Universidad Autónoma de Madrid

Escuela Politécnica Superior

21|22

Trabajo fin de grado

Combinación de diferentes fuentes de datos en la recomendación de puntos de interés

Jaime Pascual Francés

Escuela Politécnica Superior
Universidad Autónoma de Madrid
C\Francisco Tomás y Valiente nº 11

excelencia Campus Internacional UAM CSIC

www.uam.es

UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Combinación de diferentes fuentes de datos en la recomendación de puntos de interés

Autor: Jaime Pascual Francés
Tutor: Pablo Sánchez Pérez
Ponente: Alejandro Bellogín Kouki

junio 2022

Jaime Pascual Francés
Combinación de diferentes fuentes de datos en la recomendación de puntos de interés

Jaime Pascual Francés
C\ Francisco Tomás y Valiente Nº 11

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

Si necesito ver más lejos
me tengo que subir a los hombros de un gigante.

Isaac Newton

Agradecimientos

En primer lugar quiero expresar mi agradecimiento a mi tutor Pablo Sánchez por su compromiso. Su disponibilidad ha sido esencial para el desarrollo de este proyecto.

Quiero agradecer también el apoyo de mi familia y sobre todo a Carmen por su apoyo y paciencia a lo largo de la carrera.

Por último, destacar a mis compañeros de la facultad y trabajos en grupo. En especial a Alberto Cabellos, Guillermo Moya y Ángel Sánchez, con los que he pasado tantas horas de estudio y me han ayudado a lo largo de estos años.

Resumen

Visitar nuevos lugares y tener nuevas experiencias es algo que siempre ha llamado la atención de la gente. Poder ir a un museo que no habías visitado antes, un restaurante donde no habías comido e incluso visitar sitios de una ciudad en la que nunca habías estado. A la mayoría de las personas les interesa conocer nuevos productos que se adecúen a sus gustos y necesidades. En consecuencia, muchas empresas han desarrollado algoritmos para poder recomendar a sus usuarios productos que les puedan interesar en función de su perfil.

Los sistemas de recomendación están presentes en la mayoría de las plataformas. Estos sistemas se encargan de recomendar a un usuario específico el tipo de artículo que le pueda interesar, ya sea una película en el caso de plataformas de streaming (Netflix, HBO...) con su sección de 'películas que te puedan gustar', incluso usuarios o publicaciones en el caso de las redes sociales (Instagram, Facebook...). Además de la recomendación clásica han surgido otros dominios de recomendación como la de puntos de interés. Es decir, recomiendan a los usuarios lugares que no han visitado todavía, de una ciudad o una región en específico, que les podría interesar

Este Trabajo de Fin de Grado consiste en el diseño de un framework que trabaja con los datos de las redes sociales basadas en localización (Foursquare y Gowalla) para efectuar recomendaciones. En concreto, con el objetivo de paliar algunos problemas como la escasez de los datos inherentes a este dominio de recomendación, en este TFG se propone combinar diferentes tipos de datos de las plataformas de Gowalla y Foursquare. Para ello se emplean técnicas de dominio cruzado para analizar el efecto que tiene dicha combinación en las recomendaciones producidas.

Para el desarrollo se han utilizado diferentes algoritmos de recomendación, como puede ser uno basado en la popularidad de cada punto de interés, otro basado de vecinos próximos o un híbrido que incorpora la información geográfica del usuario. Se han incluido diferentes algoritmos con el objetivo de estudiar la mejora de los diferentes modelos de recomendación cuando se aplica el dominio cruzado. Sin embargo, se ha observado en los resultados obtenidos que las recomendaciones no han mejorado en términos de acierto, aunque se ha reducido el sesgo de popularidad que puede haber en los datos.

Palabras clave

Sistema de recomendación, dominio cruzado, métricas de evaluación, punto de interés, Foursquare, Gowalla

Abstract

Visiting new places and having new experiences is something that people have always been attracted to. Go to see a museum you had not visited before, to a restaurant where you had not eaten and even visit places in a city you have never been to. Most people are interested in learning about new products that suit their tastes and needs. Consequently, many companies have developed algorithms to recommend products to their users that they may be interested in based on their profile.

Recommendation systems are very common on most platforms. These systems are responsible for recommending a specific user the type of item that may interest him, it could be a movie in the case of streaming platforms (Netflix, HBO...) with their 'movies that you may like' sections, including users or publications in the case of social networks (Instagram, Facebook...). In addition to the classic recommendation, other recommendation domains have emerged, such as point of interest recommendation. They recommend points to users that they have not yet visited, of a specific city, which might interest them.

This Bachelor Thesis consists of the design of a framework that works with data from location-based social networks (Foursquare and Gowalla) to make recommendations. Specifically, with the aim of reducing some problems such as data scarcity, this TFG proposes to combine different types of data from the Gowalla and Foursquare platforms. For this, cross-domain techniques are used to analyze the effect that this combination has on the recommendations produced.

For the development, different recommendation algorithms have been used. An algorithm based on the popularity of each point of interest, based on nearest neighbors or a hybrid algorithm which mixes these last two with the proximity to the user midpoint. Different algorithms have been included with the aim of studying the improvement of the different recommendation models when applying the cross domain. However, we have seen in the results that the recommendations have not improved in terms of accuracy, although the popularity bias that may exist in the data has been reduced.

Keywords

Recommender system, cross-domain, evaluation metrics, point of interest (POI), Foursquare, Gowalla

Índice

Listas

Lista de ecuaciones

Lista de figuras

Lista de tablas

1

Introducción

En las últimas décadas el uso de Internet está aumentando considerablemente. Según [1] en 2016 el 45.79 % de la población tenía acceso a Internet y en 2019 un 57 %, lo que supone un aumento del 11.21 % que equivale a 863 millones de personas en 3 años. Actualmente se cuenta con 4,620 millones de usuarios en redes sociales según [2]. En consecuencia, las grandes empresas cada vez dan más importancia al servicio online. De acuerdo con [3] en 2017 el comercio electrónico incrementó un 25.7 % con respecto al año anterior. Además el constante desarrollo de la economía ha facilitado que los usuarios tengan acceso a una mayor variedad de productos. Para poder paliar esta sobrecarga de información, existen actualmente los sistemas de recomendación.

Un sistema de recomendación es una herramienta software que recomienda a un usuario de una plataforma artículos que le puedan ser de su interés. La investigación en estos sistemas ha crecido a lo largo de los años, creando un amplio campo de estudio donde se están desarrollando numerosas técnicas. El tipo de artículos que se recomiendan al usuario están sujetos a la empresa que los presenta. Pueden ser libros, películas, electrodomésticos o incluso otros usuarios en las redes sociales.

Hoy en día la mayoría de empresas con servicio online disponible cuentan con un sistema de recomendación. Algunas de las más conocidas son Spotify [1], Netflix [2] o Amazon [3]. Todas buscan tener el mejor recomendador, ya que puede aumentar el registro de nuevos usuarios además de mantener a los ya presentes y por lo tanto generar más beneficios.

El turismo y la hostelería también se pueden beneficiar de estos sistemas, como se puede ver en las redes sociales basadas en localización o LBSN (Location Based Social Networks) donde los usuarios registran los diferentes check-ins que realizan en los lugares que visitan. Posteriormente, el sistema puede realizar recomendaciones en función a esos datos. Las redes sociales Foursquare [4] y Gowalla [5] son un ejemplo de LBSN.

[1] Spotify: https://www.spotify.com. Accedido: 2022-06-01

[2] Netflix: https://www.netflix.com. Accedido: 2022-06-01

[3] Amazon: https://www.amazon.es. Accedido: 2022-06-01

[4] Foursquare: https://foursquare.com. Accedido: 2022-06-01

[5] Gowalla: https://go.gowalla.com. Accedido: 2022-06-01

1.1. Motivación del proyecto

Actualmente se están aplicando los sistemas de recomendación en todo tipo de plataformas, lo que supone su uso a diario por parte de los usuarios. Es por ello que se quiere aplicar algunas de estas técnicas en la recomendación de puntos de interés.

En este trabajo la motivación por una parte es recomendar a los usuarios puntos de interés que puedan ser de su gusto. Asimismo, debido a la escasez de datos en las redes sociales basadas en localización, se quiere combinar la información obtenida de diferentes fuentes de datos para ver el efecto que tiene en las recomendaciones.

1.2. Objetivos

El objetivo principal de este trabajo es estudiar el cambio de los puntos de interés recomendados en las redes sociales basadas en localización aplicando la técnica del dominio cruzado, es decir, combinando información de diferentes aplicaciones. Otro objetivo es familiarizarse con algunas de las técnicas de recomendación más populares y el manejo de conjuntos de datos. Para ello, se implementarán algunos algoritmos de recomendación conocidos. Aparte se implementará otro algoritmo menos conocido basado en la ubicación del usuario. Además, se quiere entender el funcionamiento de las métricas de evaluación empleadas para medir el acierto de las recomendaciones y estudiar otras dimensiones complementarias como la novedad y la diversidad.

1.3. Estructura del trabajo

El documento está dividido en cuatro partes bien diferenciadas. El estado del arte (Capítulo 2), expone un estudio teórico de los algoritmos y métricas que se van a desarrollar a lo largo del trabajo. El capítulo de diseño e implementación (Capítulo 3) cuenta con la estructura general de este proyecto, así como una descripción de cada uno de sus módulos. Después se describen las pruebas y resultados (Capítulo 4) donde se expone la salida de nuestro sistema junto con una explicación. Por último, se expone una conclusión y el trabajo futuro (Capítulo 5), donde se interpretan los resultados finales y se desarrollan posibles líneas de investigación futuras.

2

Estado del arte

2.1. Sistemas de recomendación

Un sistema de recomendación es una herramienta software que se encarga de sugerir nuevos artículos al usuario a través de su experiencia anterior, sus gustos y necesidades. Estos sistemas se utilizan, sobre todo, en aquellas aplicaciones en las que existe una gran cantidad de información y productos que pueden ser consumidos. De esta manera se filtran todos los artículos disponibles en el servicio de una manera coherente, basándose en el usuario.

Actualmente, los sistemas de recomendación se pueden encontrar en un gran número de plataformas. Un ejemplo que vemos a diario puede ser una plataforma de música como Spotify o de streaming como Netflix la cual nos recomienda películas o series basándose en nuestra experiencia.

Este campo está en pleno auge debido a los beneficios que generan algunas plataformas según la efectividad de sus algoritmos. Esto se debe a que las empresas propietarias de dichas plataformas, con el objetivo de buscar nuevos clientes y mantener los que ya tienen, quieren ayudar al consumidor a que su experiencia sea más personalizada y ahorrarle tiempo en la búsqueda de los productos que le interesan.

En función de cómo se realizan estas recomendaciones podemos distinguir diversos tipos de sistemas de recomendación, cada uno con sus ventajas e inconvenientes [4] [5], que pueden depender del propio usuario o la experiencia y valoración de otros usuarios. Estos son de filtrado colaborativo, basados en contenido, basados en conocimiento, demográficos e híbridos.

Todos ellos cumplen con dos fases esenciales en cualquier sistema de recomendación:

- **Predicción.** Se encarga de estimar, según el algoritmo con el que se esté trabajando, la valoración que el usuario le daría a cada ítem del sistema, también conocido como score.
- **Recomendación.** Extrae un conjunto de artículos para recomendar al usuario. Esta fase es muy importante ya que solo se seleccionan un subconjunto pequeño de artículos.

2.1.1. Filtrado colaborativo

Este tipo de sistemas emplean las preferencias de los usuarios y de los artículos (interacciones entre los usuarios y los ítems) analizándolas para efectuar las recomendaciones. Estas preferencias se pueden medir de varias maneras, ya sea con el rating específico que le da el usuario a cierto ítem o las acciones que realiza un usuario en relación al ítem.

Poniendo un ejemplo más claro. En el caso de una plataforma de streaming como Netflix, el rating sería la valoración que le da el usuario a cierta película o serie y la acción tomada podría ser, entre otras, el tiempo de reproducción. No sería lo mismo que el usuario hubiera terminado la película o que la hubiera parado en los primeros minutos.

Ventajas

- Pueden ofrecer ítems al usuario fuera de lo familiar, ya que se basa en lo que han visto otros usuarios, aunque en ocasiones tiende a recomendar ítems populares.
- Cuanta más información se tenga del usuario, más capacidad tiene el algoritmo de determinar la relevancia del ítem.
- No es necesario que el usuario determine qué valoración le proporciona a cada ítem, sino que se puede extraer información a través de su actividad. En el caso de Netflix, el hecho de ver una película significa que al usuario se ha visto interesado por el contenido.

Desventajas

- Es necesario un volumen elevado de datos para que la recomendación sea lo más efectiva posible, además de tener un gran coste computacional.
- Tiene lugar lo que se conoce como arranque en frío. A la hora de introducir nuevos usuarios o artículos en el sistema, no se podrán recomendar hasta que no tengan interacciones con otros elementos.
- Existen usuarios que sufren del problema de “oveja negra”. Esto quiere decir que al tener gustos muy diferentes al resto es difícil encontrar una recomendación efectiva.
- Este sistema depende de las valoraciones de los usuarios. Puede ser un problema ya que los usuarios muchas veces no valoran los artículos de forma seria y conlleva a la recomendación errónea a usuarios vecinos.

Dentro del filtrado colaborativo podemos distinguir dos grandes grupos a la hora de calcular la relación entre usuarios [6] [7], los basados en memoria y los basados en modelos.

Basados en memoria

Es también conocido como basado en vecinos próximos o en inglés k nearest neighbors (knn). Se encarga de generar predicciones teniendo en cuenta similitudes entre usuarios [8]. Existen dos tipos de relaciones:

- Las basadas en el **usuario.** Se predice la valoración del usuario teniendo en cuenta únicamente las valoraciones que han realizado los vecinos del usuario al artículo que se va a recomendar.
- Las basadas en el **producto.** Se predice la valoración del usuario teniendo en cuenta únicamente las valoracio-

nes que ha hecho el usuario a los vecinos del artículo que se va a recomendar.

De los dos tipos de sistemas de recomendación basados en memoria vamos a centrarnos en los primeros, los basados en usuario.

A la hora de calcular la relación entre dos usuarios, se utilizan las valoraciones que han dado ambos a los ítems. Para hallar dicha relación se pueden emplear diferentes funciones de similitud.

Similitud coseno

Recibe este nombre ya que emplea el coseno del ángulo que forman los dos vectores en un espacio vectorial. En inglés se conoce como *Cosine Vector* (CV). La fórmula utilizada sería la siguiente:

$$CV(u,v) = \frac{\sum_{i \in \mathcal{I}_{uv}} r_{ui} r_{vi}}{\sqrt{\sum_{i \in \mathcal{I}_u} r_{ui}^2 \sum_{j \in \mathcal{I}_v} r_{vj}^2}} \tag{2.1}$$

Siendo $\mathcal{I}_{uv}$ los elementos valorados por ambos usuarios, mientras que I_u representa los elementos puntuados por el usuario u e I_v representa los elementos puntuados por el usuario v. El resultado oscila en el rango de 0 y 1, donde 0 indica una similitud nula mientras que 1 la similitud perfecta.

Correlación de Pearson

Esta métrica incorpora la media de valoraciones que un usuario le ha dado a cada elemento. En inglés se conoce como *Pearson Correlation* (PC).

$$PC(u,v) = \frac{\sum_{i \in \mathcal{I}_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in \mathcal{I}_{uv}} (r_{ui} - \bar{r}_u)^2 \sum_{j \in \mathcal{I}_{uv}} (r_{vj} - \bar{r}_v)^2}} \tag{2.2}$$

Hay que recalcar que con la distancia coseno, se tienen en cuenta todos los ítems visitados por cada usuario, mientras que con la correlación de Pearson se tienen en cuenta únicamente los ítems valorados por ambos usuarios.

Con la correlación de Pearson pueden darse dos tipos de resultados. El primero es que la correlación sea menor a cero. Esto supondría que es una correlación negativa (si la correlación es igual a -1, nos encontramos con una correlación negativa perfecta). También podría darse que sea mayor que cero, lo que significa que la correlación es positiva, por lo que la similitud sería alta, siendo esta perfecta igual a 1. Por último que sea igual a cero, que se produce cuando no se puede determinar la similitud que hay entre ambos usuarios.

Una vez calculada la similitud entre usuarios se calcula el score para cada ítem siguiendo la siguiente fórmula:

$$\hat{r}_{ui} = \frac{\sum_{v \in \mathcal{N}_i(u)} w_{uv} r_{vi}}{\sum_{v \in \mathcal{N}_i(u)} |w_{uv}|} \tag{2.3}$$

Donde $N_i(u)$ representa a los vecinos del usuario u que han puntuado el ítem i, w_{uv} la similitud que hay entre los usuarios u y v (empleando la similitud coseno o la correlación de Pearson entre otras) y r_{vi} la valoración del usuario v al ítem i.

Métodos basados en modelos

Los algoritmos basados en modelos se llaman así debido a que se encargan de crear un modelo de valoraciones para el usuario. Para esto se utiliza una matriz de valoraciones, donde cada fila representa un usuario y cada columna un ítem, de forma que la celda (u, i) contiene el rating que le haya dado el usuario u al artículo i, siendo 0 en caso de que no lo haya puntuado.

Básicamente, utiliza este conjunto de entrenamiento para encontrar patrones y crear modelos de los ítems que le interesan al usuario. Los algoritmos más conocidos son los de factorización de matrices, aunque también se pueden incluir en esta familia los modelos basados en redes neuronales.

El modelo basado en redes neuronales se encarga de calcular un valor de salida a partir de un conjunto de neuronas artificiales a las que le llega información de entrada, una función de activación y un umbral. Hay distintos tipos de redes neuronales, como pueden ser perceptrón multicapa, prealimentadas, recurrentes o redes neuronales convolucionales [9].

Varias ventajas que tienen los métodos basados en modelos, son que el tiempo de respuesta es más rápido al estar ya entrenado el modelo y que mejora el rendimiento de la predicción al ser un tipo de predicción estadística. Sin embargo, necesita tiempo de aprendizaje, el cual puede ser bastante largo e intenso. Además las formas de generar estas recomendaciones pueden ser menos explicables para los usuarios. Hay varios algoritmos que se encargan de esto pero nos vamos a centrar en el más común.

Factorización de matrices

Consiste en descomponer la matriz de valoraciones inicial (M_O) de tamaño $U \times I$, siendo U el número de usuarios e I el número total de artículos del sistema en otras que representen a cada uno de los elementos [10]. Es decir, una matriz para los usuarios (M_U) de tamaño $U \times K$ y otra para los elementos (M_I) de tamaño $I \times K$, siendo K el número de factores latentes. Mediante la multiplicación de estas matrices, resulta una matriz similar a la original, como se muestra en la Figura 2.1. Esto se resume en la siguiente ecuación:

$$M_O \approx M_U \cdot {M_I}^T \tag{2.4}$$

El objetivo es predecir los valores que están inicialmente a 0 en M_O. Para ello, se inicializan las otras matrices con valores aleatorios para que se optimicen hasta que la multiplicación de éstas sea lo más cercana posible a la original. Hay varias formas de optimizar las matrices, pero el algoritmo más conocido es el descenso por gradiente, donde los factores latentes se van optimizando hasta que se

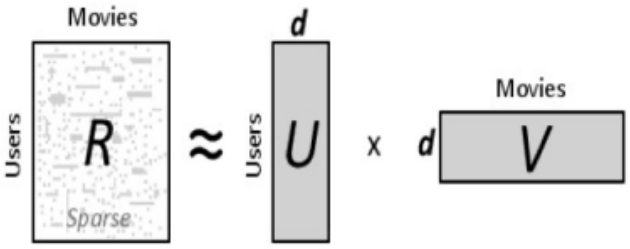

Figura 2.1: Esquema explicativo de la factorización de matrices siendo d el número de factores latentes. *Extraído de [7]*

alcanza la convergencia minimizando el error o hasta un número de iteraciones particular, minimizando la siguiente fórmula:

$$min_{U,I}\|M_O - M_U M_I^T\|_F^2 + \lambda\|U\|_F^2 + \lambda\|I\|_F^2 \tag{2.5}$$

2.1.2. Basado en contenido

Este método analiza únicamente los artículos que ha consumido anteriormente el usuario. Se estudian los ítems que han interesado al usuario anteriormente y se relaciona con el resto que no ha visto. Para ello se utilizan características del ítem seleccionado, ya sean la descripción, el tipo o el género entre otros.

Un ejemplo de este filtrado sería una tienda online como Amazon. Si el usuario únicamente busca artículos de cocina y videojuegos, se le recomendarán artículos relacionados con estos ámbitos. Además se puede intuir que un libro no es de su interés. No obstante, con el filtrado colaborativo que hemos visto anteriormente, ambos métodos podrían recomendarle un libro si un vecino cercano lo ha valorado bien.

A pesar de que el mecanismo de recomendación sea distinto, como se muestra en la Figura 2.2, los sistemas de recomendación basados en contenido tienen unas ventajas y desventajas similares al filtrado colaborativo, las cuales se explican a continuación.

Ventajas

- A medida que pasa el tiempo, el acierto del recomendador tiende a aumentar. Esto se debe a que cuantos más ítems haya valorado el usuario, más información se tendrá de él.
- No sufre de arranque en frío a nivel de ítems. Si la tienda online pone a la venta un nuevo videojuego, éste podría ser recomendado al usuario si cumple con las mismas características que los videojuegos comprados anteriormente.

Desventajas

- El volumen de datos requerido debe ser bastante grande para que la recomendación sea lo más efectiva posible.

- La variedad de contenido a recomendar es escasa, ya que el usuario no suele descubrir nuevos ítems que están fuera de su círculo.

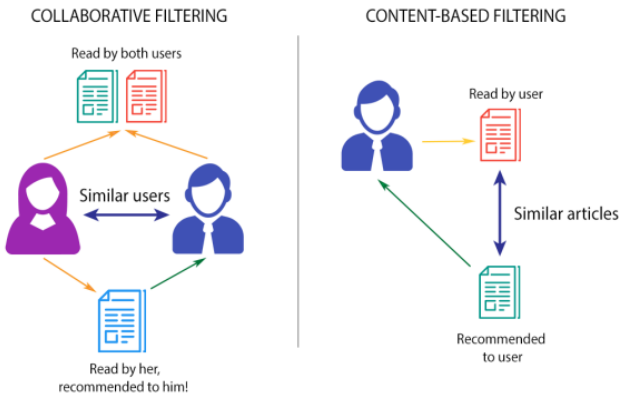

Figura 2.2: Esquema explicativo con las diferencias entre el filtrado colaborativo y el basado en contenido. *Extraído de [11]*

2.1.3. Híbridos

Un sistema de recomendación híbrido es aquel que combina diferentes técnicas de recomendación con el objetivo de hacer más exacta su eficacia, además de disminuir los problemas que pueda presentar algún sistema en concreto. Para la combinación de estos sistemas se utilizan diferentes técnicas [12].

- **Por pesos.** Se encarga de ponderar los pesos obtenidos por ambos sistemas de recomendación.
- **Conmutados.** Utiliza únicamente un recomendador. Se puede seleccionar antes de realizar la recomendación o seguir un criterio de selección posterior a ella.
- **Mezclados.** Con esta técnica se presentan los resultados obtenidos de ambos sistemas de manera independiente.
- **Combinación de propiedades.** Las propiedades de un sistema de recomendación se usan dentro del algoritmo de otro sistema para mejorar los resultados. Esto quiere decir que ambas características se utilizan en un solo algoritmo.
- **Cascada.** Un segundo sistema de recomendación mejora los resultados del otro. Es decir, primero se usa un sistema para recomendar al usuario y después otro sistema aumenta la eficacia del anterior.
- **Aumento de cualidades.** Las recomendaciones que realiza un sistema de recomendación sirven de entrada para el otro. Por ejemplo, un sistema de recomendación en una plataforma de streaming recomienda ciertas películas. Se puede utilizar el género o actores de esas películas como entrada del otro algoritmo de manera que mejore su rendimiento.
- **Meta niveles.** Al igual que el aumento de cualidades se utiliza lo aprendido en un sistema de recomendación como entrada el otro. Sin embargo, se reemplaza la fuente de conocimiento.

2.1.4. Otros recomendadores

Existen otros tipos de recomendadores, unos basados en conocimiento y otros basados en el perfil demográfico del usuario.

Basados en conocimiento

Se basa en la información que se tiene sobre el usuario y los productos. Se utilizan las necesidades del usuario, sin obligación de que éste haya valorado cualquier producto, además de la información que se tiene sobre este producto. Un ejemplo es una aplicación de cursos online. Esta aplicación nos pregunta cuál es nuestro campo de trabajo para poder recomendarnos cursos adecuados a nuestro perfil profesional [4].

Demográficos

En este tipo de sistemas se recomienda según el perfil demográfico del usuario, es decir, teniendo en cuenta la información del usuario que se ha recolectado previamente. Este filtro se puede realizar en base a la ciudad donde vive el usuario, su edad, sexo, profesión, estado civil u otra característica del usuario que le incluya en cualquier tipo de grupo social [13].

2.2. Dominio cruzado (Cross-domain)

Hoy en día muchas de las plataformas más conocidas en el mundo, utilizan recomendaciones de un solo dominio, como puede ser Netflix que recomienda artículos de streaming o Spotify que recomienda únicamente música. Aunque este tipo de recomendaciones pueden tener un rendimiento bastante alto, en ocasiones es necesario combinar diferentes fuentes de datos en algunos dominios para recomendar artículos a ciertos usuarios sobre los que no se dispone de demasiada información.

En las diferentes plataformas que conocemos, a los usuarios se les puede recomendar distintos tipos de artículos. Esto se debe a la gran variedad de dominios que existen. Además, hay muchas plataformas que ofrecen al usuario el mismo tipo de artículos, como el caso de HBO [1] y Netflix con películas y series. Así mismo se pueden encontrar plataformas con artículos de diferentes dominios disponibles. Un ejemplo muy claro puede ser Amazon, que contiene artículos desde cocina hasta libros.

A raíz de esto se ha creado lo que conocemos como el intercambio de recursos de origen cruzado o cross-domain. Es un mecanismo que permite mejorar las recomendaciones realizadas al usuario sobre un ítem en concreto a través de información que se ha obtenido de otros ítems de distinto dominio. En resumen, se puede utilizar información de otras plataformas u otro tipo de artículos para aumentar la eficacia del recomendador [14] [15].

[1] HBO: https://www.hbomax.com. Accedida: 2022-06-01

Ventajas

- Puede paliar parcialmente el problema de arranque en frío, el cual hemos visto en la Sección 2.1
- Mejora la precisión a la hora de hacer recomendaciones ya que se tiene más información del usuario.

Desventajas

- La cantidad de información difiere según la plataforma en la que se esté. Es por ella por lo que si introducimos datos de una plataforma de origen a otra, siendo la primera más rica en información, puede hacer que se esté aprendiendo a recomendar únicamente teniendo en cuenta los datos de la de origen.
- Puede ser que los modelos de los usuario sean completamente diferentes, provocando así un ruido en la plataforma de destino.

Tipos de dominio cruzado

Dentro del dominio cruzado se encuentran diferentes niveles de intercambio de dominio según el tipo de ítem a recomendar [16]:

- **Nivel atributo.** Ambos ítems son del mismo tipo y tienen los mismos atributos, simplemente varían en algún elemento dentro de los atributos. El ejemplo más claro puede ser el género de una película dentro de una plataforma de streaming.
- **Nivel tipo.** Se da cuando los ítems difieren en el tipo, pero comparten los mismo atributos, es decir, están relacionados únicamente por su "temática". Este es un claro ejemplo que vemos en las películas y las series. Pueden compartir muchos atributos, no obstante, no son el mismo tipo de artículo.
- **Nivel ítem.** Aquí no coinciden en el tipo de ítem, pero pueden coincidir en algún elemento. Un ejemplo es la relación entre un libro y una película, los cuales pueden tener atributos en común como puede ser el título o la fecha de publicación.
- **Nivel sistema.** Este nivel se da cuando ambos ítems son del mismo tipo pero están en diferentes plataformas. Es un ejemplo que podemos ver en una película de Netflix con respecto a una película de HBO. Ambos son del mismo tipo y comparten los mismos atributos, aunque puede diferir en la forma de almacenarlos. A lo largo de este trabajo vamos a profundizar sobre este tipo de cross domain.

Métodos de recomendación sobre dominios cruzados

Existen dos tipos de métodos de recomendación sobre dominios cruzados según como se trata la información obtenida del dominio de origen [16].

Agregación de conocimiento

Dentro de este tipo de recomendación encontramos tres métodos de agregación diferentes:

- **Agregación de preferencia del usuario.** Se encarga de añadir las preferencias que tienen los usuarios sobre un cierto conjunto de ítems. Este tipo de agregación beneficia a los nuevos usuarios.
- **Mediación de modelos del usuario.** Se basa en la recomendación de elementos a través de similitudes con el usuario del dominio de destino.
- **Combinación de recomendaciones.** Realiza recomendaciones en el dominio de origen, las cuales son transmitidas al dominio destino para realizar su respectiva recomendación. El problema de este tipo de agregación es

que podría haber solapamiento de usuarios.

Transferencia de conocimiento

- **Enlace de dominios.** Se encarga de crear una red multidominio en la que se establecen relaciones entre los distintos atributos de ambos dominios. Un inconveniente que tiene es que no supone una solución para todos los tipos de dominios cruzados.
- **Compartición de factores latentes.** Ambos dominios están relacionados a través de factores latentes compartidos (véase Subsección 2.1.1).
- **Transferencia de patrones de rating.** Analiza la información de las valoraciones para elementos del mismo nivel de ambos dominios para encontrar un patrón de preferencias. El problema de este tipo es que es computacionalmente costoso.

2.3. Puntos de interés

Un punto de interés o POI (point of interest) es un punto de ubicación geográfica específico el cual puede resultar de interés a una persona determinada. Esto puede ir desde sitios turísticos como pueden ser museos, monumentos o palacios, hasta sitios de ocio como el cine, estadio de fútbol, salón recreativo y muchos más. Un sistema de recomendación de POIs se encarga de recomendar a un usuario un lugar geográfico que visitar. Para ello se usan las mismas técnicas que en un recomendador clásico pero con algunas variaciones o especificaciones que veremos más adelante.

Hoy en día se conocen muchas aplicaciones que tratan con estos datos, una de las más conocidas es TripAdvisor [2], basada en opiniones de los usuarios. También se pueden encontrar otras aplicaciones como Foursquare y Gowalla, las cuales son conocidas como redes sociales basadas en la ubicación o location-based social networks (LBSN) [17], donde los usuarios registran los diferentes check-ins que realizan en los lugares que visitan. Sin embargo, hay otras aplicaciones que usan estos datos de diferente manera, puede ser de reserva de alojamiento como Airbnb [3] o incluso de restaurantes como TheFork [4].

En las aplicaciones relacionadas con este tipo de sistemas, es común usar la información de los usuarios para realizar recomendaciones. Esta información suele relacionarse con los check-ins que han realizado los usuarios a ciertos POIs. Un check-in es la visita de un usuario a un determinado POI en una hora determinada. Además, también se encuentran las valoraciones de los usuarios a un POI, un herramienta con la que por lo general este tipo de aplicaciones cuenta y es conocida como las reseñas [18].

En esta sección se habla de cómo se aplica lo visto con respecto a los sistemas de recomendación llevado a los POIs y la diferencia con la recomendación clásica según [19].

[2] TripAdvisor: https://www.tripadvisor.es. Accedido: 2022-06-01

[3] Airbnb: https://www.airbnb.es. Accedida: 2022-06-01

[4] TheFork: https://www.thefork.es. Accedida: 2022-06-01

2.3.1. Diferencias con la recomendación clásica

Aunque el objetivo de los sistemas de recomendación de puntos de interés sea el mismo, proporcionar POIs que puedan gustar al usuario a través de su experiencia anterior, hay diferencias importantes entre ambas áreas.

La cantidad de información que se tiene es bastante escasa. Hay que tener en cuenta que al contrario de otros sistemas, por ejemplo el de la compra online, en los sistemas basados en puntos de interés, no muchos usuarios suelen hacer uso de la misma. Esto provoca que la cantidad de información relevante que tengamos sea inferior y por lo tanto el recomendador sea menos efectivo.

En los recomendadores de POIs podemos encontrar muchos valores externos al usuario y el POI, como puede ser la posición geográfica o el momento del check-in. Esto ayuda a solventar el problema que hemos visto anteriormente, ya que al haber más valores útiles, podemos aumentar esa escasa información. Esto hace que los tipos de sistemas de recomendación vistos en la Sección 2.1 puedan variar con respecto a la recomendación clásica, aunque se empleen las mismas familias de algoritmos.

Los algoritmos más utilizados según [19] son la similitud de usuarios y factorización de matrices vistos en la Subsección 2.1.1, los híbridos visto en la Subsección 2.1.3 y otros como pueden ser el deep learning, métodos probabilísticos y basados en grafos [20].

2.4. Evaluación de los sistemas de recomendación

En todo algoritmo tiene que haber algún tipo de evaluación, es decir, alguna manera de comprobar, además de que los resultados son los esperados, que está bien implementado el algoritmo y ver su efectividad. En el caso de los sistemas de recomendación, es una manera de comprobar si tiene más efectividad un sistema basado en vecinos que uno basado en contenido, o incluso uno híbrido que se haya implementado para mejorar estos dos. Para ello, podemos encontrar distintos mecanismos de evaluación [21].

Online

Se basa en el uso del sistema por parte de usuarios reales realizando acciones reales, de esta manera se tiene en cuenta al usuario directamente. Consiste en redirigir parte del tráfico de una plataforma a otro sistema. Esto es algo que se realiza en la mayoría de las plataformas actuales ya que es una forma muy eficaz de evaluar múltiples algoritmos.

Estudio de usuarios

Esta evaluación es similar a la evaluación online, ya que se hace a través de un usuario real. Pero a diferencia de la online, se mide explícitamente el grado de satisfacción que tiene el usuario con la recomendación a través de un cuestionario. Este es un sistema bastante eficaz ya que, al igual que el online, cuenta con la respuesta directa del usuario al recomendador.

Además, este sistema de evaluación puede pedir al usuario que califique otros aspectos más específicos, como puede ser la novedad del artículo recomendado o incluso si está fuera de lo familiar. Sin embargo, tiene algún inconveniente, uno de los más importantes es que cuenta únicamente con la respuesta del usuario, y para que se realice la evaluación con la mayor exactitud posible el usuario tiene que ser sincero.

Offline

Se llama offline ya que se utiliza la información que se tiene previamente del usuario. Este tipo de evaluación puede no llegar a ser tan exacto ya que hay que confiar en que la reacción de los usuarios con respecto a la recomendación va a ser la misma. No obstante, es beneficioso ya que no requiere contacto directo con el usuario, además que se pueden comparar múltiples algoritmos con un coste bajo. Por eso nos vamos a centrar en este método.

En este método es necesario, como se ha explicado antes, tener información de las valoraciones de los usuarios a cada ítem. Esto es lo que se conoce como dataset, que suele ser un conjunto de datos perfectamente tabulados y consiste en separarlos en entrenamiento y test, de manera que usando los datos de entrenamiento, se puedan crear recomendaciones que predigan los datos de test. Además, a veces se emplea un conjunto auxiliar de validación para optimizar los parámetros de los recomendadores.

Para realizar la evaluación existen diferentes métricas vistas en [22] [23]. Éstas inicialmente medían únicamente el acierto del sistema de recomendación. Sin embargo, se vio que la evaluación es más efectiva si se tienen en cuenta otros factores como pueden ser la diversidad o la novedad [24], aunque es complicado encontrar un buen equilibrio entre las tres dimensiones. El objetivo de cada una de ellas es el siguiente:

- **Relevancia.** Tiende a medir el acierto de las recomendaciones, es decir, cuáles de las recomendaciones producidas les han gustado a los usuarios.
- **Diversidad.** Mide la cantidad de ítems que se están recomendando además de su diferencia, de manera que sean muy variados y que no se esté centrando solo en una pequeña parte del dataset.
- **Novedad.** Un recomendador novedoso es aquel que recomienda al usuario ítems que son poco conocidos.

Hay que recalcar que muchas métricas vienen definidas con abreviaciones:

k: cutoff que representa el número de ítems que se tienen en cuenta de la recomendación,

es decir, si $k = 10$, se escogen únicamente los 10 ítems con más score.

$R(u)_k$: ítems recomendados al usuario u delimitados por el cutoff.

$|\mathcal{U}|$: representa al número de usuarios recomendados.

$T(u)$: ítems que ha valorado el usuario u en la parte de test o dicho de otra forma, los que se deberían de haber recomendado.

Precisión

Ésta es una métrica de relevancia. Se basa en la calidad del sistema de recomendación, es decir, el porcentaje de elementos que hemos recomendado correctamente frente a todos los que se han recomendado. Básicamente, es una manera de medir el acierto ya que, si conseguimos una precisión alta, significa que el usuario ha recibido buenas recomendaciones. Se define por la siguiente fórmula:

$$Precision = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{|R(u)_k \cap T(u)|}{k} \tag{2.6}$$

Recall

Básicamente se representa la exhaustividad del sistema de recomendación, es decir, el número de ítems que se han recomendado correctamente, frente a los ítems que se debería de haber recomendado. Al igual que la precisión, es una métrica que mide la relevancia. Esta se define por:

$$Recall = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{|R(u)_k \cap T(u)|}{T(u)} \tag{2.7}$$

Aggregate Diversity

Esta métrica, al contrario que las anteriores, no es una métrica de relevancia, sino de diversidad [25]. Con ella se comprueba si se están recomendando elementos que son distintos. La métrica viene dada por la siguiente función:

$$AD = \cup_{u \in \mathcal{U}} R_u \tag{2.8}$$

Donde R_u representa a las recomendaciones que se le han realizado al usuario u y $\cup_{u \in \mathcal{U}}$ la unión de las recomendaciones de todos los usuarios.

User Coverage

Hay algunos recomendadores que pueden no estar devolviendo recomendaciones para algunos usuarios. Esto sucede en los recomendadores basados en vecinos, si el usuario no tiene similitud con ningún otro usuario, a ese usuario no se le recomienda nada. Para calcular la efectividad del sistema en cuanto al número de usuarios a los que se recomienda se usa el coverage, el cual te devuelve el

número de usuarios a los que se ha conseguido efectuar una recomendación.

Expected Popularity Complement

Esta métrica mide la novedad, es decir, calcula dentro de los elementos recomendados si estos han sido valorados por el resto de usuarios, por lo que si es así, serían los más populares o si en su lugar, son elementos más novedosos. Para cada usuario el EPC se define por:

$$EPC(u) = \frac{1}{k} \sum_{i_k \in R_u} (1 - p(seen|i_k)) \tag{2.9}$$

Siendo $p(seen|i_k)$ la popularidad de dicho ítem, por ello, cuanto más alto, menos novedosa es la recomendación. Sin embargo, en la formulación original se incluyen algunos componentes adicionales como un modelo de relevancia y un modelo de descuento visto en [26]. El modelo de descuento le da una mayor importancia a los ítems situados en posiciones más altas de la recomendación y tiene en cuenta la relevancia de los artículos para el usuario. No obstante, en nuestro caso nos centraremos en medir únicamente la novedad.

3

Diseño e implementación

Para explicar lo visto en el Capítulo 2, se ha implementado un framework que trabaja con los datos de dos redes sociales basadas en localización, Foursquare y Gowalla, para efectuar recomendaciones. El objetivo es crear inicialmente, diferentes algoritmos que recomienden a los usuarios de Foursquare nuevos POIs que visitar. Más adelante se agrega la información de Gowalla, para estudiar la mejora de las recomendaciones.

En este capítulo se van a nombrar los diferentes requisitos funcionales y no funcionales con los que cuenta dicho sistema además de explicar el diseño que se ha seguido para crearlo. Finalmente se va a indagar en su desarrollo, así como la organización del trabajo.

3.1. Requisitos

Para el desarrollo de este framework se eligió el lenguaje Python por las siguientes razones:

- Es un lenguaje de alto nivel.
- Tiene una gran cantidad de librerías accesibles.
- Puede utilizarse en diversos sistemas operativos.
- Es un lenguaje bastante familiar.

3.1.1. Requisitos funcionales

General

RF1. El lenguaje utilizado será Python.

RF2. El sistema será capaz de notificar cualquier tipo de error.

Tratamiento del dataset

RF3. Los datos de entrada deberán ser ficheros de texto.

RF4. Se podrán tratar los datos de diferentes datasets de entrada.

Recomendación

RF5. El sistema deberá hacer recomendaciones a un usuario dado.

RF6. El sistema podrá hacer recomendaciones a cualquier número de usuarios solicitados.

RF7. Ninguna recomendación se detendrá hasta recomendar a todos los usuarios indicados.

RF8. Se implementará un recomendador de popularidad, uno basado en en vecinos próximos (knn), un híbrido que combina el basado en usuarios, la popularidad e influencia geográfica y un último aleatorio.

RF9. Se podrán utilizar y añadir diferentes algoritmos de recomendación.

RF10. Las recomendaciones se realizarán a través de ficheros de texto.

RF11. Se podrán introducir diferentes parámetros para los recomendadores necesarios.

Evaluación

RF12. El sistema podrá evaluar cualquier tipo de recomendación, independientemente del número de recomendaciones y de usuarios recomendados.

RF13. Se implementarán diferentes métricas de evaluación, precision, recall, expected popularity complement, use coverage y aggregate diversity. Además se podrá añadir otras deseadas.

3.1.2. Requisitos no funcionales

RNF1. El sistema estará documentado en español.

RNF2. El sistema deberá ser accesible desde un repositorio Git.

RNF3. El sistema podrá funcionar en cualquier sistema operativo.

RNF4. El sistema será eficiente, realizando recomendaciones de la manera más certera y en el menor tiempo posible.

3.2. Diseño

El objetivo de esta sección es exponer el diseño que se ha seguido en este sistema. Para ello se va a exponer su estructura general así como la relación que hay entre los diferentes módulos. De igual forma se muestra el ciclo de vida que se ha empleado durante su desarrollo.

3.2.1. Estructura general

La estructura general se muestra en la Figura 3.1, en la cual se pueden diferenciar tres bloques, dataset, recomendación y evaluación, los cuales tienen funcionalidades distintas y marcan varias etapas en la ejecución. Se han creado diferentes partes con el fin de reducir el tiempo de recomendación y marcar distintas etapas, de manera que un fallo en la ejecución no signifique volver a iniciar sino retomarlo desde el último punto.

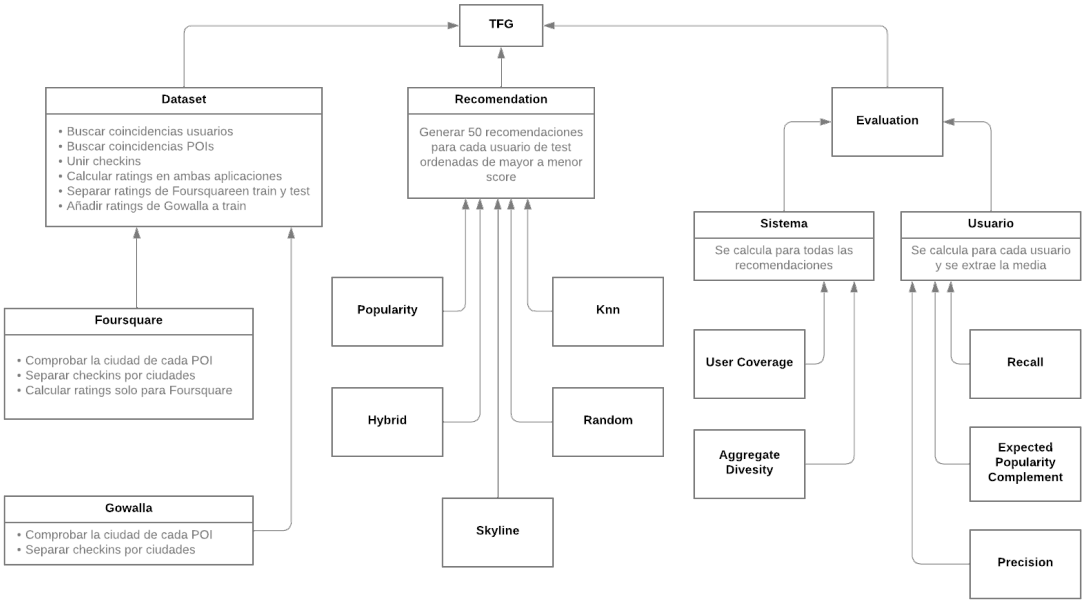

Figura 3.1: Estructura del sistema y relación entre los diferentes bloques.

Dataset

Es la primera parte en ejecutarse. Recibe un dataset en a través de ficheros de texto y se encarga de transformar su formato en el solicitado por parte de los bloques de recomendación y evaluación. Es decir, procesa los datos del dataset de forma que sean legibles por parte del recomendador.

Recomendación

Recibirá como parámetro de entrada el dataset procesado por el bloque anterior. Su tarea es realizar las recomendaciones solicitadas a través de los distintos algoritmos seleccionados, de manera que devuelva las recomendaciones en ficheros de texto. Esta parte de la implementación se basa en lo explicado en la Sección 2.1.

Evaluación

Es el último módulo del sistema. Realiza las distintas evaluaciones explicadas en la Sección 2.4. Por lo tanto se comparan los resultados de las recomendaciones devueltas por el segundo bloque,

teniendo en cuenta los diferentes parámetros que se hayan elegido para cada uno de los algoritmos.

3.2.2. Ciclo de vida

En cuanto al ciclo de vida, durante todo el desarrollo se ha seguido un modelo iterativo, mostrado en la Figura 3.2. La razón por la que se ha decidido seguir este ciclo de vida es la presencia de distintos bloques en el sistema. Cada uno de estos tiene una funcionalidad diferente, como se ha explicado anteriormente, además es muy importante el buen funcionamiento de un bloque para los posteriores en la ejecución. Es por ello que se ha escogido para cada módulo una iteración en el ciclo de vida.

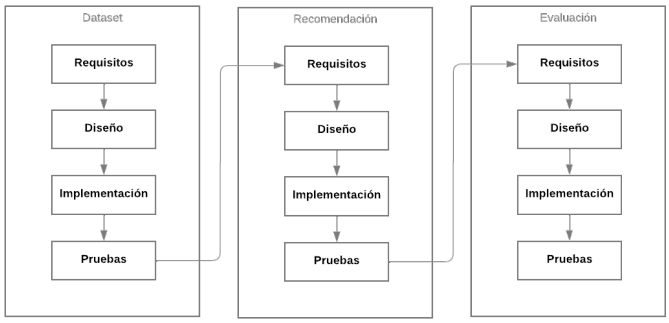

Figura 3.2: Diagrama del ciclo de vida iterativo que se ha seguido durante el desarrollo del sistema.

3.3. Desarrollo

En esta sección se va a entrar en detalle en el desarrollo de cada bloque, las funciones realizadas y la implementación de cada algoritmo. En la Figura 3.3 se muestra el flujo de desarrollo de nuestro sistema. En él podemos observar los bloques descritos en la Sección 3.2.

3.3.1. Dataset

Foursquare

cities

Este programa recibe dos ficheros de texto. Uno contiene las coordenadas y datos de las distintas ciudades como vemos en la Tabla 3.1 y otro la información de cada punto de interés de la aplicación de Foursquare mostrado en la Tabla 3.2.

Su funcionalidad es crear un fichero que contenga la información que queremos de cada POI como en la Tabla 3.3. Para ello se usó al principio una función de la librería geopy, la cual devuelve la

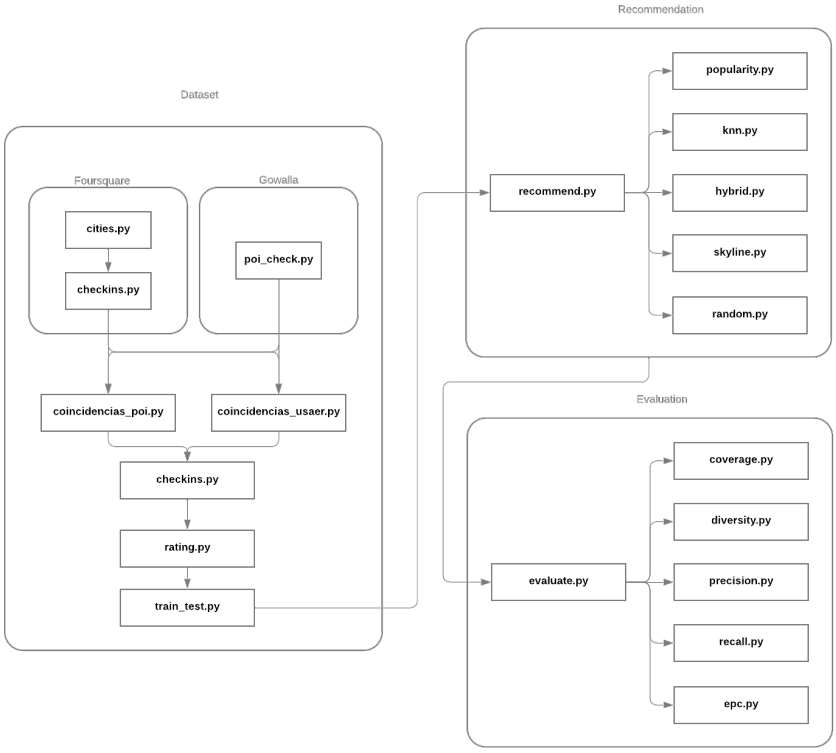

Figura 3.3: Flujo de desarrollo y ejecución de los distintos algoritmos de la aplicación.

Ciudad	Latitud	Longitud	Código país	País	Tipo ciudad
Cuiaba	-15.615000	-56.093004	BR	Brazil	Provincial capital
Veracruz	19.185003	-96.122996	MX	Mexico	Other

Tabla 3.1: Ejemplo del fichero del dataset de Foursquare con los datos de las ciudades.

POI foursquare	Latitud	Longitud	Tipo POI	Código país
3fd66200f964a52000e71ee3	40.733596	-74.003139	Jazz Club	US
4f5d15ece4b036907198c242	-33.366264	-70.678417	Bookstore	CL

Tabla 3.2: Ejemplo fichero del dataset de Foursquare con los datos de los POIs.

distancia entre dos coordenadas en km. Sin embargo, el tiempo de ejecución era muy elevado por lo que se decidió finalmente usar la fórmula Haversine.

Esta función calcula la distancia de círculo máximo entre dos puntos de la tierra conociendo la latitud y la longitud de ambos. De este modo, asignamos a cada POI la ciudad que devuelve un menor resultado. Por último, debido a la dificultad que supone trabajar con el identificador asignado por Foursquare, este programa genera un nuevo identificador de tipo entero a cada POI.

Id asignado	POI foursquare	Latitud	Longitud	Ciudad
1	3fd66200f964a52000e81ee3	40.758102	-73.975734	US_NewYork
124543	522ed80111d202c56ce7542e	35.658709	139.699850	JP_Tokyo

Tabla 3.3: Ejemplo del fichero generado con los datos de cada POI de Foursquare.

checkins

Este programa recibe un archivo con los datos de todos los check-ins de Foursquare, similar a la Tabla 3.4 y se encarga de separarlos según la ciudad a la que pertenece el POI. Para realizar esto, se localiza el POI correspondiente en el archivo generado por el anterior programa a través de su identificador.

Con el objetivo de manejar cómodamente la fecha del check-in, se decidió convertirla a formato timestamp usando la función strptime de la librería datetime. Debido al cambio horario se utiliza como fecha estándar la hora universal coordinada (UTC). Finalmente, se crea un fichero para cada ciudad con sus respectivos check-ins. Un ejemplo se puede ver en la Tabla 3.5. Estos ficheros tienen como nombre país_ciudad.

Usuario	POI	Fecha	Offset
50756	4f5e3a72e4b053fd6a4313f6	Tue Apr 03 18:00:06 +0000 2012	240
190571	4b4b87b5f964a5204a9f26e3	Tue Apr 03 18:00:07 +0000 2012	180

Tabla 3.4: Ejemplo del fichero del dataset de Foursquare con los datos de los check-ins.

Usuario	POI	Timestamp
221021	3662	1333476008
49932	7457	1333476052

Tabla 3.5: Ejemplo de los ficheros generados para cada ciudad con los sus respectivos check-ins.

Gowalla

poi_check

Esta sección se encarga de crear para el dataset de Gowalla los mismos ficheros descritos en

Foursquare. Para ello únicamente ha hecho falta un programa que recibe un fichero con todos los check-ins de Gowalla, cuyo contenido sigue el formato de la Tabla 3.6.

Usuario	Fecha	Latitud	Longitud	POI
0	2010-10-19T23:55:27Z	30.2359091167	-97.7951395833	22847
134481	2010-06-05T05:20:27Z	35.65566313	139.73511368	1226120

Tabla 3.6: Ejemplo del fichero del dataset de Gowalla con los datos de los check-ins.

Éste usa la función Haversine para calcular la distancia con las ciudades, estableciendo una distancia de 30 km para que un POI pertenezca a una ciudad dada. Para ello se utiliza el fichero con los datos de cada ciudad, visto en la Tabla 3.1, creando los ficheros correspondientes para los check-ins, al igual que Foursquare en 3.5. También se utiliza la función strptime para cambiar a formato timestamp.

Además se crea el fichero que contiene los datos de todos los POIs, con la diferencia que en Gowalla no hay que generar identificadores nuevos ya que más adelante se calculará la correspondencia con los POIs de Foursquare. Un ejemplo del contenido de este fichero es el que se muestra en la Tabla 3.7.

POI	Latitud	Longitud	Ciudad
225462	40.7268065214	-73.9837789536	US_NewYork
2040386	35.7525137647	139.641726315	JP_Tokyo

Tabla 3.7: Ejemplo del fichero generado con los datos de cada POI de Gowalla.

Coincidencias y dataset

coincidencias_poi

Este programa se encarga de crear las correspondencias de los diferentes POIs de Gowalla con los de Foursquare, que se hace a través de los ficheros de salida de programas anteriores, vistos en las Tablas 3.3 y 3.7.

Para calcular la correspondencia, se utiliza la función Haversine. Se asume que un POI de Gowalla es el mismo que un POI de Foursquare si la distancia con este es igual o inferior a 5 metros. En el caso de que haya varios POIs que cumplan esta restricción se coge el más cercano. Por último, se crea un archivo con estas correspondencias como se muestra en la Tabla 3.8.

POI Gowalla	POI Foursquare
908534	35961
245003	42196

Tabla 3.8: Ejemplo de fichero generado con las correspondencias de los POIs.

coincidencias_user

Al juntar los check-ins de Foursquare y Gowalla hay que encontrar las coincidencias de POIs entre ambas aplicaciones. No obstante, en cuanto a los usuarios, hay que saber diferenciar cual es de cada aplicación. A raíz de esto, hay que considerar que pueden coincidir los identificadores al tratarse ambos de números enteros.

Para crear nuevos identificadores, hay que calcular el número de usuarios en la aplicación de Foursquare. Después se van generando nuevos identificadores para los de Gowalla siguiendo el orden. De este modo, se crea un fichero que contiene las correspondencias de los usuarios como aparece en la Tabla 3.9.

Usuario Gowalla	Id asignado
124	266884
175	266885

Tabla 3.9: Ejemplo de fichero generado con las correspondencias de los usuarios.

all_checkins

Una vez se tienen las correspondencias y check-ins de ambas aplicaciones, hay que unirlos en sus respectivos ficheros. Para ello se recorren los check-ins de Gowalla añadiendo cada uno de ellos a los de Foursquare, mostrados en la Tabla 3.5, teniendo en cuenta las correspondencias vistas en las Tablas 3.8 y 3.9.

rating

A la hora de preparar los datos para la recomendación se necesita que los usuarios hayan valorado los POIs visitados. Con este fin se ha creado este programa que asigna como rating el número de veces que el usuario ha visitado un determinado POI. Realmente no es un rating pero como método de simplificación lo consideraremos como tal.

Una vez hecho esto, se genera un fichero para cada ciudad de Foursquare y de Foursquare con Gowalla. En la Tabla 3.10 se puede ver que se le asigna un timestamp, que corresponde con la fecha del último check-in realizado por el usuario.

Usuario	POI	Timestamp	Rating
267728	122560	1269352036	3
12494	56200	1378388196	10

Tabla 3.10: Ejemplo de fichero generado con el rating de los usuarios.

train_test

El último paso para preparar el dataset es dividir en Foursquare los ratings en training y test. La

parte de training contendrá los ratings para entrenar el modelo y test los usuarios para los que se generarán recomendaciones.

De todos los datos disponibles se seleccionó un 80 % de manera aleatoria para entrenar los recomendadores y el 20 % restante para test. Para separar estos datos se utiliza la función train_test_split de la librería de sklearn, a la que hay que indicarle como parámetro los datos en forma de dataframe. Por esta razón, se acabó usando la librería pandas para leer los ficheros con los ratings.

Finalmente, este programa genera los mismos ficheros de training para cada ciudad pero añadiendo los ratings de Gowalla, con el objetivo de comparar las recomendaciones al añadir estos datos. Los ficheros de training y test tienen el mismo formato que los mostrados en la Tabla 3.10.

3.3.2. Recomendación

En esta sección se va a explicar la implementación de los distintos algoritmos de recomendación. Todos ellos generan un fichero para cada ciudad con las recomendaciones a cada usuario que se encuentra en el conjunto de test. Se hacen teniendo en cuenta las visitas del usuario, de manera que no se puedan recomendar en ningún momento un POI que ya haya visitado.

Popularity

Es bastante usado en las plataformas de hoy en día. Se encarga de recomendar los POIs que más veces hayan visitado los usuarios en el conjunto training. Este algoritmo se ha programado de dos formas diferentes:

Con elementos repetidos. Se tiene en cuenta el número de veces que un usuario ha visitado un ítem, es decir, si un usuario ha visitado 7 veces un POI, este contabiliza como 7.

Sin elementos repetidos. No se tiene en cuenta el número de veces que un usuario ha visitado un ítem. Si un usuario ha visitado 7 veces un POI, este contabiliza como 1. Generalmente, este es el método más utilizado.

En resumen, crea un fichero para cada variación del algoritmo, con las 50 recomendaciones que tienen mayor score. Un ejemplo lo podemos ver en la Tabla 3.11.

Knn

Se encarga de calcular, para cada usuario de test, la similitud con cada usuario de training. Para calcular la similitud se decidió utilizar la distancia coseno, definida por la fórmula 2.1. Este algoritmo, como bien se define en la Subsección 2.1.1, utiliza la técnica de k-vecinos, la cual viene definida por un parámetro k representando al número de vecinos más próximos.

Usuario	POI	Score
1	5077	1,991
1	852	1,457
...	...	...
196610	971	2,704
196610	5077	1,991

Tabla 3.11: Ejemplo del fichero con las recomendaciones basadas en popularidad.

A raíz de esto, se predice el score para cada POI en función del rating de los k usuarios con más similitud. Para ello, se ha simplificado eliminando la normalización indicada en la ecuación 2.3, resultando lo siguiente:

$$\hat{r}_{ui} = \sum_{v \in \mathcal{N}_i(u)} w_{uv} r_{vi} \tag{3.1}$$

Finalmente, se genera un fichero con los 50 POIs que tienen mayor score tal y como muestra la Tabla 3.12. No obstante, puede haber usuarios que no lleguen a tener similitud con k número de vecinos o incluso con ninguno. Si se da este último caso, no se recomienda ningún POI al usuario ya que no se tiene información suficiente.

Usuario	POI	Score
1	104990	1.3172956827981417
1	4748	0.3068897124939791
...	...	...
196610	971	1.0822344613180532
196610	503	0.8964566247106883

Tabla 3.12: Ejemplo del fichero con las recomendaciones basadas en knn.

Hybrid

Para dar más variedad a este trabajo, se quiso incorporar un algoritmo híbrido, como hemos visto en la Subsección 2.1.3. Su finalidad es combinar las anteriores técnicas de recomendación (popularity y knn) junto con la influencia geográfica.

De este modo se ha desarrollado un algoritmo que calcula para cada usuario su ubicación media en la ciudad seleccionada. Una vez se tiene ese dato, se mide la distancia que tiene con cada POI con Haversine y finalmente, se seleccionan los 50 POIs con mayor score.

Como podemos comprobar, el rango que resulta en los algoritmos son muy distintos, pudiendo llegar en popularidad a 2,704 como vemos en la Tabla 3.11 y en knn a 1.31 como muestra la Tabla

3.12. Por ese motivo se han normalizado los resultados de tal manera que oscilan entre 0 y 1.

Una vez normalizados, se incluyen todas las recomendaciones en una lista. En el caso de que un POI aparezca en más de una recomendación, se suman los resultados. Para acabar, se crea un fichero recomendando los 50 POIs con más score para cada usuario. Un ejemplo de este fichero lo podemos ver en la Tabla 3.13.

Usuario	POI	Score
32768	7330	2.0
32768	6537	1.0704504213734747
...	...	...
2	6537	1.1816114727409721
2	11547	1.0

Tabla 3.13: Ejemplo del fichero con las recomendaciones basadas en el sistema híbrido.

Random

Para comprobar que están bien implementados los algoritmos, se ha generado una recomendación que devuelve 50 POIs aleatorios para cada usuario, siempre y cuando el usuario no los haya visitado en el conjunto de training. Al devolver resultados aleatoriamente, no tiene en cuenta ningún score, por lo que el fichero de recomendación sería como el de la Tabla 3.14.

Usuario	POI
32768	7330
32768	6537
...	...
2	6537
2	11547

Tabla 3.14: Ejemplo del fichero con las recomendaciones aleatorias.

Skyline

El objetivo es comparar los algoritmos implementados con otro que realice recomendaciones perfectas, así es más fácil evaluar los resultados. Concretamente, a cada usuario se le recomiendan todos los POIs que ha visitado en el conjunto de test pero, al igual que el resto de algoritmos, sólo si el usuario no ha visitado cierto POI en el conjunto de training. El fichero generado es similar al de la Tabla 3.14.

3.3.3. Evaluación

En este bloque se detalla la implementación de las distintas métricas que se han desarrollado para evaluar los algoritmos de recomendación, descritas en la Sección 2.4. En esta se explica que hay tres tipos de evaluaciones: online, estudio de usuarios y offline. Para este sistema nos hemos centrado en la evaluación offline, ya que no tenemos contacto directo con los usuarios de Foursquare ni de Gowalla, pero si disponemos de un dataset para cada una de las redes sociales basadas en localización.

Además, observando la Figura 3.1, se ve que hay dos tipos de métricas. Las primeras son a nivel de usuario (precisión, recall y EPC), que se calcula con cada usuario para después aplicar la media con todos los usuarios que hemos conseguido recomendar. Las segundas son a nivel de sistema (user coverage y aggregate diversity), las cuales se realizan para todos los usuarios recomendados. Para estudiar estas métricas, se ha creado el programa evaluate que llama a cada una indicando el algoritmo que se quiere evaluar y enviando los ficheros con las recomendaciones correspondientes. En la Tabla 3.15 podemos ver un ejemplo con la salida, en concreto del recomendador knn.

Métrica	Valor
Precisión	0.02724797738025263
Recall	0.07438767201678988
EPC	0.9725571694179922
Aggregate Diversity	7,194
User Coverge	2,268

Tabla 3.15: Ejemplo de salida de la evaluación de los recomendadores, en concreto del recomendador knn.

4

Pruebas y resultados

En este capítulo se analizan los resultados obtenidos en el framework desarrollado. Para ello se comparan los distintos algoritmos implementados a través de las métricas de evaluación. El objetivo es conseguir una mejora de las recomendaciones al utilizar la técnica del dominio cruzado, vista en la Sección 2.2.

Para realizar estas pruebas se ha utilizado un dispositivo con las siguientes características:

Sistema operativo: Ubuntu 20.04.2 LTS.

Procesador: Intel® Core™ i7-7500U CPU @ 2.70GHz × 4.

Versión de Python: 3.8.10

4.1. Dataset

Como se ha comentado anteriormente, se han seleccionado las aplicaciones de Gowalla y de Foursquare para estudiar nuestros algoritmos. Ambas aplicaciones ofrecen información acerca de los check-ins que los usuarios han realizado. Sin embargo, en la Tabla 4.1 podemos encontrar diferencias en ambos datasets en cuanto al número de datos que podemos obtener.

	Usuarios	POIs	Check-ins	Ciudades
Foursquare	266,909	3,680,126	33,263,633	415
Gowalla	196,586	1,280,969	6,442,892	-

Tabla 4.1: Número de datos que podemos encontrar en los datasets de cada aplicación.

En esta tabla se ve que el número de ciudades que contiene Gowalla es desconocido. No supone ningún problema ya que obtendremos la correspondencia analizando los datos de Foursquare.

Aparte del número de datos que proporciona cada aplicación, hay un cambio en la forma de presentar estos datos, como se explica en la Subsección 3.3.1. Foursquare proporciona un dataset separado en 3 ficheros diferentes, uno con los datos de cada ciudad, otro con los datos de cada POI, y el último

con la información de cada check-in. Mientras tanto, Gowalla nos proporciona únicamente un fichero cuyo contenido equivale al de la Tabla 3.6.

Filtro de ciudades

Los datos que se han mostrado corresponden con el dataset entero. Sin embargo, al haber una gran cantidad de información, se han seleccionado únicamente tres ciudades. El objetivo es reducir el tiempo de ejecución en los recomendadores ya que con 415 ciudades algunos algoritmos podrían ser intratables para el entorno de pruebas. Con este fin se realizó un estudio de qué ciudades cuentan con más check-ins. En la Tabla 4.2 se muestra el ranking de las ciudades en función de los check-ins que proporcionan. De todas las ciudades se han seleccionado Tokio, Nueva York y San Francisco.

Ranking	Ciudad
7	Tokio
12	Nueva York
53	San Francisco

Tabla 4.2: Ranking de algunas ciudades con mayor número de check-ins.

La razón de estudiar estas ciudades, en el caso de Nueva York y Tokio, es que son bastante turísticas y cuentan con muchos datos. En cuanto a San Francisco, se realizó un filtro entre tres ciudades (Chicago, Los Ángeles y San Francisco) analizando el número de datos que cada una podría añadir de Gowalla.

Foursquare

En la Tabla 4.3 se puede observar el número de datos que se obtienen de cada ciudad, los cuales, son más elevados en Tokio.

	Usuarios	POIs	Check-ins	Rating
Tokio	12,464	83,189	1,030,090	474,150
Nueva York	15,785	41,386	380,247	210,192
San Francisco	6,682	13,104	106,648	81,419

Tabla 4.3: Datos que se obtienen de cada ciudad de Foursquare.

No obstante, puede variar a la hora de incluir los datos de Gowalla debido a la cantidad de ratings que pueda aportar en el conjunto de training. Este incremento de datos se muestra en la Tabla 4.4, donde Ratings Foursquare representa al número de ratings que se asignan al conjunto de training. Cada ciudad aporta los datos de una forma diferente por lo que tendremos diferentes opciones a la hora de interpretar los resultados. Mientras que Foursquare tiene mucha información sobre Tokio, Gowalla aporta un gran porcentaje con San Francisco.

	Ratings Foursquare	Aumento	% de aumento
Tokio	379,320	4,100	1.08 %
Nueva York	168,153	10,698	6.36 %
San Francisco	52,340	15,994	30.55 %

Tabla 4.4: Número y porcentaje de datos que añade Gowalla a Foursquare para cada ciudad.

Gowalla

A lo largo de las pruebas, por razones que se explicarán más adelante, se evaluó además la mejora de las recomendaciones al añadir los datos de Foursquare a los de Gowalla, es decir, a la inversa de antes. Para ello, se realizó el mismo procedimiento a la hora de preparar los datasets. En el caso de Gowalla, se tienen inicialmente los datos mostrados en la Tabla 4.5, lo que puede explicar por qué en el caso anterior San Francisco aumenta un 30.55 % los ratings en el conjunto de training, visto en la Tabla 4.4.

	Usuarios	POIs	Check-ins	Rating
Tokio	849	7,309	29,784	16,290
Nueva York	6,111	13,755	97,154	71,106
San Francisco	7,113	14,485	185,941	120,406

Tabla 4.5: Datos que se obtienen de cada ciudad de Gowalla.

En cuanto al aumento del rating, se puede observar en la Tabla 4.6 (donde Ratings Gowalla representa a los ratings que se encuentran en el conjunto de test), que la ciudad de San Francisco no mejora tanto como en el anterior caso. Sin embargo, Tokio y Nueva York aumentan considerablemente.

	Rating Gowalla	Aumento	% de aumento
Tokio	13,032	2,795	21.45 %
Nueva York	56,884	16,272	28.61 %
San Francisco	96,324	9,366	9.72 %

Tabla 4.6: Número y porcentaje de datos que añade Foursquare a Gowalla para cada ciudad.

4.2. Selección de parámetros

Los algoritmos se han implementado de manera que pueda cambiar la recomendación en función de algunos parámetros. Por ello se ha estudiado el rendimiento de cada uno de estos para escoger únicamente un valor para cada uno. Además, como se ha explicado en la Subsección 3.3.3 algunas métricas de evaluación, en concreto aquellas que son a nivel de usuario, dependen de un cutoff, que indica el top de recomendaciones que se van a evaluar. Para elegir el parámetro a utilizar, se ha

pensado que lo más conveniente sería tener en cuenta la precisión, ya que esta se basa en la calidad del recomendador.

4.2.1. Número de recomendaciones a estudiar

Para la elección del cutoff se analizó cómo mejora el algoritmo según los diferentes valores (10, 20, 30, 40 y 50). Además la media de check-ins en test es de 4 por usuario, por lo que decidió usar un cutoff de 10. Además en recomendación trabajar con un cutoff bajo es una medida estándar para no sobresaturar de información al usuario.

4.2.2. Popularidad repetida

En la Subsección 3.3.2 se observa que para el caso de popularidad se han desarrollado dos variaciones dependiendo de si se tienen en cuenta los elementos repetidos. En la Figura 4.1 se muestran los diferentes resultados de precisión. Viendo esta evaluación, se llegó a la conclusión de que es más efectivo el algoritmo si no se tienen en cuenta los elementos repetidos. Además, esta versión corresponde con la popularidad clásica.

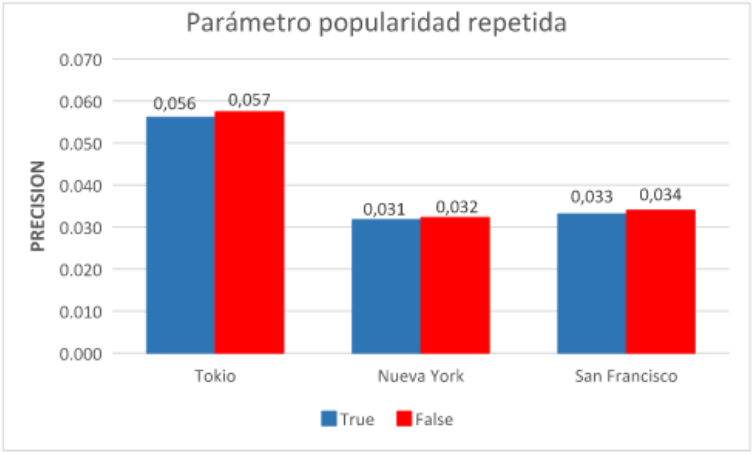

Figura 4.1: Evaluación de los distintos parámetros para cada ciudad en popularidad.

4.2.3. K vecinos

En el caso del recomendador knn e híbrido, es necesario un parámetro k que indique el número de vecinos con más similitud con los que se efectúa la recomendación. Para escoger su valor se ha realizado el mismo procedimiento que en popularidad. En la Figura 4.2 se muestra la precisión obtenida para cada valor de k en knn. En ella podemos ver que en el caso de Tokio y Nueva York, cuantos más vecinos se tienen en cuenta, más precisión tenemos con el recomendador. Por esa razón, se ha

decidido usar un k de 100 para realizar las pruebas.

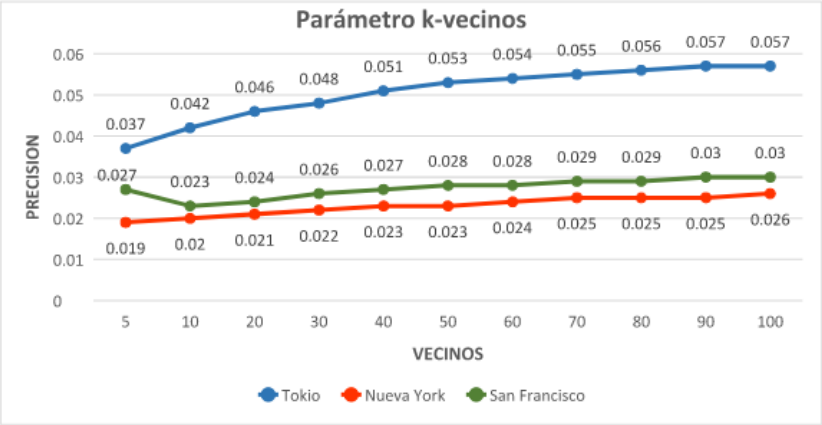

Figura 4.2: Evaluación de los distintos números de vecinos para el recomendador knn.

4.3. Resultados

En esta sección vamos a exponer y explicar los resultados de la evaluación descrita en la Subsección 3.3.3 aplicando el dominio cruzado, descrito en la Sección 2.2, además de las dificultades que se han encontrado a lo largo del desarrollo. En un principio se han realizado recomendaciones para los datos de Foursquare y se ha visto cómo afectaba añadir las correspondencias de Gowalla. Sin embargo, no se obtuvieron los resultados esperados, los cuales veremos a continuación, por lo que también se analizó a la inversa.

4.3.1. Resultados obtenidos al integrar los datos de Gowalla en Foursquare

En primer lugar, se estudiaron las recomendaciones para Tokio y Nueva York, las cuales cuentan respectivamente con 10,612 y 9,925 usuarios en el conjunto de test.

En la Tabla 4.7 figuran los resultados que se han obtenido al evaluar los recomendadores en la ciudad de Tokio. En ella se ve que no hay ninguna mejora en términos de precisión y recall. En cuanto a aggregate diversity y EPC, sí que hay un aumento, ya que hay más variedad en la recomendación y los usuarios nuevos tienden a visitar POIs que son poco populares.

User coverage va a ser similar para todas las ciudades. Es una métrica que no va a llegar a empeorar ya que solo cambia para dos casos. En skyline hay algunos POIs que no se recomiendan porque no aparecen en el conjunto de training. Al haber nuevos check-ins, puede que estos usuarios hayan visitado algún POI que antes no se había visitado antes, por lo que pasa a ser recomendado. Para

knn, existen usuarios que no tienen similitud con otros, por lo que no se le llega a recomendar nada. Al aparecer nuevos usuarios, puede que estos sean similares.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	7.5386355e-05	7.8790169e-05	0.9995846	74,380	10,612
Skyline	0.5794882	0.8377626	0.972587	24,351	10,472
Popularity	0.0574727	0.0904425	0.8536096	95	10,612
Knn	0.0573521	0.0887028	0.9045899	13,351	10,378
Hybrid	0.0504994	0.0817951	0.8915544	46,234	10,612

(a) Evaluación para Foursquare en Tokio.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	1.3192612e-04	1.2714543e-04	0.9996105	74,489	10,612
Skyline	0.5799179	0.8385205	0.9738657	24,454	10,472
Popularity	0.0575009	0.0907488	0.8604287	95	10,612
Knn	0.0573562	0.0886455	0.9092653	13,464	10,379
Hybrid	0.0505089	0.0817766	0.8966666	46,284	10,612

(b) Evaluación para Foursquare agregando Gowalla en Tokio.

Tabla 4.7: Resultados obtenidos de la evaluación en Tokio añadiendo Gowalla a Foursquare.

En la Tabla 4.8 aparecen los resultados que se han obtenido al evaluar los recomendadores en la ciudad de Nueva York. Se puede observar que los recomendadores empeoran en precisión y recall. No obstante aumenta en EPC y aggregate diversity. Esto se debe a que los usuarios nuevos de Nueva York tienden a visitar POIs menos populares, por lo que aumenta la variedad y la novedad, pero recomendando POIs menos acertados.

Vistas ambas ciudades, en ninguna encontramos lo que esperamos, por lo que se decidió añadir una ciudad más. Las elecciones eran Chicago, Los Ángeles y San Francisco y se hizo un estudio sobre qué ciudad aumentaría más el número de ratings de tres ciudades. Finalmente, se decidió incluir San Francisco, el cual cuenta con 3,682 usuarios en el conjunto de test a los cuales se va a recomendar.

En la Tabla 4.9 se observan los resultados que se han obtenido al evaluar los recomendadores en la ciudad de San Francisco. En esta tabla se ve que ocurre lo mismo que en Tokio y Nueva York. Únicamente mejora el EPC, aggregate diversity y user coverage. Sin embargo, precision y recall empeoran.

Como se puede ver, no se han encontrado las mejoras que buscábamos en ninguna de las tres ciudades, por lo que se decidió realizar el mismo experimento a la inversa. Para ello se utilizaron los ficheros vistos en las Tablas 3.6 y 3.7, con los que se realizó el mismo procedimiento para separar estos datos en los conjuntos de training y test. Después se realizaron las correspondencias de los datos de Foursquare y se juntaron para estudiar el cambio en las recomendaciones.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	9.0680101e-05	2.3381712e-04	0.9996865	36,604	9,925
Skyline	0.336668	0.9170562	0.9808087	11,278	9,676
Popularity	0.0323526	0.1390475	0.9251687	71	9,925
Knn	0.0253652	0.09184149	0.9683149	19,088	8,488
Hybrid	0.0273552	0.1189315	0.9481526	31,703	9,925

(a) Evaluación para Foursquare en Nueva York.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	5.0377834e-05	8.5660342e-05	0.9997322	36,718	9,925
Skyline	0.3370611	0.9181243	0.9841562	11,392	9,684
Popularity	0.0322921	0.1390092	0.9382898	72	9,925
Knn	0.0251765	0.0913275	0.9740890	19,328	8,496
Hybrid	0.0272544	0.1185395	0.9574267	31,570	9,925

(b) Evaluación para Foursquare agregando Gowalla en Nueva York.

Tabla 4.8: Resultados obtenidos de la evaluación en Nueva York añadiendo Gowalla a Foursquare.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	2.1727322e-04	3.5251896e-04	0.9992889	11,707	3,682
Skyline	0.2829165	0.9300513	0.9456978	4,065	3,559
Popularity	0.0340576	0.1760944	0.937286	63	3,682
Knn	0.0287088	0.1314895	0.9632036	6,831	2,859
Hybrid	0.0269419	0.1510288	0.9580159	11,594	3,682

(a) Evaluación para Foursquare en San Francisco.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	2.9875067e-04	9.1629808e-04	0.9994389	11,800	3,682
Skyline	0.284119	0.9332986	0.964609	4,158	3,559
Popularity	0.0314503	0.1658245	0.9443126	71	3,682
Knn	0.0274222	0.1271734	0.9726978	7,133	2,912
Hybrid	0.025774	0.1471707	0.9623470	11,534	3,682

(b) Evaluación para Foursquare agregando Gowalla en San Francisco.

Tabla 4.9: Resultados obtenidos de la evaluación en San Francisco añadiendo Gowalla a Foursquare.

4.3.2. Resultados obtenidos al integrar los datos de Foursquare en Gowalla

De esta manera parecía que iba a dar mejores resultados ya que el dataset de Gowalla es más pequeño, por lo que al añadir los datos de Foursquare, éste aumenta considerablemente. Para ello se han utilizado las mismas ciudades descritas anteriormente aunque cuentan con menos usuarios a recomendar. En Tokio se recomiendan a 551 usuarios, en Nueva York a 3,607 y en San Francisco a 4,539.

En la Tabla 4.10 figuran los resultados que se han obtenido al evaluar los recomendadores en la ciudad de Tokio. En la Tabla 4.7 vimos que para Tokio no había ninguna mejora en términos de precisión en los resultados obtenidos. Lo mismo pasa cuando añadimos los ratings de Foursquare a Gowalla.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	1.6333938e-03	4.1474538e-03	0.9973428	6,283	551
Skyline	0.3290581	0.7996931	0.9832436	1,373	499
Popularity	0.03411978	0.0740604	0.9233216	95	551
Knn	0.0373016	0.0889116	0.9535372	2,163	504
Hybrid	0.0333938	0.0858767	0.9433859	4,165	551

(a) Evaluación para Gowalla en Tokio.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	3.6297641e-04	3.7269899e-04	0.9993243	6,439	551
Skyline	0.338189	0.8166799	0.9956476	1,538	508
Popularity	0.007441	0.0147034	0.9281864	67	551
Knn	0.032874	0.0806071	0.981821	2,281	508
Hybrid	0.0174229	0.0523354	0.954302	4,148	551

(b) Evaluación para Gowalla agregando Foursquare en Tokio.

Tabla 4.10: Resultados obtenidos de la evaluación en Tokio añadiendo Foursquare a Gowalla.

En la Tabla 4.11 se observan los resultados que se han obtenido al evaluar los recomendadores en la ciudad de Nueva York. Sigue dando malos resultados, al igual que con Tokio. Pero queda por comprobar San Francisco por lo que todavía no se pueden extraer conclusiones.

En la Tabla 4.12 aparecen los resultados que se han obtenido al evaluar los recomendadores en la ciudad de San Francisco. La mejora sigue siendo negativa. Podemos concluir con que con el aumento de los datos de Gowalla a través de Foursquare no devuelve el resultado esperado.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	3.0496257e-03	7.9266318e-04	0.9992413	12,764	3,607
Skyline	0.2821307	0.940221	0.9810545	5,332	3,520
Popularity	0.0380926	0.1467304	0.9256439	77	3,607
Knn	0.0428059	0.13615	0.959412	7,490	3,065
Hybrid	0.0372054	0.1430849	0.9448733	11,794	3,607

(a) Evaluación para Gowalla en Nueva York

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	2.772387e-04	9.5812299-04	0.9995265	12,853	3,607
Skyline	0.2834043	0.9438676	0.9907201	5,421	3,525
Popularity	0.031023	0.1238669	0.9624801	93	3,607
Knn	0.0416368	0.1318707	0.981303	7,803	3,067
Hybrid	0.0370945	0.142112	0.9735638	11,579	3,607

(b) Evaluación para Gowalla agregando Foursquare en Nueva York

Tabla 4.11: Resultados obtenidos de la evaluación en Nueva York añadiendo Foursquare a Gowalla.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	4.4062568e-04	1.0065721-03	0.9989523	13,678	4,539
Skyline	0.348474	0.9363187	0.9740478	7,211	4,489
Popularity	0.0320115	0.1124272	0.9294165	94	4,539
Knn	0.0403656	0.117811	0.9613333	7,036	3,879
Hybrid	0.0320776	0.1106193	0.9562269	12,445	4,539

(a) Evaluación para Gowalla en San Francisco.

	Precision	Recall	EPC	Aggregate diversity	User coverage
Random	2.4234413e-04	2.9015108e-04	0.9992082	13,716	4,539
Skyline	0.348553	0.9368733	0.9810598	7,249	4,492
Popularity	0.0292575	0.1039796	0.9422459	96	4,539
Knn	0.0393658	0.1135853	0.9711517	7,552	3,939
Hybrid	0.0307997	0.1086237	0.9637322	12,517	4,539

(b) Evaluación para Gowalla agregando Foursquare en San Francisco.

Tabla 4.12: Resultados obtenidos de la evaluación en San Francisco añadiendo Foursquare a Gowalla.

4.3.3. Análisis de los resultados

No se ha conseguido una mejora en los recomendadores en relación con el acierto al introducir nuevos datos como se esperaba. Sin embargo, se ha conseguido por lo general una mejora en cuanto a diversidad y novedad. A raíz de esto, se ha hecho un estudio para poder explicar por qué no encontramos los resultados esperados. Este estudio se ha realizado solo para el aumento de los datos de Foursquare.

Lo primero fue comprobar si los lugares más visitados en el conjunto de training cambiaban al añadir nuevos check-ins, por lo que se seleccionaron los 30 POIs más populares. Se obtuvo un cambio bastante alejado del top en el caso de Tokio, donde el top 20 seguía siendo el mismo. En el caso de Nueva York no cambiaba hasta el top 16. Mientras tanto, en San Francisco el número 3 ya es diferente. También se ha calculado cual es el porcentaje de coincidencia en los POIs más visitados. En él no se ha tenido en cuenta el orden, por lo que si un POI se encuentra en el top 2 y pasa a ser el top 25 no se aprecia ningún cambio. En Tokio se ha visto un 93.33 % de coincidencia, en Nueva York un 80 % y en San Francisco un 33.33 %.

Además, para poder explicar porqué no mejora el recomendador knn, se ha hecho un análisis de los usuario que cambian los vecinos. Primero se ha comprobado el porcentaje de usuarios cuyos k vecinos varían, devolviendo el porcentaje de estos. En el caso de Tokio cambia para un 12.54 % de los usuarios, en Nueva York un 14.79 % y en San Francisco un 28.98 %. Después se ha comprobado los usuarios que cambian mayor número de vecinos por ciudad. En Tokio un usuario cuenta con 36 vecinos diferentes al añadir los datos de Gowalla, mientras que en Nueva York llega a haber una variación de 100 y en San Francisco de 93.

En vista de estos cambios, se ha comprobado que los nuevos lugares que entran en el top son de uso diario como pueden ser tiendas, estaciones, oficinas o urbanizaciones residenciales. Esto nos puede demostrar que los check-ins de Gowalla están relacionados con acciones de la vida cotidiana, mientras que en Foursquare, los sitios más visitados son aeropuertos, monumentos o estadios entre otros. Se ha estudiado el caso de San Francisco ya que genera mayores cambios, cuyo ejemplo podemos ver en la Tabla 4.13. Cabe destacar que las coordenadas del POI 2527 corresponden con las del Golden Gate y que al añadir los datos de Gowalla no aumenta su popularidad.

POI	Top	Descripción
737	1	Aeropuerto
24820	2	Estadio baseball
2527	3	Puente

(a) Top POIs en la recomendación de Foursquare.

POI	Top	Descripción
1043	3	Tienda electrónica
57364	4	Gym
19101	6	Plaza

(b) Nuevos POIs en el top al añadir Gowalla.

Tabla 4.13: Ejemplos de POIs que se encuentran en el top para ambas aplicaciones en San Francisco.

5

Conclusiones y trabajo futuro

En esta sección se exponen las conclusiones a las que se han llegado a lo largo del proyecto. Además se hace un breve resumen de los posibles trabajos a realizar en el futuro.

5.1. Conclusiones

El avance de la tecnología hace que crezca cada vez más el servicio online en la mayoría de empresas. Esto conlleva a una gran competencia entre ellas y por lo tanto, que sea más común querer disponer del sistema de recomendación más efectivo. A raíz de esto, la investigación en los sistemas de recomendación resulta especialmente importante y el continuo estudio sobre este campo, ha hecho que aparezcan nuevas técnicas como el dominio cruzado.

Durante este trabajo se ha diseñado un framework que permita realizar recomendaciones de puntos de interés. Además se ha desarrollado un sistema que evalúe estas recomendaciones mediante distintas métricas. El objetivo era demostrar que a través del dominio cruzado aumenta la eficacia de los recomendadores al disponer de más datos de usuarios de otras aplicaciones, en este caso usando Foursquare y Gowalla. Sin embargo, no se han obtenido los resultados esperados. Esto se debe a la diferencia de los check-ins que se dan en ambas aplicaciones. Mientras que en Foursquare los POIs más populares son sitios de visita turística, en Gowalla equivalen a sitios de la vida cotidiana.

A pesar de este resultado negativo en términos de relevancia, se ha visto que los recomendadores son bastante eficaces, devolviendo una mejor respuesta que al recomendar puntos de interés de forma aleatoria. Además, los resultados parecen indicar que las recomendaciones mejoran en términos de diversidad y novedad.

Asimismo se ha comprobado que un recomendador basado en popularidad es muy efectivo ya que los usuarios tienden a visitar los lugares conocidos, aunque esto haga que no descubran nuevos lugares. Por consecuencia se ha evaluado el algoritmo basado en similitud de usuarios, el cual ha respondido de una forma esperada, recomendando puntos de interés que al usuario le interesan con una variedad y novedad bastante elevada. Por último se ha desarrollado un algoritmo híbrido simple el cual se basa en la posición geográfica media del usuario. Este ha demostrado que junto con los otros

recomendadores, es eficiente y aumenta considerablemente la variedad de recomendación.

De igual manera, el desarrollo de las diferentes métricas de evaluación ha resultado ser efectivo. No solo se han implementado las más conocidas como la precisión o el recall, sino que también se han estudiado otros componentes como pueden ser la diversidad, la novedad e incluso el número de usuarios a los que se ha conseguido realizar la recomendación, lo cual ha supuesto un estudio más completo de los diferente recomendadores.

Por último, el código de el framework desarrollado se encuentra en el siguiente repositorio: `https://github.com/pascuu7/TFG`.

5.2. Trabajo futuro

Como trabajo futuro se ha planteado continuar aplicando esta técnica utilizando otros datos, ya sea otras ciudades u otras aplicaciones que se basen en puntos de interés, de tal forma que se consiga realizar una mejora efectiva y notoria. Para empezar se podría hacer una comparativa entre ciudades dependiendo del número de POIs que proporcionen y hacer un estudio de la mejora de los recomendadores a medida que aumenta el tamaño del dataset. Además se podría hacer el mismo análisis con los distintos parámetros que se han explicado de los recomendadores desarrollados.

Otra opción sería aplicar otros tipo recomendadores que hemos visto en el Capítulo 2, como redes neuronales o factorización de matrices. Igualmente, en cuanto al recomendador basado en la similitud de los vecinos, una alternativa interesante sería comparar los diferentes métodos de calcular la distancia entre usuarios.

También se ha planteado aplicar este proyecto a recomendación de rutas turísticas, de manera que se tenga en cuenta la cercanía de los puntos de interés. Un trabajo muy interesante acerca de esto es [7] que estudia la recomendación de puntos de interés a raíz del análisis de los patrones de movimiento de los usuarios.

Bibliografía

[1] "¿cómo es el acceso a internet en el mundo?." https://blog.orange.es/noticias/acceso-internet-mundo/. Accessed: 2022-06-12.

[2] "Digital report 2022: El informe sobre las tendencias digitales, redes sociales y mobile." https://wearesocial.com/es/blog/2022/01/digital-report-2022-el-informe-sobre-las-tendencias-digitales-redes-sociales- Accessed: 2022-06-12.

[3] "El imparable crecimiento de la tienda online." https://www.activions.com/es/blog/el-imparable-crecimiento-de-la-tienda-online. Accessed: 2022-06-12.

[4] P. Sánchez Pérez *et al.*, "Estudio y aplicación de algoritmos y estructuras de datos a los sistemas de recomendación," B.S. thesis, 2016.

[5] A. Bellogín, "Recommender systems," in *Recommender system performance evaluation and prediction: Information retrieval perspective*, pp. 17–35, 2012.

[6] S. M. G. Nieto, "Filtrado colaborativo y sistemas de recomendación," *Inteligencia en Redes de Comunicaciones. Madrid*, 2007.

[7] S. Torrijos López de la Manzanara, "Estudio de métodos de detección de patrones de movimiento para sistemas de recomendación turístico," B.S. thesis, 2020.

[8] C. Desrosiers and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods," *Recommender systems handbook*, pp. 107–144, 2011.

[9] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Computing Surveys (CSUR)*, vol. 52, no. 1, pp. 1–38, 2019.

[10] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[11] L. N. Tondji, "Web recommender system for job seeking and recruiting," *Partial Fulfillment of a Masters II at AIMS*, 2018.

[12] M. Seguido Font, "Sistemas de recomendación para webs de información sobre la salud," Master's thesis, Universitat Politècnica de Catalunya, 2009.

[13] M. J. Pazzani, "A framework for collaborative, content-based and demographic filtering," *Artificial intelligence review*, vol. 13, no. 5, pp. 393–408, 1999.

[14] I. Cantador, I. Fernández-Tobías, S. Berkovsky, and P. Cremonesi, "Cross-domain recommender systems," in *Recommender systems handbook*, pp. 919–959, Springer, 2015.

[15] I. Cantador and A. Bellogín, "Sistemas de recomendación sobre dominios cruzados," 2020.

[16] I. Fernández-Tobías, "Matrix factorization models for cross-domain recommendation: Addressing the cold start in collaborative filtering," 2017.

[17] M. Ye, P. Yin, W.-C. Lee, and D.-L. Lee, "Exploiting geographical influence for collaborative point-of-interest recommendation," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 325–334, 2011.

[18] M. Hang, I. Pytlarz, and J. Neville, "Exploring student check-in behavior for improved point-of-interest prediction," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 321–330, 2018.

[19] P. Sánchez and A. Bellogín, "Point-of-interest recommender systems based on location-based social networks: A survey from an experimental perspective," *ACM Computing Surveys (CSUR)*, 2022.

[20] M. Caro Martínez, "Sistemas de recomendación basados en técnicas de predicción de enlaces para jueces en línea," p. 46, 2017.

[21] J. Beel and S. Langer, "A comparison of offline evaluations, online evaluations, and user studies in the context of research-paper recommender systems," in *International conference on theory and practice of digital libraries*, pp. 153–168, Springer, 2015.

[22] A. Bellogín, "Evaluating performance in recommender systems," in *Recommender system performance evaluation and prediction: Information retrieval perspective*, pp. 37–76, 2012.

[23] P. Sanchez Pérez *et al.*, "Recommender systems," in *Exploring attributes, sequences, and time in Recommender Systems: From classical to Point-of-Interest recommendation*, pp. 13–34, 2021.

[24] A. Gunawardana, G. Shani, and S. Yogev, "Evaluating recommender systems," in *Recommender systems handbook*, pp. 547–601, Springer, 2022.

[25] G. Adomavicius and Y. Kwon, "Maximizing aggregate recommendation diversity: A graph-theoretic approach," in *Proc. of the 1st International Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011)*, pp. 3–10, Citeseer, 2011.

[26] S. Vargas and P. Castells, "Rank and relevance in novelty and diversity metrics for recommender systems," in *Proceedings of the fifth ACM conference on Recommender systems*, pp. 109–116, 2011.


UÁM
Universidad Autónoma
de Madrid