Universidad Autónoma de Madrid

Escuela Politécnica Superior

21|22

Trabajo fin de máster

Recomendación de localizaciones de negocio usando aprendizaje automático

Jorge Gutiérrez Díaz

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

Proyecto para la obtención del título de
Máster en Investigación e Innovación en
Inteligencia Computacional y Sistemas Interactivos
por la Universidad Autónoma de Madrid

Tutor del trabajo fin de máster:
Alejandro Bellogín Kouki

Recomendación de localizaciones de negocio usando aprendizaje automático

Jorge Gutiérrez Díaz

Todos los derechos reservados.
Queda prohibida, salvo excepción prevista en la Ley, cualquier
forma de reproducción, distribución comunicación pública
y transformación de esta obra sin contar con la autorización
de los titulares de la propiedad intelectual.
La infracción de los derechos mencionados puede ser constitutiva de
delito contra la propiedad intelectual *(arts. 270 y sgts. del Código Penal).*

DERECHOS RESERVADOS
© 3 de Septiembre de 2021 por UNIVERSIDAD AUTÓNOMA DE MADRID
Francisco Tomás y Valiente, n^o 1
Madrid, 28049
Spain

Jorge Gutiérrez Díaz
Recomendación de localizaciones de negocio usando aprendizaje automático

Jorge Gutiérrez Díaz
C\ Francisco Tomás y Valiente Nº 11

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

A mi familia y amigos por hacer posible ser lo que soy hoy

La experiencia es el nombre que le damos a nuestros errores.

Oscar Wilde

Agradecimientos

En primer lugar me gustaría dar las gracias a Alejandro Bellogín por confiar en mí para el desarrollo de este proyecto y en su apoyo a lo largo de este.

Por otro lado, me gustaría agradecer a mis amigos y, en especial, a Patricia, Pablo y Ricardo por acompañarme en este año tan atípico y animarme en momentos difíciles.

Por último, agradecer a mi familia por su apoyo y cariño, y por estar siempre a mi lado en todo momento.

Este trabajo se ha desarrollado en el marco del proyecto “América en Madrid. Patrimonios interconectados e impacto turístico en la Comunidad de Madrid (AmerMad-CM), financiado por la Comunidad de Madrid” (ref. H2019/HUM-5694).

Me gustaría agradecer a la Universidad Autónoma de Madrid la concesión de la Ayuda para el fomento de la Investigación en Estudios de Máster-UAM 2020/2021 que me ha permitido realizar este curso y trabajo.

Resumen

La selección de una ubicación para abrir un nuevo negocio se convierte en un desafío dentro del sector empresarial. Tradicionalmente esta selección se realiza a través de estudios de la zona y de conocimientos abstractos de los propios inversores, lo que supone un gran esfuerzo analizar y obtener resultados significativos para tomar una decisión. El principal problema de elegir un buen lugar es que se deben tener en cuenta varios factores para que la tienda pueda obtener beneficios, en el caso contrario un mal lugar puede precipitar el cierre de este al poco de abrir.

En este Trabajo Final de Máster se propone un algoritmo basado en un artículo reciente donde se analizan diferentes ubicaciones a través de características basadas en la localización de la tienda y de características comerciales relacionadas con los tipos de negocio que hay alrededor para recomendar qué tipo de negocio es más factible en esa ubicación. Para lograr esto, utilizamos la técnica de la factorización de matrices para obtener factores latentes de otras ubicaciones y tipos de negocio, y junto a otras características predecir la valoración de la ubicación con el tipo de negocio candidato.

Para comprobar la eficiencia del algoritmo hemos utilizado el conjunto de datos de Foursquare que ofrece información sobre el posicionamiento de los diferentes negocios y registros de usuarios que incluyen información sobre cuándo acceden a estos. Para evaluar los resultados del modelo que hemos desarrollado, se han utilizado diferentes algoritmos clasificadores como *baselines*: KNN, SVM, árboles de decisión y regresión logística. La evaluación de los resultados se ha elaborado a través de las métricas de *precision*, *recall* y nDCG.

Como conclusión, se ha observado que la eficiencia del algoritmo desarrollado se encuentra limitada por el conjunto de datos con el que se entrene, teniendo peores resultados cuando no se tiene suficiente información (algo que ocurre en alguna de las ciudades incluidas en nuestros experimentos). Por otro lado, el algoritmo es capaz de obtener factores latentes a partir de conjuntos de datos dispersos y obtener resultados que superan a los *baselines* en la mayoría de situaciones.

Palabras clave

Servicios basados en la localización, recomendación de tipos de negocio, factorización de matrices

Abstract

The selection of a location to open a new business becomes a challenge within the business sector. Traditionally this selection is made through studies of the area and abstract knowledge of the investors themselves, which involves a great effort to analyze and obtain significant results to make a decision. The main problem of choosing a good location is that several factors must be taken into account so that the store can make a profit, otherwise a bad location can precipitate the closure of the store shortly after opening.

In this Master's thesis we propose an algorithm based on a recent article where different locations are analyzed through characteristics based on the location of the store and commercial characteristics related to the types of business that are around to recommend what type of business is more feasible in that location. To achieve this, we used the technique of matrix factorization to obtain latent factors of other locations and business types and, together with other features, predict the score of the location for the candidate business type.

To test the efficiency of the algorithm we have used the Foursquare dataset that provides information about the location of different businesses and user check-ins that register when they access these. To evaluate the results of the model we have developed, we have used different classifier algorithms such as KNN, SVM, decision trees, and logistic regression. The evaluation of the results was performed with the following metrics: *precision*, *recall*, and nDCG.

In summary, we have observed that the efficiency of the developed algorithm is limited by the data set with which it is trained, having worse results when there is not enough information (something that occurs in some of the cities included in our experiments). On the other hand, the algorithm is able to obtain latent factors from sparse datasets and obtain results that outperform those from the baselines in most situations.

Keywords

Location-based services, type-shop recommendation, matrix factorization

Índice

Listas

Lista de algoritmos

Lista de ecuaciones

Lista de figuras

Lista de tablas

1

Introducción

En esta sección se especifica la motivación por la cual se ha decidido hacer este trabajo, los objetivos del mismo y la estructura del documento.

1.1. Motivación del proyecto

En las últimas décadas, los sistemas de recomendación han sufrido una gran evolución debido a su aplicación a diferentes ámbitos como en las compras a través de páginas web (por ejemplo, Amazon) o recomendación de contenido multimedia (por ejemplo, Netflix). La recomendación que suele llevarse a cabo en este tipo servicios utiliza las valoraciones que dan los usuarios a los diferentes artículos para ir refinando las futuras recomendaciones o buscar usuarios similares y recomendar artículos que les hayan gustado a estos y que todavía no haya probado el usuario.

Otras tecnologías que han dado mucho de que hablar en los últimos años son los Location-based Services (LBS), los cuales junto con el aumento de la popularidad de los teléfonos móviles o *smartphones* permiten registrar la localización de donde se encuentre el usuario. Algunos ejemplos de empresas que implementan este tipo de servicios son Instagram o Twitter. Esta tecnología ha permitido explorar numerosas investigaciones que tienen en cuenta la localización geoespacial de los usuarios con el objetivo de estudiar los comportamientos de movilidad de estos.

Teniendo presente estas dos ideas anteriores, varios investigadores han abierto un nuevo área de investigación donde se estudia cómo se pueden utilizar los registros de localización de los usuarios para poder recomendarles lugares que podrían ser de su interés (por ejemplo, cafeterías, restaurantes, etc.), también conocidos como Point-Of-Interest (POI). En este tipo de recomendaciones se suelen extraer características de las localizaciones de otros POIs que hay en los alrededores y de las localizaciones de donde se haya encontrado el usuario previamente para encontrar POIs similares en base a esas características y recomendar esos nuevos POIs al usuario final.

Este tipo de investigaciones pueden ser clave dentro del mundo empresarial, en concreto para inversores que esperan grandes beneficios cuando se disponen a abrir un nuevo negocio. De hecho, la recomendación de POIs puede adaptarse para que se centre en encontrar la mejor localización para

un tipo de negocio concreto con el objetivo de atraer al mayor número de clientes posible. En artículos recientes se emplean diferentes características basadas en la localización de los clientes de la zona y de los negocios vecinos, como la distancia a transporte públicos desde la localización candidata o la cantidad de registros que han hecho los usuarios en los POIs vecinos, junto con algoritmos de aprendizaje automático como KNN.

Con el objetivo de darle un cambio al paradigma de la recomendación de localizaciones para abrir un nuevo negocio dado su tipo, en este trabajo se plantea la recomendación de tipos de negocio dada una localización. Se propone el uso de factorización de matrices basado en los modelos de filtrado colaborativo para predecir el *rating* que tendrá un tipo de tienda en una localización determinada, utilizando características de movilidad de los usuarios y las características comerciales y de localización de los negocios. Para comprobar el rendimiento del sistema de recomendación se utilizarán como *baselines* algunos de los clasificadores más populares.

1.2. Objetivos y enfoque

Los objetivos que se proponen en este trabajo son los siguientes:

O-1.– Planteamiento de un nuevo sistema de recomendación basado en la explotación de características de movilidad del usuario y las características comerciales y de localización de los negocios.

O-2.– Desarrollo de un framework con distintos modelos de aprendizaje automático para poder realizar una comparación con el modelo propuesto.

O-3.– Evaluación de los modelos estudiados mediante medidas de ranking como *precision* y *recall* y medición del error con las métricas Mean Absolute Error (MAE) y Root Mean Squared Error (RMSE). El principal conjunto de datos será el de Foursquare versión de 2013, aunque se plantea utilizar otros conjuntos de datos disponibles que ofrezcan información complementaria.

Los objetivos propuestos se cumplirán gracias a los conocimientos obtenidos en las siguientes asignaturas del Máster Universitario en Investigación e Innovación en Inteligencia Computacional y Sistemas Interactivos (MUI2-ICSI): Aprendizaje Automático (para el tratamiento de datos y algoritmos de aprendizaje automático que se utilizarán en los sistemas de recomendación) y otras asignaturas como Dirección y Gestión de Proyectos Científicos y Tecnológicos (para la búsqueda y lectura de artículos que ayudarán a entender conceptos y aplicar nuevas metodologías), que serán cruciales en el trabajo.

1.3. Estructura del documento

El documento se encuentra dividido de la siguiente manera:

Introducción Descripción breve de la motivación del proyecto, los objetivos a completar en

este trabajo y la estructura del documento.

Estado del arte Se procede a realizar un repaso de las áreas de conocimiento estudiadas, haciendo especial hincapié en las técnicas utilizadas en los sistemas de recomendación y en los servicios basados en la localización.

Diseño e implementación El capítulo detalla los conjuntos de datos utilizados, el flujo de trabajo realizado en el proyecto y la descripción del modelo.

Pruebas y resultados Se analizan los resultados obtenidos con diferentes conjuntos de datos y se comparan con los resultados obtenidos con los *baselines*.

Conclusiones y trabajo futuro Se exponen las conclusiones obtenidas y las posibles investigaciones que se pueden realizar en base a este trabajo.

2

Estado del arte

Este capítulo realiza una breve introducción a los sistemas de recomendación, los servicios basados en la localización y las métricas que se utilizan habitualmente en los sistemas de recomendación.

2.1. Sistemas de recomendación

Los sistemas de recomendación son algoritmos que proveen sugerencias de una serie de *items* que podrían interesarle a un usuario en concreto [1]. Los *items* pueden ser muchas cosas como tipos de tiendas, películas, libros, videojuegos, etc. La recomendación puede realizarse de diferentes maneras, pero los métodos más populares suelen crear perfiles de usuarios a través de las descripciones de los *items* y, por otro lado, predecir el *rating* del nuevo producto en base a la calificación que hayan dado otros usuarios que sean similares en gusto [2]. En esta sección se realiza una descripción de varios algoritmos de sistemas de recomendación (filtrado colaborativo, modelos basado en contenido e híbridos) con el objetivo de tener una visión global de este campo aunque en el trabajo desarrollado se ha enfocado en un único tipo de algoritmo: filtrado colaborativo.

2.1.1. Conceptos básicos

Dentro de los sistemas de recomendación el conjunto de usuarios que participan en el sistema se define como $\{u_1, u_2, ..., u_n\} \in U$ y el conjunto de *items* como $\{i_1, i_2, ..., i_m\} \in I$. Los datos que relacionan los usuarios con los *items* se denotan en una matriz de valoraciones R de tamaño $|U| \times |I|$. Cada valoración $r \in R$ se define como la puntuación r_{ui} que da el usuario u sobre el *item* i [3].

La forma más habitual a la hora de representar la matriz de valoraciones es colocar los usuarios en las filas y los *items* en las columnas, de tal forma que la intersección entre estas muestren la valoración que ha dado el usuario al *item*. La valoración que se encuentra dentro de la matriz se encuentra acotada en determinados rangos de valores, como por ejemplo [1,10], [1,5] o [0.1, 1.0]. La imagen 2.1 muestra un ejemplo de matriz de valoraciones.

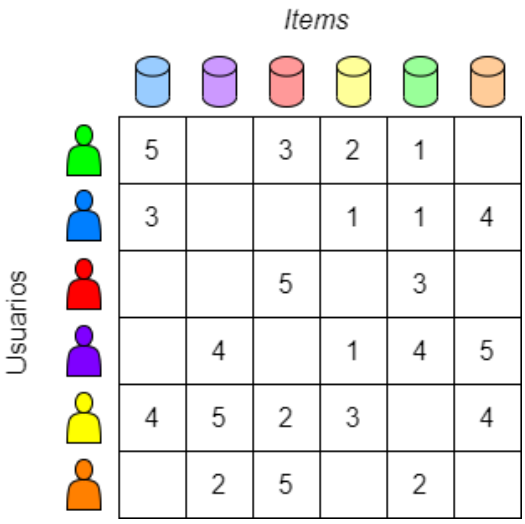


Figura 2.1: Matriz de valoraciones. Los usuarios se encuentran en las filas, los *items* en las columnas y las valoraciones se encuentran en el rango [1,5]

2.1.2. Problemas y desafíos

Los sistemas de recomendación, al igual que otros algoritmos, presentan problemas que pueden alterar los resultados de estos. Entre los problemas más recurrentes se encuentran los siguientes:

Escasez de datos Uno de los problemas más comunes en estos sistemas es la escasez de información debido a que los usuarios no hayan valorado suficientes artículos debido a que el sistema se encuentra en una fase temprana o cuando la proporción de artículos es muy elevada. Algunos métodos que evitan la escasez consisten en utilizar medias globales o dar valores por defecto pero con el inconveniente de estar modificando la matriz, o utilizar modelos de recomendación multidimensional como la factorización de matrices [4].

Cold start problem Este problema aparece cuando se añaden nuevos artículos o usuarios al sistema. Al no tener una valoración las recomendaciones se vuelven más imprecisas lo que ocasiona peores resultados [5]. La escasez de información en artículos nuevos afecta a las estrategias de filtrado colaborativo, porque un artículo no se recomienda hasta que otro usuario lo haya valorado previamente, mientras que la escasez de información por usuarios nuevos afecta a todas las estrategias por no poder construir un perfil de este. En investigaciones recientes se ha comprobado que el problema de artículos nuevos puede solucionarse utilizando información de otros artículos que sean similares dentro del sistema [6], y para usuarios nuevos utilizar fuentes de información adicionales que pueden obtenerse de las redes sociales [7].

Fraude El uso de los sistemas de recomendación en los *e-commerce* es un factor que puede ayudar en el crecimiento de una tienda [8], y un ejemplo donde se puede observar su funcionamiento es en la web de Amazon. Debido a esto han surgido métodos suspicaces con

el fin de aprovechar estos algoritmos para beneficiar a las propias empresas dando valoraciones positivas a sus propios productos, y perjudicar a las empresas de la competencia dando valoraciones negativas. Estas técnicas suelen ser muy dañinas para algoritmos que predicen valoraciones, pero los algoritmos de filtrado colaborativo basado en elementos suelen ser muy robustos para estos ataques [9].

Disponibilidad de datos Un problema recurrente dentro de los sistemas de recomendación es la falta de datos públicos para poder realizar pruebas. Debido a que muchos conjuntos de datos son de propiedad privada o restringidos en un determinado país, la investigación dentro del área de los sistemas de recomendación se ve mermada debido a que los datos que se encuentran públicos no siempre son suficientemente completos o no se encuentran actualizados [10].

2.1.3. Filtrado colaborativo

Los sistemas de recomendación basados en filtrado colaborativo son modelos que utilizan las interacciones pasadas que se han producido anteriormente entre los usuarios y los artículos para realizar nuevas recomendaciones en el presente. La idea principal de esta estrategia es encontrar similitudes entre los usuarios y los artículos del sistema, o factores latentes a partir de las valoraciones anteriores y aprovechar estas características para realizar recomendaciones. Los métodos que se encuentran dentro de los filtrados colaborativos pueden dividirse en dos grupos: basado en modelo y basado en memoria.

Basados en modelo

Se asume la existencia de factores latentes como consecuencia de la interacciones entre los usuarios y los artículos. Los modelos entrenan para reconstruir los valores reales de las valoraciones a partir de una representación que ha generado el propio modelo a partir de los factores latentes. En teoría, estos modelos presentan un mayor sesgo pero una menor varianza en comparación con los modelos que no asumen la existencia de factores latentes. Existen una gran variedad de técnicas basadas en modelo, pero una de las más populares es la factorización de matrices.

La factorización de matrices se basa en la creación de *embeddings* a través de los patrones de valoraciones que existan dentro del sistema [11]. La idea de estos modelos es que cada usuario y artículo puede representarse en un subespacio de factores latentes e inferir las valoraciones a través de estos. Estos modelos han causado impacto en las últimas décadas debido a su buena escalabilidad además de sus buenos resultados a la hora de predecir [12, 13].

Los *embeddings* pueden definirse como vectores de propiedades implícitas que componen a los usuarios y a los artículos. Por ejemplo, si se define una dimensión de 3 para estos vectores y el sistema

recomienda videojuegos a usuarios, los factores que podrían representar a los videojuegos podrían ser (i) la calidad de los gráficos, (ii) el nivel de violencia y (iii) la dificultad del juego. A su vez, los factores que compondrían a los usuarios pueden ser (i) cuánto le gusta al usuario los buenos gráficos, (ii) cuánto le gusta la violencia y (iii) cuánto le gusta los juegos difíciles. Estos factores explicados son suposiciones teóricas ya que es difícil averiguar en general a qué están haciendo referencia. Con esto se logra que estos vectores puedan actualizar sus valores en una fase de entrenamiento para representar al usuario o artículo lo más parecido posible a los datos observados.

La forma en la que la factorización de matrices obtiene los *embeddings* es mediante la descomposición de matrices [14], los cuales crean dos (o más dependiendo del algoritmo) componentes matriciales cuyo multiplicación escalar es igual a la matriz original. Uno de los primeros algoritmos que surgió dentro de la factorización de matrices fue Singular Value Decomposition (SVD) que descompone la matriz original en un producto escalar de tres matrices de dimensiones inferiores. La figura 2.2 muestra una representación visual de la descomposición.

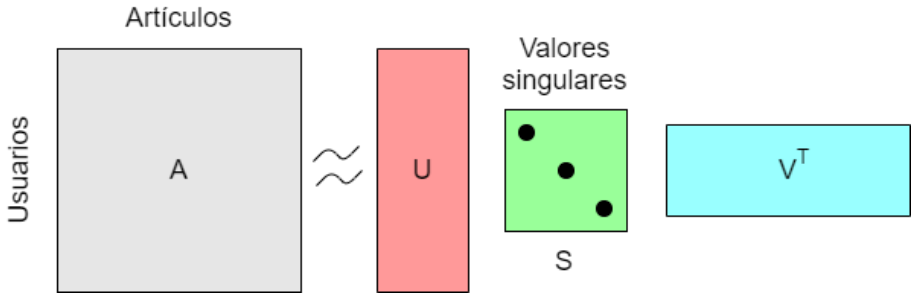


Figura 2.2: Descomposición SVD

Sea $A \in \mathbb{R}^{n \times m}$ la matriz original con las valoraciones de los usuarios y artículos, $U \in \mathbb{R}^{n \times k}$, $V^t \in \mathbb{R}^{m \times k}$ las matrices unitarias y $S \in \mathbb{R}^{k \times k}$ la matriz diagonal con los valores singulares de A ordenados de mayor a menor, el producto escalar de las matrices unitarias y la diagonal da como resultado la matriz original. El tamaño de los vectores de características implícitas se define con k siendo $k < \text{mín}(n, m)$. Aunque SVD consigue resultados muy precisos también presenta diversos problemas, entre estos se encuentra el sobreentrenamiento producido cuando la matriz es muy dispersa, y el coste de computación y distorsión de los datos cuando se decide rellenar los *missing value* que haya en la matriz.

La predicción de la valoración de un usuario $u \in U$ sobre un artículo $i \in I$ se realiza mediante una multiplicación escalar de los factores latentes que calcula la interacción entre ambos

$$\hat{r}_{u,i} = q_i^t p_u \tag{2.1}$$

siendo q_i el vector de factores latentes correspondientes al artículo i y p_u el vector de factores

latentes correspondientes al usuario u.

Una vez que se han calculado los factores latentes q_i, p_u, el siguiente paso es entrenar estos factores para minimizar el error y conseguir mejores predicciones. La manera en la que la factorización de matrices entrena estos factores es disminuyendo el error cuadrático regularizado en el conjunto de las valoraciones conocidas.

$$min_{q*,p*} \sum_{(u,i)\in K} (r_{ui} - q_i^t p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2) \tag{2.2}$$

siendo $K = \{(u,i)|r_{u,i} \neq 0\}$, la constante λ se trata de un parámetro que regula el aprendizaje y evita el sobreentrenamiento. Existen dos técnicas para disminuir el error en los factores: Stochastic Gradient Descent (SGD) y Alternating Least Squares (ALS).

Stochastic Gradient Descent: se itera sobre el conjunto de $(u,i) \in K$ y se calcula la predicción $\hat{r}_{ui}$ sobre cada uno de los elementos. Por cada valoración r_{ui} se obtiene el error e asociado a la predicción, y se varía el valor de los parámetros en una magnitud proporcional a γ. Destaca por su sencillez a la hora de implementar y su rapidez en la ejecución.

$$\begin{aligned} e_{ui} &= r_{ui} - \hat{r}_{ui} \\ q_i &\leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i) \\ p_u &\leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u) \end{aligned} \tag{2.3}$$

Alternating Least Squares: los factores p_u y q_i son desconocidos y por lo tanto el problema a optimizar no presenta mínimos globales. Este técnica consiste en rotar entre ambos factores dejando uno siempre fijo, lo que convierte al problema en un problema cuadrático disminuyendo de forma monótona la función de coste global. A diferencia de la técnica anterior, ALS es más difícil de implementar pero se puede paralelizar y, si no tiene datos dispersos, es más eficiente porque no tiene que iterar sobre cada valoración.

Los sesgos que aparecen en este tipo de filtrado colaborativo es debido a los efectos que causan los usuarios y artículos dentro del sistema independientemente de la interacción que ocurra entre ellos. Existen usuarios que otorgan mayor o menor valoración a los artículos por su forma de ser y existen artículos que reciben mayor o menor valoración debido a ciertas tendencias externas al sistema. Estas pautas se pueden agregar a la predicción del sistema y se conocen como sesgos.

Al incluir estos sesgos en la predicción, el cálculo de la estimación cambia siendo μ la valoración promedio, b_i la desviación respecto a la media del artículo i y b_u la desviación respecto a la media del usuario u (ecuación 2.4).

$$\hat{r}_{u,i} = \mu + b_i + b_u + q_i^t p_u \tag{2.4}$$

De forma similar al caso sin sesgos, el aprendizaje de los parámetros se realiza mediante la disminución del error cuadrático regularizado (ecuación 2.5) y un método para obtener el valor de los sesgos b_i, b_u es mediante la resolución del problema de mínimos cuadrados por SGD (ecuación 2.6).

$$min_{q*,p*,b*} \sum_{(u,i)\in K} (r_{ui} - \mu + b_i + b_u + q_i^t p_u)^2 + \lambda(b_i^2 + b_i^2 + \|q_i\|^2 + \|p_u\|^2) \tag{2.5}$$

$$\begin{aligned} e_{ui} &= r_{ui} - \hat{r}_{ui} \\ b_i &\leftarrow b_i + \gamma \cdot (e_{ui} \cdot b_u - \lambda \cdot b_i) \\ b_u &\leftarrow b_u + \gamma \cdot (e_{ui} \cdot b_i - \lambda \cdot b_u) \end{aligned} \tag{2.6}$$

Basados en memoria

Al contrario que los métodos anteriores, estos modelos no asumen la existencia de factores latentes y, en consecuencia, trabajan directamente con las interacciones entre los usuarios y los artículos. Un subconjunto de usuarios son elegidos en base a su similitud con el usuario al que se quiere recomendar un producto y se obtiene una valoración determinada de un producto en base a la puntuación que dieron los usuarios similares. Igualmente se puede utilizar este método pero en lugar de buscar usuarios similares se buscan artículos similares. Al no asumir factores latentes, los modelos presentan un sesgo bajo pero una varianza alta. La ventaja de estos métodos es que son fáciles de construir y de interpretar, pero la eficiencia decae rápidamente si existe escasez en los datos.

Dentro de este grupo uno de los algoritmos más populares es K-Nearest Neighbor (KNN) [15] debido a su eficiencia en crear *clusters* al calcular la similitud entre los diferentes datos. Se encuentran dos variantes de este algoritmo:

KNN basado en usuarios: la predicción de la valoración de un usuario sobre un artículo está basado en la valoración que hayan dado a ese artículo los usuarios similares:

$$\hat{r}_{ui} = \frac{\sum_{v\in\kappa(u)} sim(u,v) \cdot r_{vi}}{\sum_{v\in\kappa(u)} |sim(uv)|} \tag{2.7}$$

donde u es el usuario al que se desea predecir su valoración sobre el artículo i, $\kappa(u)$ son los vecinos del usuario u y $sim(u,v)$ es la función de similitud entre usuarios.

KNN basado en artículos: la predicción se realiza de forma similar al caso anterior pero en lugar de calcular la similitudes de un usuario dado con el resto de usuarios, se calculan las similitudes de un artículo dado con el resto de artículos que hayan sido valorados por el usuario:

$$\hat{r}_{ui} = \frac{\sum_{j \in \iota(u)} sim(i,j) \cdot r_{uj}}{\sum_{j \in \iota(u)} |sim(i,j)|} \tag{2.8}$$

donde u es el usuario al que se desea predecir su valoración sobre el artículo i, $\iota(u)$ son los artículos que han sido valorado por el usuario u y $sim(i,j)$ es la función de similitud entre artículos.

El calculo de la similitud entre artículos o usuarios puede calcularse a través de diferentes algoritmos. A continuación se muestran los algoritmos que se utilizan de forma más recurrente para calcular similitudes entre usuarios, las similitudes entre artículos se calculan de manera análoga:

$$cos(u,v) = \frac{\sum_{i \in I} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I} r_{ui}^2 \cdot r_{vi}^2}} \tag{2.9}$$

$$pearson(u,v) = \frac{\sum_{i \in I} (r_{ui} - \bar{r}_u) \cdot (r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I} (r_{ui} - \bar{r}_u)^2 \cdot \sum_{i \in I} (r_{vi} - \bar{r}_v)^2}} \tag{2.10}$$

$$jaccard(u,v) = \frac{I_u \cap I_v}{I_u \cup I_v} \tag{2.11}$$

$$euclidean(u,v) = \sqrt{\sum_{i \in I} (u_i - v_i)^2} \tag{2.12}$$

La **similitud coseno** (ecuación 2.9) se basa en medir el coseno del ángulo producido por dos vectores en un espacio multidimensional. En este caso, el tamaño de los vectores corresponde al número de artículos que dos usuarios han valorado en común. Esta similitud es de las más utilizadas en los problemas de Natural Language Processing (NLP) porque permite calcular cómo de cercanos son dos textos independientemente de su longitud.

La similitud por **correlación de Pearson** (ecuación 2.10) se basa mucho en la similitud de coseno, pero Pearson tiene en cuenta la valoración media que da el usuario (o la valoración media de un artículo si se calcula la similitud entre artículos) debido a que un usuario, por lo general, tiende a dar puntuaciones más altas o más bajas de forma habitual.

En las anteriores similitudes, los algoritmos solo tienen en cuentan el conjunto de valoraciones

que tengan en común dos usuarios, de tal forma que dos usuarios pueden ser muy similares aunque solo coincida una de sus valoraciones en un solo artículo. La similitud de **Jaccard** (ecuación 2.11) no presenta este inconveniente debido a que mide el solapamiento entre las valoraciones de los usuarios. Siendo I_u los artículos que han sido valorados por el usuario u y I_v los artículos que han sido valorados por el usuario v, el algoritmo se basa en la división de la intersección entre la unión de artículos que han sido interactuados por ambos usuarios.

Por último, la similitud por **distancia Euclidea** (ecuación 2.12) mide cómo de cerca se encuentran dos puntos en un espacio multidimensional. Este algoritmo se presenta como una alternativa a los algoritmos anteriores cuando los puntos en el espacio se encuentran alineados y los ángulos que se formarían entre ellos es 0.

2.1.4. Basado en contenido

La recomendación basada en contenido se basa en el análisis de las descripciones de los artículos e información de los usuarios para crear representaciones de artículos y perfiles de usuarios para sugerir artículos que sean similares a artículos que le hayan gustado al usuario en el pasado. El proceso para realizar las recomendaciones consiste en analizar los atributos que componen el perfil del usuario donde se guarda las preferencias e intereses de estos con los atributos de los artículos. Como resultado se obtiene una puntuación que mide el interés del usuario por el artículo. Los atributos que describen a los artículos dentro del sistema son características extraídas de los metadatos del propio artículo, o características extraídas de las descripciones del artículo. Habitualmente la extracción por metadatos es difícil debido a que existe poca información de estos y no se consiguen los suficientes para definir de forma correcta los intereses de los usuarios, mientras que el uso de las descripciones trae consigo una serie de problemas en la ambigüedad del lenguaje natural como puede ser el uso de sinónimos o expresiones diferentes para indicar lo mismo.

El proceso de recomendación se lleva a cabo en tres módulos:

Analizar información: se analiza y procesa la información procedente de los artículos para poder ser utilizada en los siguientes pasos. Un ejemplo de esto es cuando se debe extraer información procedente de la descripción del artículo en forma de texto plano. Los datos obtenidos en este paso sirven como entrada para los siguientes módulos.

Elaborar perfil de usuario: el módulo reúne datos de los gustos y preferencias del usuario para construir un perfil de este a través de la generalización de los datos. Una estrategia para generalizar los datos es a través de algoritmos de aprendizaje automático como KNN.

Búsqueda de similitud: el último módulo se encarga de recomendar artículos que pueden ser relevantes a un usuario por la similitud que haya entre estos artículos y el perfil del usuario. El cálculo de la similitud se realiza a través de funciones como la similitud coseno. Por otro lado, los resultados

obtenidos en la recomendación sirven para seguir moldeando los perfiles de los usuarios.

En primer lugar la información del artículo pasa por el módulo que analiza la información y aplica diferentes métodos para extraer características y producir una representación clara de estos. El proceso de creación de un perfil de usuario se lleva a cabo con las valoraciones que ha puntuado el usuario y la representación de los artículos de estas valoraciones. Este conjunto de datos es utilizado para crear un modelo que sea capaz de predecir la relevancia que tendrá un nuevo artículo para el usuario. De manera alternativa, los usuarios pueden indicar sus gustos o áreas de interés para la creación de un perfil inicial sin tener que utilizar sus valoraciones pasadas.

Existen dos formas de crear la representación del usuario, de forma explícita y de forma implícita. En la primera de ellas, el sistema requiere que el usuario ponga una valoración, por ejemplo una puntuación de 1 a 5 a una serie de artículos, lo que implica una forma sencilla de obtener datos exactos pero no es adecuada para saber qué es lo que realmente le gusta al usuario del artículo. La segunda opción, la forma implícita, se trata de marcar un artículo como relevante a través de las acciones que haga el usuario, por ejemplo el tiempo que esté mirando una película. La principal ventaja de esta opción es que no requiere la cooperación del usuario de forma directa, pero su inconveniente es que pueden aparecer sesgos como por ejemplo marcar una película como relevante porque el usuario la ha visto entera, pero que en realidad se haya quedado dormido al inicio de esta.

La figura 2.3 muestra un esquema de este proceso.

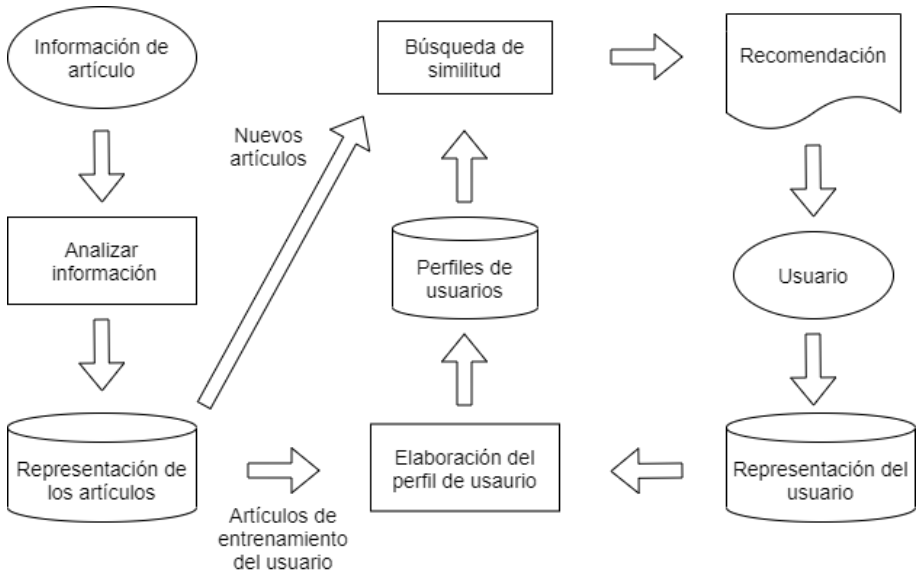


Figura 2.3: Esquema de recomendación basada en contenidos

El algoritmo que se emplea normalmente para encontrar representaciones de artículos similares en este tipo de sistema de recomendación es mediante la **similitud coseno** (ecuación 2.13) donde x, y son vectores representativos de los artículos.

$$cos(x,y) = \frac{\sum_{k \in K} x_k \cdot y_k}{\sqrt{\sum_{k \in K} x_k^2} \cdot \sqrt{\sum_{k \in K} y_k^2}} \tag{2.13}$$

El aprendizaje de perfiles de usuarios se realiza a través de algoritmos de aprendizaje automático, y entre los algoritmos más populares para este objetivo se encuentran modelos probabilísticos como *Naïves Bayes*, *Relevance Feedback* o KNN.

2.1.5. Híbridos

Los sistemas de recomendación híbridos nacen como la combinación de dos o más métodos de recomendación con el objetivo de obtener mejores resultados arreglando los problemas que originan los métodos combinados por separado. Un ejemplo muy común de esto es cómo varios métodos de filtrado colaborativo se combinan con otros métodos para evitar los problemas de *cold-start* que presentan estos métodos por sí solos [16].

Una forma directa de emplear este tipo de sistema de recomendación es ejecutar de forma independiente un recomendador basado en contenido y uno basado en filtrado colaborativo y combinar sus predicciones utilizando un sistema de votación [17], o utilizar un método basado en contenidos que emplee información procedente de un método de filtrado colaborativo para ayudar en la búsqueda de similitud entre las representaciones de los artículos.

Aunque se ha descrito de forma breve dos sistemas de recomendación híbridos simples, existen métodos más elaborados:

Ponderado: este método puntúa un artículo en base a los resultados obtenidos de los otros métodos de recomendación que hay en el sistema. Para calcular la valoración final se emplean diferentes algoritmos como por ejemplo una combinación lineal donde, en un principio, los pesos de los resultados de los otros métodos comienzan siendo iguales pero se irán ajustando dependiendo si la valoración final es acorde a la valoración del usuario o no. La principal ventaja de este método es que es fácil de implementar y de ajustar los pesos de las puntuaciones que realicen el resto de métodos, pero implica que todos los métodos reciban la misma información lo que puede suponer un inconveniente debido a que ciertos métodos necesitan de más información que otros para obtener resultados decentes.

Alterno: este método implementa una serie de criterios para cambiar entre los diferentes métodos de recomendación. Si el método de filtrado colaborativo no es capaz de calcular una predicción acertada entonces el sistema híbrido cambia a un método basado en contenido. Este método alterno introduce complejidad adicional dentro del sistema debido a que se necesita definir un criterio para ir alternando entre los diferentes métodos, pero su ventaja es que el método dispone de los puntos fuerte de los métodos que lo componen para compensar los puntos débiles.

Mezclado: este tipo de método utiliza los métodos basados en contenidos para crear la representación de los artículos y los métodos de filtrado colaborativo para averiguar la preferencia de los usuarios. El método de mezclado evita los problemas al introducir un nuevo artículo en el sistema porque el método basado en contenido puede utilizar los metadatos o elementos textuales y/o visuales de este para la recomendación, pero no resuelve el problema de la falta de información de un nuevo usuario cuando este se une al sistema. Al igual que el método alterno, el punto fuerte de este método es que dispone de los puntos fuertes de los otros sistemas de recomendación para compensar los puntos débiles que tendrían estos por separado.

Combinación de características: este método consiste en tratar la información procedente de los métodos de filtrado colaborativo como características adicionales en los métodos basado en contenido. Esta combinación de características permite tener en cuenta la información de los métodos de filtrado colaborativo sin considerar sus resultados, lo que permite a los métodos basados en contenido centrarse únicamente en los usuarios que han valorado un artículo. Si se aplica de forma inversa, los métodos de filtrado colaborativo tendrían en cuenta la información sobre la similitud de los artículos propios de los métodos basado en contenido.

Cascada: a diferencia de los métodos descritos anteriormente el método en cascada no combina los métodos, sino que realiza un proceso por etapas. En primer lugar se emplea un método para predecir una serie de recomendaciones y después se aplica otra segunda recomendación con otro método para refinar el conjunto de candidatos. La ventaja de este método permite que el segundo sistema de recomendación que se utilice se centre únicamente en aquellos candidatos que han sido elegidos por el primer sistema de recomendación.

Aumento de características: el siguiente método consiste en utilizar la valoración que predice un método como datos de entrada para el siguiente método. Lo destacable de este método es que ofrece un medio para mejorar el rendimiento del método principal sin tener que modificar nada de su funcionalidad interna, pero hay que recurrir a funcionalidades intermedias para poder procesar la información adicional.

Meta-level: este método es similar al método anterior, pero en lugar de utilizar los datos de salida del primer modelo como datos de entrada del segundo modelo, el primer modelo en sí es utilizado como dato de entrada para el segundo modelo. La ventaja de este método es que el primer modelo que se utiliza como dato de entrada para el segundo se puede tratar como una representación comprimida de los gustos de los usuarios, lo que supone una información más fácil de procesar a diferencia de si se utilizase como dato de entrada un texto plano.

2.2. Sistemas basados en la localización

La normalización del uso del teléfono móvil o *smartphone* en nuestra vida diaria ha precipitado una conexión entre nuestra vida real y cotidiana con un mundo virtual creado a través de Internet. Hoy en día cualquier persona es capaz de buscar información dentro de este mundo virtual con unos pocos toques de su teléfono móvil, en lugar de tener que consultar su ordenador personal. Debido a los avances tecnológicos, los teléfonos móviles están dejando de ser poco a poco simples portales a Internet y convertirse en herramientas que permiten al usuario explorar, descubrir e interactuar con aquello que le rodea.

Debido a la adopción del Global Positioning System (GPS) por parte de los teléfonos móviles se ha producido un alza de los servicios basados en la localización, los cuales tienen en cuenta la localización geográfica de una entidad pudiendo ser esta una persona o no [18]. Este tipo de servicios se encuentran en aplicaciones de mapeado, búsqueda de rutas, búsqueda de servicios cercanos como pueden ser centros de ocio o restaurantes, o compartir tu ubicación con amigos en tiempo real.

Gracias a los teléfonos móviles y los servicios basados en la localización, tecnologías como los sistemas de recomendación han sufrido un gran avance tecnológico. Hoy en día gracias al acceso de la posición geográfica del usuario (tanto presentes como pasadas), los sistemas de recomendación pueden tener en cuenta la movilidad de los usuarios y recomendar servicios de acuerdo a los patrones de este. Debido a que se utiliza este tipo de información, las recomendaciones que no tengan que ver con lugares (libros, películas, series, música, etc) quedarían excluidos.

2.2.1. Datos en sistemas basados en la localización

La principal diferencia de los datos que se obtienen a raíz de los servicios basados en la localización con los datos obtenidos de forma tradicional es que los primeros están centrados en registrar la movilidad de los usuarios. A través de los teléfonos móviles los usuarios pueden realizar las mismas acciones que si estuvieran en una página web (dar me gusta, valorar un artículo, compartir, etc.), pero aprovechando los servicios que ofrecen los teléfonos móviles y las Location Based Social Network (LBSN) de distintas aplicaciones (Instagram, Foursquare, Yelp); uno de los datos más significativos que se puede obtener son los *check-ins*. Estos *check-ins* son registros de lugares que ha visitado un usuario y que guardan estas aplicaciones para que el usuario pueda compartirlo con sus conocidos.

A través de los *check-ins* se puede obtener multitud de información. El registro de un *check-in* conlleva que el usuario ha entrado en un determinado establecimiento, si se presentan varios registros de un mismo establecimiento se puede intuir que al usuario le ha gustado el establecimiento y cuando se registra el *check-in* además se guarda la hora y fecha de cuando se realizó, por lo que se puede averiguar en qué épocas del año o momentos del día el usuario acude con más frecuencia.

Las redes sociales que utilizan estos registros deben mantener una base de datos donde guardar los distintos establecimientos o lugares de interés que puedan visitar los usuarios. Estos lugares son conocidos como Point of Interest (POI) y en los últimos años se han convertido en una fuente de datos para diferentes sistemas de recomendaciones por el cual se pretende predecir qué lugares podrían visitar los usuarios [19, 20]. Algunos de los datos que pueden tener estos POIs son el nombre del mismo, su posición geográfica, la categoría a la que pertenece o información multimedia como fotos o vídeos. La figura 2.4 muestra un ejemplo de POIs situados en un mapa.

Figura 2.4: Mapa de POIs. Fuente: https://www.openstreetmap.org

El uso de los *check-ins* y la información adicional que puede obtenerse de las LBSN han permitido crear diferentes grupos de datos:

Espaciales: este tipo de características infieren factores del lugar donde se encuentran los POIs relacionados con los comportamientos de movilidad de los usuarios que acuden a estos. Se puede calcular la popularidad general de la zona o la popularidad de la zona en un momento del día, la semana o del año en concreto.

Temporales: las características de este tipo se centran en los instantes temporales de cuando se realizan los *check-ins*. Debido a la existencia de una rutina, los usuarios se trasladan de un punto a otro a determinadas horas del día reflejando un comportamiento en la movilidad del usuario. Por ejemplo, un usuario puede visitar diariamente una cafetería antes de incorporarse en su puesto de trabajo a las 9:00, o ir al gimnasio por las tardes a las 18:00.

Usuarios: las características que se extraen de este tipo de datos se centran en los patrones de movilidad de los usuarios. Entre estas características encontramos el número de veces que el usuario visita diferentes POIs, la probabilidad de que un usuario vaya a visitar un POI de una categoría

determinada o la distancia de los *check-ins* realizados a determinadas zonas geográficas como el centro de la ciudad o de paradas de transporte público. Si además la red social que tiene los datos dispone de funcionalidades para crear conexiones entre sus usuarios, como peticiones de amistad, se puede tener en cuenta las características de los usuarios que sean amigos entre ellos.

2.2.2. Recomendación de POIs

Debido al aumento de la popularidad de las LBSNs, el interés de estas redes sociales por obtener un mayor número de usuarios ha llevado consigo la investigación de nuevas formas de recomendar POIs a los usuarios de estas redes sociales con el objetivo de que se mantengan activos y recomienden la aplicación para atraer nuevos usuarios.

La recomendación de POIs ha generado gran interés dentro de la comunidad investigadora y a nivel empresarial, pero a día de hoy la recomendación de POIs supone un desafío debido a la cantidad de problemas que surgen en su recomendación. Uno de los problemas principales es la falta de interacciones que se registran de los usuarios con los distintos POIs, debido a que los usuarios tienden a interactuar con unos pocos POIs si se compara con el número total de POIs que mantienen las LBSN. Esto origina que los *check-ins* representen una fracción realmente pequeña entre las conexiones de los usuarios y los POIs. A esto hay que sumarle el hecho de que los usuarios no registran todos los POIs en los que han estado, lo que aumenta aun más la escasez de datos.

Algunos métodos que se han utilizado para la recomendación de POIs son las técnicas de filtrado colaborativo que utilizan los *check-ins* para predecir la preferencia de los usuarios ante determinados POIs. Existen trabajos que han planteado tanto el filtrado colaborativo basado en modelos [21, 22] como el basado en memoria [23]. Otros métodos que se han propuesto utilizan las fotografías que guardan los POIs para buscar similitudes entre estos y recomendar, a partir de esa similitud, nuevos POIs a los usuarios [24].

Además del uso de *check-ins* para sustentar la recomendación de POIs, el uso de la información espacial, temporal y social que puede obtenerse de las LBSN puede ayudar en gran medida a la recomendación en diferentes modelos como los filtrados colaborativos [25] o la búsqueda de similitud en grafos [26].

2.2.3. Recomendación de ubicación de POIs

La recomendación de POIs para usuarios se trata de un problema que está siendo investigado actualmente y del cual ya se han obtenido métodos fiables para su uso. A su vez, dentro de este área de investigación está surgiendo una nueva vertiente que plantea en qué ubicación es preferible colocar un nuevo POI, que implica cambiar la forma de pensar con respecto a lo que se tenía por sentado en la recomendación de POIs a usuarios, ya que ahora se estaría recomendando lugares (una localización

concreta) a los POIs, es decir, el *agente que recibe la recomendación* no es el usuario sino el propio POI.

El desafío de encontrar un lugar óptimo para colocar un nuevo negocio se ha convertido en un foco de estudio en los últimos años. Tradicionalmente, los inversores deciden si una ubicación es buena para un tipo concreto de negocio basándose en su opinión subjetiva sobre los alrededores de la ubicación, los negocios del alrededor o la gente que pasa por la ubicación. Por ello, encontrar un método que utilice la minería o ciencia de datos puede ayudar a facilitar este trabajo en lugar de explotar el conocimiento subjetivo de una persona. El desarrollo de las LBSN y de los teléfonos móviles han originado la aparición de diferentes tipos de datos los cuales se han convertido en la fuente principal para encontrar soluciones a este problema. Los conocimientos que se tenían sobre la recomendación de POIs a usuarios ha servido como punto inicial para este problema ya que, al igual que en la recomendación de POIs los datos sobre la movilidad de los usuarios, la localización de los POIs en los alrededores o el uso de los *check-ins* son algunos de los datos que se utilizan actualmente para entrenar métodos que sean capaces de ofrecer opciones. Además de los datos también se están utilizando métodos que ya se utilizaban en la recomendación de POIs como las técnicas de filtrado colaborativo.

Los datos que emplean estos tipos de problema tienen relación con las características geográficas de la ubicación candidata y con las características comerciales del vecindario:

Características geográficas: este tipo de características describen el entorno alrededor de la ubicación candidata. Con estas características se plantea tener en cuenta la popularidad de la zona a través de los *check-ins* totales e individuales de los negocios de la zona, y la distribución del resto de POIs en la zona como por ejemplo la cercanía a transportes públicos o a carreteras que puede reflejar el buen acceso a la zona. En estas características se puede encontrar la densidad de usuarios que hay en la zona, la popularidad de la zona, la distancia al centro de la ciudad o la densidad de transición, entre otros [27, 28].

Características comerciales: de manera similar a como se extraen características geográficas, las características comerciales son obtenidas a través del análisis de las relaciones entre los POIs dada su categoría. Estas características se basan en cómo un determinado tipo de negocio puede estar influenciado por el resto de negocios de la zona, por ejemplo que sea el único negocio y por lo tanto satisfaga una demanda, o que la relación con otro tipo de negocio ayude a aumentar su popularidad como podría ser una relación de una biblioteca y a pocos metros una cafetería. Un ejemplo concreto de estas características es la competitividad de la zona, que consiste en ver cuántos negocios del mismo tipo hay en la zona [29].

Con respecto a los métodos que se emplean en las recomendaciones, los métodos más comunes emplean las técnicas de filtrado colaborativo [30] o KNN [31], pero existen trabajos que implementan

nuevos métodos (por ejemplo, ver qué servicios y/o productos son más buscados a través de palabras claves en una zona determinada y ver qué zona puede satisfacer más al tipo de negocio que se pretende abrir) [32] o utilizan clasificadores para estimar el mejor lugar [27, 28].

Además de la recomendación de ubicaciones para un determinado POI, recientemente se ha propuesto un problema similar pero que pone el enfoque desde otra perspectiva. El objetivo de estos estudios es, dada una ubicación, intentar averiguar qué tipo de POI le conviene mejor a esa ubicación. Aunque el planteamiento es similar, la hipótesis inicial cambia: el usuario final del sistema de recomendación no tiene claro qué POI quiere abrir pero dispone de una localización y quiere saber qué tipo de negocio le viene mejor para obtener el mayor beneficio posible. Un caso donde puede darse este problema es cuando se acaba de construir un nuevo barrio y un propietario de un local no se decanta entre posibles opciones de negocios que puede abrir, o incluso podemos imaginar que el usuario es algún asesor de urbanismo que use un sistema de este tipo para detectar el potencial de cada zona y proponer licencias de uno u otro tipo. Hoy en día hay pocos estudios que intentan resolver este problema desde esta perspectiva pero cada vez hay más investigadores que buscan posibles soluciones [29, 33, 34]. La figura 2.5 muestra una representación gráfica de ambos problemas.

Figura 2.5: La figura de la izquierda muestra una representación de la recomendación de localizaciones para un POI y la de la derecha la recomendación de POI dada una localización. Fuente: https://www.openstreetmap.org

2.3. Evaluación de los sistemas de recomendación

En los sistemas de recomendación existen distintas métricas que permiten saber cómo de precisas han sido las predicciones de un sistema con respecto al usuario que lo esté utilizando. Entre estas métricas se encuentran evaluaciones que permiten medir cómo de real han sido las valoraciones predichas con respecto a las valoraciones reales, y otras que miden la posición de los artículos dentro de un *ranking* de predicciones. Algunas de las métricas más habituales son las siguientes:

Mean Absolute Error (MAE): el error absoluto medio mide la diferencia media entre la valora-

ción predicha y la valoración real del artículo. Cuanto más cercano sea su valor a 0 implica que las predicciones de las valoraciones realizadas son similares a las valoraciones reales.

$$MAE = \frac{1}{|K|} \sum_{(u,i) \in K} |r_{ui} - \hat{r}_{ui}| \tag{2.14}$$

Root Mean Squared Error (RMSE): el error cuadrático medio mide la diferencia media entre la valoración predicha y la valoración real del artículo al igual que el error absoluto medio, pero en lugar de utilizar el valor absoluto de la diferencia se utiliza el cuadrado de la diferencia y posteriormente la raíz cuadrada. La ventaja de este método con respecto al anterior es que penaliza más cuanto mayor sea la diferencia.

$$RMSE = \frac{1}{|K|} \sum_{(u,i) \in K} \sqrt{(r_{ui} - \hat{r}_{ui})^2} \tag{2.15}$$

Aunque MAE y RMSE son métricas útiles para ver cómo de precisas son las estimaciones de las valoraciones, a la hora de evaluar recomendaciones no sirven de mucho porque lo que de verdad importa es ver qué artículos aparecen en un *ranking* de predicciones y cómo el usuario se comporta ante tales recomendaciones.

En los sistemas de recomendación se busca crear un *ranking* de artículos con una profundidad N para seleccionar entre los artículos cuáles de ellos le podría interesar a un usuario. Es por ello que las métricas *precision*, *recall* y otras tantas más se pueden calcular para los N primeros artículos ordenados de mayor a menos por la valoración obtenida del sistema de recomendación.

$$P@N = \frac{|Rel@N|}{N} \tag{2.16}$$

$$R@N = \frac{|Rel@N|}{|Rel|} \tag{2.17}$$

$Rel@N$ es el número de artículos que son relevantes y recomendados para un usuario hasta la posición N del *ranking* y $|Rel|$ el número de artículos relevantes para el usuario. **Precisión con profundidad N** (ecuación 2.16) muestra el ratio de artículos relevantes y recomendados hasta esa profundidad y ***Recall* con profundidad N** (ecuación 2.17) muestra el ratio de artículos relevantes y recomendados hasta esa profundidad de entre todos los artículos relevantes.

$$nDCG@N = \frac{1}{IDCG} \sum_{p=1}^{N} f_{dis}(rel(u, i_p), p) \tag{2.18}$$

Normalized Discounted Cumulative Gain (nDCG)(ecuación 2.18) usa la relevancia acumulada en la cima del *ranking* y cómo puede ser reducida en los puestos más bajos. La función f_{dis} es definida comúnmente como $f_{dis}(x, y) = 2^x - 1/log(1+y)$ o $f_{dis}(x, y) = x - 1/log(y)$ si $y > 1$. Ideal Discounted Cumulative Gain (IDCG) denota el valor de DCG en un ranking perfecto y sirve para normalizar el valor anterior, de manera que esta métrica siempre esté entre 0 y 1.

3

Diseño e implementación

En este capítulo se muestra un breve diagrama del flujo de trabajo realizado en el proyecto, una presentación de las fuentes de datos utilizadas y, por último, una descripción del modelo desarrollado.

3.1. Diseño

En esta sección se expone el diagrama de trabajo realizado y se muestra las fuentes de datos empleadas.

3.1.1. Flujo de trabajo

La figura 3.1 muestra un diagrama del trabajo llevado a cabo.

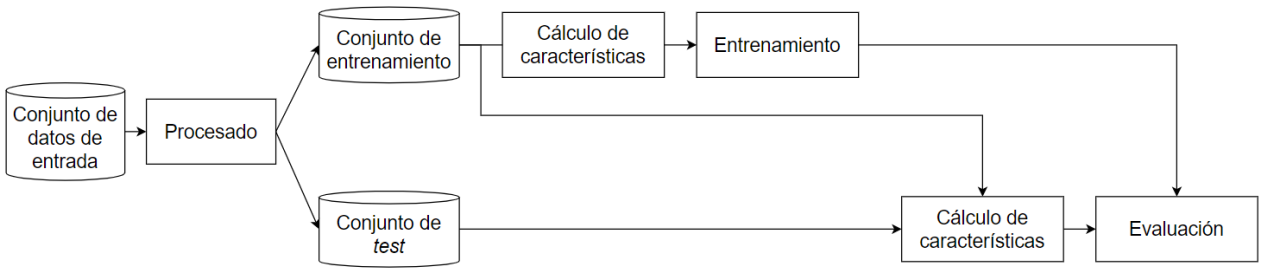


Figura 3.1: Diagrama del flujo de trabajo realizado

El primer paso es realizar un procesado del conjunto de datos de entrada debido a que los ficheros donde residen estos contienen información que es irrelevante, como por ejemplo POIs de otras ciudades que estén fuera de nuestro análisis, y por lo tanto se necesita establecer un criterio para seleccionar aquellos que interesan.

Una vez seleccionados los datos en base a un determinado criterio, el siguiente paso consiste en dividir el conjunto de datos procesado en dos: conjuntos de entrenamiento y conjuntos de *test*. Con el conjunto de datos de entrenamiento se calcula una serie de características que se implementan

dentro del modelo y posteriormente se realiza el aprendizaje de este. Una vez haya terminado el entrenamiento del modelo se calculan las características pertinentes para el conjunto de datos de *test* y se pasa a evaluar su eficacia.

3.1.2. Fuentes de datos

El conjunto de datos que se ha utilizado para este proyecto está formado por un registro de POIs y *check-ins* de usuarios anónimos del año 2013 de Foursquare que se encuentran públicos [1]. Estos datos fueron recopilados por investigadores [35,36] a través de llamadas a la web de Foursquare donde se obtuvo la información de los diferentes POIs y llamadas a Twitter donde se consiguieron los *check-ins* de los usuarios en los respectivos POIs. Además, como fuente de datos adicional se emplearon datos sobre la contaminación lumínica que hay en diferentes zonas del mundo en el año 2013 [2].

La razón por la que se decidió utilizar este primer conjunto de datos (Foursquare) se debe a su fácil interpretación, su popularidad en el área investigación sobre recomendación de POIs, su facilidad de obtención a diferencia de otras fuentes como las de Dianping y a que se dispone de información de diferentes partes del mundo. La base de datos se compone de tres ficheros:

Ciudades: el fichero está formado por las ciudades que se encuentran registradas en Foursquare junto con las coordenadas del centro de la ciudad y el país al que pertenece.

Negocios: en este fichero se encuentran los diferentes POIs que hay registrados en la aplicación junto con sus coordenadas geográficas y la categoría de negocio correspondiente. Las categorías que se registran en Foursquare se agrupan en diferentes niveles de profundidad, por lo que sería necesario transformar todas las categorías a un mismo nivel. Por ejemplo, a profundidad 2 la categoría *Dessert Shop* y dentro de esta categoría a profundidad 3 existen categorías más concretas como *Cupcakes Shop*, *Frozen Yogurt Shop* o *Ice Cream Shop* que implicarían un aumento de complejidad en la recomendación.

Check-ins: el fichero guarda todos los registros que han hecho los usuarios a lo largo del año en los diferentes POIs. Cada registro guarda el ID del usuario que ha hecho el registro, la fecha y hora de cuando se realizó y el identificador del POI que visitó el usuario.

Las figuras 3.2, 3.3 y 3.4 muestran un ejemplo de cada fichero respectivamente.

Por otro lado, los datos sobre la contaminación lumínica se obtuvieron desde otra web debido a que esta información no se encontraba en los conjuntos de datos anteriores. Se trata de una web de uso libre, lo que implica que no es necesario un registro previo ni esperar a una confirmación para habilitarte el acceso, y posee una interfaz amigable donde solo es necesario desplazarse en el mapa

[1] https://sites.google.com/site/yangdingqi/home/foursquare-dataset

[2] https://www.lightpollutionmap.info/

```
Rochester       43.210467       -77.635611      US      United States   Other
Newark  40.719996       -74.199996      US      United States   Other
New York        40.707864       -73.905237      US      United States   Other
Harrisburg      40.265001       -76.904001      US      United States   Provincial capital
Trenton 40.224996       -74.781000      US      United States   Provincial capital
```

Figura 3.2: Fichero de ciudades

```
3fd66200f964a52001e91ee3        40.730957       -73.954400      Thai Restaurant US
3fd66200f964a52001ed1ee3        37.779837       -122.494471     Seafood Restaurant      US
3fd66200f964a52001ef1ee3        34.092793       -118.281469     Bar     US
3fd66200f964a52001f01ee3        34.010366       -118.493357     Bar     US
3fd66200f964a52001f11ee3        40.591334       -73.960725      Nightclub       US
```

Figura 3.3: Fichero de POIs

y seleccionar la zona donde se desea obtener la información de contaminación lumínica. La razón por la que se decidió utilizar los datos de esta web fue porque encontramos varios artículos que usaban estos mismos datos para resolver problemas similares y nos pareció interesante emplear los datos de la web para nuestro problema. La figura 3.5 muestra un ejemplo de la página web.

3.2. Implementación

El trabajo está basado en el artículo *Shop-Type Recommendation Leveraging the Data from Social Media and Location-Based Services* [29] en el cuál se utiliza Daiping (aplicación de valoración de comercios en China) y Baidu LBS (servicio LBS de China) como fuente de datos. Al no disponer de estas fuentes de datos, en nuestro trabajo vamos a emplear como fuente de datos los registros cedidos por Foursquare y adaptar el modelo del artículo a este.

La idea principal es emplear en nuestro sistema de recomendación la factorización de matrices utilizando la descomposición SVD para recomendar qué tipo de negocio es más óptimo para un lugar determinado. Se emplean los tipos de negocio que se encuentran en el conjunto de datos de *test* y el sistema devuelve un *ranking* del éxito de cada tipo de negocio ordenados de mejor a peor. Además, se introduce unos sesgos que se calculan en base a la posición geográfica de la tienda y al tipo de negocio que se esté recomendando para mejorar los resultados de la recomendación.

```
1729064 4e298ed0c65ba11f4c67a615        Tue Apr 03 18:00:10 +0000 2012  120
1633579 4dfc825bc65b31579b2e7679        Tue Apr 03 18:00:11 +0000 2012  180
159670  4e3ccf9a52b1a04aff106021        Tue Apr 03 18:00:11 +0000 2012  -300
211326  4b4f9d38f964a5209e0d27e3        Tue Apr 03 18:00:11 +0000 2012  -300
854109  4c64815ef07e2d7f3c7c8f50        Tue Apr 03 18:00:12 +0000 2012  -420
```

Figura 3.4: Fichero de *check-ins*

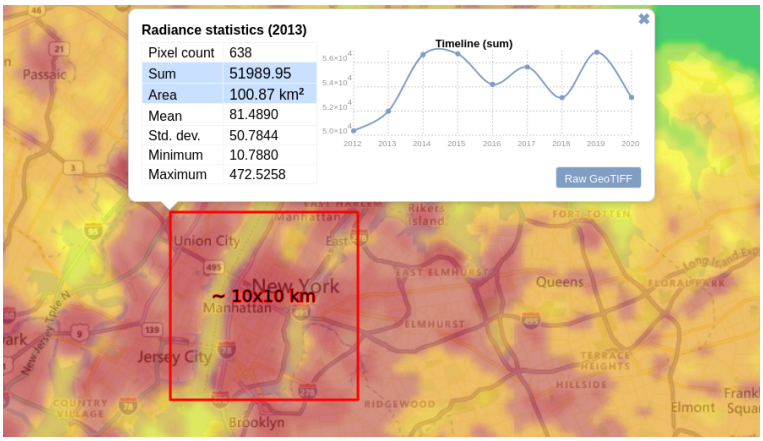


Figura 3.5: Datos de contaminación lumínica

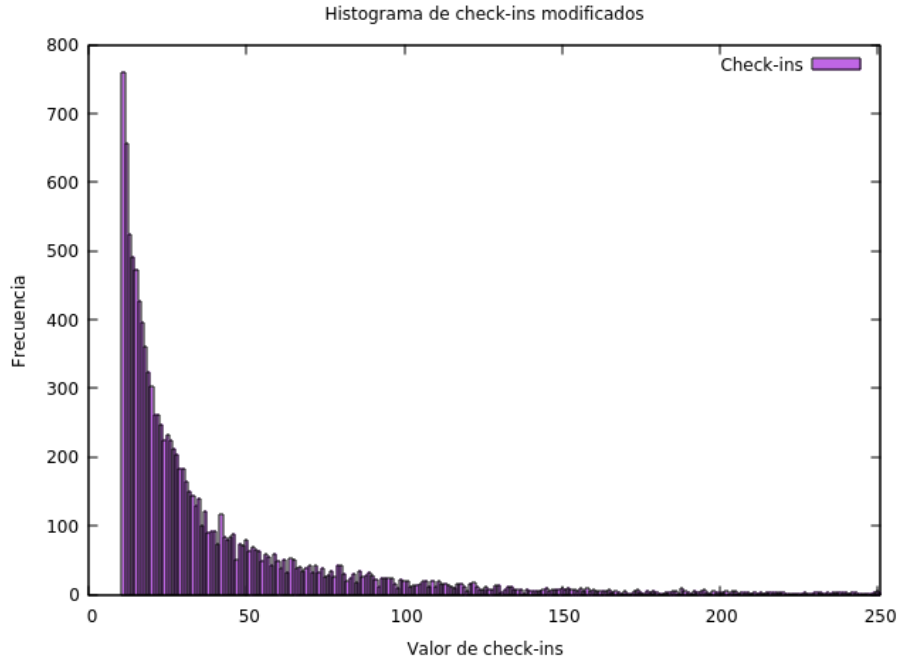
3.2.1. Procesado de datos

En primer lugar se deben filtrar los *check-ins* y POIs de la ciudad donde se realiza la recomendación para calcular las características que se emplearán como sesgos dentro de la recomendación. Se seleccionan los POIs de la ciudad correspondiente y se guardan los ID de cada uno para poder filtrar los *check-ins* que hacen los usuarios en esas tiendas. Para evitar guardar POIs que no haya visitado ningún usuario se eliminan aquellos que no se encuentren en ningún *check-in*.

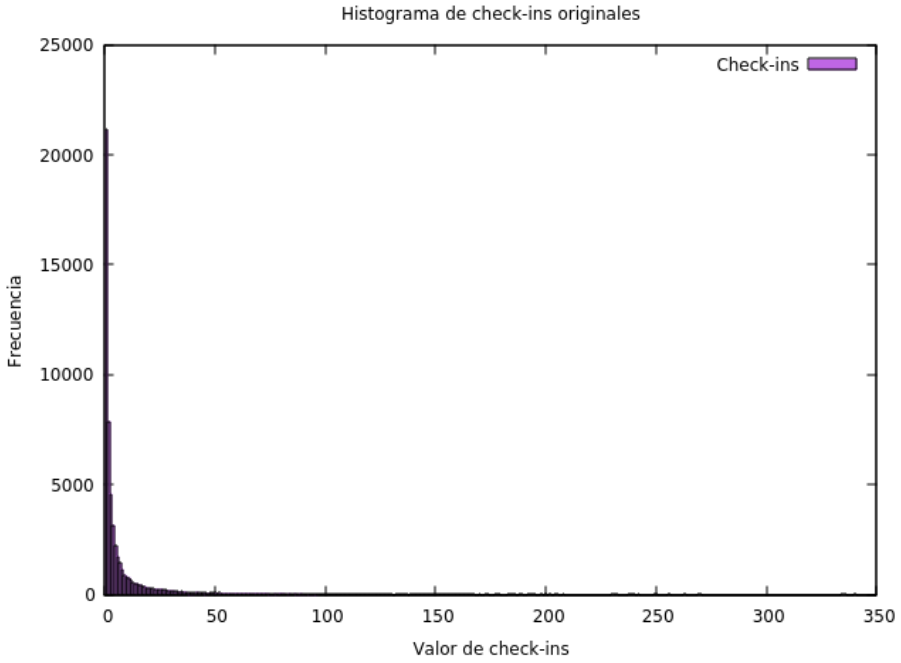
El siguiente paso es crear las valoraciones de cada POI para que el sistema de recomendación pueda entrenar. Como en los ficheros no se encuentra ningún tipo de valoración real de los negocios, se utilizan los *check-ins* que ha recibido cada negocio como valoración de tal forma que aquellos negocios con más cantidades de *check-ins* tendrán una puntuación más alta. Las valoraciones de los POIs se asignaron en un rango de valores de [1,10] a través de una técnica de discretización que agrupase los diferentes valores de *check-ins* dentro del rango definido.

Uno de los problemas que se encontraron fue la diferencia de *check-ins* que había entre los diferentes negocios, causando que unos pocos negocios tuvieran un gran número de *check-ins* en comparación con el resto de negocios y por lo tanto que las valoraciones que se originaban fueran únicamente el valor máximo y mínimo del rango de valores definidos. Para solucionar este inconveniente se decidió definir un número de *check-ins* máximo para que los negocios con mayor número de *check-ins* no causaran problemas al momento de asignar la valoración mediante la agrupación, y además se decidió eliminar aquellos POIs con menos de 10 *check-ins* debido a que había muchos negocios con esta condición que causaban problemas en el sistema de recomendación haciendo que el sistema se inclinase más por dar valoraciones bajas.

Los histogramas de la figura 3.6 muestran la frecuencia de aparición de los diferentes valores de los *check-ins* cuando se definen los límites máximos y mínimos y cuando se mantiene sus valores reales.



(a) Histograma con la frecuencia de aparición de los *check-ins* al definir un máximo y mínimo en sus valores



(b) Histograma con la frecuencia de aparición de los *check-ins* originales

Figura 3.6: En la figura 3.6(a) se puede observar que la frecuencia de aparición entre los diferencia valores es más compacta y permite agrupar sus valores en un rango de [1,10], mientras que en la figura 3.6(b) los datos se encuentran más esparcidos y en su agrupamiento la mayoría de los valores se acumularía en el grupo 1.

3.2.2. Cálculo de características

Una vez que se han procesado los datos y se han dividido en los conjuntos de entrenamiento y *test*, el siguiente paso consiste en calcular las características que utilizará el sistema de recomendación en las predicciones. En este trabajo se ha considerado el uso de los mismos tipos de características que en el artículo original: localización y comerciales. Las características de localización se centran en aquellos factores que tienen que ver con los alrededores de la zona, y las características comerciales en la relación de los tipos de negocio que se ubican alrededor.

Características de localización

La información geográfica que hay alrededor de un negocio puede aportar muchos datos sobre la influencia que tiene ese negocio en la zona. Las siguientes características son en base a esos datos:

Distancia al centro de la ciudad Se mide la distancia de la localización al centro de la ciudad. Esto no implica que cuanto menor sea la distancia más popular será la tienda, pero los negocios que tienen cierta influencia suelen encontrarse cerca del centro. La ecuación 3.1 calcula el valor de esta característica siendo d_s la distancia de la localización al centro de la ciudad en metros.

$$DC(s) = 1/\ln(d_s) \tag{3.1}$$

Accesibilidad del tráfico Que un negocio sea lo más accesible permite que más clientes puedan visitar el negocio lo que implica una mayor influencia. Una forma que permite medir esta accesibilidad es tener en cuenta la cantidad de transportes públicos que hay cerca del negocio. La ecuación 3.2 define la accesibilidad del transporte público a un negocio siendo $N(s;r,b), N(s;r,m)$ el número de paradas de autobuses y de metro que hay cerca de la localización s en un radio de r metros respectivamente, y d_b, d_m la distancia mínima a la parada más cercana de autobús y metro respectivamente.

$$AT(s;r) = \frac{\log_2(N(s;r,b)+1)}{\log_2(d_b)} + \frac{\log_2(N(s;r,m)+1)}{\log_2(d_m)} \tag{3.2}$$

Diversidad de tiendas Se mide la variedad de tiendas que hay en los alrededores de la localización. Por lo general si existe diversidad, la localización se encontraría cerca de una zona de comercio donde la demanda podría estar saciada y donde el resto de negocios podrían tener influencia. Por otro lado si no existe diversidad, la localización se situaría en una zona

residencial o recién construida que podría tener demanda de ciertos servicios, pero podría no tener la suficiente influencia. La ecuación 3.3 define la diversidad de tiendas donde $t \in T$ son los tipos de tienda que hay en el sistema, $N(s;r,t)$ el número de tiendas de tipo t en un radio de r metros y $N(s;r)$ el número de tiendas totales que hay en un radio de r metros.

$$Div(s;r) = -\sum_{t \in T} \frac{N(s;r,t)}{N(s;r)} \cdot \ln \frac{N(s;r,t)}{N(s;r)} \tag{3.3}$$

Densidad de clientes Una localización por donde pasa mucha gente puede suponer un factor clave a la hora de elegir si abrir un nuevo negocio en la zona. Zonas donde la densidad de clientes es grande puede suponer que haya una mayor probabilidad de que los potenciales clientes se fijen en el negocio. La ecuación 3.4 calcula la densidad de clientes que hay en la zona siendo $l \in L(s,r)$ las tiendas que hay próximas a localización s dentro del radio r y $R(l)$ el número de *check-ins* de la tienda.

$$Den(s;r) = \ln \sum_{l \in L(s;r)} R(l) \tag{3.4}$$

Características de comercio

Además de las características de localización, se calculan características basadas en la relación de los tipos de las tiendas. A la hora de abrir un nuevo negocio hay que tener en cuenta la competitividad que se puede encontrar con otros negocios así como las relaciones de complementariedad.

Competitividad Negocios de la misma categoría que se encuentren en una misma zona tendrán que competir para conseguir clientes. Por lo general, la competencia es buena para los clientes ya que les permite comparar precios o elegir entre distintos estilos si por ejemplo hablamos de tiendas de ropa. En cambio para los negocios, la competitividad suele ser mala porque conlleva tener que dividir potenciales clientes entre el resto de negocios del mismo tipo. La ecuación 3.5 mide la competitividad para un negocio de tipo t siendo $N(s,t;r)$ el número de negocios de ese mismo tipo que se encuentre dentro de la localización s en un radio de r metros y $N(s;r)$ el número de negocios totales que se encuentran dentro de la localización s en un radio de r metros.

$$Compe(s,t;r) = \frac{N(s,t;r)}{N(s;r)} \tag{3.5}$$

Complementariedad Esta característica consiste en medir el beneficio mutuo que pueden conseguir dos tipos de negocio situados en una misma zona. Por ejemplo, en Tokio es común después de cenar en un restaurante ir a un karaoke para disfrutar el resto de la noche, lo que conlleva un beneficio para ambos negocios. La ecuación 3.6 muestra cómo se calcula la complementariedad para un tipo t siendo $N_{set}(t;r)$ el número de tipos de negocio no repetidos distintos de t que se encuentran dentro de un radio de r metros y N_T el número de tipos de tiendas.

$$Compl(s,t;r) = \frac{2N_{set}(t;r)}{N_T(N_T+1)} \tag{3.6}$$

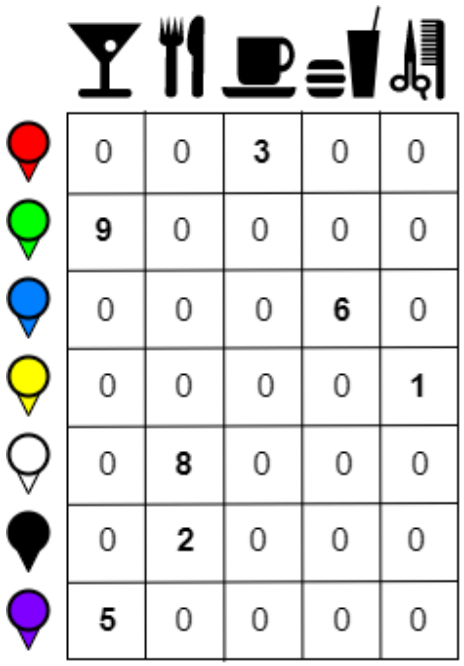
3.2.3. Descripción del modelo

Matriz de valoraciones

En primer lugar, el modelo en el que nos basamos emplea la factorización de matrices SVD para extraer los factores latentes y poder realizar predicciones en base a estos. La matriz de valoraciones se compone de las localizaciones de los negocios de la ciudad (usuarios) y los tipos de negocio (artículos) de tal forma que dada una tienda s y un tipo de negocio t su valoración dentro de la matriz es p_{st}. Como en los ficheros de datos no se encuentra la valoración real de las tiendas con su tipo de negocio correspondiente, se emplea el número de *check-ins* que tenga cada tienda (una medida de su *popularidad*) y se les otorga una valoración en el rango [1,10] a través de una división por bloques teniendo en cuenta los *check-ins* de todas las tiendas que aparecen en el conjunto de entrenamiento, y a los pares restantes se les otorga una valoración de 0. Al tratarse de tiendas y tipos de negocio, la matriz será muy dispersa debido a que una tienda solo puede tener un tipo de negocio cada vez. La figura 3.7 muestra un ejemplo de una matriz reducida.

Factorización de matrices

Gracias a la descomposición SVD el modelo propuesto en el artículo crea representaciones de las tiendas y de los tipos de negocio a través de factores latentes de k dimensiones, siendo el conjunto de vectores S las representaciones de las tiendas y T el conjunto de vectores que representan a los tipos de negocios. Con estas representaciones es posible predecir la valoración (recordemos que, según lo explicado anteriormente, esta valoración es la popularidad normalizada de una tienda medido según su número de check-ins) que tendrá la localización de una tienda con un tipo de negocio concreto a través del producto escalar de las representaciones: $\hat{p}_{st} = S_s T_t^T$.



0	0	**3**	0	0
9	0	0	0	0
0	0	0	**6**	0
0	0	0	0	**1**
0	**8**	0	0	0
0	**2**	0	0	0
5	0	0	0	0

Figura 3.7: Matriz de valoraciones de tiendas y tipos de negocio

Integración de características

Siguiendo las técnicas comunes de la factorización de matrices, el algoritmo del artículo emplea sesgos dentro de la predicción para poder integrar las características de localización y comercio dentro de este. La ecuación 3.7 muestra cómo se calcula la valoración de una tienda con un tipo determinado al incluir los sesgos.

$$\hat{p}_{st} = g + F_s + F_t + S_s T_t^T \tag{3.7}$$

El sesgo global g corresponde a la valoración media de las valoraciones que hay registradas en el sistema y F_s, F_t a los sesgos de la tienda y los sesgos del tipo de negocio respectivamente y sirven para integrar las características ya calculadas. Los sesgos de la tienda F_s se calcula mediante el producto escalar de las características de localización $l_s = \{DC(s), AT(s;r), Div(s;r), Den(s;r)\}$ con el vector del sesgo de la tienda α_s, mientras que el sesgo del tipo de tienda F_t se calcula con el producto escalar de las características de comercio $c_{st} = \{Compe(s,t;r), Compl(s,t;r)\}$ con el vector del sesgo del tipo de negocio β_t. La tabla 3.1 muestra un resumen de las notaciones.

El aprendizaje de los parámetros que aparecen en la tabla 3.1 se realiza mediante la disminución del error cuadrático regularizado (ecuación 2.5) en el cual se introduce un parámetro λ para evitar el sobreentrenamiento. El aprendizaje se realiza sobre aquellas tiendas y tipos de negocio cuya valoración $p_{st} > 0$ y se utiliza el algoritmo de SGD (ecuación 2.3) para disminuir el error en los factores. Una vez que se hayan calculado los valores óptimos para cada parámetro, el algoritmo es capaz de

Anotación	Dimensión	Descripción
S	$N \times K$	Factores latentes de las localizaciones
T	$M \times K$	Factores latentes de los tipos de negocio
F_s	$N \times 4$	Sesgos de las localizaciones
F_t	$M \times 2$	Sesgos de los tipos de negocio
l_s	$N \times 4$	Características de localización de la tienda
c_{st}	$M \times 2$	Características comerciales del tipo de negocio
α	$N \times 4$	Parámetros de las características de la localización
β	$M \times 2$	Parámetros de las características del tipo de negocio

Tabla 3.1: Notaciones que aparecen en el modelo

predecir valoraciones para nuevas tiendas con distintos tipos de negocio.

Aprendizaje de sesgos para nuevas tiendas

El problema principal de este algoritmo se encuentra a la hora de introducir un nueva localización en el sistema de recomendación, ya que no se puede averiguar el valor de los sesgos asociados a este porque no tiene ninguna interacción con algún tipo de negocio. Para solucionar este problema, se implementó un algoritmo para aprender los posibles valores de los sesgos que tendría una nueva localización si se abriese con un tipo de negocio en concreto. El aprendizaje se realiza siguiendo los siguientes pasos:

1.– Se introduce la localización s y el tipo de negocio t a predecir y se calculan las características de localización l_s y las características de comercio c_{st} asociados a estos.

2.– Se buscan todas las localizaciones de las tiendas que sean del mismo tipo de negocio que t que hayan sido utilizados durante el entrenamiento del algoritmo y se guardan en la lista LT.

3.– Se crea un diccionario DT cuyos valores son listas donde se agrupan las localizaciones de las tiendas. Como claves del diccionario se emplean las valoraciones de las localizaciones, en nuestro caso las valoraciones se encuentran en el rango [1,10] por lo que el diccionario separa las localizaciones en 10 grupos.

4.– Se comprueba a qué grupo de DT es más cercana la localización de la tienda s midiendo la distancia media de esta localización a cada grupo calculando la distancia Euclidea (ecuación 2.12) utilizando las características de localización.

5.– Se toma el grupo cuya distancia es menor y se realiza una media de los parámetros de todas las localizaciones que haya en el grupo con el objetivo de calcular los parámetros Θ para la predicción.

6.– Se predice la valoración $\hat{p}_{st}$ de la localización de la tienda s y el tipo de negocio t con los parámetros calculados Θ.

El pseudocódigo 3.1 muestra una representación de los pasos listados anteriormente.

De esta forma el algoritmo de recomendación puede predecir la valoración que tendrá la localización de una tienda dado un tipo de negocio. Su finalidad es que un usuario final introduzca como

input : Localización de la tienda s, Tipo de negocio t
output: Valoración de la localización s con el tipo de negocio t

$l_s \leftarrow DC_s, AT_s, Div, Den_s, L_s$;
$c_{st} \leftarrow Compe_t, Compl_t$;
$TL \leftarrow$ conjuntos de entrenamiento con tipo de negocio t;
for lt *in* LT **do**
    $DT[lt.valoracion] \leftarrow lt$;
end
for $i = 1$ **to** *10* **do**
    $dist_i = \sum_{j \in DT[i]} euclidean(l_s, l_j)/|DT[i]|$;
end
$clase = \text{mín}(dist)$;
$\Theta = clase.parametros/|clase|$;
$\hat{p}_{st} = predecir(s, t, \Theta)$;

Algoritmo 3.1: Pseudocódigo para la predicción de la valoración de una localización con un tipo de tienda concreto

datos de entrada una localización junto con una lista de posibles tipos de negocio para que el sistema de recomendación elabore un *ranking* y así el usuario pueda elegir el tipo de negocio que más le convenza.

4

Pruebas y resultados

Este capítulo está formado por secciones donde se describen los entornos de pruebas y los conjuntos de datos utilizados. Además, se muestran las pruebas realizadas, una reflexión de los resultados obtenidos en las mismas y una comparación con los resultados de los *baselines*.

4.1. Entorno de trabajo

La implementación y las pruebas del proyecto se han realizado sobre dos entornos de trabajo, uno de ellos sobre un ordenador personal Lenovo Z50-70 y el segundo sobre una máquina virtual de Google a través de Google Colaboratory. La tabla 4.1(a) muestra las especificaciones del ordenador personal y la tabla 4.1(b) las especificaciones de la máquina virtual.

Recursos	Características
CPU	Interl(R) Core(TM) i7-4510U
GPU	Intel(R) HD Graphics Family
RAM	15GB
S.O	Ubuntu 18.04
Python	3.6

(a) Lenovo Z50-70

Recursos	Características
CPU	Intel(R) Xeon(R)
GPU	NVIDIA Tesla K80/T4/P4/P100
RAM	13GB
S.O	Ubuntu 18.04
Python	3.7

(b) Google Colaboratory

Tabla 4.1: Especificaciones de los entorno de trabajo

4.2. Descripción de los datos

Dentro de los datos públicos de Foursquare se encuentran registrados POIs de varios lugares del mundo así como un registro de *check-ins* que han pasado por estos. Debido a que el proyecto se centra en el análisis de zonas comerciales en las ciudades, se decidió elegir tres ciudades del mundo para comprobar el funcionamiento de nuestro modelo desarrollado y comparar los resultados entre estos.

Una de las zonas que se ha seleccionado para comprobar el funcionamiento del trabajo ha sido Nueva York porque se trata de una ciudad con mucha actividad, está formado por varias zonas comerciales y se encuentra en el mismo país donde se desarrolló Foursquare, por lo que tanto el número de POIs como de *check-ins* tendría que ser elevado en este conjunto de datos.

La siguiente ciudad que se decidió utilizar para el estudio fue Shanghai debido a que el artículo en que nos basamos para este estudio utiliza esta ciudad para mostrar sus resultados. Shanghai se trata de la ciudad más poblada de China, entonces al igual que Nueva York esta ciudad tendrá mucha movilidad y, por consiguiente, mucha zona comercial que satisfaga las necesidades de los consumidores.

La última ciudad que se eligió para el trabajo fue Madrid porque se quería experimentar su funcionamiento en una ciudad con zonas que nos son conocidas para poder confirmar si los algoritmos funcionan.

La tabla 4.2 muestra una comparativa de usuarios, POIs y *check-ins* del total y las ciudades. Como se puede apreciar Nueva York es la ciudad con más POIs, usuarios y *check-ins* de las ciudades por lo que nuestro modelo tendrá datos suficientes como para entrenarse correctamente y por ello centraremos las pruebas de búsqueda de mejores parámetros en este conjunto. Por otro lado, Shanghai se trata de la ciudad con menos datos en comparación con el resto de las ciudades por lo que el modelo puede que no sea capaz de entrenarse correctamente.

Conjunto	**POIs**	**Usuarios**	***Check-ins***
Todo	11,180,160	86,855,390	90,048,627
Nueva York	58,396	631,920	760,966
Shanghai	9,570	35,932	44,139
Madrid	20,802	135,308	171,055

Tabla 4.2: Conjunto de datos de las ciudades

Las pruebas de recomendación se realizaron sobre unos conjuntos compuestos de cinco tipos de negocio cada uno, con los cuales el algoritmo debía crear un *ranking* de esos tipos para cada localización. Los conjuntos se crearon teniendo en cuenta la ciudad de Nueva York y seleccionando tipos de negocios que fueran similares, además de un conjunto con tipos de negocios similares al que se usaba en el artículo en que está basado nuestro trabajo y un conjunto donde el número de POIs de cada tipo fuera lo más similar posible. En la tabla 4.3 se muestra cada conjunto con el número de POIs de cada tipo de negocio en las ciudades estudiadas.

Como se puede observar en las tablas 4.3(a), 4.3(b), 4.3(c), 4.3(d) y 4.3(e), la ciudad de Nueva York dispone de más POIs de cada tipo en comparación con el resto de ciudades y existen casos como en el conjunto de la tabla 4.3(c) donde una ciudad no presente POIs de un determinado tipo de negocio. Por estas razones, se emplean los conjuntos de Nueva York como ciudad para entrenar

Ciudad	Café	T. postres	S. vinos	T. miscelánea	T. sándwiches
Nueva York	187	154	120	22	169
Shanghai	18	1	9	5	0
Madrid	58	16	11	5	37

(a) Conjunto similar al del artículo

Ciudad	Bar	Café	T. postres	T. miscelánea	T. sándwiches
Nueva York	1291	187	154	22	169
Shanghai	40	18	1	5	0
Madrid	186	58	16	5	37

(b) Conjunto del artículo

Ciudad	L. deportes	T. ropa	T. cosméticos	Bodega	T. miscelánea
Nueva York	450	266	45	136	22
Shanghai	19	8	0	0	5
Madrid	123	25	5	3	5

(c) Conjunto de datos con tiendas de ropa y moda

Ciudad	T. ropa	T. postres	E. rápida	Comercio	T. sándwiches
Nueva York	266	154	76	217	169
Shanghai	8	1	6	18	0
Madrid	25	16	50	55	37

(d) Conjunto de datos para ir a tomar algo

Ciudad	T.ropa	E. rápida	Comercio	L. entretenimiento	L. escénicas
Nueva York	266	76	217	114	233
Shanghai	8	6	18	4	5
Madrid	25	50	55	25	38

(e) Conjunto de datos con numero de POIs similares

Tabla 4.3: Conjuntos de *test* utilizados en el trabajo. Los conjuntos están formados por los tipos de negocio: Café, Tienda de postres (T. postres), Salón de vinos (S. vinos), Tienda de miscelánea (T. miscelánea), Tienda de sándwiches (T. sándwiches), Bar, Lugar de deportes (L. deportes), Tienda de ropa (T. ropa), Tienda de cosméticos (T. cosméticos), Bodega, Establecimiento de comida rápida, (E. rápida), Tienda, Comercio, Lugar de entretenimiento (L. entretenimiento) y Lugar de artes escénicas (L. escénicas)

los parámetros del modelo desarrollado y evaluar su funcionamiento aunque también se analizarán las ciudades de Madrid y Shanghai para ver cómo reacciona ante la escasez y ausencia de tipos de datos respectivamente a la hora de entrenar.

4.3. Estrategia experimental

Las pruebas se realizan utilizando todos los POIs que haya en una determinada ciudad, separando un 70 % de los datos para la fase de entrenamiento y un 30 % para *test*. El conjunto de entrenamiento utiliza todos los POIs independientemente del tipo de negocio que sea mientras que el conjunto de *test* solo utiliza los POIs que pertenezcan a alguno de los conjuntos de datos que se vio en la sección anterior.

El recomendador crea un *ranking* en base a los tipos de negocio que conforma el conjunto de datos de *test* pero no será capaz de puntuar un tipo de negocio que no haya aparecido en la fase de entrenamiento. Una vez obtenido el *ranking* se calcula el valor de las métricas *precision*, *recall* y nDCG (ver sección 2.3).

La evaluación de la eficacia del algoritmo desarrollado se realiza comparando los resultados obtenidos con los resultados de los *baselines*, los cuales están conformados por modelos clasificadores que se utilizan de forma habitual en investigaciones. Para que la comparación de resultados sea justa, se buscan los mejores parámetros para cada clasificador. La tabla 4.4 muestra los parámetros que se han utilizado en cada clasificador y los valores que se han probado en cada uno.

Baseline	**Parámetros**
SVM	decision_function_shape = {'ovo', 'ovr'}, C={3, 1, .5, .1, .01}, kernel={'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'}, degree={3, 2, 1}, gamma={'scale', 'auto'}
Decision Tree	criterion={'gini', 'entropy'}, splitter={'best', 'random'} min_samples_split={.2, .4, .6, .8}, min_samples_leaf={2, 4, 6, 8}, max_depth={2, 3, 4}
Logistic regression	penalty={'l1', 'l2', 'elasticnet', 'none'}, C={3, 1, .5, .1, .01}, solver={'newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'}, multi_class={'auto', 'ovr', 'multinomial'}, max_iter={100, 200, 300}
KNN	n_neighbors={5, 10, 15, 20}, weights={'uniform', 'distance'}

Tabla 4.4: Parámetros de los *baselines* y sus posibles valores

La selección de los mejores parámetros se realizó utilizando la ciudad de Nueva York y utilizando el conjunto de datos similar al artículo (tabla 4.3(a)) para *test*, y la métrica que se seleccionó para comprobar la optimización de las métricas fue $precision@1$. La tabla 4.5 muestra los valores que se

seleccionaron para cada parámetro.

Baseline	**Parámetros**
SVM	decision_function_shape='ovr', C=3, kernel='rbf', gamma='scale'
Decision Tree	criterion='gini', splitter='random', min_samples_split=0.4, min_samples_leaf=2, max_depth=3
Logistic regression	penalty='l1', C=.5, solver='liblinear', multi_class='ovr', max_iter=100
KNN	n_neighbors=10, weights='uniform'

Tabla 4.5: Parámetros de los *baselines* y sus valores elegidos

Además de comparar los resultados del algoritmo con los *baselines*, para comprobar si añadiendo características de localización y comercio se mejora la recomendación, se han realizado pruebas utilizando una versión simple de la factorización de matrices, y pruebas utilizando solo uno de los dos tipos de características al mismo tiempo.

Por último, se han comparado los resultados obtenidos del algoritmo desarrollado con otra versión del algoritmo en el cual se añade el factor de luminosidad dentro de las características de localización. Durante el desarrollo del modelo se pensó que incluir esta característica podría mejorar los resultados de la recomendación y se decidió realizar pruebas sobre un conjunto de datos de test para comprobar si esta hipótesis era cierta, pero hay que recordar que estos datos ni se usan ni se plantean en el artículo de referencia.

4.4. Pruebas

En esta sección se describen los parámetros que se han elegido para el recomendador, las pruebas realizadas y se comparan los resultados obtenidos con los resultados de los *baselines* y otras versiones del recomendador.

4.4.1. Parámetros del recomendador

El recomendador implementado se compone de cuatro parámetros: número de iteraciones en la fase de entrenamiento, dimensión de los factores latentes, la tasa de aprendizaje y un parámetro regularizador del error. La manera en la que se obtuvo el valor óptimo para cada parámetro fue comprobar

qué valores de los parámetros ayudaban a reducir el MAE y RMSE (ver sección 2.3) en las predicciones de las valoraciones de todos los pares de localizaciones y tipos de negocio, cuya valoración fuera mayor que 0, con respecto a sus valoraciones originales al finalizar la fase de entrenamiento tal y como se explicó en la sección 3.2.3.

En primer lugar se comprobó a partir de cuántas iteraciones en la fase de entrenamiento el recomendador dejaba de reducir el error. La figura 4.1 muestra la progresión del error según se añadían más iteraciones. El error decrece desde el principio y comienza a estabilizarse a las 80 iteraciones, pero al aumentar el número de iteraciones el RMSE empieza a crecer y alcanza un pico en las 160 iteraciones para volver a decrecer. Con respecto al MAE, el error se mantiene más estable pero sufre un pequeño aumento a las 160 iteraciones. Visto los resultados, se decidió utilizar 80 iteraciones para el entrenamiento.

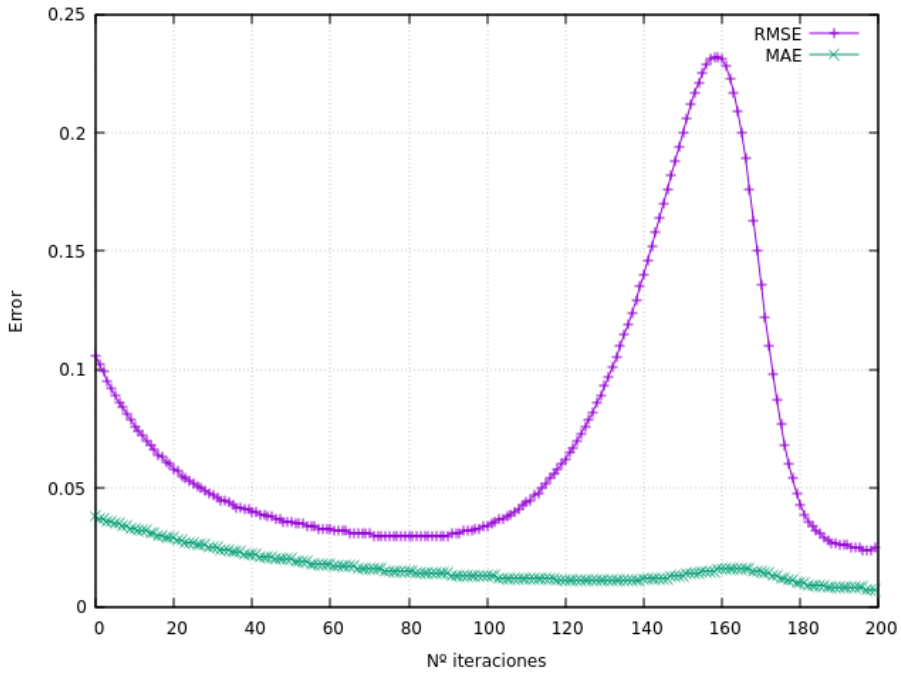


Figura 4.1: Error en el número de iteraciones de la fase de entrenamiento

El siguiente parámetro que se estudió fue la dimensión de los factores latentes que determina la dimensión de las matrices que se crearán al descomponer con la técnica de SVD. La figura 4.2 muestra el error al aumentar la dimensión de los factores latentes. El error va decreciendo a medida que se aumenta la dimensión y comienza a estabilizarse al cabo de una dimensión de 50, por lo que se decidió utilizar ese valor de dimensión ya que elegir un valor mayor podría causar sobreentrenamiento y causar problemas en la recomendación.

Por último, se analizaron los valores de tasa de aprendizaje y del parámetro regularizador del error de manera conjunta porque ambos se utilizan dentro de SGD. Como se observa en la tabla 4.6 el error RMSE se reduce al mínimo cuando los valores de λ y γ son 0.004 y 0.002 respectivamente, mientras

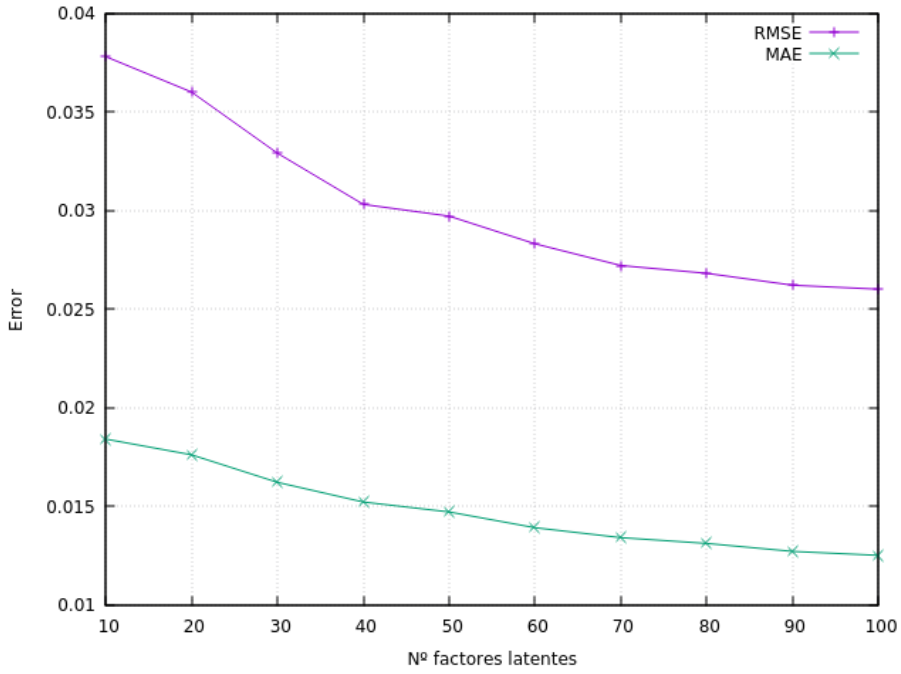


Figura 4.2: Error en la dimensión de los factores latentes

que en la tabla 4.7 se muestra el error MAE y los valores de λ y γ que alcanzan un error mínimo son 0.005 y 0.002 respectivamente. Con estos resultados se optó por utilizar los valores $\lambda = 0{,}004$ y $\gamma = 0{,}002$ como parámetros para el entrenamiento del modelo debido a que RMSE penaliza más el error que MAE y a que la diferencia del error por el valor de lambda en MAE es muy pequeña.

RMSE	0.002	0.004	0.006	0.008	0.010
0.001	0.0470	0.0481	0.0498	0.0517	0.0538
0.002	0.0264	0.0289	0.0319	0.0349	0.0377
0.003	0.0197	0.0230	0.0265	0.0297	0.0326
0.004	**0.0178**	0.0233	0.0303	0.0383	0.0472
0.005	0.0209	0.0395	0.0657	0.0959	0.1308

Tabla 4.6: RMSE en la tasa de aprendizaje y parámetro regularizador del error. Las filas indican la tasa de aprendizaje λ y las columnas el parámetro regularizador del error γ. En negrita se resalta el error mínimo.

4.4.2. Comparación con baselines

La forma de probar si el modelo desarrollado en este trabajo (M) funciona es comparando sus resultados con los resultados de algoritmos de aprendizaje automáticos populares que se utilizan normalmente en las investigaciones. Los algoritmos seleccionados son: Support Vector Machine (SVM), árboles de decisión (DT), regresión logística (LR) y KNN. Estos algoritmos se tratan de algoritmos cla-

MAE	0.002	0.004	0.006	0.008	0.010
0.001	0.0190	0.0217	0.0243	0.0267	0.0289
0.002	0.0110	0.0141	0.0168	0.0191	0.0211
0.003	0.0076	0.0105	0.0129	0.0147	0.0162
0.004	0.0060	0.0087	0.0107	0.0122	0.0133
0.005	**0.0053**	0.0078	0.0095	0.0108	0.0121

Tabla 4.7: MAE en la tasa de aprendizaje y parámetro regularizador del error. Las filas indican la tasa de aprendizaje λ y las columnas el parámetro regularizador del error γ. En negrita se resalta el error mínimo.

sificadores, por lo que la comparación de resultados entre el sistema de recomendación implementado y estos algoritmos no sería posible al tratarse de sistemas de diferentes tipos, pero gracias a la librería de Scikit-learn [1] de Python estos clasificadores pueden predecir la probabilidad que tiene un tipo de negocio concreto de pertenecer a una localización concreta (y no sólo predecir qué tipo de negocio es más adecuado para una localización) mediante la llamada a la función *pred_proba()*, y de esta forma hacer posible una comparación justa de resultados.

Ciudad	Algoritmo	Prec@1	Prec@2	Recall@1	Recall@2	nDCG@1	nDCG@2
Nueva York	M	**0.365**	0.270	**0.365**	0.540	**0.365**	**0.475**
	SVM	0.320	**0.277**	0.320	**0.555**	0.320	0.468
	DT	0.250	0.255	0.250	0.510	0.250	0.414
	LR	0.320	0.260	0.320	0.520	0.320	0.446
	KNN	0.300	0.237	0.300	0.475	0.254	0.379
Shanghai	M	**0.700**	**0.450**	**0.700**	**0.900**	**0.700**	**0.826**
	SVM	0.600	0.400	0.600	0.800	0.600	0.726
	DT	0.600	**0.450**	0.600	**0.900**	0.600	0.757
	LR	0.600	0.400	0.600	0.800	0.600	0.726
	KNN	**0.700**	**0.450**	**0.700**	**0.900**	0.450	0.639
Madrid	M	0.305	0.222	0.305	0.444	0.305	0.392
	SVM	0.417	**0.333**	0.417	**0.666**	**0.417**	0.574
	DT	**0.444**	**0.333**	**0.444**	**0.666**	0.400	0.556
	LR	**0.444**	**0.333**	**0.444**	**0.666**	0.444	**0.584**
	KNN	0.417	0.292	0.417	0.583	0.236	0.354

Tabla 4.8: Resultados de nuestro modelo y de los baselines con el conjunto similar al del artículo. En negrita se muestran los mejores resultados en cada caso.

La tabla 4.8 muestra los resultados de las pruebas en las tres ciudades utilizando el conjunto de *test*

[1] https://scikit-learn.org/stable/

similar al utilizado en el artículo en que nos basamos. En las pruebas con los POIs de Nueva York se puede observar que nuestro algoritmo obtiene mejores resultados que el resto de algoritmos excepto en *precision* y *recall* cuando la profundidad del *ranking* es de tamaño 2, que en ese caso SVM obtiene mejores resultados aunque la diferencia no sea muy grande. Con respecto a las pruebas con los POIs de Shanghai, nuestro algoritmo obtiene los mejores resultados en cada métrica pero los algoritmos DT y, en especial, KNN llegan a igualar los resultados de nuestro modelo. Por último, en las pruebas con los POIs de Madrid nuestro modelo obtiene peores resultados que los *baselines*, los cuales obtienen resultados similares en varias de las métricas.

Con estos resultados se puede afirmar que nuestro algoritmo necesita grandes cantidades de POIs de cada tipo para que sus predicciones puedan superar a los predicciones de otros algoritmos. Por otro lado, si no se dispone de suficientes POIs para poder entrenar, los algoritmos KNN o árboles de decisión pueden utilizarse para resolver este tipo de problemas ya que logran buenos resultados en ciertas situaciones.

El resto de pruebas con los diferentes conjuntos de *test* aparecen en el Anexo A. Como resumen de estas pruebas podemos concluir que el recomendador es muy sensible con respecto a la falta de información de los tipos de negocios, de forma que consigue buenos resultados si tiene mucha información de estos y su eficiencia decae si escasea la información.

4.4.3. Análisis del efecto de las características

En estas pruebas vamos a comprobar cómo la integración de las características de localización y comercio dentro de la factorización de matrices influye en la predicción de tipos de negocio. Se realizan pruebas utilizando únicamente la factorización de matrices e integrando una de las características al mismo tiempo con el conjunto de *test* similar al del artículo que nos basamos. Recordemos que las características de localización están compuestas por la distancia al centro de la ciudad, accesibilidad del tráfico, diversidad de tiendas y densidad de clientes, mientras que las características de comercio están formadas por la competitividad y complementariedad (subsección 3.2.2)

La tabla 4.9 muestra los resultados de utilizar solo SVD, SVD integrando características de comercio (SVD+C) y SVD integrando características de localización (SVD+L) en cada una de las tres ciudades. Analizando los resultados en los conjuntos de datos de las tres ciudades se puede observar que se obtienen mejores resultados cuando se integran exclusivamente las características de comercio, lo que puede indicar que funciona mejor si solo se tiene en cuenta la competencia que existe alrededor de la zona y los tipos de negocio complementarios, aunque en los conjuntos de datos de Nueva York y Madrid, cuando se tienen en cuenta las dos primeras posiciones del *ranking*, el modelo sin integrar ningún grupo de características obtiene mejores resultados.

Ciudad	Tipo	Prec@1	Prec@2	Recall@1	Recall@2	nDCG@1	nDCG@2
	M	*0.365*	*0.270*	*0.365*	*0.540*	*0.365*	*0.475*
	SVD	0.265	**0.272**	0.265	**0.545**	0.165	0.442
Nueva York	SVD+C	**0.350**	0.265	**0.350**	0.530	**0.350**	**0.463**
	SVD+L	0.260	0.202	0.260	0.405	0.260	0.359
	M	*0.700*	*0.450*	*0.700*	*0.900*	*0.700*	*0.826*
	SVD	0.200	0.200	0.200	0.400	0.200	0.326
Shanghai	SVD+C	**0.700**	**0.450**	**0.700**	**0.900**	**0.700**	**0.826**
	SVD+L	0.600	0.400	0.600	0.800	0.600	0.726
	M	*0.305*	*0.222*	*0.305*	*0.444*	*0.305*	*0.392*
	SVD	0.222	**0.333**	0.222	**0.667**	0.222	**0.503**
Madrid	SVD+C	**0.277**	0.208	**0.277**	0.417	**0.278**	0.352
	SVD+L	0.222	0.250	0.222	0.500	0.208	0.306

Tabla 4.9: Resultados de nuestro modelo utilizando distintas combinaciones de las características. En negrita se muestran los mejores resultados en cada caso. Nótese que se ha incluido el modelo completo de las pruebas anteriores (M) para comparar, pero no se tiene en cuenta al remarcar con negrita el mejor valor.

4.4.4. Usando datos de luminosidad

En esta prueba se pretende comparar la eficiencia del recomendador al añadir una característica extra al conjunto de características de localización original del artículo en que nos basamos. Esta característica se basa en el hecho de que la contaminación lumínica es un claro indicador de la actividad de las personas, y por lo tanto los negocios con mayor influencia buscan situarse en lugares con alta contaminación lumínica o que la distancia a estas zonas sean lo más pequeña posible, lo que acaba contribuyendo a aumentar todavía más esta contaminación. La ecuación 4.1 muestra la medición de la luminosidad siendo $lum(s)$ la luminosidad de la localización s y $lumMin, lumMax$ la luminosidad mínima y máxima de la ciudad respectivamente.

$$L(s) = \frac{lum(s) - lumMin}{lumMax - lumMin} \tag{4.1}$$

En la tabla 4.10 se muestran los resultados al incluir la característica de luminosidad dentro de las características de localización (Luz) y los resultados al no incluir esta característica manteniendo la versión original del modelo (No Luz). Al igual que las pruebas anteriores, estas pruebas fueron hechas utilizando el conjunto de *test* similar al artículo original. Como se puede observar, al utilizar los POIs de las ciudades de Nueva York y Shanghai esta nueva característica no consigue mejorar los resulta-

Ciudad	Tipo	Prec@1	Prec@2	Recall@1	Recall@2	nDCG@1	nDCG@2
	No Luz	**0.365**	**0.270**	**0.365**	**0.540**	**0.365**	**0.475**
Nueva York	Luz	**0.365**	**0.270**	**0.365**	**0.540**	**0.365**	**0.475**
	No Luz	**0.700**	**0.450**	**0.700**	**0.900**	**0.700**	0.826
Shanghai	Luz	**0.700**	**0.450**	**0.700**	**0.900**	**0.700**	**0.900**
	No Luz	**0.305**	**0.222**	**0.305**	**0.444**	**0.305**	**0.392**
Madrid	Luz	0.277	**0.222**	0.277	**0.444**	0.278	0.382

Tabla 4.10: Resultados del modelo añadiendo la característica de luminosidad. En negrita se muestran los mejores resultados en cada caso.

dos, mientras que en las pruebas de Madrid el añadir esta característica empeora los resultados del recomendador desarrollado. Algunos de los motivos que pueden causar que no mejore los resultados respecto a la versión normal pueden deberse a que los POIs de este conjunto de *test* no utilicen esta característica, debido a que se tratan de tipos de negocios que tengan mayor actividad de día y que por la noche cierren, o que la luminosidad sea redundante con otra de las característica de localización como la distancia al centro de la ciudad.

Otra de las pruebas que se realizaron fue comprobar si esta característica mejoraba los resultados con respecto a la versión original, pero en este caso se prescindía de las características de comercio como se hacía en la subsección anterior (ver 4.4.3).

Ciudad	Tipo	Prec@1	Prec@2	Recall@1	Recall@2	nDCG@1	nDCG@2
	SVD+L	0.260	0.202	0.260	0.404	0.260	0.359
Nueva York	SVD+LL	**0.280**	**0.215**	**0.280**	**0.430**	**0.280**	**0.375**
	SVD+L	**0.600**	**0.400**	**0.600**	**0.800**	**0.600**	**0.726**
Shanghai	SVD+LL	0.200	0.200	0.200	0.400	0.200	0.326
	SVD+L	**0.222**	**0.250**	**0.222**	**0.500**	0.208	0.306
Madrid	SVD+LL	**0.222**	0.194	**0.222**	0.389	**0.222**	**0.327**

Tabla 4.11: Resultados del modelo añadiendo la característica de luminosidad a las características de localización y sin utilizar características de comercio. En negrita se muestran los mejores resultados en cada caso.

En la tabla 4.11 se muestra la comparación de los resultados utilizando la factorización de matrices sin la característica de luminosidad ni características de comercio (SVD+L) y la factorización de matrices sin características de comercio e integrando la característica de luminosidad (SVD+LL) utilizando el conjunto de *test* similar al artículo en que nos basamos. Utilizando los POIs de la ciudad de Nueva York la característica de luminosidad ayuda a mejorar los resultados cuando se integra dentro de las características de localización, lo que puede indicar que su uso sí sea relevante para ayudar en las

recomendaciones. Por otro lado, la recomendación con los datos de Shanghai es mejor cuando no se usa la característica de luminosidad, y en las pruebas de la ciudad de Madrid los resultados son muy similares cuando se integra tal característica y cuando no.

Partiendo de que el conjunto de datos de Nueva York es más completo que el de las otras dos ciudades, la característica de luminosidad puede ser un factor relevante a la hora de predecir qué tipo de negocio es más recomendable para una localización, pero hay que tener en cuenta que la granularidad de la luminosidad puede que no fuera suficiente como para que los datos de luminosidad se aprovechen dentro del modelo y, además, qué tipos de negocios se van a utilizar ya que a ciertos tipos de negocios puede no influirles tal característica si su actividad se rige más en momentos diurnos que nocturnos.

5

Conclusiones y trabajo futuro

En este capítulo se expone un resumen de todos los resultados e ideas que se han obtenido y se proponen una serie de ideas que podrían plantearse como futuras investigaciones en base a este trabajo.

5.1. Conclusión

En este proyecto se ha abarcado el desafío de recomendación de localizaciones para POIs, en concreto la creación de un *ranking* de tipos de POI para una determinada localización a partir de una lista de posibles opciones que es creada por el usuario final que utilice el sistema de recomendación. Para ello, hemos desarrollado un algoritmo basado en el artículo *Shop-Type Recommendation Leveraging the Data from Social Media and Location-Based Services* [29] el cual emplea la técnica de factorización de matrices, perteneciente a los sistemas de recomendación basados en filtrado colaborativo, junto con una serie de características de localización y comerciales generadas a partir de datos LBS. Las pruebas realizadas con este modelo fueron realizadas utilizando datos públicos de Foursquare, los cuales permitían analizar el comportamiento del algoritmo mediante conjuntos de datos diferentes entre ciudades.

Se ha podido observar que el algoritmo desarrollado es capaz de obtener factores latentes a partir de conjuntos de datos muy dispersos y obtener resultados que superan a otros métodos famosos. Hay que tener en cuenta que ante ciertas circunstancias los resultados del modelo desarrollado no consiguen superar los resultados de otros modelos e incluso quedan por debajo. Por otro lado, además del uso de factores latentes, la integración de características de localización, como la distancia al centro de la ciudad o la cercanía a diferentes paradas de transporte público, y en especial las características comerciales, en los cuales se representa la competitividad o los tipos de negocio complementarios en la zona, logran ayudar al modelo a la hora de recomendar el mejor tipo de negocio.

Buscando nuevas maneras de mejorar el modelo del artículo original, se han realizado pruebas para comprobar cómo puede afectar al funcionamiento del modelo original la integración de una característica que tiene en cuenta la luminosidad de la localización dentro del grupo de características

de localización, logrando obtener mejores resultados cuando solo se utilizan este grupo por sí solo, pero no cuando se utilizan junto con el grupo de características de comercio. Una de las conclusiones que se ha podido obtener es que esta característica apenas influye cuando se predice sobre tipos de negocio con actividad diurna.

Por último, tener en cuenta que la eficiencia del algoritmo desarrollado se encuentra muy limitado al conjunto de datos con el que se entrene ya que, como se ha podido ver en las pruebas, los resultados en diferentes ciudades utilizando los mismos tipos de negocio como conjunto de *test* son muy dispares entre ellos.

5.2. Trabajo futuro

Con perspectiva al futuro sería interesante realizar más pruebas con distintos conjuntos de *test*, en especial con tipos de negocio cuya actividad sea exclusivamente nocturna. Además, ya que solo se han realizado pruebas con ciudades localizadas en continentes diferentes sería interesante realizar pruebas con otras ciudades localizadas en esos mismos continentes (o países) y comparar las diferencias de los resultados entre estos. Otra investigación que se podría realizar en base a este trabajo es comparar resultados utilizando fuentes de datos distintas como las de Yelp.

Sería interesante también comprobar cómo otros sistemas de recomendación como los modelos basados en memoria o basado en contenidos son capaces de afrontar este mismo problema. Recientemente se ha publicado un artículo donde los autores enfocaban este mismo problema a través de los modelos basados en memoria, de manera que se basaba en encontrar localizaciones similares a una localización dada a través de características como las que se han visto en este trabajo y recomendar el tipo de negocio con mejor puntuación entre las localizaciones similares [33].

Por último, como en este trabajo se ha estudiado cómo la contaminación lumínica puede influir en la recomendación de localizaciones de POIs, se podría investigar cómo otros tipos de contaminaciones pueden influir en la recomendación como lo hace la lumínica. Recientemente están surgiendo estudios que investigan el nivel de polución de las zonas, como por ejemplo la huella de carbono que se produce en los viajes de taxis [37], lo que podría convertirse en nuevos datos a tener en cuenta para este tipo de problemas ya que zonas muy contaminadas pueden convertirse en zonas no deseables para abrir un nuevo negocio. Los datos sobre el nivel de polución de ciertas ciudades pueden obtenerse de sus respectivos ayuntamientos, como por ejemplo Madrid [1], que ofrecen datos anuales de la calidad del aire en los diferentes distritos de la ciudad.

[1] https://datos.madrid.es/portal/site/egob

Bibliografía

[1] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, *Recommender Systems Handbook*. Berlin, Heidelberg: Springer-Verlag, 1st ed., 2010.

[2] L. Candillier, K. Jack, F. Fessant, and F. Meyer, *State of the Art Recommender System*, pp. 1–22. 04 2009.

[3] M. D. Ekstrand and J. A. Konstan, "Recommender systems notation: Proposed common notation for teaching and research," *CoRR*, vol. abs/1902.01348, 2019.

[4] C. Sirikayon, P. Thusaranon, and W. Sukpongthai, "A preprocessing matrix factorization on collaborative filtering based library book recommendation system," in *Proceedings of the 2018 International Conference on Data Science and Information Technology*, DSIT '18, (New York, NY, USA), p. 33–37, Association for Computing Machinery, 2018.

[5] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '02, (New York, NY, USA), p. 253–260, Association for Computing Machinery, 2002.

[6] S. D. Barman, M. Hasan, and F. Roy, "A genre-based item-item collaborative filtering: Facing the cold-start problem," in *Proceedings of the 2019 8th International Conference on Software and Computer Applications*, ICSCA '19, (New York, NY, USA), p. 258–262, Association for Computing Machinery, 2019.

[7] S. Sedhain, S. Sanner, D. Braziunas, L. Xie, and J. Christensen, "Social collaborative filtering for cold-start recommendations," in *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, (New York, NY, USA), p. 345–348, Association for Computing Machinery, 2014.

[8] M. B. Dias, D. Locher, M. Li, W. El-Deredy, and P. J. Lisboa, "The value of personalised recommender systems to e-business: A case study," in *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, (New York, NY, USA), p. 291–294, Association for Computing Machinery, 2008.

[9] P. Melville and V. Sindhwani, *Recommender Systems*, pp. 829–838. Boston, MA: Springer US, 2010.

[10] S. Khusro, Z. Ali, and I. Ullah, *Recommender Systems: Issues, Challenges, and Research Opportunities*, pp. 1179–1189. 02 2016.

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

[12] P. D. Meo, "Trust prediction via matrix factorisation," *ACM Trans. Internet Technol.*, vol. 19, Sept. 2019.

[13] V. Nikolaenko, S. Ioannidis, U. Weinsberg, M. Joye, N. Taft, and D. Boneh, "Privacy-preserving matrix factorization," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, (New York, NY, USA), p. 801–812, Association for Computing Machinery, 2013.

[14] C.-M. Cheng and X.-Q. Jin, *Matrix Decomposition*, pp. 1280–1290. New York, NY: Springer New York, 2018.

[15] C. Zeng, C.-X. Xing, and L.-Z. Zhou, "Similarity measure and instance selection for collaborative filtering," WWW '03, (New York, NY, USA), p. 652–658, Association for Computing Machinery, 2003.

[16] R. D. Burke, "Hybrid recommender systems: Survey and experiments," *User Model. User Adapt. Interact.*, vol. 12, no. 4, pp. 331–370, 2002.

[17] G. Popescu, "Group recommender systems as a voting problem," in *Online Communities and Social Computing - 5th International conference, OCSC 2013, Held as Part of HCI International 2013, Las Vegas, NV, USA, July 21-26, 2013. Proceedings* (A. A. Ozok and P. Zaphiris, eds.), vol. 8029 of *Lecture Notes in Computer Science*, pp. 412–421, Springer, 2013.

[18] I. A. Junglas and R. T. Watson, "Location-based services," *Commun. ACM*, vol. 51, p. 65–69, Mar. 2008.

[19] Y. Liu, C. Liu, X. Lu, M. Teng, H. Zhu, and H. Xiong, "Point-of-interest demand modeling with human mobility patterns," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, (New York, NY, USA), p. 947–955, Association for Computing Machinery, 2017.

[20] H. Li, Y. Ge, R. Hong, and H. Zhu, "Point-of-interest recommendations: Learning potential check-ins from friends," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, (New York, NY, USA), p. 975–984, Association for Computing Machinery, 2016.

[21] J.-B. Griesner, T. Abdessalem, and H. Naacke, "Poi recommendation: Towards fused matrix factorization with geographical and temporal influences," in *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, (New York, NY, USA), p. 301–304, Association for Computing Machinery, 2015.

[22] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, and Y. Rui, "Geomf: Joint geographical modeling and matrix factorization for point-of-interest recommendation," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, (New York, NY, USA), p. 831–840, Association for Computing Machinery, 2014.

[23] C. Song, J. Wen, and S. Li, "Personalized poi recommendation based on check-in data and geographical-regional influence," in *Proceedings of the 3rd International Conference on Machine Learning and Soft Computing*, ICMLSC 2019, (New York, NY, USA), p. 128–133, Association for Computing Machinery, 2019.

[24] H. Katsumi, W. Yamada, and K. Ochiai, "Generic poi recommendation," in *Adjunct Proceedings*

of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers, UbiComp-ISWC '20, (New York, NY, USA), p. 46–49, Association for Computing Machinery, 2020.

[25] X. Li, D. Han, J. He, L. Liao, and M. Wang, "Next and next new poi recommendation via latent behavior pattern inference," *ACM Trans. Inf. Syst.*, vol. 37, Sept. 2019.

[26] S. Oppokhonov, S. Park, and I. K. E. Ampomah, "Current location-based next poi recommendation," in *Proceedings of the International Conference on Web Intelligence*, WI '17, (New York, NY, USA), p. 831–836, Association for Computing Machinery, 2017.

[27] D. Karamshuk, A. Noulas, S. Scellato, V. Nicosia, and C. Mascolo, "Geo-spotting: Mining online location-based services for optimal retail store placement," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, (New York, NY, USA), p. 793–801, Association for Computing Machinery, 2013.

[28] M. Xu, T. Wang, Z. Wu, J. Zhou, J. Li, and H. Wu, "Demand driven store site selection via multiple spatial-temporal data," in *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPACIAL '16, (New York, NY, USA), Association for Computing Machinery, 2016.

[29] Z. Yu, M. Tian, Z. Wang, B. Guo, and T. Mei, "Shop-type recommendation leveraging the data from social media and location-based services," *ACM Trans. Knowl. Discov. Data*, vol. 11, July 2016.

[30] B. Eravci, N. Bulut, C. Etemoglu, and H. Ferhatosmanoğlu, "Location recommendations for new businesses using check-in data," in *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, pp. 1110–1117, 2016.

[31] W. Cai, Y. Wang, R. Lv, and Q. Jin, "An efficient location recommendation scheme based on clustering and data fusion," *Comput. Electr. Eng.*, vol. 77, pp. 289–299, 2019.

[32] A. K. P, S. S. G, P. K. R. Maddikunta, T. R. Gadekallu, A. Al-Ahmari, and M. H. Abidi, "Location based business recommendation using spatial demand," *Sustainability*, vol. 12, no. 10, 2020.

[33] X. Mao, X. Zhao, J. Lin, and E. Herrera-Viedma, "Utilizing multi-source data in popularity prediction for shop-type recommendation," *Knowl. Based Syst.*, vol. 165, pp. 253–267, 2019.

[34] Y. Xu, Y. Shen, Y. Zhu, and J. Yu, "Ar^{2}net: An attentive neural approach for business location selection with satellite data and urban data," *ACM Trans. Knowl. Discov. Data*, vol. 14, no. 2, pp. 20:1–20:28, 2020.

[35] D. Yang, D. Zhang, L. Chen, and B. Qu, "Nationtelescope: Monitoring and visualizing large-scale collective behavior in lbsns," *J. Netw. Comput. Appl.*, vol. 55, pp. 170–180, 2015.

[36] D. Yang, D. Zhang, and B. Qu, "Participatory cultural mapping based on collective behavior data in location-based social networks," *ACM Trans. Intell. Syst. Technol.*, vol. 7, no. 3, pp. 30:1–30:23, 2016.

[37] M. Sun, C. Xue, Y. Cheng, L. Zhao, and Z. Long, "Analyzing spatiotemporal daily travel source carbon emissions based on taxi trajectory data," *IEEE Access*, vol. 9, pp. 107012–107023, 2021.

Definiciones

Alternating Least Squares Mínimos Cuadrados Alternados.

cluster Agrupamiento.

e-commerce Comercio electrónico.

embedding Espacio de dimensiones reducidas al que se pueden trasladar vectores de dimensiones altas.

item Artículo.

missing value Dato faltante.

rating Calificación.

Relevance Feedback Retroalimentación de relevancia.

Stochastic Gradient Descent Descenso por Gradiente Estocástico.

Acrónimos

ALS Alternating Least Squares.

GPS Global Positioning System.

IDCG Ideal Discounted Cumulative Gain.

KNN K-Nearest Neighbor.

LBS Location-based Services.

LBSN Location Based Social Network.

MAE Mean Absolute Error.

MUI2-ICSI Máster Universitario en Investigación e Innovación en Inteligencia Computacional y Sistemas Interactivos.

nDCG Normalized Discounted Cumulative Gain.

NLP Natural Language Processing.

POI Point-Of-Interest.

RMSE Root Mean Squared Error.

SGD Stochastic Gradient Descent.

SVD Singular Value Decomposition.

SVM Support Vector Machine.

Apéndices

A

Otras pruebas

En este apéndice se muestran los resultados del resto de conjunto de *test* que se hicieron para las pruebas de la subsección 4.4.2.

Ciudad	Algoritmo	Prec@1	Prec@2	Recall@1	Recall@2	nDCG@1	nDCG@2
Nueva York	M	0.683	0.407	0.683	0.814	0.683	0.766
	SVM	**0.736**	**0.422**	**0.736**	**0.855**	**0.736**	**0.805**
	DT	0.699	0.380	0.699	0.795	0.699	0.760
	LR	0.680	0.403	0.687	0.807	0.687	0.762
	KNN	0.677	0.393	0.677	0.787	0.635	0.703
Shanghai	M	0.550	0.450	**0.550**	0.900	**0.550**	0.770
	SVM	**0.600**	0.450	**0.600**	0.900	**0.600**	0.789
	DT	**0.600**	0.450	**0.600**	0.900	**0.600**	0.789
	LR	**0.600**	0.450	**0.600**	0.900	**0.600**	**0.790**
	KNN	0.400	**0.475**	**0.400**	**0.950**	0.408	0.603
Madrid	M	0.607	0.376	0.607	0.753	0.607	0.700
	SVM	**0.640**	**0.410**	**0.640**	**0.820**	**0.640**	**0.754**
	DT	0.607	**0.410**	0.607	**0.820**	0.607	0.731
	LR	**0.640**	**0.410**	**0.640**	**0.820**	**0.640**	**0.754**
	KNN	0.629	0.404	0.629	0.810	0.510	0.600

Tabla A.1: Resultados de nuestro modelo y de los baselines con el conjunto del artículo. En negrita se muestran los mejores resultados en cada caso.

Ciudad	Algoritmo	Prec@1	Prec@2	Recall@1	Recall@2	nDCG@1	nDCG@2
Nueva York	M	0.565	0.384	0.565	0.768	0.565	0.693
	SVM	0.586	0.401	0.586	0.803	0.586	0.723
	DT	0.540	**0.417**	0.540	**0.835**	0.540	0.726
	LR	**0.593**	0.405	**0.593**	0.810	**0.593**	**0.730**
	KNN	0.586	0.381	0.586	0.761	0.436	0.540
Shanghai	M	0.330	0.400	0.333	0.800	0.333	**0.828**
	SVM	**0.667**	**0.433**	**0.667**	**0.867**	**0.667**	0.793
	DT	0.600	0.367	0.600	0.733	0.622	0.720
	LR	**0.667**	**0.433**	**0.667**	**0.867**	**0.667**	0.793
	KNN	0.364	0.364	0.364	0.727	0.242	0.481
Madrid	M	0.588	0.333	0.588	0.667	0.588	0.638
	SVM	**0.804**	0.451	**0.804**	0.902	**0.804**	**0.866**
	DT	0.784	**0.461**	0.784	**0.921**	0.784	0.865
	LR	**0.804**	0.451	0.803	0.902	**0.804**	**0.866**
	KNN	0.470	0.451	0.470	0.902	0.504	0.587

Tabla A.2: Resultados de nuestro modelo y de los baselines con el conjunto de tiendas de ropa. En negrita se muestran los mejores resultados en cada caso.

Ciudad	Algoritmo	Prec@1	Prec@2	Recall@1	Recall@2	nDCG@1	nDCG@2
Nueva York	M	0.361	0.297	0.361	0.595	0.361	0.508
	SVM	0.428	0.327	0.428	0.654	0.428	0.571
	DT	0.416	0.305	0.416	0.610	0.416	0.538
	LR	**0.450**	**0.333**	**0.450**	**0.665**	**0.450**	**0.586**
	KNN	0.349	0.264	0.349	0.528	0.350	0.460
Shanghai	M	**0.455**	0.364	**0.455**	0.727	**0.455**	0.627
	SVM	**0.455**	0.364	**0.455**	0.727	**0.455**	0.627
	DT	**0.455**	**0.409**	**0.455**	**0.818**	**0.455**	**0.655**
	LR	**0.455**	0.364	**0.455**	0.627	**0.455**	0.627
	KNN	0.364	0.364	0.364	0.727	0.242	0.481
Madrid	M	0.283	0.283	0.283	0.565	0.283	0.461
	SVM	0.413	0.304	0.413	0.609	0.413	0.536
	DT	0.326	0.293	0.326	0.587	0.326	0.491
	LR	**0.435**	**0.315**	**0.435**	**0.630**	**0.435**	**0.558**
	KNN	0.370	0.250	0.370	0.500	0.264	0.392

Tabla A.3: Resultados de nuestro modelo y de los baselines con el conjunto de tipos para tomar algo. En negrita se muestran los mejores resultados en cada caso.

Ciudad	Algoritmo	Prec@1	Prec@2	Recall@1	Recall@2	nDCG@1	nDCG@2
Nueva York	M	**0.477**	**0.365**	**0.477**	**0.730**	**0.477**	**0.637**
	SVM	0.138	0.114	0.138	0.228	0.138	0.195
	DT	0.126	0.109	0.126	0.218	0.125	0.184
	LR	0.113	0.100	0.113	0.200	0.113	0.168
	KNN	0.156	0.108	0.156	0.215	0.154	0.192
Shanghai	M	0.167	0.167	0.167	0.333	0.167	0.272
	SVM	**0.278**	0.222	**0.278**	0.444	**0.278**	0.383
	DT	**0.278**	**0.250**	**0.278**	**0.500**	**0.278**	**0.456**
	LR	**0.278**	**0.250**	**0.278**	**0.500**	**0.278**	0.418
	KNN	0.111	0.194	0.111	0.389	0.111	0.242
Madrid	M	0.283	0.283	0.283	0.565	0.283	0.461
	SVM	0.413	0.304	0.413	0.609	0.413	0.536
	DT	0.326	0.293	0.326	0.587	0.326	0.491
	LR	**0.435**	**0.315**	**0.435**	**0.630**	**0.435**	**0.558**
	KNN	0.370	0.250	0.370	0.500	0.264	0.392

Tabla A.4: Resultados de nuestro modelo y de los baselines con el conjunto de tipos variados pero con un numero de POIs similar. En negrita se muestran los mejores resultados en cada caso.


UAM
Universidad Autónoma
de Madrid