makes it difficult to generate relevant recommendations. For this reason, it is common to combine data from different information sources to partially alleviate this data sparsity.

2. Project definition

In this thesis we have developed a recommendation pipeline that works with data from the location-based social network (LBSN) Foursquare to make recommendations and then evaluate them using various metrics. In addition, the effect of combining different datasets (other social networks) on the quality of recommendations has been analyzed using cross-domain techniques.

3. Model description

In this project, a complete recommendation pipeline has been developed which consists of three main phases: data preprocessing, recommendation generation and evaluation.

To generate the recommendations, data provided by location-based social networks (LBSN) was used. Foursquare was chosen as the main data source as it was the dataset with the highest number of check-ins. In total, 33 million data (interactions between users and points of interest) were processed, which were combined with other data sources such as Gowalla or Brightkite that also contained several million records. This integration process enriched the data and increased the amount of information available to generate recommendations.

The first phase consists of preprocessing the data and separating it into temporal data in the validation and test training sets to be applied to the model. Also, in this phase, the search for equivalences between the main data source selected (Foursquare) and the other additional data sources will be carried out.

In the second stage of the process, recommendations were generated using different algorithms such as popularity-based, nearest neighbor or hybrid. For the nearest neighbor-based algorithms, the number of neighbors was chosen as a parameter to optimize the accuracy of the recommendations.

Finally, in the evaluation stage, the quality of the recommendations provided was measured based on three properties: relevance (accuracy of recommendations), novelty (recommendation of little-known articles) and diversity (recommendation of varied articles). In turn, the metrics were divided into metrics calculated per user and per system (considering all recommendations as a whole).

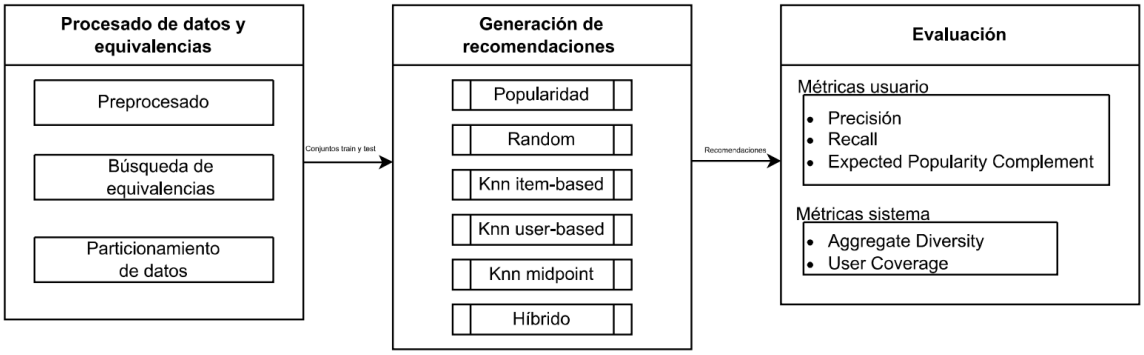


Ilustración 1-2 Outline of the Recommendation Framework

4. Results

- In general, the metrics obtained for all the algorithms had low values, something common in this recommendation domain, due to the high dispersion of the data sets used. Nevertheless, it was observed that the developed algorithms were effective as they demonstrated higher performance than the random recommender.

- A popularity bias was observed in the recommendations, this being the algorithm with the best results, despite not being personalized. On the other hand, the hybrid algorithm, which combined the recommendations provided by the popularity, knn and knn midpoint algorithms, presented a good balance between the relevance metrics and the novelty and diversity metrics, making this the most suitable recommender in general terms.

- Including additional data sets resulted in a decrease in the average accuracy of recommendations of up to 18%. However, improvements in diversity and novelty metrics were observed. Specifically, aggregate diversity increased by up to 14% for some algorithms.

5. Conclusions

The main objective of the work was to demonstrate that, by using cross-domain, that is, combining data from different sources, the effectiveness of the recommendation systems could be increased by having more information. In terms of accuracy, we saw the opposite effect to that expected, i.e., the accuracy of the recommendations decreased when new data were included; this may be attributed to the lack of comparability between the data sets used. This may also indicate that the results reported in the analyzed articles from the POI recommendation domain are not as comparable with each other as one might first assume.

However, when incorporating new data, metrics such as Aggregate Diversity or Expected Popularity Complement increased in all cases, which means that, even if they were less successful, users were recommended more novel and varied POIs. These promising results may encourage the emergence of new algorithms capable of providing different recommendations to users.

As future work, we plan to explore new sources of information such as the weather, the number of visitors in real time at the point of interest or the schedule of each point of interest. These data will be incorporated with the aim of providing a more complete and satisfactory experience for users, since the weather, the influx of people and the schedule of the points of interest are factors that influence the preferences and decisions of users.

Índice de la memoria

Índice de figuras

Índice de tablas

Índice de ecuaciones

Capítulo 1. INTRODUCCIÓN

El uso de Internet ha transformado completamente la forma en la que las personas interactúan entre ellas, realizan compras y se divierten. Actualmente, el 62.5% de la población mundial es usuaria de Internet [1] , que con una previsión de crecimiento anual del 4%, se estima que en 2030 habrá alrededor de 7.5 mil Millones de usuarios. El crecimiento masivo del uso de Internet ha sido uno de los impulsores clave del comercio electrónico, el cual ha experimentado un crecimiento inigualable en los últimos años. Se calcula que la cifra de negocio a nivel mundial de comercio electrónico alcanzará en torno a los 6 billones de euros en 2023 y superará los 7 billones en 2025 [2]. Este crecimiento, ha despertado el interés entre las empresas de crear una experiencia para el usuario atractiva y personalizada, que permita solucionar problemas como el de la sobrecarga de información ofertada al usuario. Para lograrlo, los sistemas de recomendación se han convertido en un componente crucial para un gran número de servicios de Internet.

Los sistemas de recomendación son herramientas de software que, utilizando distintas técnicas y algoritmos de aprendizaje automático, generan recomendaciones que puedan ser de interés para los usuarios de una determinada plataforma. Estos pueden recomendar una amplia variedad de artículos, dependiendo del contexto y dominio en el que se utilicen. Algunos ejemplos comunes son la recomendación de películas, libros, música, artículos de comercio electrónico, noticias y puntos de interés, es decir, lugares de interés turístico de una región.

La investigación en este tipo de sistemas de recomendación experimentó un crecimiento a partir del “Netflix Prize” organizado en 2006 por la conocida compañía de Streaming, el cual tenía como objetivo encontrar un algoritmo de recomendación de películas que incrementara la precisión de las predicciones del sistema en un 10%. El desafío consistía en

crear un algoritmo capaz de recomendar películas a un usuario utilizando su historial de visualización.

Este concurso suscitó interés a nivel mundial en los sistemas de recomendación y condujo a avances significativos en el área. También, hizo que todas las empresas a nivel mundial se dieran cuenta de que los sistemas de recomendación tenían una gran aplicación más allá de la recomendación de películas [3].

Actualmente, la mayoría de las empresas que cuentan con comercio en línea utilizan los datos de navegación e historial que tienen sobre sus usuarios para ofrecerles recomendaciones personalizadas de contenido, productos o servicios, aumentando así la calidad de la experiencia del usuario en la plataforma. Algunos ejemplos de estas empresas incluyen: Amazon[1], Spotify[2] o LinkedIn[3].

Por otro lado, los sistemas de recomendación tienen un gran potencial en otras industrias como la del turismo y hostelería, ya que pueden ayudar tanto a los viajeros que buscan planificar su viaje de forma más eficiente y personalizada como a los propios habitantes de una ciudad para que descubran nuevos lugares que visitar. Para ello, pueden hacer uso de los datos proporcionados por las Redes Sociales Basadas en Localización o LBSNs (Location-Based Social Networks), las cuales permiten a los usuarios registrar los lugares que visitan mediante check-ins (la visita de un usuario a un lugar determinado, que se registra en una

[1]Amazon: https://www.amazon.com/

[2]Netflix: https://www.netflix.com/

[3]LinkedIn: https://www.linkedin.com/

base de datos). Ejemplos de redes sociales basadas en la ubicación incluyen Foursquare, Gowalla y Yelp. En ocasiones, también se han empleado datos de redes sociales como Instagram o Flickr para desarrollar sistemas de recomendación de POIs o de rutas [4].

Debido a la amplia aplicación y utilidad de estos sistemas, en este Trabajo de Fin de Grado, se explorará el área de los sistemas de recomendación, concretamente en la recomendación de puntos de interés (POIs del inglés Point-Of-Interest). En el ámbito del turismo y la hostelería, los usuarios buscan orientación para descubrir nuevos lugares y experiencias personalizadas, por estas razones, en este TFG nos centraremos en este tipo de recomendación.

El objetivo principal de este TFG es comprender a fondo el problema de la recomendación, para lo que se llevará a cabo un estudio del estado del arte de los sistemas de recomendación clásicos, la recomendación de puntos de interés y la evaluación offline de los mismos.

Para realizar este estudio, se procesarán los datos obtenidos de las Redes sociales basadas en localización (LBSNs) con el objetivo de construir un pipeline sólido de recomendación. Se examinará cómo se pueden extraer, transformar y cargar datos relevantes que permitan generar recomendaciones personalizadas.

Finalmente, se llevará a cabo una comparación de rendimiento entre los recomendadores desarrollados utilizando datos de una LBSN principal, que en este caso será Foursquare, y esos mismos datos complementados con datos adicionales provenientes de otras LBSN como Gowalla o BrightKite. Esta comparación permitirá analizar cómo las diferentes fuentes de datos impactan en el rendimiento de los recomendadores y qué aspectos deben considerarse al utilizar datos de distintas plataformas.

Capítulo 2. ESTADO DEL ARTE

2.1 SISTEMAS DE RECOMENDACIÓN

Los sistemas de recomendación son herramientas software que generan recomendaciones de artículos y/o servicios que pueden ser útiles para los usuarios de una plataforma en particular. Utilizando diferentes algoritmos, y analizando datos como las necesidades, gustos y preferencias de cada usuario, generarán recomendaciones personalizadas.

Estos sistemas surgieron como respuesta a la sobrecarga de información y la necesidad de contar con herramientas de personalización de contenidos para cada usuario. Para el usuario, resulta tedioso y complejo seleccionar la fracción de información que le interesa de entre toda la disponible en la red. En la era digital, la personalización de contenidos se ha convertido en una herramienta indispensable para las empresas que buscan mejorar la experiencia del usuario y así aumentar la satisfacción del usuario.

Los sistemas de recomendación utilizan principalmente tres tipos de datos: ítems, usuarios y las interacciones entre ellos. Los ítems representan los artículos que el sistema recomienda, que en el contexto de este trabajo, serían los puntos de interés (POIs) de una región. El usuario es el destinatario principal de las recomendaciones. Por último, encontramos las interacciones entre usuarios e ítems, que en el ámbito de la recomendación de puntos de interés serían los check-ins o registros de visitas de un usuario a distintos lugares y en un entorno de recomendación clásico de películas o libros, sería la nota (generalmente acotada entre 1 y 5) que le ha dado el usuario a ese artículo.

2.1.1 FORMULACIÓN TEÓRICA

El sistema de recomendación tratará de recomendar al usuario aquellos productos que le proporcionen una mayor utilidad. La utilidad de un artículo estará representada generalmente por un rating, Estas interacciones pueden ser explícitas, es decir, mediante ratings (generalmente entre y podrá ser especificada por el usuario o calculada por la aplicación.

El objetivo por tanto es recomendar al usuario aquellos artículos con mayor utilidad para él, pudiendo formalizar el problema de la recomendación de la siguiente forma [5]:

$$\forall u \in U, \qquad i'_u = argmax_{\mathrm{i}\in I}\, S(c,i) \qquad \textit{2-1}$$

Donde U representa el conjunto de todos los usuarios, I el conjunto de todos los ítems candidatos a ser recomendados y $S(c,i)$ la función de utilidad.

El objetivo de la función será proporcionar al usuario los artículos que maximicen dicha función de utilidad. Este score no tiene por qué estar acotado como un rating, sino que puede ser un valor que represente cómo de probable es que el usuario consuma el artículo en cuestión, según un determinado algoritmo.

La problemática radica en que la función de utilidad no es conocida para todo el espacio IxU, es decir, no se conocerán los ratings para todas las relaciones artículo/usuario, sino que solo conoceremos el valor de dicha función para un subconjunto de todo el espacio. Por ejemplo, en el premio de Netflix se conocían 100.480.507 ratings para 17.770 películas, proporcionados por 480.189 usuarios, aplicando la ecuación 2-2, el porcentaje de datos conocidos representa un 1,18% del total. Como se verá más adelante, en la recomendación de puntos de interés, el porcentaje de datos conocidos es incluso menor.

$$\%\ datos\ conocidos\ = \frac{nº\ ratings\ conocidos}{nºartículos\ \cdot\ nºusuarios} \cdot 100 \qquad 2\text{-}2$$

Los motores de recomendación deben ser capaces de predecir los ratings para todo el espacio. Para ello, podrán hacer uso de métodos de Machine Learning, aproximaciones o establecer reglas heurísticas que definan la función de utilidad. Cuando se tengan los ratings para los ítems desconocidos, podremos generar recomendaciones a los usuarios, se recomendarán aquellos ítems que tengan los mayores ratings estimados.

Existen diversos tipos de sistemas de recomendación, los cuales se clasificarán según la forma en que realizan las recomendaciones. Estos tipos incluyen los sistemas de recomendación basados en filtrado colaborativo, basados en contenido, demográficos e híbridos. En la siguiente sección se explicarán las características de cada uno de ellos.

2.2 TIPOS DE SISTEMAS DE RECOMENDACIÓN

2.2.1 FILTRADO COLABORATIVO

Los algoritmos de filtrado colaborativo analizan las interacciones entre los usuarios y artículos de un sistema y mediante ese análisis, efectúan recomendaciones. Estos algoritmos tratarán de predecir la utilidad de los ítems para un usuario basándose en los ítems que haya valorado, así como las interacciones de otros usuarios con el resto de los artículos.

Es utilizado por varias compañías dentro del e-commerce como Netflix, eBay o Amazon para analizar las interacciones entre usuarios y artículos y así realizar recomendaciones personalizadas a sus usuarios. Por ejemplo, si los usuarios A y B han dado una alta calificación para un libro, el sistema podrá recomendar dicho libro a un nuevo usuario C con gustos similares a los de A y B.

Ventajas:

- Personalización: se generarán recomendaciones personalizadas a cada usuario en función de sus preferencias.

- A medida que se dispone de más información sobre el usuario, las recomendaciones adquieren mayor calidad, por lo que el sistema se va volviendo cada vez más efectivo.

- No es necesaria la valoración explícita del usuario a cada uno de los productos. (Asignar una puntuación numérica por ejemplo de 0-5 estrellas). La valoración implícita, es más fácil de obtener y menos invasiva para el usuario. Está basada en el comportamiento que tiene el usuario con el artículo, por ejemplo, según el número de clics que ha realizado en un artículo determinado.

Desventajas:

- Su rendimiento puede degradarse en conjuntos que contengan pocos datos o sean dispersos. También podrían presentarse problemas de sesgo en los datos, puesto que los usuarios tienden a evaluar los artículos que les gustan y a ignorar los que no.

- Problema de arranque en frío o Cold Start Problem: Para poder crear recomendaciones personalizadas, el sistema de recomendación necesitará recopilar información sobre el histórico de preferencias del usuario. Por ello, si un usuario es nuevo en el sistema o no tiene un gran histórico de acciones, será difícil recomendarle productos o servicios que puedan interesarle. Una solución inmediata a este problema sería pedir/forzar a los usuarios a puntuar un conjunto de artículos.

Se distinguen dos tipos de filtrado colaborativo, el basado en memoria y el basado en modelos, que se explicarán en detalle a continuación.

2.2.1.1 Filtrado colaborativo basado en memoria

En este tipo de filtrado colaborativo, se emplearán los conocidos como algoritmos de vecinos próximos (o k-nn de k-nearest neighbors) en los que se generarán predicciones considerando las similitudes entre los usuarios o artículos. Algunas de las ventajas de utilizar esta técnica son su simplicidad, eficiencia y capacidad de generar recomendaciones novedosas para los usuarios.

Existen dos tipos de algoritmos de filtrado colaborativo, los basados en el usuario y los basados en el artículo.

Algoritmos basados en el usuario

Se predice la valoración de un usuario sobre un producto teniendo en cuenta las valoraciones que usuarios similares (vecinos) le han dado a dicho artículo en el pasado.

Podremos calcular la valoración que un usuario le dará a un ítem considerando los ratings que sus vecinos más próximos le han dado a ese ítem. Al igual que en los algoritmos basados en el artículo, no todos los vecinos contribuirán de la misma forma al rating, por lo que su contribución se verá escalada por la similitud que tengan con el usuario [6].

En la fórmula, el término $\widehat{r_{ui}}$ será el rating aproximado por el sistema que el usuario *u* le da al producto *i*, $N_i(u)$ será el conjunto de vecinos del usuario u que han puntuado el artículo i. Por último, w_{uv} será la similitud existente entre el usuario *u* y el usuario *v*.

$$\widehat{r_{ui}} = \frac{\sum_{v \in N_i(u)} w_{uv} r_{vi}}{\sum_{v \in N_i(u)} | w_{uv} |} \qquad 2\text{-}3$$

Existen diversas fórmulas que pueden ser utilizadas para calcular las similitudes entre usuarios. Algunas de estas fórmulas se explican a continuación.

Similitud coseno

$$\cos(u, v) = \frac{\sum_{i \epsilon I_{uv}} r_{u_i} r_{v_i}}{\sqrt{\sum_{i \epsilon I_u} r_{u_i^2} \sum_{j \epsilon I_v} r_{v_j^2}}} \qquad 2\text{-}4$$

En esta fórmula se representará a cada usuario como un vector, por lo que se puede obtener el coseno del ángulo que forman ambos vectores en un espacio vectorial y tomar este valor como la similitud. Esta tendrá un valor comprendido entre 0 y 1, siendo 1 la máxima similitud posible.

En la fórmula, el término I_u representa el conjunto de ítems que ha visitado el usuario u, I_v representa el conjunto de ítems visitados por el usuario v, por tanto, I_{uv} será la intersección entre los dos anteriores, representando los ítems que han sido consumidos por ambos usuarios.

Correlación de Pearson

En la fórmula de la correlación de Pearson, deberán aislarse los casos de evaluación conjunta. Es decir, solo se tendrán en cuenta los ítems que hayan sido valorados por ambos usuarios. (I_{uv} se trata del conjunto de ítems visitados por ambos usuarios).

El valor del coeficiente de correlación de Pearson variará entre -1 y 1. Un valor positivo indicará una alta similitud entre usuarios. En el caso contrario, un valor negativo indicará que no hay similitud entre los usuarios. Por último, un valor de 0 implicará que hay falta de correlación y por tanto no se podrá determinar la similitud existente entre los usuarios.

$$\mathrm{PC}(u,v) = \frac{\sum_{i \epsilon I_{uv}} (r_{ui} - \bar{r_u})(r_{vi} - \bar{r_v})}{\sqrt{\sum_{i \epsilon I_{uv}} (r_{ui} - \bar{r_u})^2 \sum_{j \epsilon I_{uv}} (r_{vj} - \bar{r_v})^2}} \qquad 2\text{-}5$$

Índice de Jaccard

Este coeficiente, también conocido como coeficiente de Tanimoto, es la cantidad de elementos por la que dos usuarios expresan cierta preferencia dividida por la cantidad de elementos por los cuales el usuario expresa cierta preferencia [7].

Es decir, es la relación entre el tamaño de la intersección y el tamaño de la unión de los elementos preferidos de ambos usuarios. (En la ecuación, I_u representará el conjunto de elementos preferidos del usuario de prueba u, e I_v representará los elementos preferidos del usuario v).

Cuando los elementos favoritos de dos usuarios se superponen completamente, el valor del coeficiente será 1 y por tanto tendrán una similitud perfecta. En el caso contrario, cuando no tienen nada en común, el coeficiente tomará un valor igual a 0.

$$J(u,v) = \frac{I_u \cap I_v}{I_u \cup I_v} \qquad 2\text{-}6$$

Algoritmos basados en el artículo

En este caso, se predice la valoración de un usuario sobre un artículo teniendo en cuenta las valoraciones que dio en el pasado a artículos similares (vecinos).

En la ecuación 2-7 se muestra el cálculo de la predicción de rating que un usuario dará a un artículo, en la que $N_u(i)$ representará el conjunto de artículos valorados por el usuario u que son más similares al artículo *i*. Siendo w_{ij} la similitud entre los artículos y r_{uj} el rating que ha dado el usuario u al artículo j, el rating predicho $\widehat{r_{ui}}$ que u dará a i se obtendrá como un promedio ponderado de los ratings otorgados por u a los elementos de $N_u(i)$.

$$\widehat{r_{ui}} = \frac{\sum_{j \in N_u(i)} w_{ij} r_{uj}}{\sum_{j \in N_u(i)} | w_{ij} |} \qquad 2\text{-}7$$

Se podrán utilizar las mismas fórmulas que en el apartado anterior para calcular la similitud entre artículos simplemente sustituyendo los usuarios por artículos.

Por ejemplo, la similitud entre dos artículos utilizando la fórmula del coseno quedaría así:

$$\cos(i,j) = \frac{\sum_{u \epsilon U_{ij}} r_{u_i} r_{u_j}}{\sqrt{\sum_{u \epsilon U_i} r_{u_i^2} \sum_{u \epsilon U_j} r_{u_j^2}}} \quad 2\text{-}8$$

En la fórmula, el término U_i representa el conjunto de usuarios que han consumido el ítem i, U_j el conjunto de usuarios que consumieron j. Por último, U_{ij} será la intersección de todos aquellos usuarios que hayan consumido ambos artículos i y j.

2.2.1.2 Filtrado colaborativo basado en modelos

Por otro lado, el filtrado colaborativo basado en modelos es una técnica centrada en construir modelos que capturen patrones y relaciones entre usuarios e ítems para así poder generar recomendaciones personalizadas.

Para construir estos modelos, se utilizarán técnicas y algoritmos de aprendizaje automático como las redes neuronales, clasificadores bayesianos, árboles de decisión o modelos de factores latentes.

Modelos de factores latentes

Esta técnica consiste en descomponer una matriz de interacciones usuario-elemento en dos matrices de menor tamaño, llamadas de factor latente, que al multiplicarse se aproximen a la matriz original de interacciones. Estas matrices, capturarán las características o factores latentes ocultos de los usuarios y los ítems. Las matrices tienen k factores latentes, cuantos más factores existan, más personalizadas serán las recomendaciones, pero más aumentará el riesgo de que haya overfitting en el modelo [6].

Dado un conjunto de usuarios U, un conjunto de ítems I y una matriz R de dimensiones |U| x |I| que contiene los ratings que los usuarios les han dado a los ítems, el objetivo de la factorización de matrices será encontrar dos matrices P y Q cuyo producto se aproxime a R. En las columnas de estas matrices se encontrarán los factores latentes, que definirán a los usuarios en el caso de la matriz P, y a los artículos en el caso de Q. Por tanto, una fila de la matriz Q representará el grado de asociación de un usuario con esas características [8].

$$R \approx P \, x \, Q^T \qquad 2\text{-}9$$

Una vez descompuestas las matrices, se podrá encontrar el score que un usuario dará a un ítem calculando el producto escalar entre los vectores que corresponden al usuario y al artículo. Por tanto, para un usuario *u*, se podrá predecir la valoración que dará a un ítem *j* de la siguiente manera:

$$r_{ij} = p_i^T q_j \qquad 2\text{-}10$$

Para descomponer la matriz inicial se utilizarán técnicas como la descomposición en valores singulares (SVD) que permitirá encontrar las matrices de factor latente optimizando una función de pérdida como el error cuadrático medio existente entre las interacciones reales y las predicciones.

Después de calcular el error, se optimizarán los parámetros de los vectores de usuarios y artículos utilizando técnicas como el descenso de gradiente o los mínimos cuadrados alternantes [9].

Los ganadores del *Netflix Prize*, propusieron un modelo basado en factorización de matrices, concluyendo que, para esos datos, este tipo de modelos funcionaban mejor que los basados en vecinos. Gracias a esta solución propuesta, se convirtieron en estado del arte los sistemas de recomendación, integrando datos adicionales como información temporal o geográfica en el caso de la recomendación de puntos de interés.

Matriz inicial: interacción usuario-ítem

	POI 1	POI 2	POI 3
User 1	1.37	0.7	?
User 2	1.53	?	1.32
User 3	?	2.07	2.76

≈

Sub-Matriz usuario

	Característica 1	Característica 2
User 1	0.2	0.5
User 2	0.3	0.4
User 3	0.7	0.8

X

Sub-Matriz ítems traspuesta

	POI 1	POI 2	POI 3
Característica 1	3.1	2.5	1.2
Característica 2	1.5	0.4	2.4

Ilustración 2-1 Factorización de matrices. Fuente: Elaboración propia

En la ilustración 2-1 podemos ver un ejemplo que demuestra el resultado de la factorización de una matriz inicial de ratings. Al utilizar esta técnica, se consigue reducir la dimensión de la matriz inicial dispersa para crear dos sub-matrices más fáciles de computar. Además, es fácilmente escalable para ser utilizada en grandes volúmenes de datos.

Modelos de redes neuronales

Las redes neuronales permiten encontrar relaciones no lineares en los datos y así encontrar patrones ocultos en ellos. Una red neuronal estará compuesta de neuronas que se ubiquen en las distintas capas de la red (capa de entrada, capas ocultas, capa de salida). Estas redes pueden entrenarse utilizando los datos históricos de la interacción de los usuarios para así generar recomendaciones.

En los últimos años, se han hecho muchas propuestas para mejorar los sistemas de recomendación utilizando redes neuronales, por ejemplo, se han empleado las redes recurrentes para los sistemas de recomendación basados en sesiones [10]. Estos sistemas tratarán de predecir la siguiente acción de un usuario dado su historial de acciones en la sesión actual, de este modo, ofrecen recomendaciones personalizadas en tiempo real, mejorando así la experiencia del usuario y aumentando la probabilidad de que encuentre contenido relevante y de interés.

En [11] se presenta un modelo que, utilizando las redes neuronales convolucionales (CNN Convolutional Neural Networks), tratará de predecir los top-N artículos con los que el usuario interactuará en un futuro cercano. Por otro lado, en [12] se presentaban nuevas funciones de pérdida adaptadas a las RNNs (Recurrent Neural Networks) en el contexto de los sistemas de recomendación. Estas funciones hicieron que dichos modelos tuvieran un mayor rendimiento con respecto a los de filtrado colaborativo, llegando a alcanzar mejoras de hasta el 53% en métricas como el Recall.

No obstante, aunque los modelos basados en redes neuronales han experimentado un gran crecimiento en los últimos años, algunos estudios alertan que estos modelos podrían obtener un rendimiento peor que los modelos clásicos basados en reglas heurísticas como los de vecinos próximos [13] .

2.2.2 FILTRADO BASADO EN CONTENIDO

Este tipo de sistemas de recomendación recomendarán al usuario artículos que son similares a los que valoró positivamente en el pasado. Se establecerá un perfil del usuario basado en las características de los artículos que el usuario haya consumido anteriormente. Por ejemplo, en el caso de la recomendación de películas o libros, se analizarán las categorías de los artículos que los usuarios han consumido y se compararán con las categorías del artículo objetivo para determinar si hay coincidencias [14].

Estos sistemas tienen una implementación sencilla y son muy efectivos para productos con contenido explícito, que tengan atributos muy claros y definidos como las películas, la música o los libros [15]. Algunos ejemplos de los mecanismos utilizados en estos sistemas son el filtrado basado en características, el análisis de contenido textual o las recomendaciones basadas en el historial del usuario.

Un ejemplo de aplicación podría ser una plataforma de reproducción de música como Spotify, la cual generará sugerencias basándose en el tipo de música que el usuario ya ha escuchado. Por ejemplo, si un usuario escucha regularmente jazz, el algoritmo de Spotify podría sugerirle canciones o listas del mismo género.

2.2.3 HÍBRIDOS

Los sistemas de recomendación híbridos combinan las diferentes técnicas de recomendación con el objetivo de obtener un mejor resultado disminuyendo las limitaciones que tienen dichas técnicas cuando actúan por separado.

Según [16] se han identificado distintos tipos de recomendadores híbridos:

- **Conmutado:** el sistema elegirá entre diferentes componentes de recomendación y aplicará el seleccionado.

- **Por pesos:** varios recomendadores generan scores para artículos candidatos, que luego se combinan numéricamente. Si varios recomendadores proponen el mismo artículo, se calcula su score como una combinación lineal. Los pesos para esta combinación se suelen determinar mediante pruebas empíricas.

- **Mixto:** cada componente presentará su propia lista ordenada de recomendaciones. Estos rankings propuestos se integrarán basándose en factores como la confianza en cada recomendador, creando así una lista final de recomendaciones.

- **Combinación de propiedades:** las características derivadas de diferentes fuentes de conocimiento se combinan y se entregan a un solo algoritmo de recomendación.

- **Aumento de características:** Se utiliza una técnica de recomendación para calcular una característica o conjunto de características. Después se pasan estas características como entrada a otro recomendador.

- **Cascada:** se le asignará una prioridad a los diferentes recomenadores. Aquellos que cuenten con menor prioridad resolverán los empates en la puntuación.

2.2.4 OTROS ENFOQUES DE SISTEMAS DE RECOMENDACIÓN

Además de los mencionados anteriormente, existen otros tipos de sistemas de recomendación:

1. **Basados en conocimiento (knowledge-based)**

   Estos sistemas de recomendación utilizarán el conocimiento adquirido sobre los usuarios y los productos para generar recomendaciones teniendo en cuenta qué productos cumplen con los requisitos del usuario. Estos sistemas utilizan una función de utilidad que estimará cuanto necesita un usuario un producto en concreto. Su principal ventaja es que evitan problemas como el de arranque en frío ya que sus recomendaciones no están basadas en las valoraciones del usuario para los distintos artículos.

   Esto los convierte en sistemas muy valiosos tanto para trabajar individualmente como para complementar a otros sistemas de recomendación [17]. Estos sistemas son utilizados principalmente en dominios de los que no se dispone apenas de información sobre ratings de los usuarios, como es el caso del mercado inmobiliario o automovilístico. Además, en este tipo de escenarios, el sistema debe ser capaz de ajustar sus recomendaciones teniendo en cuenta posibles restricciones impuestas por el usuario como por ejemplo la fijación de un presupuesto máximo.

2. **Demográficos**

   Estos sistemas de recomendación clasificarán a los usuarios en clases demográficas teniendo en cuenta los atributos del usuario, como pueden ser su localización, edad, género u otra información demográfica que les haga formar parte de un grupo [18]. Por ejemplo, si un usuario está localizado en una ciudad costera, el sistema de recomendación podrá sugerirle destinos de playa o actividades acuáticas populares en su ubicación.

2.3 RECOMENDACIÓN DE PUNTOS DE INTERÉS

Un punto de interés o POI (Point of Interest) es una ubicación específica que puede resultar de interés para los visitantes de un lugar. Algunos ejemplos de puntos de interés pueden ser áreas recreativas, lugares de interés histórico, atracciones turísticas o comercios y restaurantes.

Los sistemas de recomendación basados en puntos de interés proporcionarán al usuario sugerencias sobre lugares que visitar cuando llega a una ciudad, basándose en información extraída sobre los requerimientos y preferencias del usuario.

Las redes sociales basadas en localización o LBSNs (Location-Based Social Networks) como Gowalla, Foursquare, Yelp o BrightKite, permiten al usuario registrar check-ins cuando visita un punto de interés. Estas redes sociales permiten obtener grandes conjuntos de datos de los que se puede extraer información útil que servirá para generar futuras recomendaciones. En el ámbito de los sistemas de recomendación basados en puntos de interés (POIs), la información geográfica adquiere una relevancia fundamental a la hora de proporcionar recomendaciones personalizadas a los usuarios. Estos sistemas consideran la ubicación física tanto de los usuarios como de los distintos POIs con el objetivo de ofrecer recomendaciones más precisas, ya que es común que los usuarios tiendan a visitar lugares de interés cercanos entre ellos.

Existen algunas particularidades que según [19] diferencian a la recomendación de puntos de interés de la recomendación clásica:

- **Sparsity o dispersión:** En este contexto encontramos una gran escasez de datos disponibles sobre las interacciones de los usuarios. Usualmente, los usuarios únicamente han interactuado con un pequeño subconjunto de POIs, resultando en una

matriz de interacciones muy dispersa. Esto dificulta hacer recomendaciones precisas y genera sesgo hacia la recomendación de elementos populares. Por ejemplo, en el conjunto de datos utilizados en el premio de Netflix encontrábamos una dispersión del 99.8% mientras que en los datos de Gowalla o Foursquare, utilizados en este trabajo, encontramos alrededor de un 99.997% de dispersión.

- **Información implícita:** En los sistemas de recomendación clásicos, la matriz de interacciones se rellena utilizando los ratings que los usuarios proporcionaron. Sin embargo, las redes sociales basadas en localización únicamente proporcionarán datos sobre los check-ins de los usuarios. Estos check-ins solo contendrán datos sobre el momento en que un usuario visita un punto de interés, pero no incluirán una valoración explícita del lugar por parte del usuario. Como un usuario podrá haber visitado el mismo POI varias veces, una solución común para generar un rating será utilizar como puntuación el número de veces que un usuario haya visitado un POI, esto es conocido en el área como matriz de frecuencias.

- **Influencias externas:** En los sistemas de recomendación clásicos la única información utilizada para generar información es la matriz de interacciones del usuario. Por el contrario, en el ámbito de la recomendación de puntos de interés, resulta muy útil incluir otros factores como la localización geográfica, el factor temporal o la influencia social a la hora de generar recomendaciones. En concreto, la información geográfica es muy importante en la recomendación de puntos de interés, aunque existen distintas formas de incorporar esta información a los algoritmos de recomendación, todas las técnicas coinciden en una regla básica: cuanto menor sea la distancia, mayor será la relación existente [20].

2.4 *EVALUACIÓN DE LOS SISTEMAS DE RECOMENDACIÓN*

La evaluación de los sistemas de recomendación es necesaria durante todo el ciclo de vida de estos. Con ella, se busca comprobar que el algoritmo de recomendación funciona correctamente y cumple los fines para los cuales fue diseñado.

Existen tres tipos de experimentos fundamentales, los offline, en los que se comparan algoritmos sin necesidad de interacción del usuario, los estudios de usuario (user studies) donde una pequeña muestra experimenta con el sistema y proporciona feedback, y por último, los experimentos online, que se realizan cuando el sistema ya ha sido lanzado y poblaciones de usuarios reales interactúan con el sistema.

2.4.1 ESTUDIO DEL USUARIO

Este experimento consiste en tomar una pequeña muestra de la población que experimentará con el sistema y proporcionará feedback en base a su experiencia. Se utilizará cuando sea difícil crear una simulación de las interacciones reales que tendrían los usuarios con el sistema. Una ventaja de este tipo de evaluación es que proporciona retroalimentación real y directa de los usuarios en base a su experiencia con el sistema de recomendación, permitiendo identificar problemas o mejoras que no hubieran sido capturados en una simulación. Por otro lado, existe la posibilidad de que las opiniones y preferencias de los participantes escogidos para el estudio no sean representativas y por ello no reflejen la diversidad de la población total.

2.4.2 EVALUACIÓN ONLINE

En este caso, se evaluará el rendimiento del sistema de recomendación en tiempo real mientras hace recomendaciones a usuarios reales. Se analizarán distintas métricas como la tasa de clicks en los artículos (CTR o Click-Through Rate), o la tasa de engagement. Esta evaluación es necesaria para analizar la calidad de las recomendaciones hechas al usuario y poder ajustar y mejorar el rendimiento del sistema.

2.4.3 EVALUACIÓN OFFLINE

Consiste en realizar una comparación de los resultados obtenidos al pasarle el mismo conjunto de datos a distintos algoritmos. Suele ser utilizada en la fase de diseño del sistema cuando se debe decidir cuál es el mejor enfoque. Para que los experimentos offline se consideren útiles, deben contar con las siguientes características:

- ✓ Se utilizarán conjuntos de datos previamente recolectados sobre usuarios que valoran distintos ítems. Utilizando estos datos podemos simular el comportamiento de los usuarios que interactuarán con el sistema.

- ✓ Es requerido tomar una muestra representativa de la población donde se instalará el sistema puesto donde se asumirá, que el comportamiento que los usuarios mostraron en el momento de recoger la muestra será similar al que mostrarán cuando el sistema se lance.

- ✓ Es una opción de bajo coste, pues no requiere que usuarios reales interaccionen con el sistema, con ella, por ello solo se podrán contestar cuestiones relacionadas con el poder predictor del algoritmo.

- ✓ Sigue el estándar clásico de los modelos de Machine Learning, se dividirá el conjunto de datos en entrenamiento, validación y test.

Inicialmente, la calidad de los recomendadores se evaluaba según su capacidad predictiva, es decir, la capacidad para predecir con precisión las elecciones del usuario. Sin embargo, en muchas aplicaciones, los usuarios que interactúan con un sistema de recomendación valoran distintas propiedades como por ejemplo descubrir nuevos artículos, mantener su privacidad u obtener una variedad diversa de recomendaciones [21].

En el contexto de los sistemas de recomendación basados en puntos de interés, evaluaremos los algoritmos de recomendación basándonos en tres propiedades: relevancia, novedad y diversidad.

- **Relevancia**: Se refiere a la calidad de las recomendaciones ofrecidas por el sistema, es decir, la capacidad que tiene el sistema para recomendar elementos al usuario que le hayan gustado.

- **Novedad**: Consideraremos novedad como recomendar al usuario artículos poco populares. Es decir, que no sean conocidos por la mayoría de los usuarios.

- **Diversidad**: Se refiere a la capacidad del sistema de recomendación para sugerir elementos que sean variados y con distintas características o categorías, por lo que el sistema de recomendación debe crear recomendaciones que cubran todo el espectro de ítems disponible.

En el ámbito de los sistemas de recomendación se distinguen dos categorías a la hora de medir la precisión de las recomendaciones, la predicción de ratings y la predicción de rankings [22].

Inicialmente, para evaluar la precisión de los sistemas de recomendación enfocados en la predicción de valoraciones, se utilizaban métricas como el RMSE (Root Mean Square Error) o el MAE (Mean Absolute Error). Con ellas, se mide la diferencia entre las valoraciones predichas por el sistema y las valoraciones proporcionadas por los usuarios en el conjunto de test. De este modo, cuanto menor sea el valor del RMSE o MAE, mejor será la precisión del sistema de recomendación.

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{(u,i)\in T}(\widehat{r_{ui}} - r_{ui})^2} \qquad 2\text{-}11$$

$$MAE = \sqrt{\frac{1}{|T|}\sum_{(u,i)\in T}|\widehat{r_{ui}} - r_{ui}|} \qquad 2\text{-}12$$

Aunque en el premio de Netflix, se utilizaron las métricas mencionadas anteriormente, posteriormente se produjo un cambio en la forma de evaluar los recomendadores, lo que llevó a la implementación de sistemas de recomendación de rankings o predicción de top-N. Esto se debió a que recibir un conjunto de buenas recomendaciones brindaba una experiencia superior para el usuario comparada con la predicción de ratings particulares.

Los sistemas basados en rankings se enfocan en predecir el orden en el que el usuario calificaría un grupo de ítems, generando una lista de n recomendaciones a cada usuario. De este modo, se trata de predecir la preferencia relativa que tiene el usuario por esos ítems. Para medir la precisión de estos sistemas se implementan métricas como el Recall.

En primer lugar, se definirán tres métricas que serán calculadas para cada usuario, para obtener datos sobre el performance del algoritmo se hará la media del valor obtenido para todos los usuarios. Estas métricas son la precisión, el recall y el EPC.

Precisión

Es una métrica de relevancia que mide la eficacia del recomendador al proporcionar recomendaciones relevantes al usuario. La precisión se determina al medir la proporción de las recomendaciones que fueron relevantes entre todas las recomendaciones hechas por el modelo. Para ello, se medirá la exactitud de las recomendaciones generadas en relación con las preferencias del usuario. Por lo tanto, si un sistema tiene una precisión alta, significará que las recomendaciones que proporciona son útiles y relevantes para el usuario.

$$\text{Precisión@}k = \frac{1}{|U|}\sum_{u \in U}\frac{|L(u)_k \cap T(u)|}{k} \qquad \textit{2-13}$$

- k: Cut off, será el número de recomendaciones que se tendrán en cuenta de entre todas las proporcionadas por el sistema. Cuando se hacen recomendaciones a un usuario, generalmente se le proporcionará una lista pequeña con 5 o 10 artículos para evitar sobresaturarle de información.
- U: Número de usuarios recomendados (generalmente, los usuarios en el conjunto de test sobre los que los recomendadores pueden efectuar sugerencias).
- $L(u)_k$→ top k ítems recomendados por el sistema al usuario de destino U. (Serán los k ítems recomendados con mayor score)
- T(u) → artículos que el usuario de test valora como relevantes, en este caso serán los POIs que el usuario visitó en test, es decir, los que tendrían que haber sido recomendados.
- Threshold → Usualmente se emplea un valor de threshold que permite indicar si un artículo es relevante o no en test- Este valor representa la nota que le dio el usuario al artículo. De esta forma, se considerarán relevantes solo aquellos artículos cuya nota supere un determinado valor umbral.

Recall

Esta métrica de relevancia medirá la capacidad del modelo para sugerir elementos relevantes al usuario. Para ello, se calculará la proporción de recomendaciones relevantes que el sistema fue capaz de recuperar en relación con todos los elementos que el usuario consideró relevantes.

$$Recall = \frac{1}{|U|}\sum_{u \in U}\frac{|L(u)_k \cap T(u)|}{|T(u)|} \qquad 2\text{-}14$$

Donde cada una de las componentes de la fórmula anterior ya han quedado explicadas anteriormente.

Expected popularity complement (EPC):

Esta métrica medirá cómo de novedosas son las recomendaciones hechas al usuario. Recordemos que la novedad se suele interpretar como artículos que son menos conocidos por la comunidad (menos populares). Para ello, se calculará el número de veces que ha sido visitado un punto de interés por usuarios distintos. Si un punto de interés ha sido visitado muchas veces será considerado como popular, en el caso contario, se considerará como novedoso. En la ecuación, la popularidad de un artículo se calculará dividiendo el número de usuarios que han puntuado un artículo $|U_i|$ entre el número de usuarios totales del sistema $|U|$.

$$EPC = \frac{1}{\mathrm{k}}\sum_{i \in R(u)_k}\left(1 - \frac{|U_i|}{|U|}\right) \qquad 2\text{-}15$$

En su formulación original, el expected popularity complement permitía no sólo tener en cuenta la relevancia de las recomendaciones, sino también la posición en el ranking, de forma que cuanto más altas estén en las posiciones en el ranking, mejor [23].

Como se mencionó anteriormente, las métricas anteriores son calculadas para cada usuario del conjunto de prueba para luego calcular la media entre todos. Por el contrario, el Aggregate Diversity y User Coverage son métricas que se calcularán para el recomendador, no a nivel de usuario.

Aggregate diversity

Como su nombre indica, esta métrica medirá la diversidad que tienen los artículos sugeridos por un sistema de recomendación. En esta forma de Aggregate Diversity se mide como el tamaño del conjunto de todos los elementos que un sistema de recomendación ha podido recomendar en su totalidad a sus usuarios [24].

$$aggregate\ diversity\ = \left| \bigcup_{u \in U} R_u \right| \qquad 2\text{-}16$$

User coverage

Medirá a cuantos usuarios de todos los posibles se ha podido proporcionar recomendaciones. Puede ocurrir que el sistema no haya podido sugerir artículos si, por ejemplo, en un algoritmo de vecinos próximos no se han encontrado vecinos para usuario en cuestión. Siendo U el

conjunto de usuarios a los que se les han podido efectuar recomendaciones, el user coverage se representará como la cardinalidad de este conjunto.

$$User\ Coverage\ = |U_{rec}| \qquad 2\text{-}17$$

A mayor resultado obtenido en todas las métricas descritas, los recomendadores estarán obteniendo valores más altos. (Más acierto, novedad, diversidad y cobertura).

2.5 DOMINIO CRUZADO (CROSS-DOMAIN)

Un dominio es un campo de pensamiento, actividad o interés particular [25]. Los dominios cruzados consisten en transferir conocimiento entre dominios considerando que existirá un solapamiento de información de ítems y usuarios. En el contexto de la recomendación, donde se enfrenta a problemas de baja densidad de datos y falta de conocimiento, resulta útil explorar la posibilidad de combinar información proveniente de diferentes dominios para poder paliar la escasez de datos de la matriz de interacciones y así mejorar las recomendaciones.

El dominio cruzado o cross-domain consiste en utilizar información proveniente de distintas fuentes para incrementar el rendimiento de un sistema. Es una solución para abordar problemas como el de arranque en frío. Al introducir nuevos datos de otras fuentes al sistema, se aumentará la densidad de ratings disponible, incrementando la precisión de las recomendaciones hechas. También, recomendar ítems empleando información de distintos dominios ofrece un valor añadido a las recomendaciones, aportándoles diversidad o novedad.

Según [26] , existen distintos niveles de dominio cruzado:

- ✓ **Nivel de atributo**
  En este caso, se trataría del mismo tipo de ítems pero que toman valores diferentes en cierto atributo. Un ejemplo podría ser el género de una película dentro de una plataforma de streaming. (Comedia – Thriller).

- ✓ **Nivel de tipo ítem**
  Los artículos tendrán tipos similares y compartirán algunos atributos. Por ejemplo, las películas y las series, aunque pertenezcan a distintos dominios, comparten la mayoría de sus atributos.

- ✓ **Nivel ítem**
  A este nivel, los artículos no son del mismo tipo, por lo que diferirán en todos o en la mayoría de sus atributos. Por ejemplo, una asociación entre películas y restaurantes.

- ✓ **Nivel de sistema**
  En este caso, los artículos recomendados pertenecerán a distintos dominios. Se tratará del mismo tipo de ítems, pero almacenados de distinta forma o mediante distintas operaciones.

Por otro lado, existen muchas taxonomías a la hora de clasificar las técnicas de cross-domain. En este caso las categorizaremos según la forma que tienen de explotar el conocimiento [26]:

- ✓ **Técnicas de agregación de conocimiento**

  Podemos distinguir entre tres casos de uso: fusión de preferencias del usuario, mediación de datos de modelado de usuario y combinación de recomendaciones.

- ✓ **Técnicas de transferencia de conocimiento entre dominios**
  Se distinguirá en tres variantes: transferencia de patrones de datos, transferencia de factores latentes y dominios conectados.

En este trabajo, se desarrollará el nivel de intercambio de dominio de sistema ya que se utilizarán datos sobre check-ins obtenidos de distintas fuentes y que por ello estarán almacenados con distinto formato.

Por otro lado, se empleará la agregación de conocimiento en un escenario en el que se superpondrán los elementos, pero no los usuarios. Es decir, se buscará establecer una correspondencia entre los puntos de interés de los check-ins de diferentes conjuntos de datos, pero no se considerará la coincidencia de usuarios.

En resumen, se utilizará la información de diversas redes sociales basadas en la localización para evaluar si aplicando estas técnicas y enriqueciendo así los datos en un conjunto de destino, se pueden proporcionar mejoras en el rendimiento de los sistemas de recomendación.

Capítulo 3. DISEÑO

3.1 PIPELINE DE LA APLICACIÓN

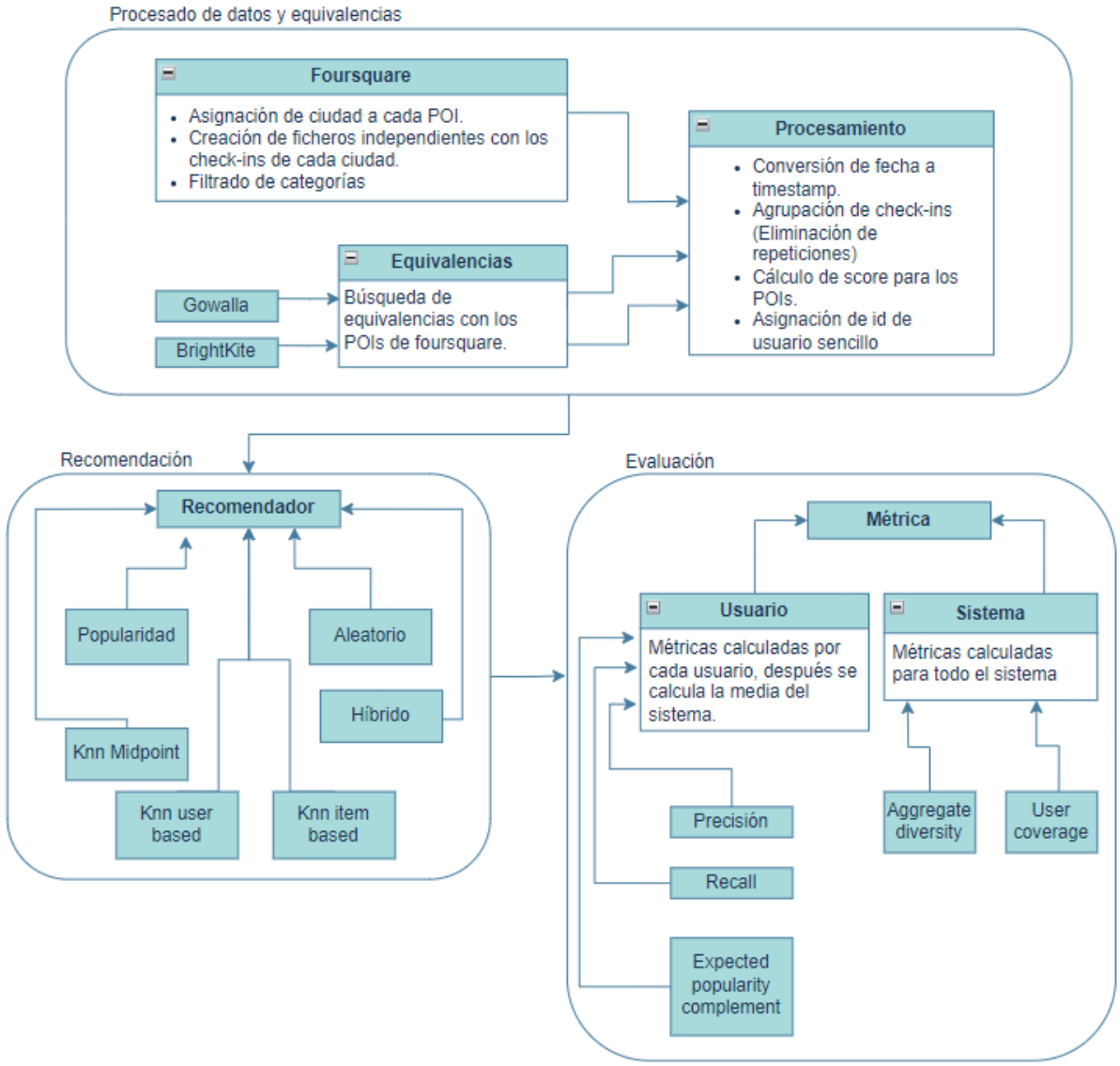


Ilustración 3-1 Esquema de la aplicación

La aplicación está desarrollada en el lenguaje Python y se encuentra dividida en tres partes: procesado de datos, recomendación y evaluación.

1. **Procesado de Datos:** En esta parte del código, se llevará a cabo el procesamiento de los datos provenientes de diferentes fuentes (diferentes conjuntos de datos). También se generarán los conjuntos de entrenamiento, validación y prueba que serán utilizados en la etapa de recomendación.

    a. **Equivalencias:** Se buscarán las equivalencias entre los POIs de Foursquare y los POIs de los conjuntos de datos de las otras LBSN. Para ello, se calculará la distancia de Haversine entre cada POI del nuevo conjunto y cada POI de Foursquare. Si la distancia obtenida es menor a 10 metros, se considerará que se trata del mismo punto de interés y se añadirá la equivalencia a un fichero intermedio. Después, utilizando este fichero se complementará el conjunto de train con los check-ins de los conjuntos de datos adicionales.

2. **Recomendación:** recibe los conjuntos de datos proporcionados por la fase anterior y genera recomendaciones utilizando los distintos algoritmos según lo que se le solicite, creando archivos con las recomendaciones generadas. En todos los casos, las recomendaciones propuestas al usuario estarán compuestas por POIs que no haya visitado previamente en el conjunto de entrenamiento.

3. **Evaluación:** este módulo recibe los archivos con las recomendaciones generadas y realiza una evaluación basándose en distintas métricas, tanto de relevancia, novedad y diversidad, para así obtener una visión completa del rendimiento de cada recomendador.

3.2 *DATOS EMPLEADOS*

Se utilizaron conjuntos de datos procedentes de cuatro redes sociales basadas en localización: Foursquare, Gowalla, BrightKite y Yelp. En la Tabla 1 se proporciona información sobre dichos conjuntos. Se decidió utilizar Foursquare como dominio principal dado que cuenta con el mayor número de usuarios, POIs y check-ins.

Fuente	Nº usuarios	NºPOIs	NºCheck-ins	Año	Sparsity (%)
Foursquare	266.909	3.680.126	33.278.683	2012-2013	99,996612
Gowalla	196.585	1.280.969	6.442.892	2009-2010	99,9974415
Yelp	30.887	18.995	860.888	2016	99,8532657
BrightKite	58.227	772.967	4.747.287	2008-2010	99,9894522

Tabla 1 Información sobre conjuntos de datos

3.2.1 FOURSQUARE

El conjunto de datos de Foursquare fue obtenido de Dingqi YANG's Homepage - Foursquare Dataset está compuesto de 3 ficheros, con información sobre las ciudades, los puntos de interés y los check-ins, respectivamente.

Como se puede observar en la tabla, Foursquare es el conjunto de datos con más check-ins, además de ser la red social más empleada por la comunidad para efectuar recomendaciones. Se seleccionará este conjunto de datos para que sea el conjunto base que posteriormente será ampliado con los otros.

3.2.1.1 Preprocesado

El preprocesado de los datos es un paso fundamental en los proyectos de Machine Learning, al hacerlo garantizamos la calidad de los datos de entrada y los preparamos para que se adecúen al modelo. Así, obtendremos un conjunto de datos útil y consistente para la extracción de conocimiento.

En primer lugar, los Puntos de Interés (POIs) se asignaron a las ciudades en función de su proximidad utilizando la fórmula de distancia de Haversine. Después, se elaboró un fichero que contuviera todos los check-ins asociados a cada ciudad.

En la Ilustración 3-2 se representan las ciudades con mayor número de check-ins. En este caso, se decidió centrar el análisis en las ciudades de Nueva York y Tokio debido a su popularidad turística y la disponibilidad de una buena cantidad de datos relacionados. Tras hacer un análisis inicial obtuvimos los datos mostrados en la Tabla 2 .

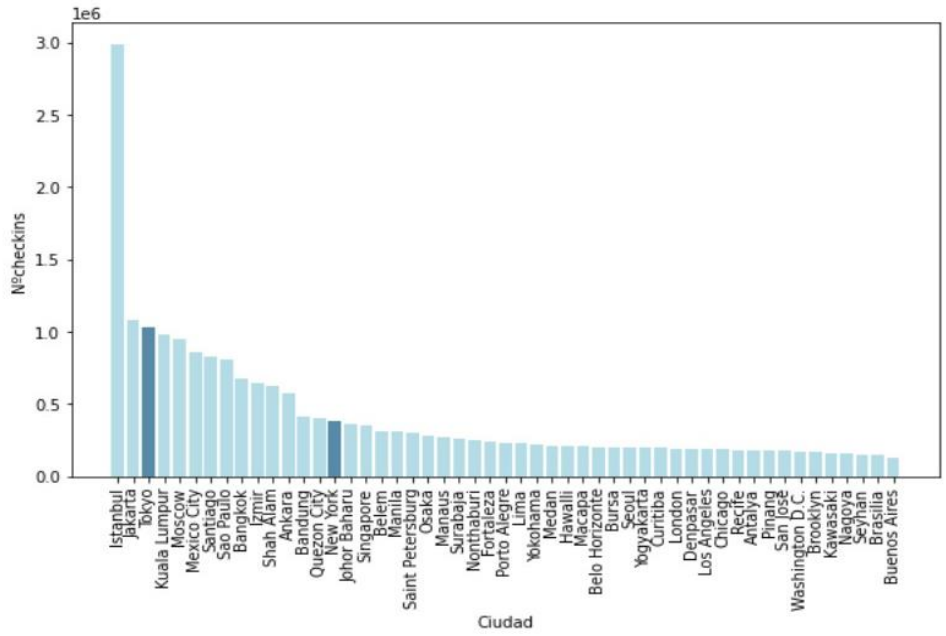


Ilustración 3-2 Número de check-ins por ciudad

En la Tabla 2 se encuentran dos distinciones para cada ciudad: "con repeticiones" y "sin repeticiones". En la distinción "con repeticiones", se considerarán todos los check-ins realizados por cada usuario, por lo que si un usuario ha visitado el mismo POIs en distintas ocasiones, existirá un check-in individual para cada visita.

Por el contrario, en la distinción "sin repeticiones", los check-ins se agruparán. En este caso, un usuario que ha visitado varias veces un lugar contará únicamente con un check-in que guardará como timestamp el de su última visita. El porcentaje de repeticiones representaba un 54% de los check-ins de Tokio y un 44,7% de los de Nueva York.

Ciudad	Nºusuarios	Nºcheck-ins	NºPOIs	Avg. check-ins /usuario	Repeticiones
Tokio	12.464	1.030.090	83.189	82,65	Con repeticiones
Tokio	12.464	474.150	83.189	38,04	Sin repeticiones
New York	15.785	380.247	41.386	24,09	Con repeticiones
New York	15.785	210.192	41.386	13,32	Sin repeticiones

Tabla 2 Análisis estadístico datos iniciales Foursquare

En las siguientes figuras, podemos ver la distribución de los usuarios en función del número de check-ins que han registrado sin tomar en cuenta las repeticiones. Se decidió filtrar los datos seleccionando únicamente los usuarios que hubieran realizado más de 5 check-ins y los POIs que hubieran sido visitados al menos 5 veces.

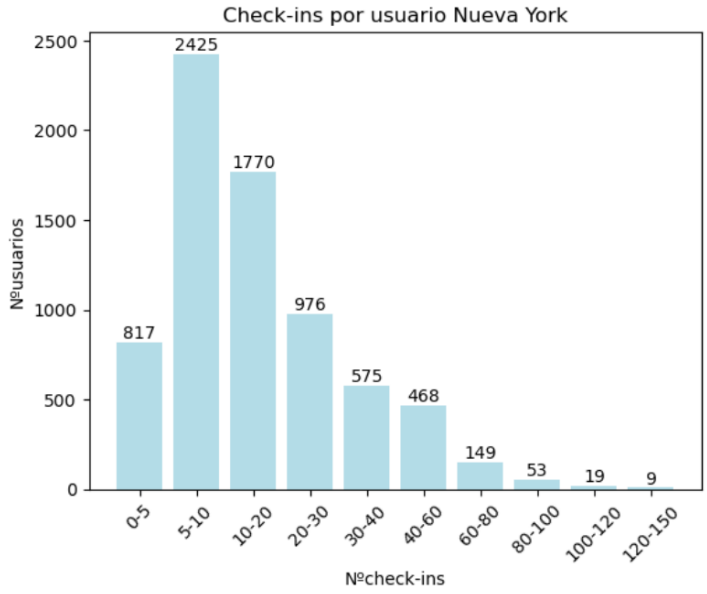


Ilustración 3-3 Distribución check-ins por usuario Nueva York

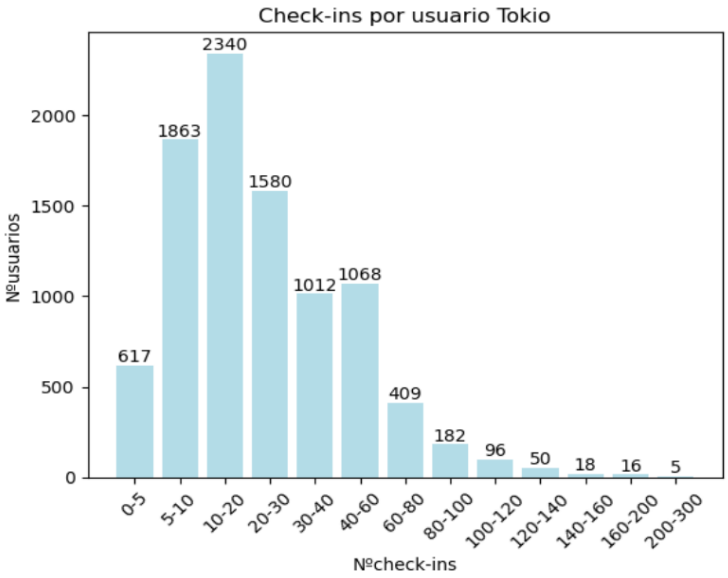


Ilustración 3-4 Distribución check-ins por usuario Tokio

También se filtró por categorías, seleccionando únicamente aquellas que pudieran ser interesantes a la hora de hacer recomendaciones turísticas a un usuario (Se eliminaron aquellas categorías correspondientes a casas privadas, hospitales, carreteras, oficinas, etc.). En las siguientes figuras podemos ver las categorías más y menos populares de cada ciudad.

Distribución de categorías Nueva York

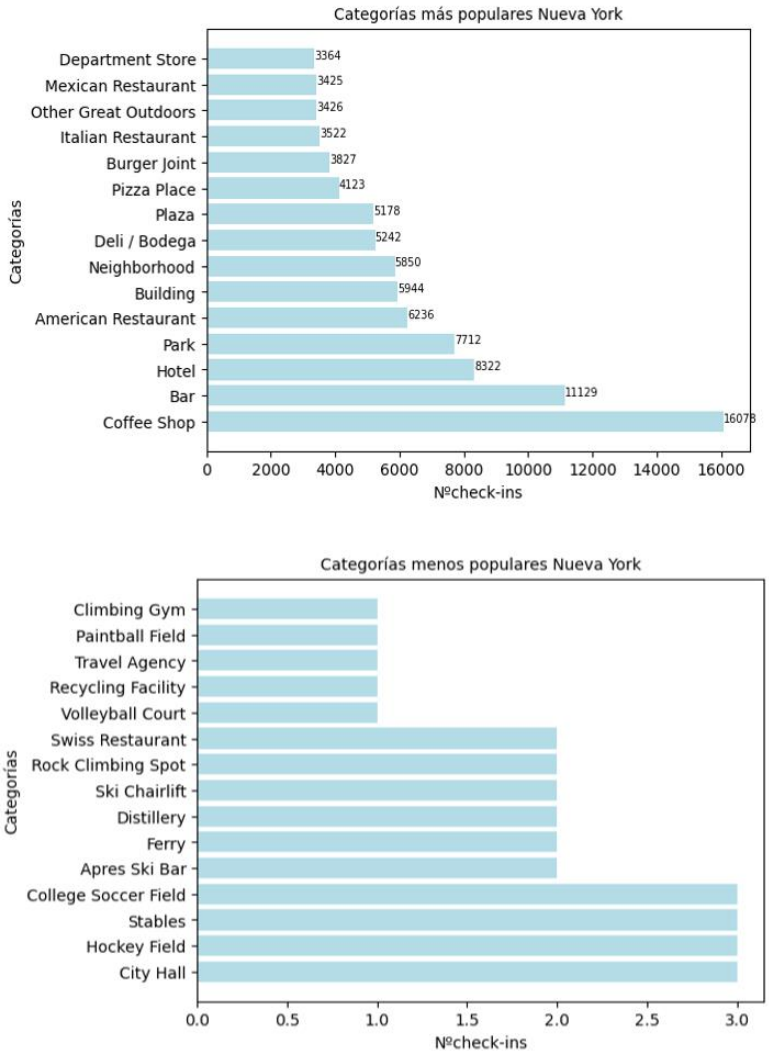


Ilustración 3-5 Categorías Nueva York

Distribución de categorías Tokio

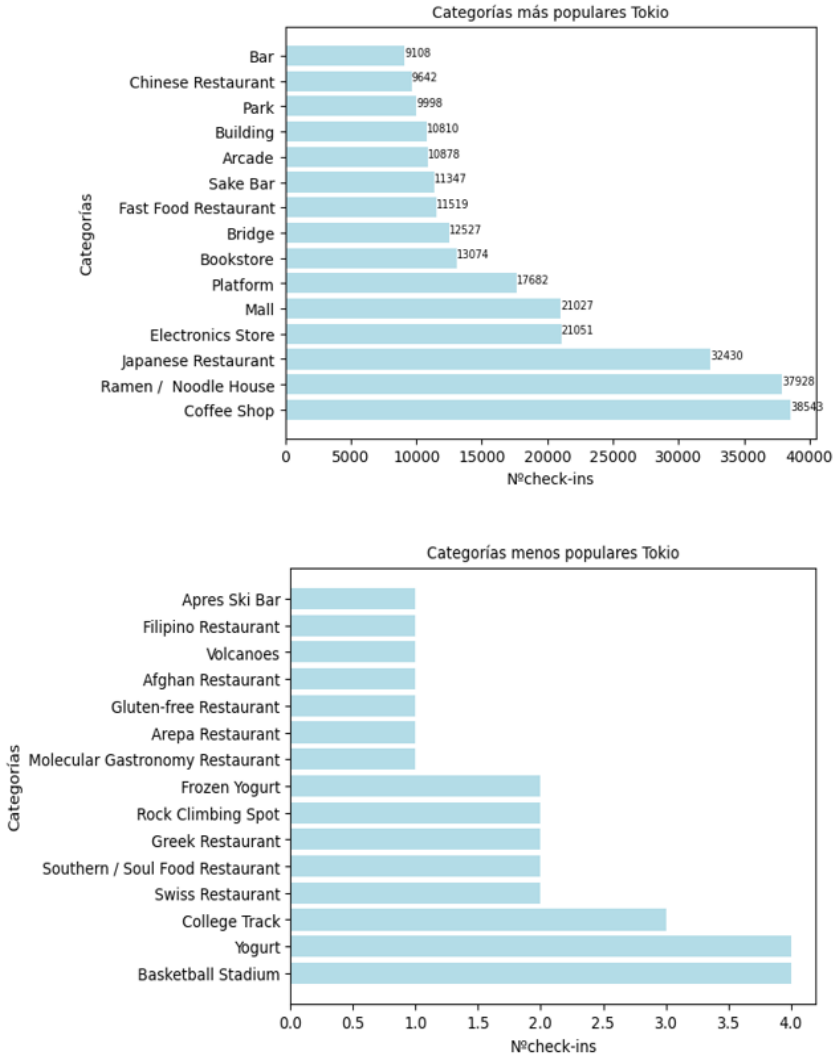


Ilustración 3-6 Categorías Tokio

Para simplificar el conjunto de datos, se sustituyeron los identificadores de los POIs por identificadores numéricos y se modificó el formato de fecha a timestamp, para que fuese más sencillo y rápido efectuar las recomendaciones.

Por último, se agruparon los check-ins, es decir, si un usuario había visitado más de 1 vez un lugar, se agruparían todos los check-ins en uno, tomando como TimeStamp el de la última visita. En la

Tabla 3 se muestra el mismo análisis estadístico realizado anteriormente, pero utilizando los datos finales.

Ciudad	**Nºusuarios**	**Nºcheck-ins**	**NºPOIs**	**Avg. check-ins /usuario**
Nueva York	7266	127606	9883	17.56
Tokio	9257	239948	19908	25.92

Tabla 3 Análisis estadístico con datos finales

Importancia de la información geográfica

- ✓ El geographic midpoint es un punto que se calcula encontrando el centro de gravedad de las coordenadas geográficas de todos los POIs que un usuario ha visitado. La latitud y longitud de cada POI se convertirán a coordenadas cartesianas que luego se multiplicarán por un factor de ponderación y se sumarán. Se puede trazar una línea desde el centro de la tierra hasta estas nuevas coordenadas x, y z. El Midpoint será la intersección entre dicha línea y la superficie de la Tierra. (Se puede ver el cálculo específico del midpoint en el anexo ii de este documento).

Se decidió analizar la influencia de la información geográfica en la recomendación de puntos de interés. En este caso se calculó la distancia media entre los check-ins de un usuario y su midpoint.

En las figuras podemos ver que las distancias calculadas suelen ser pequeñas para la mayoría de los usuarios. Esto quiere decir que los check-ins registrados por los usuarios estarán cerca unos de otros, demostrando la importancia de la influencia geográfica en este dominio de recomendación.

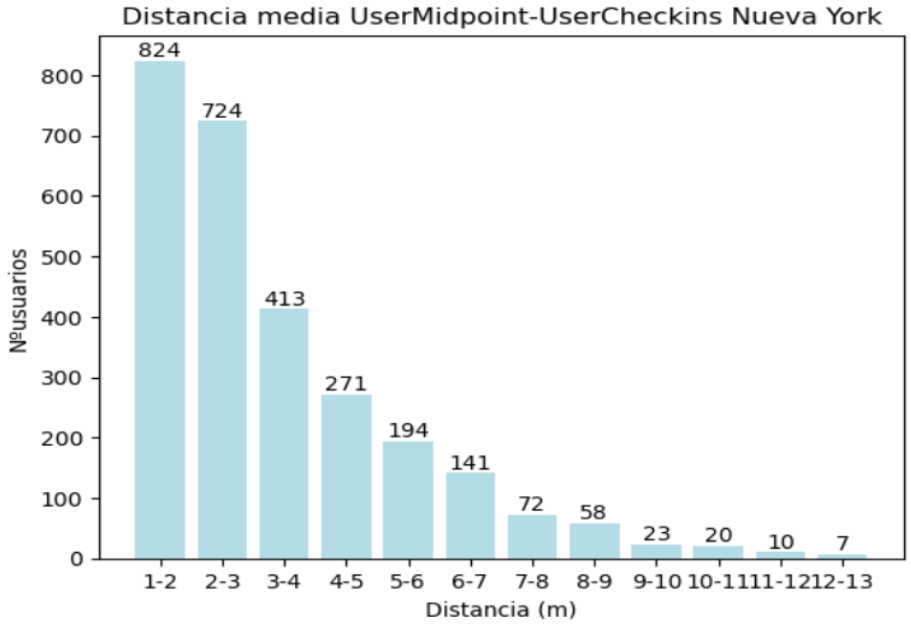


Ilustración 3-7 Distancia media UserMidpoint-User Check-ins Nueva York

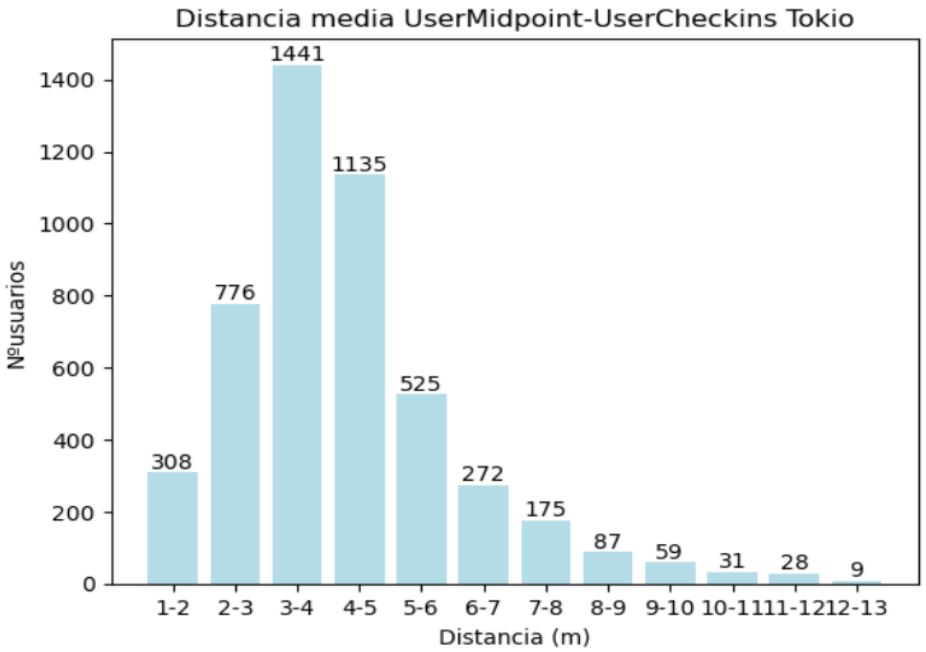


Ilustración 3-8 Distancia media UserMidpoint-User Check-ins Tokio

3.2.2 GOWALLA

Gowalla es una red social que se centra en el uso de la geolocalización para que los usuarios puedan registrar y compartir los lugares que han visitado [27]. Los datos utilizados fueron obtenidos de: SNAP: Network datasets: Gowalla (stanford.edu).

Tras calcular las equivalencias entre los POIs de Fourquare y los de Gowalla, se encontraron 2250 POIs equivalentes para Nueva York y 1483 para Tokio.

3.2.3 YELP

Yelp es una plataforma en línea que funciona como un directorio de negocios locales. Permite a los usuarios escribir reseñas y calificar su experiencia cada vez que visitan un establecimiento. Esto ayudará a futuros usuarios a tomar decisiones informadas sobre dónde ir y qué esperar a visitar un negocio [28]. Los datos de esta red social fueron obtenidos de Yelp Dataset | Kaggle y Yelp Dataset | Github.

Igual que en el caso anterior, se trató de encontrar equivalencias entre los POIs de Yelp y los de Foursquare. En este caso no encontró ninguna equivalencia para los ficheros de Nueva York y Tokio, por lo que decidió descartarse este conjunto de datos.

3.2.4 BRIGHTKITE

BrightKite fue una red social basada en localización que permitía a los usuarios hacer check-ins en los lugares que visitaban, así como de ver qué usuarios se encontraban cerca en ese momento y qué usuarios habían visitado ese lugar en el pasado. BrightKite proporcionaba una experiencia basada en la localización y permitía a los usuarios conectar con nuevos usuarios basándose en los lugares que frecuentaban.

Los datos de BrightKite fueron obtenidos de BrightKite Dataset | Snap Stanford. Al comprobar por equivalencias se obtuvieron 2.220 POIs equivalentes para Nueva York y 10.146 para Tokio.

3.3 *PARTICIONAMIENTO DE LOS DATOS*

El conjunto de datos inicial (Foursquare) se dividió en tres subconjuntos, entrenamiento, validación y test, cada uno conteniendo un 70%, 10% y 20% de los datos respectivamente.

Para hacer dicha partición, se ordenaron los registros según su Timestamp, es decir, según el momento temporal en el que se produjeron. De esta forma, los registros más nuevos quedarían en el conjunto de prueba y los más antiguos en el conjunto de train. Según se muestra en la Tabla 1, los conjuntos de datos complementarios disponibles (Gowalla y BrightKite) solo contienen información hasta el año 2010. Por esta razón, será posible incorporar los check-ins provenientes de estas fuentes al conjunto de entrenamiento.

Esta división tiene como objetivo poder evaluar el rendimiento del modelo de manera imparcial y evitar problemas como el overfitting.

El conjunto de entrenamiento será utilizado para ajustar el modelo a los datos, el de validación, se utilizará para escoger los hiperparámetros óptimos para cada modelo. Por último, el conjunto de prueba se utilizará para medir la capacidad de generalización del modelo al introducirle nuevos datos desconocidos.

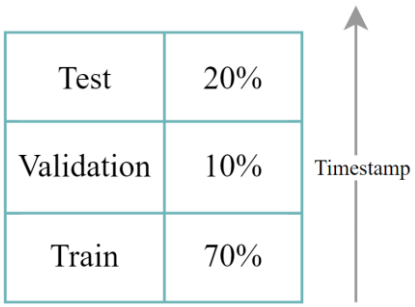


Ilustración 3-9 Particionamiento de los datos

Tras la elección de los hiperparámetros adecuados, el conjunto de entrenamiento utilizado comprenderá el 80% de los datos (Conjunto validación + Conjunto entrenamiento).

Al agregar los check-ins de Gowalla, se consiguió enriquecer el conjunto de entrenamiento para las dos ciudades, añadiendo 16.401 check-ins al fichero de Nueva York y 4.084 al de Tokio. Con los datos de BrightKite se obtuvieron 3.069 check-ins adicionales para Nueva York y 14.805 para Tokio.

Finalmente, considerando los conjuntos de entrenamiento de Nueva York y Tokio al completo, estos experimentaron un aumento del 19,07% y 9,84% respectivamente, al considerar los check-ins añadidos de Gowalla y BrightKite.

Capítulo 4. EXPERIMENTOS

4.1 ALGORITMOS DESARROLLADOS

En esta sección se explicarán los diferentes algoritmos desarrollados para la creación de sugerencias al usuario. Cada uno de ellos generará un fichero con las recomendaciones generadas para los usuarios del conjunto de prueba.

En todos los algoritmos se ha adoptado la estrategia de recomendar a los usuarios únicamente puntos de interés (POIs) que no hayan sido visitados durante el entrenamiento. En otras palabras, sin importar el algoritmo utilizado, nunca se recomendarán POIs que el usuario haya visto previamente durante el entrenamiento.

4.1.1 RANDOM

Este algoritmo, implementa un método de recomendación simple basado en la aleatoriedad. Dado un conjunto de entrenamiento y para un usuario específico, el algoritmo creará una lista de puntos de interés aleatorios que el usuario aún no haya visitado en el conjunto de entrenamiento.

Como tal, este algoritmo es un enfoque muy sencillo de recomendación ya que no tiene en cuenta las preferencias individuales. No obstante, puede resultar muy útil para identificar los valores óptimos de diversidad y novedad.

4.1.2 POPULARIDAD

Este algoritmo se apoya en la idea de que los lugares más populares serán los más interesantes para la mayoría de los usuarios, por lo que las recomendaciones de POIs hechas al usuario se calcularán basándose en la cantidad de visitas que han recibido, evitando recomendar al usuario lugares que ya haya visitado.

4.1.3 KNN BASADO EN EL USUARIO

Este algoritmo busca automatizar el "boca a boca" donde un usuario confiará en las valoraciones de usuarios que piensan como él a la hora de elegir un artículo. En este caso, se utilizó la distancia del coseno (ver ecuación 2-4) para calcular las similitudes entre usuarios. Por otro lado, para calcular los ratings se utilizó la ecuación 2-3 sin tener en cuenta el denominador.

4.1.3.1 KnnMidpoint

Este algoritmo es una variación del de KNN. En este caso, en lugar de la distancia del coseno, calcularemos el Midpoint asociado a cada uno de los usuarios utilizando las coordenadas geográficas de los puntos de interés que ya han visitado.

Utilizando la fórmula de Haversine, se calculará la distancia entre el midpoint del usuario y el de cada usuario del conjunto de entrenamiento. El score asociado a cada usuario será inversamente proporcional a dicha distancia.

Finalmente, al igual que en el algoritmo KNN clásico, se busca predecir el rating que el usuario de prueba dará a cada punto de interés ponderando las calificaciones que los usuarios vecinos han dado a ese POI en función de su similitud con ellos.

4.1.4 KNN BASADO EN EL ARTÍCULO

En primer lugar, se calculará la similitud entre todos los POIs basada en las coincidencias entre los usuarios que los visitaron. Para cada usuario, utilizando la ecuación 2-7, se calculará el rating predicho para cada uno de los POIs vecinos de los POIs que él visitó en el pasado. Finalmente, se le recomendarán aquellos POIs que tengan mayor score.

4.1.5 HÍBRIDO

El algoritmo híbrido combinará tres algoritmos distintos para generar recomendaciones al usuario. En primer lugar, se generarán recomendaciones utilizando los algoritmos popularidad, random y knnmidpoint. Se normalizarán los scores de las recomendaciones hechas por cada uno de forma que queden entre 0 y 1. Por último se seleccionarán los top n POIs con mayor score y se escribirán en el archivo de salida.

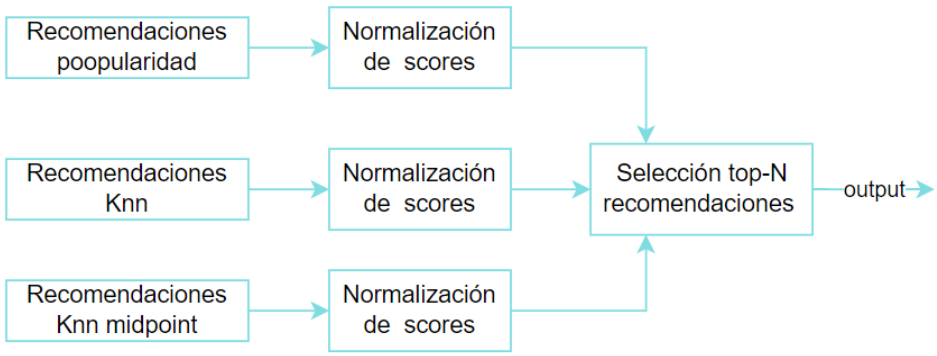


Ilustración 4-1 Estructura algoritmo híbrido

Según la taxonomía vista en la sección 2.2.3, se trataría de un sistema de recomendación híbrido por pesos dado que se combina numéricamente el score de los diferentes algoritmos de recomendación.

4.2 SELECCIÓN DE HIPERPARÁMETROS

Se utilizó el conjunto de validación para escoger los hiperparámetros óptimos. En este caso se evaluó el número de vecinos a utilizar, iterando con diferentes valores. Se entrenaron los modelos KNN user-based, KNN Midpoint y KNN item-based con esos valores en el conjunto de entrenamiento y posteriormente se utilizó el modelo para hacer predicciones sobre el conjunto de validación.

Tras completar todas las iteraciones se seleccionó el número de vecinos que proporcionaba un mayor rendimiento del sistema en términos de precisión. Después, utilizando estos parámetros se entrenará el modelo utilizando los conjuntos de entrenamiento + validación para generar recomendaciones sobre el conjunto de test.

En el caso de KNN user-based la mayor precisión se obtuvo para el número de vecinos más alto k=120. (Debido al largo tiempo de ejecución que implicaría el uso de un número mayor de vecinos y la escasa mejora resultante, no se llevaron a cabo experimentos con números más elevados de vecinos).

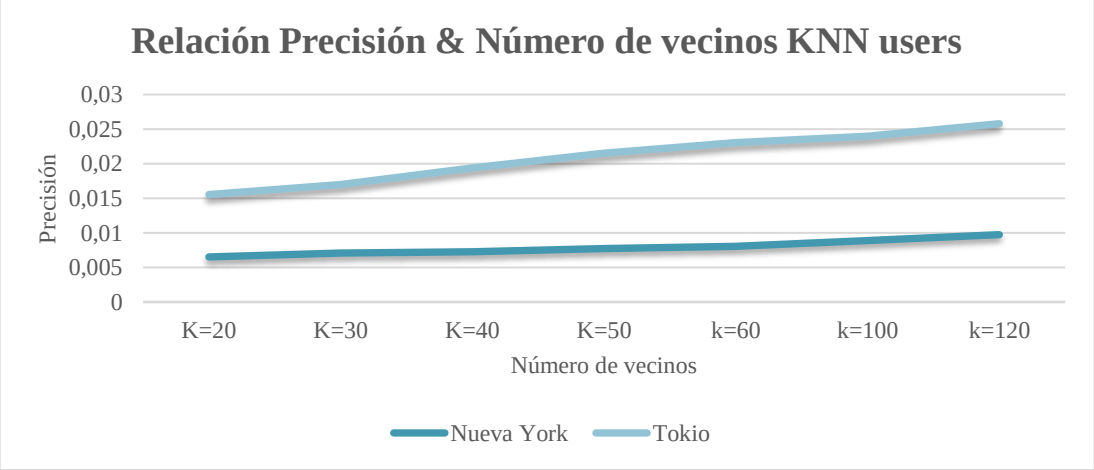


Ilustración 4-2 Evaluación del número óptimo de vecinos en Knn user-based

Para KNN Midpoint, al igual que en el caso anterior, se decidió el número de vecinos óptimos basándose en la precisión. Como se puede ver en la Ilustración 4-3, para ambas ciudades la máxima precisión se daba con el mayor número de vecinos.

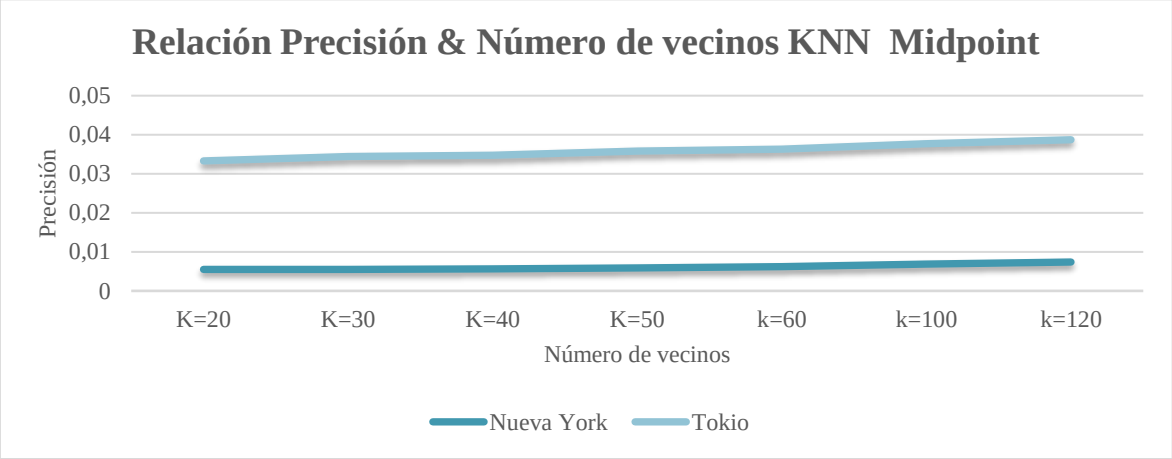


Ilustración 4-3 Evaluación del número óptimo de vecinos en KnnMidpoint

En el caso de KNN item-based, para la ciudad de Nueva York, el número óptimo de vecinos fue k=20 y para Tokio k=30.

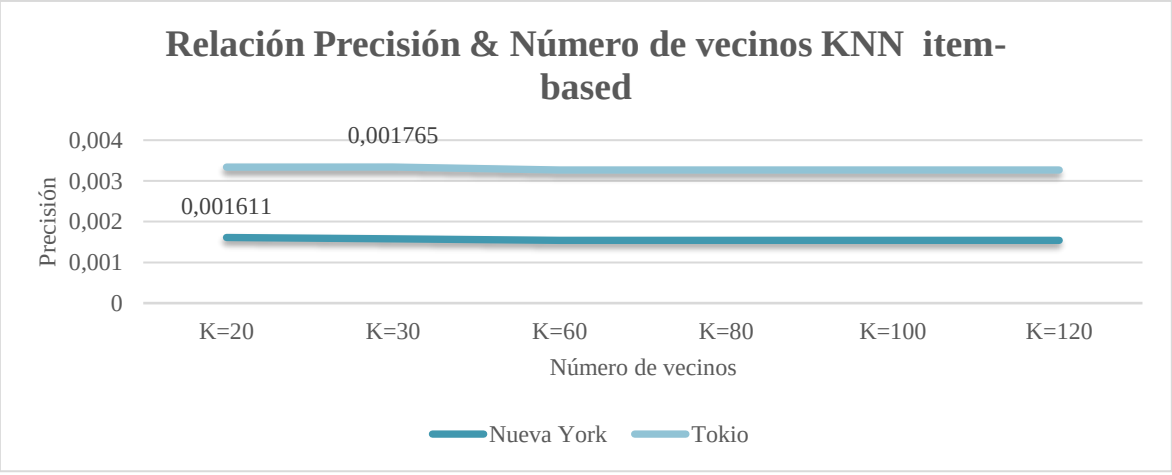


Ilustración 4-4 Evaluación del número óptimo de vecinos en Knn item-based

4.3 RESULTADOS

En esta sección se analizarán los resultados obtenidos. Para ello, emplearemos las métricas de evaluación presentadas anteriormente para así comparar los distintos algoritmos.

También, se analizará el efecto que tiene aplicar el dominio cruzado (sección 2.5) en las recomendaciones generadas. Para las dos ciudades evaluadas, Nueva York y Tokio, se evaluaron los distintos recomendadores en tres escenarios distintos: utilizando el conjunto de datos inicial de Foursquare, el conjunto de datos de Foursquare complementado con datos de Gowalla y el conjunto de Foursquare complementado con datos de Gowalla y BrightKite.

- ✓ En este caso se utilizó un valor de cutoff para las distintas métricas igual a 10.

Además, en los recomendadores aleatorio, de popularidad y de k-nn vecinos, se ha incorporado un reranking que ordenará las recomendaciones generadas según su midpoint (en las figuras, quedará representado con la palabra Reranked). Esta reordenación se hace con el objetivo de proporcionar a los usuarios artículos más relevantes y personalizados en función de su ubicación geográfica. En caso de que el usuario de test sea nuevo y no existan datos históricos suficientes para calcular su midpoint, la reordenación no se hará y en su lugar se mantendrá el ranking establecido originalmente por el algoritmo. Esto se ha hecho para evitar tener pérdidas en la cobertura de los algoritmos.

Por último, se añadió una métrica adicional en la que se calcula la precisión por categorías, con ella se busca medir la relevancia o acierto en las recomendaciones, comparando las categorías de los POIs recomendados con las categorías de los POIs que el usuario de test visitó en el pasado.

4.3.1 RESULTADOS NUEVA YORK

Foursquare

Algoritmo	Precisión	Recall	Epc	Aggregate Diversity	Coverage	Precisión Categorías
Random	0,000742	0,001659	0,997183	9123	1	0,067400
Random Reranked	0,000722	0,001743	0,996289	9053	1	0,059234
Popularity	0,033812	0,067107	0,768067	20	1	0,055479
Popularity Reranked	0,027715	0,059691	0,863392	20	1	0,054457
KNN (k=120)	0,016505	0,038841	0,916964	2940	0,890181	0,093165
KNN (k=120) Reranked	0,016505	0,038841	0,932279	2940	0,890181	0,093165
KNN Midpoint (k=120)	0,013302	0,0227796	0,947311	3448	0,890675	0,065010
Híbrido	0,025822	0,05346	0,791379	1986	1	0,06603
KNN ítems (k=20)	0,003138	0,007417	0,996413	8490	0,890675	0,087365

Tabla 4 Resultados Nueva York Foursquare

Foursquare + Gowalla

Algoritmo	Precisión	Recall	Epc	Aggregate Diversity	Coverage	Precisión Categorías
Random	0,000371	0,000820	0,996717	9162	1	0,055949
Random Reranked	0,000361	0,000841	0,995253	9091	1	0,054874
Popularity	0,028716	0,058432	0,716905	18	1	0,065941
Popularity Reranked	0,023410	0,051946	0,827493	18	1	0,058151
KNN (k=120)	0,015199	0,036005	0,909852	3186	0.890181	0,094804

KNN (k=120) Reranked	0,015199	0,036005	0,920982	3186	0.890181	0,094804
KNN Midpoint (k=120)	0,010580	0,021173	0,930160	3723	0,890675	0,062733
Híbrido	0,024610	0,05046	0,79845	1986	1	0,06590
KNN ítems (k=20)	0,003221	0,00777	0,995711	8369	0,890675	0,086504

Tabla 5 Resultados Nueva York incorporando Gowalla a Foursquare

Foursquare + Gowalla + BrightKite

Algoritmo	Precisión	Recall	Epc	Aggregate Diversity	Coverage	Precisión Categorías
Random	0,000618	0,001187	0,996771	9246	1	0,056072
Random Reranked	0,000694	0,001332	0,995171	9181	1	0,05540
Popularity	0,027776	0,056979	0,714277	18	1	0,065545
Popularity Reranked	0,022744	0,050748	0,825809	18	1	0,057706
KNN (k=120)	0,014310	0,034145	0,910303	3352	0,890181	0,094776
KNN (k=120) Reranked	0,014310	0,034145	0,923725	3352	0,890181	0,094776
KNN Midpoint (k=120)	0,009497	0,018818	0,941205	3839	0,890675	0,062760
Híbrido	0,022510	0,04589	0,8001	1986	1	0,06581
KNN ítems (k=20)	0,003055	0,007229	0,994324	8332	0,890675	0,086115

Tabla 6 Resultados Nueva York incorporando a Foursquare Gowalla + BrightKite

En los resultados obtenidos sobre el conjunto de Foursquare, pudimos observar que en términos de acierto eran muy bajos. Esto es debido a la dispersión (spartisty) de los datos, ya que dificulta obtener recomendaciones interesantes para los usuarios. El algoritmo de

popularidad obtuvo los mejores resultados en términos de precisión, pero a su vez presentó un rendimiento inferior en cuanto a novedad y diversidad.

El recomendador híbrido es muy interesante ya que, perdiendo solo un poco de acierto, fue capaz de mejorar notablemente la novedad y diversidad de sus recomendaciones.

Al añadir los conjuntos de datos adicionales, como se puede ver en la Ilustración 4-5, se observó un decrecimiento en la precisión y recall de los algoritmos. En concreto, al incluir el conjunto de datos de Gowalla se experimentó una pérdida de rendimiento media del 11,93% con respecto a los experimentos realizados únicamente con el conjunto original de Foursquare. Tras la inclusión de los datos de BrightKite, la pérdida media respecto al conjunto original aumentó hasta el 17,30%.

Estos resultados van en contra de las expectativas iniciales, puesto que se espera que la inclusión de más datos conduzca a una mejora en la calidad de las recomendaciones.

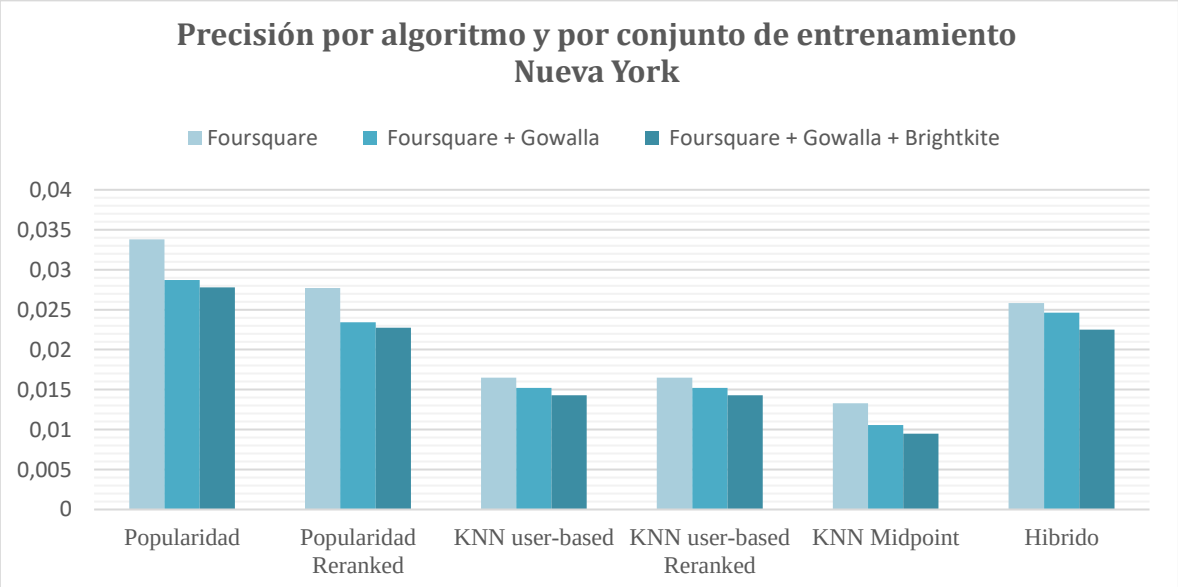


Ilustración 4-5 Variación de la precisión

El aggregate diversity aumentó tras la inclusión de nuevos datos. Para algunos algoritmos, llegó a aumentar un 8,37% al incluir los datos de Gowalla y un 14% tras incluir los de Foursquare.

En cuanto al coverage, se mantuvo constante para todos los algoritmos, lo que indica que aquellos usuarios del conjunto de prueba que inicialmente no encontraron vecinos cercanos en el conjunto de datos original continuarán sin encontrarlos incluso después de la inclusión de nuevos usuarios.

Por otro lado, el EPC aumentó para todos los recomendadores, esto quiere decir que las recomendaciones proporcionadas son más novedosas y por tanto, que los usuarios incorporados visitaban POIs menos populares. Esto también justificará el aumento en aggregate diversity dado para todos los algoritmos.

En conclusión, el aumento en el aggregate diversity y el EPC indica que se ha logrado una mayor diversidad en las recomendaciones y que se están presentando opciones menos populares a los usuarios, ofreciéndoles una experiencia más variada. Sin embargo, la precisión y el recall se vieron afectados negativamente.

4.3.2 RESULTADOS TOKIO

Foursquare

Algoritmo	Precisión	Recall	Epc	Aggregate Diversity	Coverage	Precisión Categorías
Random	0,000416	0,000923	0,998396	18.314	1	0,096006
Random Reranked	0,000431	0,000956	0,998144	18.238	1	0,096136
Popularity	0,032028	0,056503	0,793235	19	1	0,090933
Popularity Reranked	0,032061	0,056762	0,905839	19	1	0,089654
KNN (k=120)	0,031321	0,051297	0,920020	3.902	0,965459	0,144178
KNN (k=120) Reranked	0,031321	0,051297	0,950264	3.901	0,965459	0,144178
KNN Midpoint (k=120)	0,021507	0,033118	0,933262	5.920	0,965767	0,104582
Híbrido	0,024113	0,054131	0,80126	2.414	1	0,10138
KNN ítems (k=30)	0,003600	0,006002	0,997877	13.802	0,985767	0,145239

Tabla 7 Resultados Tokio Foursquare

Foursquare + Gowalla

Algoritmo	Precisión	Recall	Epc	Aggregate Diversity	Coverage	Precisión Categorías
Random	0,000401	0,000672	0,998417	18.379	1	0,097255
Random Reranked	0,000399	0,000690	0,98197	18.310	1	0,097797
Popularity	0,032197	0,057278	0,793232	19	1	0,090933
Popularity Reranked	0,032237	0,057565	0,905391	19	1	0,089654
KNN (k=120)	0,031097	0,050873	0,920217	3.932	0,965459	0,144609

KNN (k=120) Reranked	0,031097	0,050873	0,950592	3.932	0,965459	0,144609
KNN Midpoint (k=120)	0,020613	0,032004	0,938868	5.965	0,965767	0,103944
Híbrido	0,022104	0,052132	0,86126	2.508	1	0,099138
KNN ítems (k=30)	0,003608	0,006071	0,997877	13.829	0,985767	0,140093

Tabla 8 Resultados Tokio incorporando Gowalla a Foursquare

Foursquare + Gowalla + BrightKite

Algoritmo	Precisión	Recall	Epc	Aggregate Diversity	Coverage	Precisión Categorías
Random	0,000416	0,000429	0,998385	18.723	1	0,094541
Random Reranked	0,000415	0,000437	0,998155	18.635	1	0,094843
Popularity	0,033338	0,060427	0,793178	20	1	0,093015
Popularity Reranked	0,033387	0,060749	0,903030	20	1	0,091793
KNN (k=120)	0,029532	0,048626	0,920362	4.075	0,965459	0,144546
KNN (k=120) Reranked	0,029532	0,048626	0,951522	4.075	0, 965459	0,144546
KNN Midpoint (k=120)	0,017963	0,028618	0,956206	5.990	0,965767	0,102331
Híbrido	0,019984	0,050132	0,87246	3.024	1	0,092138
KNN ítems (k=30)	0,003816	0,006441	0,997789	13.787	0,965767	0,137474

Tabla 9 Resultados Tokio incorporando a Foursquare Gowalla + BrightKite

En general, los algoritmos presentaron un mayor rendimiento al utilizar los datos de Tokio en comparación con los datos de Nueva York. Esto se debe a que, en Tokio, la media de

check-ins por usuario es más alta, por lo que la base de datos estará más completa, permitiendo a los algoritmos generar recomendaciones más relevantes para el usuario.

Tras incorporar los nuevos conjuntos de datos, se observó un decremento en el rendimiento de las recomendaciones en términos de precisión y recall (Ilustración 4-6), similar a lo ocurrido en el caso de Nueva York, pero con un decremento del rendimiento mucho menor. Al incorporar el conjunto de datos de Gowalla, se registró un decremento medio de precisión del 2,2% mientras que al incorporar el conjunto de BrightKite se observó un decremento medio del 3,34% con respecto al conjunto de datos original.

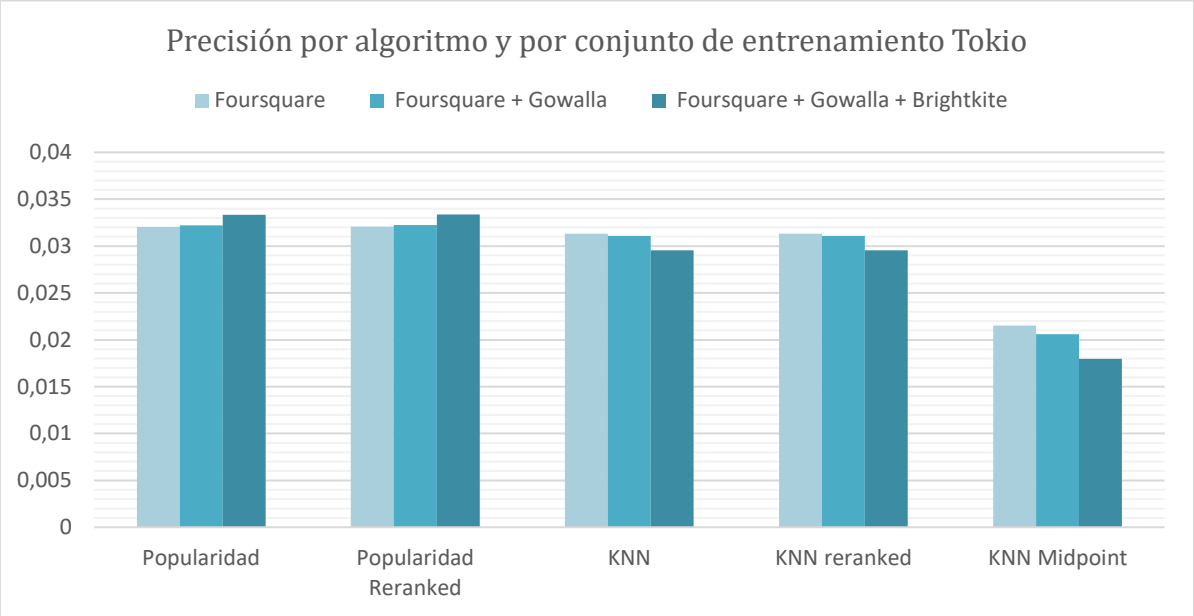


Ilustración 4-6 Variación de la precisión

En cuanto al aggregate diversity, hubo una variación muy pequeña al combinar los datos de Foursquare con los de Gowalla (Aumento medio del 0,87%). Sin embargo, se observó un mayor incremento al agregar los datos de BrightKite puesto que estos añadían una gran cantidad de check-ins al conjunto de entrenamiento, presentando un aumento medio del 6,29%.

4.3.3 DISCUSIÓN GENERAL DE LOS RESULTADOS OBTENIDOS

Durante el desarrollo de los experimentos se procesaron más de 33 millones de check-ins, lo que conllevó que el tiempo de ejecución para preprocesar todos los datos y pasar todos los conjuntos de datos por todos los algoritmos fuera muy elevado.

En general, los resultados obtenidos fueron bajos debido fundamentalmente a la dispersión de los datos, como se comentó anteriormente los conjuntos de datos empleados en estos experimentos contaban con una dispersión media del 99,9945%. También, al observar la media de check-ins por usuario, podemos obtener una idea de la escasez de datos o la falta de conocimiento. En el caso de Nueva York, la media de check-ins por usuario sin tener en cuenta las repeticiones y habiendo agrupado los datos, es de 18, mientras que en Tokio es de 26 (Estos datos se miden sin tener en cuenta las repeticiones y tras haber agrupado los check-ins). Como pudimos ver en los experimentos, cuanto mayor es este ratio, mejor es el rendimiento de los recomendadores.

Por otro lado, obtuvimos un efecto adverso al esperado al incorporar datos de nuevas fuentes dado que en ambos casos las métricas de relevancia disminuyeron. Esto puede atribuirse al bajo porcentaje de coincidencias encontradas entre las bases de datos utilizadas. En el caso de Nueva York, encontrábamos un 18,2% y 14,1% de coincidencias para Gowalla y Brighkite, mientras que para Tokio fueron del 6,6% y 25% respectivamente. Estos resultados implican que, para muchos de los experimentos mostrados, los algoritmos no sean comparables entre ellos.

A pesar de haber obtenido resultados bajos en términos de relevancia, se ha visto que los algoritmos desarrollados son bastante eficaces, ya que tienen una mejor respuesta que al recomendar puntos de interés de forma aleatoria.

Los resultados obtenidos para la precisión por categorías fueron mucho más altos que los de la precisión tradicional. Esto se debe a cómo se define la relevancia que tiene para un usuario un POI que se le ha recomendado. En la precisión normal, un POI se considerará relevante solo si el usuario lo ha visitado previamente. Sin embargo, al evaluar las recomendaciones según la precisión por categorías, se obtendrá una visión más global y amplia de lo que es relevante para el usuario, ya que no solo se tendrán en cuenta las visitas pasadas, sino que también se considerarán los intereses generales del usuario.

De manera más analítica, en la

Tabla 10, podemos observar la diferencia de magnitudes, el número de categorías disponibles será mucho menor que el número de POIs candidatos a ser recomendados.

Ciudad	Nº categorías	NºPOIs
Nueva York	321	9233
Tokio	318	18948

Tabla 10 Número de categorías y POIs a recomendar

Al comparar los dos tipos de algoritmos de vecinos próximos, para el UB (user based) obtuvimos mejores resultados en términos de relevancia y acierto, esto es debido a que en UB se generan las recomendaciones teniendo en cuenta la similitud entre usuarios con gustos similares. En el caso del algoritmo IB (item based), las recomendaciones mejoraron en términos de diversidad y novedad.

Capítulo 5. CONCLUSIONES Y TRABAJO FUTURO

5.1 CONCLUSIONES

Los sistemas de recomendación tienen aplicaciones en diversas industrias como la del entretenimiento, los servicios de streaming, el turismo y el comercio electrónico. Estos sistemas son esenciales para la personalización de información para el usuario, siendo una solución efectiva para abordar el problema de la sobrecarga de información. Esta capacidad para proporcionar recomendaciones precisas y relevantes los han convertido en una herramienta invaluable para las empresas que buscan brindar un servicio de calidad a los usuarios y obtener una ventaja competitiva frente a sus competidores.

En este trabajo nos centramos en la recomendación de puntos de interés debido al impacto que su aplicación tiene en sectores como el del turismo. Estos sistemas pueden favorecer al turismo de las ciudades fomentando un turismo sostenible. En países como España, donde el sector turístico es tan relevante, esto puede tener una importancia crucial.

Se han desarrollado diferentes algoritmos que proporcionan recomendaciones sobre puntos de interés, así como métricas de evaluación que miden la calidad y el rendimiento de dichos recomendadores. Teniendo en cuenta los resultados obtenidos, se ha observado que es difícil encontrar un equilibrio entre las diferentes dimensiones de evaluación de un sistema de recomendación. Por ejemplo, encontraremos modelos que sean superiores en términos de precisión pero que puedan carecer de novedad y diversidad en sus recomendaciones. Esto ilustra la dificultad del problema de recomendación para adaptarse a las necesidades de los usuarios.

El objetivo principal del proyecto ha sido comprobar el efecto que tiene utilizar el cross-domain en las recomendaciones generadas por el sistema. En las dos ciudades estudiadas, cuando se complementaron los check-ins de Foursquare con los de Gowalla y BrightKite, no se obtuvo el resultado esperado, sino que las recomendaciones generadas empeoraron en términos de relevancia. Sin embargo, el aggregate diversity y el EPC (Expected Popularity Complement) incrementaron, lo cual proporcionaba a los usuarios recomendaciones más novedosas y variadas.

El algoritmo de popularidad fue el que mejor resultados obtuvo en términos de relevancia, manifestando el sesgo de popularidad observado. Se tiende a recomendar lugares más populares frente a opciones más diversas o menos conocidas. Aunque esto ya se había observado previamente, alerta sobre la necesidad de innovar en este tipo de recomendación, midiendo el rendimiento de novedad y diversidad, lo cual permitirá proporcionar recomendaciones más sorprendentes a los usuarios.

Los resultados obtenidos al agregar conjuntos de datos adicionales y observar una disminución en la precisión y recall plantean interrogantes sobre la generalización de ciertos modelos de recomendación propuestos por algunos investigadores, los cuales demuestran un rendimiento alto en un conjunto de datos específico. Un modelo de datos exitoso puede no ser igualmente efectivo en otro conjunto, por lo que se deben considerar diferentes escenarios para obtener una visión general de la eficacia de un sistema de recomendación.

Por último, es importante mencionar la importancia que tiene el tiempo de ejecución del pipeline, ya que influye en la eficiencia y la escalabilidad de un sistema de recomendación. Para aminorar este tiempo, se pueden utilizar técnicas de optimización que permitan incrementar la capacidad de respuesta del sistema en la generación de recomendaciones. Por ejemplo, al emplear la técnica de almacenamiento en caché, se almacenarán los resultados

cuyos cálculos sean computacionalmente costosos y se utilicen recurrentemente, evitando recalcularlos más de una vez.

- El código desarrollado se puede consultar en el repositorio online del siguiente enlace: https://github.com/beasicilia14/TfgSistemasRecomendacion

5.2 *TRABAJO FUTURO*

Como trabajo futuro, se plantea incorporar información adicional para generar mejores recomendaciones. Por ejemplo, añadir el horario de los POIs para así generar recomendaciones más personalizadas en función de la disponibilidad y preferencias temporales de los usuarios.

Otra mejora sería analizar los diferentes tipos de usuarios del sistema, como turistas de diferentes categorías o usuarios con perfiles más locales. De este modo, se podrán comprender las características y preferencias que distinguen a estos grupos de usuarios para así adaptar las recomendaciones realizadas. Por ejemplo, si se tuvieran en cuenta las categorías de turista podríamos distinguir entre turistas aventureros, culturales o de negocios... Al comprender las actividades y experiencias preferidas por cada tipo de turista se podría influir en el proceso de recomendación y así ofrecerle lugares más relevantes. De la misma forma, al considerar a usuarios locales, se podrá tener en cuenta su conocimiento sobre el entorno y así recomendarle lugares menos conocidos, pero más auténticos o diversos.

Por último, también se propone incorporar otros datos adicionales provenientes de otras fuentes de información que muestren una mayor compatibilidad con la fuente de datos principal.

Capítulo 6. BIBLIOGRAFÍA

[1] «IT User,» 27 Abril 2022. [En línea]. Available: https://www.ituser.es/actualidad/2022/04/el-63-de-la-poblacion-mundial-es-usuaria-de-internet. [Último acceso: 2 Mayo 2023].

[2] «StackScale,» 21 Abril 2023. [En línea]. Available: https://www.stackscale.com/es/blog/crecimiento-estadisticas-ecommerce/. [Último acceso: 2 Mayo 2023].

[3] M. Maroto, «Medium,» 21 Julio 2021. [En línea]. Available: https://marianamarotob.medium.com/collaborative-filtering-and-some-history-on-the-netflix-prize-63d63c2af22b. [Último acceso: 5 Mayo 2023].

[4] M. D. Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel y C. Yu, «Automatic construction of travel itineraries using social breadcrumbs.,» de *Proceedings of the 21st ACM conference on Hypertext and hypermedia*, New York, NY, Association for Computing Machinery, 2010, pp. 35-44.

[5] G. A. Alexander Tuzhilin, «Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions,» IEEE Transactions of knowledge and Data Engineering, 2005.

[6] C. Desrosiers y G. Karypis, «A Comprehensive Survey of Neighborhood-based Recommendation Methods,» de *Recommender systems handbook*, 2011, pp. 107-144.

[7] M. C. P. d. C. Herrero, Y. L. D. Jesús y G. M. Olguín, «Métricas de similaridad y evaluación para sistemas de recomendación de filtrado colaborativo.,» *RITI Journal*, vol. 7, nº 14, pp. 230-235, 2019.

[8] A. A. Yeung, «QuuxLabs,» 16 Septiembre 2010. [En línea]. Available: http://www.quuxlabs.com/blog/2010/09/matrix-factorization-a-simple-tutorial-and-implementation-in-python/#the-mathematics-of-matrix-factorization. [Último acceso: 23 Mayo 2023].

[9] K. Yehuda, R. Bell y C. Volinsky, «Matrix Factorization Techniques for Recommender Systems,» *Computer*, vol. 42, nº 8, pp. 30-37, 2009.

[10] D. Jannach y M. Ludewig, «When Recurrent Neural Networks meet the Neighborhood for Session-Based Recommendation,» Como, 2017.

[11] T. Jiaxi y W. Ke, «Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding,» de *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, New York, NY, Association for Computing Machinery, 2018, p. 565–573.

[12] B. Hidasi y A. Karatzoglou, «Recurrent Neural Networks with Top-k Gains for Session-based Recommendations,» de *Proceedings of the 27th ACM International*

Conference on Information and Knowledge Management, Turín, Italia, Association for Computing Machinery, 2018, p. 843–852.

[13] F. Dacrema y M. Cremonesi, «Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches,» de *Proceedings of the 13th ACM Conference on Recommender Systems*, Copenhaguen, Dinamarca, 2019, p. 101–109.

[14] R. Burke, «Knowledge-based recommender systems,» de *Encyclopedia of Library and Information Science*.

[15] P. Lops, M. d. Gemmis y G. Semeraro, «Content-based Recommender Systems: State of art and trends,» de *Recommender Systems Handbook*, pp. 73-101.

[16] R. Burke, «Hybrid Web Recommender Systems,» de *The Adaptive Web*, 2007.

[17] R. Burke, «Knowledge-based recommender systems,» de *Encyclopedia of Library and Information Science*.

[18] L. Chameikho y I. Ryngksai, «Recommender Systems: Types of Filtering Techniques,» *International Journal of Engineering Research & Technology*, vol. 3, nº 11, 2014.

[19] P. Sánchez y A. Bellogín, «Point-of-Interest Recommender SYstems based on Location-Based Social Networks: A survey from an Experimental Perspective,» *ACM Comput. Surv.* , vol. 1, nº 1, 2022.

[20] Z. Tian, K. Li, H. Wei y X. He, «Relational POI recommendation model combined with geographical information,» *PLoS ONE 17*, 2022.

[21] G. Shani y G. Asela, «Evaluating Recommender Systems,» de *Recommender Systems Handbook*, 2010, pp. 257-299.

[22] H. Steck, «Evaluation of Recommendations: Rating-prediction and Ranking,» Netflix Inc, Los Gatos, California, 2013.

[23] S. Vargas y P. Castells, «Rank and relevance in novelty and diversity metrics for recommender systems,» *RecSys '11: Proceedings of the fifth ACM conference on Recommender systems*, pp. 109-116, 2011.

[24] G. Adomavicius y Y. Kwon, «Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques,» de *IEEE Transactions on Knowledge*.

[25] I. Cantador y A. Bellogín, «Sistemas de Recomendación Sobre Dominios Cruzados,» 27 Agosto 2020. [En línea]. Available: https://docplayer.es/225322007-Sistemas-de-recomendacion-sobre-dominios-cruzados.html. [Último acceso: 22 Mayo 2023].

[26] I. Cantador, I. Fernández-Tobías, S. Berkovsky y P. Cremonesi, «Cross-Domain Recommender Systems,» de *Recommender Systems Handbook*, Boston, MA, Springer, 2015, pp. 919-959.

[27] H. Williams, «TechMonitor,» 27 Marzo 2017. [En línea]. Available: https://techmonitor.ai/what-is/what-is-gowalla. [Último acceso: 27 Mayo 2023].

[28] B. Stephenson, «LifeWire,» 27 Marzo 2020. [En línea]. Available: https://www.lifewire.com/what-is-yelp-4685842. [Último acceso: 2023 Mayo 27].

[29] «The Netflix Prize and Production : An insider look.,» MathWorks, 2016.

[30] A. Ajitsaria, «RealPython,» [En línea]. Available: https://realpython.com/build-recommendation-engine-collaborative-filtering/. [Último acceso: 20 Abril 2023].

[31] R. Parmar, 11 Septiembre 2018. [En línea]. Available: https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964. [Último acceso: 9 Mayo 2023].

[32] I. Cantador, I. Fernández-Tobías, S. Berkovsky y P. Cremonesi, «Cross-Domain Recommender Systems».

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS

Este proyecto está alineado con el cumplimiento del objetivo de desarrollo sostenible número 8, el cual promueve el trabajo decente y crecimiento económico. Los sistemas de recomendación de puntos de interés pueden favorecer al cumplimiento de dicho objetivo de las siguientes formas:

- ✓ **Generación de empleo:** al alentar a los visitantes a explorar distintos puntos de interés se pueden generan oportunidades laborales en sectores como la hostelería, las actividades turísticas.

- ✓ **Impulso a la economía local:** al recomendar puntos de interés que son menos conocidos se favorece la distribución de los beneficios económicos procedentes del turismo, lo cual puede impulsar el crecimiento local de lugares que no solían ser tan conocidos.

- ✓ **Mayor productividad económica gracias a la innovación y tecnología:** los sistemas de recomendación de puntos de interés fomentan la innovación y el uso de las nuevas tecnologías en la industria del turismo. La personalización de la experiencia del usuario resulta en una mayor satisfacción del cliente, lo que a su vez conlleva una mayor rentabilidad en el sector.

ANEXO II

Cálculo Midpoint

Esta función recibe como argumento un diccionario que asocia a los usuarios con los POIs que han visitado y un diccionario que contiene la información geográfica de los POIs.

```
Def knnMidpointCalc(self, user_dicc, pois_dicc):
        dicc_midpoint = {}

        foruserinuser_dicc.keys():
            pois_visited = user_dicc[user].keys()
            x=0
            y=0
            z=0
            forpoiinpois_visited:
                lat= float(pois_dicc[poi][0])
                lon = float(pois_dicc[poi][1])

                lon = lon * math.pi / 180
                lat = lat * math.pi / 180

                x += math.cos(lat) * math.cos(lon)
                y += math.cos(lat) * math.sin(lon)
                z += math.sin(lat)

            x_weighted = x/len(pois_visited)
            y_weighted = y/len(pois_visited)
            z_weighted = z/len(pois_visited)

            lon = math.atan2(y_weighted, x_weighted)
            hyp = math.sqrt(x_weighted*x_weighted+ y_weighted*y_weighted)
            lat = math.atan2(z_weighted, hyp)

            lon = lon*180 / math.pi
            lat = lat*180 /math.pi

            #midpoint de cada usuario que está en train
            dicc_midpoint[user] = [lat,lon]

        returndicc_midpoint
```

Cálculo distancia de Haversine

Función que recibe como argumentos las coordenadas de los dos puntos entre los que se quiere hallar la distancia.

```
def haversine(lat1, lon1, lat2, lon2):
        rad=math.pi/180
        dlat=lat2-lat1
        dlon=lon2-lon1
        R=6372.795477598
        a=(math.sin(rad*dlat/2))**2 +
math.cos(rad*lat1)*math.cos(rad*lat2)*(math.sin(rad*dlon/2))**2
        distancia=2*R*math.asin(math.sqrt(a))
        return distancia
```

Código completo

Como se indicó anteriormente, el código completo se encuentra en Github TFG Sistemas de Recomendación