COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA MATEMÁTICA E INTELIGENCIA ARTIFICIAL

TRABAJO FIN DE GRADO

EVALUACIÓN DE SISTEMAS DE RECOMENDACIÓN EMPLEANDO DIFERENTES ESTRATEGIAS DE EVALUACIÓN

Autor: María Carreño Nin De Cardona

Director: Pablo Sánchez Pérez

Madrid

Mayo de 2026

Declaración de originalidad

Declaro bajo mi responsabilidad que el Proyecto presentado con el título **Evaluación de sistemas de recomendación empleando diferentes estrategias de evaluación** e la ETS de Ingeniería – ICAI de la Universidad Pontificia Comillas en el curso académico 2025-2026 es de mi autoría y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Uso de Inteligencia Artificial[1]

Declaro bajo mi responsabilidad que (indicar la opción correcta):

☐ No he utilizado Inteligencia Artificial en la elaboración del presente documento.

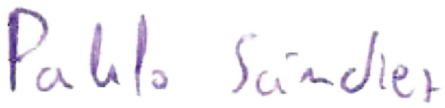
X He utilizado Inteligencia Artificial en la elaboración del presente documento y/o del Anexo B siempre en las condiciones permitidas por la Universidad Pontificia Comillas, es decir, aplicando el Nivel 2 de la Escala de Evaluación de Perkins et al. (2024): "*La IA puede utilizarse para actividades previas a la tarea, como la lluvia de ideas, la descripción y la investigación inicial. Este nivel se centra en el uso de la IA para la planificación, las síntesis y la generación de ideas, pero las evaluaciones deben hacer hincapié en la capacidad de desarrollar y refinar estas ideas de forma independiente*". En concreto, las Inteligencia Artificial ha sido empleada para:

(indicar aquí el uso concreto que se ha hecho de la Inteligencia Artificial) - Se utilizaron herramientas de inteligencia artificial como apoyo para la comprensión de referencias bibliográficas y la búsqueda de información. - Se utilizaron herramientas de inteligencia artificial para facilitar la comprensión profunda de conceptos técnicos relacionados con los algoritmos empleados en el trabajo.


Firmado (alumno): María Carreño Nin de Cardona
Fecha: 21/05/2026

Autorización para la entrega del Proyecto

[1] Esta declaración se refiere al uso de la Inteligencia Artificial generativa para realizar los documentos del Proyecto (Anexo B y Memoria). No aplica a Proyectos donde, por su naturaleza, deban emplear inteligencia artificial como parte de los mismos (aplicación de técnicas de aprendizaje automático, redes neuronales, análisis de datos…)

El Director del Proyecto	El co-Director del Proyecto (si aplica)
	
Fdo: Pablo Sánchez Pérez	Fdo:
Fecha: 21/05/2026	Fecha:

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

Evaluación de Sistemas de Recomendación Empleando Diferentes Estrategias de Evaluación

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

curso académico 2025/26 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

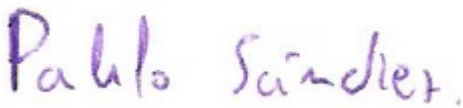
El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: María Carreño Nin de Cardona Fecha: 08/05/2026

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO



Fdo.: Pablo Sánchez Pérez Fecha: 08/05/2026

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA MATEMÁTICA E INTELIGENCIA ARTIFICIAL

TRABAJO FIN DE GRADO

EVALUACIÓN DE SISTEMAS DE RECOMENDACIÓN EMPLEANDO DIFERENTES ESTRATEGIAS DE EVALUACIÓN

Autor: María Carreño Nin de Cardona

Director: Pablo Sánchez Pérez

Madrid

EVALUACIÓN DE SISTEMAS DE RECOMENDACIÓN EMPLEANDO DIFERENTES ESTRATEGIAS DE EVALUACIÓN

Autor: Carreño Nin de Cardona, María.
Director: Sánchez Pérez, Pablo.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Este trabajo analiza cómo las estrategias de evaluación influyen en el rendimiento observado de distintos sistemas de recomendación. Se comparan varios algoritmos: Aleatorio, Popularidad, Filtrado Colaborativo (KNN), Bayesian Personalized Ranking (BPR), Redes Neuronales y un modelo híbrido, aplicados sobre múltiples datasets heterogéneos. Para ello, se emplean diferentes particiones de los datos, aleatorias y temporales, tanto globales como por usuario y se evalúan mediante métricas como: Precisión, Recall, Diversidad, Cobertura y Novedad. Los resultados permiten estudiar la robustez de los modelos y muestran que la elección de la estrategia de evaluación condiciona significativamente las conclusiones sobre su rendimiento.

Palabras clave: Sistemas de recomendación, evaluación, machine learning.

1. Introducción

El presente trabajo se enmarca en el ámbito de los sistemas de recomendación, una de las áreas clave dentro de la inteligencia artificial y el análisis de datos. Estos sistemas se utilizan ampliamente en plataformas digitales para filtrar grandes volúmenes de información y personalizar la experiencia del usuario, influyendo directamente en el contenido que consume y en sus decisiones.

En el ámbito de la investigación, uno de los problemas actuales es la dificultad para comparar resultados entre trabajos de forma objetiva. El rendimiento de los algoritmos puede variar significativamente según decisiones metodológicas como la partición de los datos o el tratamiento temporal de las interacciones, lo que genera problemas de reproducibilidad y comparabilidad. En muchos casos, las mejoras reportadas pueden deberse al protocolo experimental más que a la superioridad real del modelo propuesto.

En este contexto, los sistemas de recomendación han sido ampliamente estudiados en la literatura, destacando como referencia fundamental el *Recommender Systems Handbook* [1]. Asimismo, los trabajos clásicos de Adomavicius y Tuzhilin establecen las bases teóricas y los principales desafíos de esos sistemas [2]. Por otro lado, la evaluación de sistemas de recomendación ha sido objeto de estudio específico, especialmente en lo relativo a métricas offline y su interpretación [3], junto con enfoques más recientes que amplían en análisis hacia escenarios más complejos y contextuales [4].

2. Objetivos

El objetivo principal de este trabajo es analizar cómo varía el rendimiento de distintos sistemas de recomendación en función de la estrategia de evaluación empleada, prestando especial atención a la robustez de los resultados ante cambios en la partición de los datos.

Como hipótesis principal, se plantea que la elección de la estrategia de partición, especialmente entre enfoques aleatorios y temporales, tiene un impacto significativo en los

resultados obtenidos, pudiendo alterar las conclusiones sobre qué algoritmo es más adecuado.

Para alcanzar este objetivo, se plantean los siguientes objetivos específicos:

- Implementar varios algoritmos de recomendación representativos
- Diseñar diferentes estrategias de partición de los datos
- Evaluar los modelos mediante diversas métricas de rendimiento y comparar su comportamiento en distintos conjuntos de datos de naturaleza heterogénea.

3. Descripción del sistema

El problema se ha abordado mediante el diseño de un sistema experimental modular que permite evaluar distintos algoritmos bajo múltiples configuraciones. Se han utilizado cuatro datasets públicos: MovieLens, Foursquare y Amazon Video Games, preprocesados en términos de usuario, ítem, valoración y marca temporal.

Sobre ellos se aplican cuatro estrategias de partición de los datos, diferenciadas según si incorporan información temporal o no y si se aplican por usuario o de forma global, generando así conjuntos de entrenamiento y prueba, simulando distintos escenarios de recomendación. Los algoritmos evaluados incluyen baselines simples (Random, Popularity), métodos colaborativos (K-Nearest Neighbor basado en usuarios e ítems), un modelo de ranking (Bayesian Personalized Ranking), una red neuronal y un modelo híbrido basado en combinación ponderada. Las recomendaciones se evalúan con métricas de relevancia, novedad y diversidad, permitiendo analizar no solo el rendimiento de los algoritmos, sino cómo este varía en función de la estrategia de evaluación empleada, tal y como se muestra en la Figura 1.

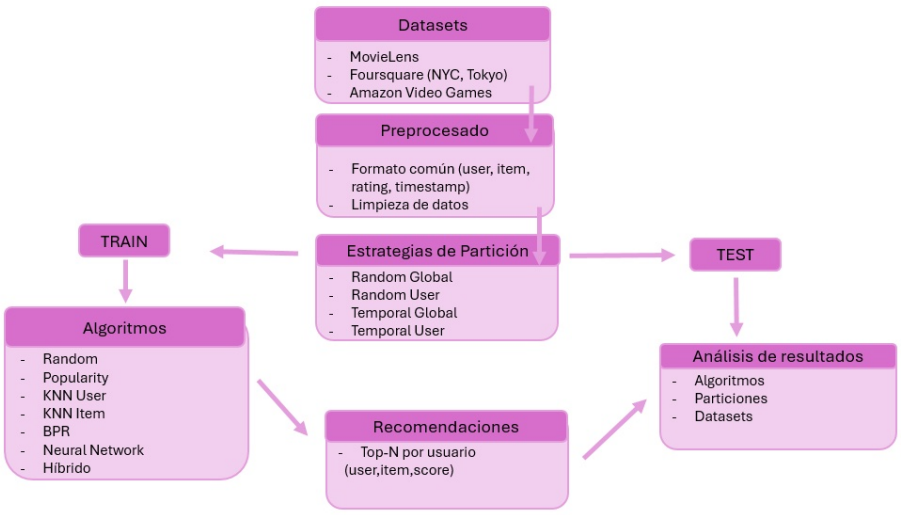


Figura 1. Esquema del sistema propuesto

El sistema se organiza como un pipeline modular en el que cada bloque se encadena con el siguiente de forma secuencial. Los datos se preprocesan, se particionan, se entrenan los modelos únicamente con el conjunto de entrenamiento y finalmente el módulo de evaluación compara las recomendaciones generadas con las interacciones reales del test. De este modo, la salida de cada módulo constituye la entrada del siguiente, lo que permite desacoplar las distintas fases del experimento y facilitar la comparación entre modelos y estrategias de evaluación.

4. Resultados

Los resultados muestran que el rendimiento depende fuertemente del dataset y la estrategia de partición, sin que exista un modelo universalmente superior. En datasets densos como MovieLens, los modelos colaborativos (especialmente KNN) ofrecen los mejores resultados bajo particiones aleatorias, mientras que en escenarios temporales el modelo de popularidad adquiere mayor relevancia. En datasets dispersos como FourSquare, los modelos basados en popularidad dominan de forma consistente, debido a la escasez de interacciones por usuario. Los modelos más complejos (BPR y NeuralCF) no presentan mejoras sistemáticas frente a enfoques más simples, mientras que los modelos híbridos muestran mayor estabilidad entre particiones. En todas las particiones se observa el mismo compromiso: mayor precisión implica mejor novedad y diversidad.

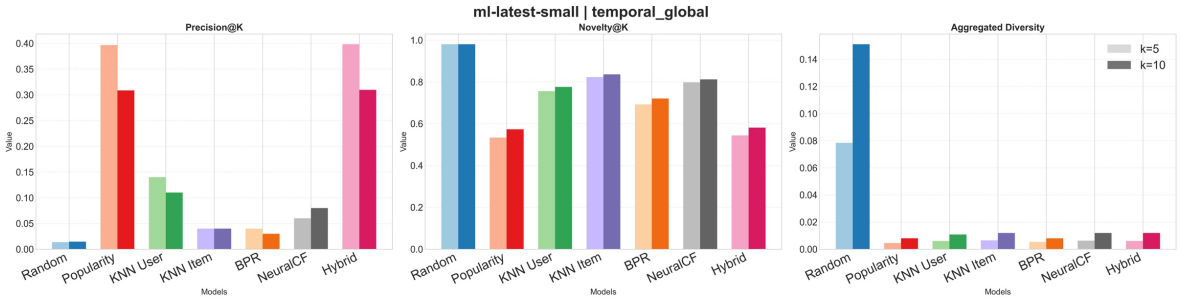


Figura 2. Resultado de MovieLens con partición Temporal Global

En conjunto, los resultados evidencian que el rendimiento está condicionado por la estructura del dataset y por el protocolo de evaluación, más que por la complejidad del modelo.

5. Conclusiones

No existe un modelo óptimo universal: el rendimiento depende del dataset, su dispersión y la estrategia de partición, más que de la complejidad del algoritmo. Las particiones temporales introducen escenarios más realistas y exigentes, mientras que las aleatorias tienden a inflar los resultados. Los modelos híbridos aportan robustez, pero requieren ajuste específico por contexto. La principal aportación es evidenciar que el protocolo experimental tiene tanto impacto en los resultados como el propio diseño del modelo, lo que pone de manifiesto la necesidad de estandarizar las prácticas de evaluación en sistemas de recomendación.

6. Referencias

Por limitaciones de espacio se recogen únicamente las referencias más relevantes. La bibliografía completa se encuentra en la memoria del proyecto.

[1] F.Ricci, L.Rokach y B.Shapira, *Recommender Systems Handbook*, 3rd ed., Springer, 2022.

[3] J. L. Herlocker et al., "Evaluating Collaborative Filtering Recommender Systems," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5–53, 2004.

EVALUATION OF RECOMMENDATION SYSTEMS USING DIFFERENT EVALUATION STRATEGIES

Author: Carreño Nin de Cardona, María.
Supervisor: Sánchez Pérez, Pablo.
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

This work analyzes how evaluation strategies influence the observed performance of different recommender systems. Several algorithms are compared: Random, Popularity, Collaborative Filtering (KNN), Bayesian Personalized Ranking (BPR), Neural Networks, and a hybrid model, applied across multiple heterogeneous datasets. To this end, different data splitting strategies are used, both random and temporal, at global and user levels, and the models are evaluated using metrics such as Precision, Recall, Diversity, Coverage and Novelty. The results allow the robustness of the models to be studied and show that the choice of evaluation strategy significantly affects the conclusions regarding their performance.

Keywords: Recommender systems, evaluation, machine learning.

1. Introduction

This work is framed within the field of recommender systems, one of the key areas in artificial intelligence and data analysis. These systems are widely used in digital platforms to filter large volumes of information and personalize user experience, directly influencing the content users consume and their decision-making processes.

In research, one of the current challenges is the difficulty of objectively comparing results across studies. The performance of algorithms can vary significantly depending on methodological choices such as data splitting or the temporal treatment of interactions, which leads to issues of reproducibility and comparability. In many cases, reported improvements may be driven by the experimental protocol rather than the actual superiority of the proposed model.

In this context, recommender systems have been widely studied in the literature, with the *Recommender Systems Handbook* [1] serving as a fundamental reference. Additionally, the classic work by Adomavicius and Tuzhilin establishes the theoretical foundations and main challenges of these systems [2]. Furthermore, the evaluation of recommender systems has been specifically studied, particularly regarding offline metrics and their interpretation [3], along with more recent approaches that extend the analysis to more complex and contextual scenarios [4].

2. Objectives

The main objective of this work is to analyze how the performance of different recommender systems varies depending on the evaluation strategy used, with special attention to the robustness of the result under different data splitting approaches.

The main hypothesis is that the choice of data splitting strategy, especially between random and temporal approaches, has a significant impact on the results, potentially altering conclusions about which algorithm is most suitable.

To achieve this goal, the following specific objectives are defined:

- To implement several representative recommender system algorithms
- To design different data splitting strategies
- To evaluate the models using multiple performance metrics and compare their behavior across heterogeneous datasets

3. System Description

The problem is addressed through the design of a modular experimental system that allows the evaluation of different algorithms under multiple configurations. Four public datsets are used: MovieLens, Foursquare and Amazon Video Games, preprocessed in terms of user, item, rating, and timestamp.

On these datasets, four data splitting strategies are applied, differentiated by whether they incorporate temporal information and whether they are applied at user or global level. This generated training and test sets simulating different recommendation scenarios. The evaluated algorithms include simple baselines (Random, Popularity), collaborative methods (user-based and item-based K-Nearest Neighbors), a ranking model (Bayesian Personalized Ranking), a neural network model, and a hybrid model based on weighted combination. Recommendations are evaluated using relevance, novelty, and diversity metrics, allowing the analysis not only of model performance but also of how it varies depending on the evaluation strategy, as shown in Figura 3.

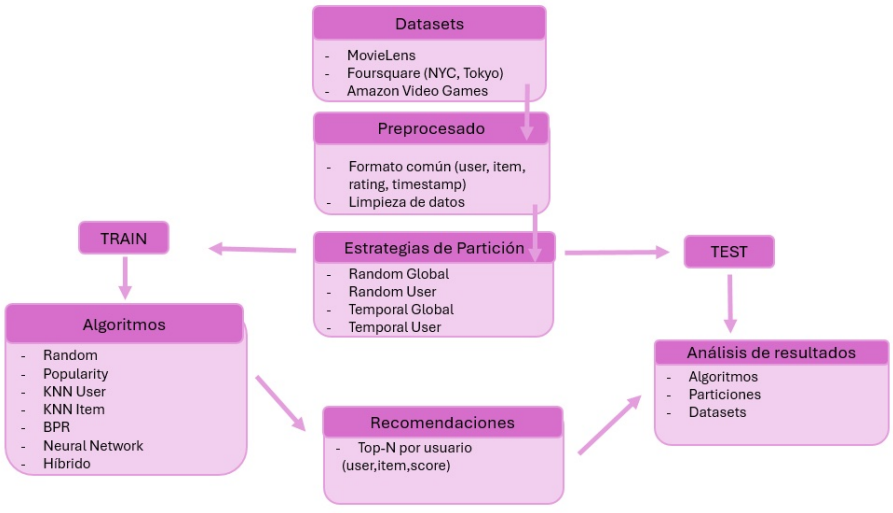


Figura 3 Diagram of the proposed system

The system is organized as a modular pipeline in which each block is sequentially connected. Data is preprocessed, split, models are trained only on the training set, and the evaluation module compares generated recommendations with the actual interactions in the test set. In this wat, the output of each module becomes the input of the next, allowing decoupling of experimental stages and facilitating comparison between models and evaluation strategies.

4. Results

The results show that performance strongly depends on the dataset and the splitting strategy, with no universally best model. In dense datasets such as MovieLens, collaborative models (especially KNN) achieve the best results under random splits, while in temporal splits the popularity model becomes more relevant. In sparse datasets such as Foursquare, popularity-based models consistently dominate due to the limited number of user interactions. More complex models (BPR and NeuralCF) do not systematically outperform simpler approaches, while hybrid models show greater stability across different splits. Across all configurations, the same trade-off is observed: higher precision tends to come at the expense of novelty and diversity.

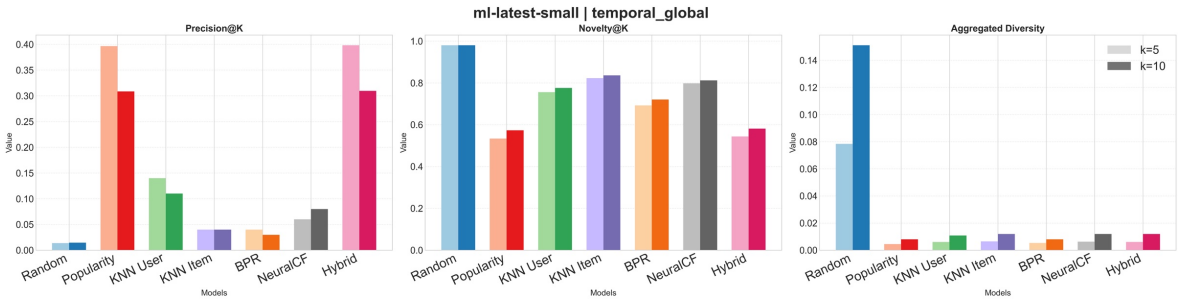


Figura 4. Results on MovieLens using Temporal Global split

Overall, the results demonstrate that performance is more strongly influenced by datasets structure and evaluation protocol than by model complexity.

5. Conclusions

There is no universally optimal model: performance depends on the dataset, its sparsity, and the evaluation strategy rather than on algorithm complexity. Temporal splits introduce more realistic and challenging scenarios, whereas random splits tend to inflate performance results. Hybrid models provide robustness but require context-specific tuning. The main contribution of this work is demonstrating that the experimental protocol has as much impact on the results as the model design itself, highlighting the need to standardize evaluation practices in recommender systems.

6. References

Due to space limitations, only the most relevant references are included. The full bibliography is available in the project report.

[1] F.Ricci, L.Rokach y B.Shapira, *Recommender Systems Handbook*, 3rd ed., Springer, 2022.

[3] J. L. Herlocker et al., "Evaluating Collaborative Filtering Recommender Systems," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5–53, 2004.

Índice de la memoria

Índice de figuras

Índice de tablas

Capítulo 1. INTRODUCCIÓN

Los sistemas de recomendación constituyen una de las aplicaciones más extendidas de la inteligencia artificial en la actualidad. Su presencia es habitual en plataformas digitales de comercio electrónico, redes sociales o en servicios de entretenimiento, donde permiten personalizar la experiencia de usuario y facilitar la toma de decisiones en entornos con un gran volumen de información. Este tipo de sistemas se emplea, por ejemplo, en plataformas como Amazon, que recomienda productos en función del historial de navegación y compra; LinkedIn, que siguiere contactos o empleos relevantes; Netflix, que personaliza el catálogo de contenidos; o TikTok, que adapta el contenido mostrado en función del comportamiento del usuario. Más allá de estos ejemplos, los sistemas de recomendación también se utilizan en ámbitos como la salud (apoyo en la toma de decisiones clínicas), la educación (sugerencia de recursos de aprendizaje) o la publicidad digital (segmentación y personalización de anuncios). Su impacto, por tanto, trasciende el entretenimiento y el consumo, influyendo en múltiples aspectos de la vida cotidiana.

No obstante, su uso plantea también desafíos importantes. Entre ellos destacan las cuestiones relacionadas con la privacidad de los datos, el sesgo algorítmico o la posible creación de burbujas de información que limitan la diversidad de contenidos a los que accede el usuario. Asimismo, en el ámbito de la investigación, uno de los problemas actuales es la dificultad para garantizar la reproducibilidad de los resultados. En particular, en los sistemas de recomendación, el rendimiento de los algoritmos puede variar significativamente en función de decisiones metodológicas como la partición de los datos, el tratamiento temporal de las interacciones o el conjunto de datos empleado.

En este contexto, el presente Trabajo de Fin de Grado analiza cómo distintas estrategias de evaluación afectan al rendimiento observado de diversos algoritmos de recomendación. Para ello, se comparan múltiples modelos sobre diferentes conjuntos de datos y bajo distintas metodologías experimentales, con el objetivo de estudiar la robustez de los resultados y la fiabilidad de las conclusiones obtenidas.

1.1 CONTEXTO Y MOTIVACIÓN

En el ámbito de los sistemas de recomendación, la evaluación de algoritmos constituye un aspecto fundamental para determinar su rendimiento y comparar distintas propuestas. Habitualmente, los trabajos académicos emplean métricas estándar como la precisión, con el objetivo de establecer qué métodos ofrecen mejores resultados.

Sin embargo, a pesar de la aparente estandarización de estas prácticas, existe una gran variabilidad en la forma en la que se diseñan los experimentos. Aspectos como la partición de los datos (aleatoria frente a temporal), la selección del conjunto de prueba, el filtrado de usuarios o ítems, o incluso pequeños detalles en la implementación, pueden influir significativamente en los resultados obtenidos.

Como consecuencia, diferentes estudios pueden llegar a conclusiones distintas sobre los mismos algoritmos, lo que dificulta la comparación justa entre métodos y plantea problemas de reproducibilidad. En muchos casos, resultados reportados como mejoras significativas pueden deberse, al menos en parte, a decisiones experimentales específicas más que a una superioridad real del modelo propuesto.

Este problema pone de manifiesto la necesidad de analizar de forma sistemática cómo influyen las estrategias de evaluación en los resultados obtenidos. En este sentido, la motivación principal de este trabajo es contribuir a una mejor comprensión del impacto del diseño experimental en la evaluación de sistemas de recomendación, con el objetivo de favorecer prácticas más robustas y comparaciones más fiables en la investigación futura.

1.2 OBJETIVOS

El objetivo principal de este trabajo es analizar cómo varía el rendimiento de distintos sistemas de recomendación en función de la estrategia de evaluación empleada, prestando especial atención a su robustez frente a cambios en la partición de los datos.

Como hipótesis principal, se plantea que la elección de la estrategia de partición, especialmente entre enfoques aleatorios y temporales, tiene un impacto significativo en los resultados obtenidos, pudiendo alterar las conclusiones sobre qué algoritmo presenta mejor comportamiento.

Para alcanzar este objetivo principal, se plantean los siguientes objetivos específicos:

- Implementar varios algoritmos representativos de sistemas de recomendación.
- Diseñar distintas estrategias de partición de datos, tanto aleatorias como temporales.
- Evaluar los modelos mediante métricas relacionadas con la relevancia, novedad y diversidad.
- Comparar el comportamiento de los algoritmos en múltiples datasets heterogéneos.
- Analizar la estabilidad de los resultados obtenidos bajo distintos escenarios experimentales.

1.3 ALINEACIÓN CON LOS ODS

Este Proyecto se alinea principalmente con varios Objetivos de Desarrollo Sostenible (ODS) definidos por las Naciones Unidas:

- ODS 4: Educación de Calidad.

El trabajo contribuye a la generación de conocimiento en el ámbito de la inteligencia artificial aplicada, promoviendo metodologías rigurosas, reproducibles y fundamentadas científicamente. Además, fomenta el aprendizaje práctico en áreas como ciencia de datos, evaluación experimental y sistemas inteligentes.

- ODS 9: Industria, innovación e infraestructura.

Los sistemas de recomendación forman parte de la infraestructura digital de numerosas empresas y plataformas tecnológicas. Mejorar su evaluación y fiabilidad contribuye al desarrollo de soluciones más eficientes, innovadoras y sostenibles en sectores estratégicos como el comercio electrónico, el entretenimiento o la movilidad.

- ODS 12: Producción y consume responsables.

De forma indirecta, los sistemas de recomendación bien diseñados ayudan a conectar a los usuarios con productos y servicios más adecuados a sus necesidades, reduciendo ruido informativo, mejorando la eficiencia en la toma de decisiones y favoreciendo patrones de consume más personalizados y racionales.

1.4 ESTRUCTURA DEL TRABAJO

La memoria se organiza en varias secciones que recogen de forma ordenada el desarrollo completo del proyecto.

En la sección 1 se presenta la introducción general del trabajo, incluyendo el contexto, la motivación, los objetivos perseguidos y su alineación con los Objetivos de Desarrollo Sostenible.

La sección 2 expone el estado del arte y los fundamentos teóricos necesarios para comprender los sistemas de recomendación, los principales algoritmos estudiados y las metodologías de evaluación.

La sección 3 describe el planteamiento del problema, la metodología seguida y el diseño de la solución desarrollada, incluyendo los conjuntos de datos empleados, las estrategias de partición implementadas, los algoritmos considerados y las métricas utilizadas.

La sección 4 recoge los resultados experimentales obtenidos y su análisis comparativo, permitiendo estudiar el impacto de las distintas estrategias de evaluación sobre el rendimiento de los modelos.

Finalmente, la sección 5 presenta las conclusiones principales del trabajo, las limitaciones encontradas y posibles líneas futuras de investigación y mejora.

Capítulo 2. ESTADO DEL ARTE

2.1 SISTEMAS DE RECOMENDACIÓN

Los sistemas de recomendación son herramientas diseñadas para sugerir ítems relevantes a los usuarios para facilitar la toma de decisiones y personalizar la experiencia del usuario en plataformas digitales. Se utilizan ampliamente en sectores como el comercio electrónico, las redes sociales y servicios de contenido, para adaptar gran parte de su contenido a los intereses individuales de cada usuario.

El problema de recomendación puede abordarse desde distintas perspectivas y mediante múltiples algoritmos. No obstante, todos ellos comparten una misma base: estimar que elementos resultarán más adecuados para cada usuario a partir de la información disponible.

2.1.1 DEFINICIÓN GENERAL

Desde un punto de vista formal, el problema de recomendación se plantea como la búsqueda de aquellos ítems que maximizan la relevancia estimada para cada usuario. Sea U el conjunto de usuarios e I el conjunto de ítems disponibles, el problema de recomendación consiste en aprender una función:

$$f: U \times I \rightarrow \mathbb{R}$$

Donde $f(u, i)$ representa la relevancia del ítem $i \in I$ para el usuario $u \in U$. A partir de esta función, el sistema busca recomendar aquellos ítems que maximicen dicha relevancia:

$$arg \max_{i \in I} f(u, i)$$

En la práctica, el sistema evalúa la función de relevancia sobre el conjunto de ítems candidatos, normalmente compuesto por aquellos ítems con los que el usuario no ha interactuado previamente, y los ordena en función de su puntuación, generando así un

ranking personalizado para cada usuario. A partir de este ranking, se seleccionan los N ítems más relevantes:

$$\text{Top-N}(u)=Top_N(\{f(u,i)|i \in C_u\})$$

Donde C_u representa el conjunto de ítems candidatos no consumidos previamente por el usuario u. La información disponible en estos sistemas suele representarse mediante una matriz de interacciones:

$$R \in \mathbb{R}^{|U|\times|I|}$$

Donde cada entrada $r_{u,i}$ refleja la interacción existente entre el usuario u y el ítem i, que puede representar una valoración, una compra, una visualización o cualquier otra señal de comportamiento. En la mayoría de los escenarios reales, esta matriz es altamente dispersa, ya que cada usuario interactúa únicamente con una pequeña fracción del catálogo disponible.

Estas interacciones suelen clasificarse en dos grandes tipos: el feedback explícito y el feedback implícito. El feedback explícito corresponde a la información proporcionada de forma directa por el usuario, como valoraciones numéricas, puntuaciones, "me gusta" o reseñas. Por su parte, el feedback implícito se obtiene a partir del comportamiento observado del usuario, sin que exista una valoración directa, incluyendo señales como el historial de compras, las visualizaciones de contenido, los clics o el tiempo de permanencia. Aunque el feedback implícito suele ser más abundante en muchos escenarios, también presenta una mayor incertidumbre, ya que una interacción no siempre implica preferencia positiva.

2.2 CRITERIOS DE CALIDAD EN SISTEMAS DE RECOMENDACIÓN

Aunque tradicionalmente los sistemas de recomendación se han centrado en maximizar la relevancia de las recomendaciones, en la práctica esta se ha evaluado habitualmente mediante métricas como la precisión (precision) o la exahustividad (recall) [3]. Sin embargo, en la actualidad se reconoce que optimizar únicamente este tipo de métricas no es suficiente

para garantizar una buena experiencia de usuario ni cumplir con los objetivos de la plataforma [9].

Un sistema que únicamente optimiza la relevancia tiende a recomendar ítems muy similares entre sí o excesivamente populares, lo que puede limitar la capacidad de descubrimiento y reducir el interés del usuario a largo plazo. Por ello, los sistemas modernos incorporan múltiples criterios de calidad que permiten evaluar distintos aspectos del comportamiento del recomendador.

Relevancia: mide hasta qué punto los ítems recomendados coinciden con los intereses reales del usuario. Es el objetivo fundamental de los sistemas de recomendación, ya que determina la utilidad inmediata de las sugerencias. Sin embargo, optimizar únicamente esta métrica puede llevar a recomendar siempre los mismos tipos de ítems o aquellos más populares.

Novedad: hace referencia a la capacidad del sistema para recomendar ítems que el usuario no conoce previamente. Este criterio es importante para favorecer el descubrimiento de contenido, evitando recomendaciones triviales o ya conocidas. Sin novedad, el sistema puede perder utilidad al no aportar información adicional al usuario.

Diversidad: evalúa el grado de variedad presente en la lista de recomendaciones. Una lista diversa reduce la redundancia entre ítems similares y mejora la experiencia de usuario al ofrecer alternativas más amplias. La diversidad resulta especialmente relevante para evitar la sobreespecialización del sistema, que podría limitarse a un único tipo de contenido.

Cobertura: mide la capacidad del sistema para generar recomendaciones sobre una parte amplia del catálogo disponible y para un elevado número de usuarios. Este criterio es importante porque un sistema con baja cobertura puede dejar sin recomendaciones a ciertos usuarios o infrautilizar gran parte de los ítems, reduciendo su efectividad global.

En la práctica, estos objetivos pueden entrar en conflicto entre sí, por lo que el diseño de un sistema de recomendación suele requerir un equilibrio entre varios criterios simultáneamente.

2.3 EVALUACIÓN GENERAL

La evaluación de los sistemas de recomendación tiene como objetivo medir el grado en que un modelo cumple con los criterios de calidad descritos anteriormente, como la relevancia, la novedad, la diversidad o la cobertura. Aunque tradicionalmente los sistemas de recomendación se han centrado en maximizar la relevancia mediante métricas como precisión o exhaustividad (recall) [3], [5], actualmente se considera insuficiente optimizar únicamente este aspecto para garantizar una buena experiencia de usuario [9].

Esta evaluación puede realizarse mediante distintos enfoques:

- Evaluación offline: se basa en datos históricos previamente registrados. El conjunto de interacciones se divide habitualmente en entrenamiento, validación y prueba, de modo que el modelo aprende con una parte de los datos y se evalúa sobre interacciones no observadas durante el entrenamiento.
- Evaluación online: se realiza en entornos reales con usuarios finales, mediante experimentos como pruebas A/B, donde se comparan distintas versiones del sistema según métricas de comportamiento o de negocio.
- Estudios de usuario: consisten en la evaluación directa mediante usuarios reales, a través de encuentras, entrevistas o experimentos controlados, con el objetivo de analizar aspectos subjetivos como la satisfacción, la utilidad percibida o la confianza en las recomendaciones.

En la evaluación offline, la forma en la que se dividen los datos entre los distintos conjuntos es un aspecto clave, ya que influye directamente en los resultados obtenidos [7], [10].

En la literatura, las estrategias de partición suelen clasificarse según dos criterios principales: el uso de información temporal y el nivel al que se aplica la división (global o por usuario) [1]. A partir de estos criterios, se distinguen cuatro enfoques principales: partición aleatoria global, aleatoria por usuarios, temporal global y temporal por usuario. Como se muestra en la Figura 5, estas estrategias pueden interpretarse como distintas formas de seleccionar subconjuntos de observaciones dentro de una matriz donde cada fila representa un usuario y

cada columna un instante temporal. En este contexto, cada celda corresponde a una interacción observada en un momento dado. Las celdas en color azul corresponden a las interacciones de entrenamiento, mientras que las celdas rojo representan las interacciones de prueba. Las celdas en blanco indican la ausencia de interacción.

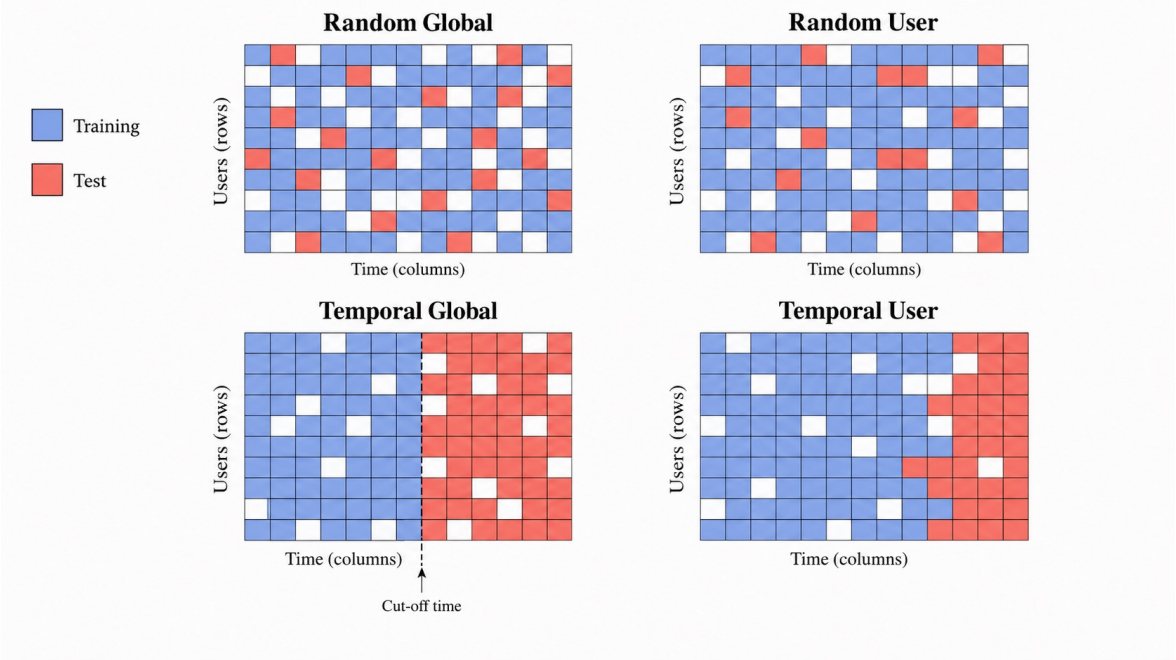


Figura 5. Estrategia de división de datos en entrenamiento y validación

- Partición aleatoria global (Random global): Todas las interacciones del conjunto de datos se distribuyen de forma aleatoria entre los subconjuntos de entrenamiento, validación y prueba, sin distinguir entre usuarios. Este enfoque es sencillo y ampliamente utilizado, aunque puede generar escenarios poco realistas al ignorar tanto el comportamiento individual de los usuarios como el orden temporal de las interacciones. Además, puede introducir el problema de *data leakage*, al mezclar información futura en el conjunto de entrenamiento [10].
- Partición aleatoria por usuario (Random user): Las interacciones se dividen de forma independiente para cada usuario, de modo que cada uno conserva aproximadamente la misma proporción de datos en los distintos subconjuntos. Esta estrategia garantiza la representación de todos los usuarios, aunque sigue sin tener en cuenta la evolución

temporal de las interacciones. Al igual que en el caso global, también puede producir *data leakage* [10].

- Partición temporal global (Temporal global): Las interacciones se orden cronológicamente a nivel global, utilizando las más antiguas para entrenamiento y las más recientes para evaluación. Este enfoque permite simular escenarios más realistas, en los que el sistema debe predecir interacciones futuras a partir de datos pasados [11]. Sin embargo, introduce una mayor dificultad, ya que pueden aparecer usuarios o ítems con muy pocas interacciones en el conjunto de entrenamiento (problema de cold-start), lo que complica la generación de recomendaciones fiables.
- Partición temporal por usuario (Temporal user): Las interacciones de cada usuario se ordenan cronológicamente de forma individual y se dividen respetando su evolución temporal. Esta estrategia combina las ventajas del enfoque temporal y del enfoque por usuario, y es especialmente adecuada para modelar cambios en las preferencias a lo largo del tiempo [11]. Esta partición también puede producir *data leakage,* ya que los usuarios pueden tener historiales temporales no alineados.

En cuanto a la evaluación cuantitativa, se emplean diversas métricas que permiten medir distintos objetivos del sistema, en línea con los criterios de calidad previamente descritos. Estas métricas suelen evaluarse sobre los K primeros ítems recomendados para cada usuario. Entre las más utilizadas destacan:

- Precisión (Precision@K): mide la proporción de ítem relevantes entre los K recomendados, evaluando la exactitud del sistema.
- Exhaustividad (Recall@K): mide la proporción de ítems relevantes recuperados respecto al total de ítems relevantes disponibles.
- Normalized Discounted Cumulative Gain (NDCG@K): mide la calidad del ranking de las recomendaciones, dando mayor peso a los ítems relevantes que aparecen en posiciones más altas de la lista ordenada.
- Diversidad agregada: evalúa el grado de variedad en las recomendaciones generadas.
- Cobertura de usuarios: mide la proporción de usuarios para los cuales el sistema es capaz de generar recomendaciones.

- Novedad (Novelty@K): cuantifica el grado en que los ítems recomendados son desconocidos o poco frecuentes para el usuario.

En este trabajo se emplea evaluación offline, utilizando diferentes estrategias de partición de los datos y métricas específicas.

2.4 ALGORITMOS DE RECOMENDACIÓN

Existen numerosos algoritmos de recomendación, con enfoques y niveles de complejidad muy diversos. De forma general, estos pueden agruparse en distintas familias según la información que utilizan y la forma en que generan recomendaciones. Entre las más relevantes se encuentran los sistemas basados en contenido, los sistemas de filtrado colaborativo, los sistemas híbridos y otros enfoques como los basados en conocimiento o en información demográfica [1].

Cada una de estas aproximaciones representa características, ventajas y limitaciones específicas, lo que condiciona su aplicación en distintos escenarios. En este trabajo, se prestará especial atención a los métodos basados en contenido, filtrado colaborativo e híbridos, por ser los más representativos y utilizados en la práctica.

2.4.1 BASADOS EN CONTENIDO

Los sistemas basados en contenido recomiendan ítems similares a los consumidos previamente por el usuario, utilizando características descriptivas como categorías, géneros o palabras clave. A partir del historial de interacciones, se construye un perfil de preferencias y se recomiendan ítems similares.

Entre sus principales ventajas destacan la capacidad de recomendar ítems nuevos sin depender de otros usuarios, y un alto grado de personalización. Sin embargo, suelen presentar menor diversidad y una tendencia a recomendar contenidos muy similares entre sí.

- Term Frecuency – Inverse Document Frecuency (TF-IDF)

Una de las representaciones más utilizadas en este tipo de sistemas es TF-IDF, que transforma cada ítem $i \in I$ en un vector numérico v_i basado en la importancia de sus términos descriptivos [1], [2].

El peso TF-IDF de un término $t \in V$ en un ítem i se define como:

$$\text{tfidf}(t,i) = \text{tf}(t,i) \cdot log\left(\frac{N}{\text{df}(t)}\right)$$

Donde $\text{tf}(t,i)$ representa la frecuencia del término en el ítem, $\text{df}(t)$ el número de ítems que contienen dicho término, N el número total de ítems y V el vocabulario del sistema.

A partir de esta representación vectorial, la similitud entre un usuario $u \in U$ y un ítem i se calcula habitualmente mediante la similitud del coseno:

$$\text{sim}(u,i) = \frac{p_u \cdot v_i}{\|p_u\|\|v_i\|}$$

Donde p_u representa el perfil del usuario construido a partir de los ítems previamente consumidos.

Esta representación convierte el problema de recomendación en un espacio vectorial donde la similitud semántica entre usuarios e ítems puede modelarse mediante técnicas de álgebra lineal.

Además de TF-IDF, existen otras aproximaciones basadas en modelos probabilísticos, como Naive Bayes [13], o en representaciones distribuidas mediante embeddings, como Word2Vec [14], capaces de capturar relaciones semánticas entre términos.

2.4.2 FILTRADO COLABORATIVO

Los sistemas basados en filtrado colaborativo generan recomendaciones a partir de patrones observados en las interacciones entre usuarios e ítems, sin necesidad de utilizar información descriptiva de los mismos. La idea fundamental es que usuarios con comportamientos

similares tienden a compartir preferencias, o que ítems consumidos conjuntamente presentan cierta relación.

A partir de esta familia de métodos, existen diferentes aproximaciones para modelar las interacciones entre usuarios e ítems. Entre ellas, una de las más tradicionales es el enfoque basado en vecinos más cercanos (K-Nearest Neighbors, KNN), que permite estimar la relevancia de un ítem a partir de la información proporcionada por usuarios o ítems similares.

- KNN basado en usuarios

El enfoque basado en usuarios estima la preferencia de un usuario a partir de las valoraciones realizadas por otros usuarios similares.

La similitud entre dos usuarios u y v se define mediante una función de similitud, como la similitud del coseno o la correlación de Pearson, siendo ambas métricas ampliamente utilizadas en la literatura de filtrado colaborativo [5].

La similitud del coseno se define como:

$$\text{sim}(u,v)=\frac{\sum_{i\in I} r_{u,i}\cdot r_{v,i}}{\sqrt{\sum_{i\in I} r_{u,i}^2}\cdot\sqrt{\sum_{i\in I} r_{v,i}^2}}$$

Alternativamente, la correlación de Pearson puede utilizarse como medida de similitud al tener en cuenta las desviaciones respecto a la media de cada usuario, siendo especialmente útil en escenarios con diferencias de escala en las valoraciones.

Para cada usuario u, se selecciona el conjunto de sus k vecinos más similares:

$$N_k(u)=Top_k(\{\text{sim}(u,v)|v\in U, v\neq u\})$$

A partir de estos vecinos, la estimación de la relevancia de un ítem i se obtiene como una combinación ponderada de las valoraciones de los usuarios vecinos:

$$f(u,i)= \sum_{v \in N_k(u)} \text{sim}(u,v) \cdot r_{v,i}$$

En su formulación clásica, este tipo de modelos suele incorporar una normalización de las valoraciones, restando la media de cada usuario y dividiendo por la suma de las similitudes, con el objetivo de corregir posibles sesgos individuales en las escalas de valoración [3], [5]. Sin embargo, en este trabajo se emplea una versión no normalizada, ya que en escenarios de recomendación Top-N se ha observado que esta variante puede ofrecer mejores resultados en términos de ranking, al preservar directamente las magnitudes de interacción.

Finalmente, las recomendaciones se obtienen seleccionando los ítems con valor de $f(u,i)$, es decir, aplicando el mismo criterio de ranking definido previamente para la generación de recomendaciones.

Este enfoque permite capturar patrones de comportamiento entre usuarios, aunque puede verse afectado por problemas de escalabilidad y por la dispersión de la matriz de interacciones, especialmente en escenarios con pocos datos.

- KNN basado en ítems

El enfoque basado en ítems estima la relevancia de un ítem a partir de su similitud con otros ítems previamente consumidos por el usuario. A diferencia del enfoque basado en usuarios, en este caso la comparación se establece entre ítems en lugar de entre usuarios.

Sea $R \in \mathbb{R}^{|U|\times|I|}$ la matriz de valoraciones, donde $r_{u,i}$ representa la valoración del usuario $u \in U$ sobre el ítem $i \in I$, la similitud entre dos ítems i y j puede calcularse mediante métricas como la similitud del coseno, de forma análoga al enfoque KNN-User. En este caso, la comparación se realiza sobre los usuarios que han interactuado con ambos ítems, en lugar de sobre los ítems valorados por usuarios.

Sea $I_u \subseteq I$ el conjunto de ítems previamente consumidos por el usuario u y sea $C_u = I \setminus I_u$ el conjunto de ítems candidatos a recomendar.

Para cada ítem candidato $i \in C_u$, se selecciona el conjunto de los k ítems más similares dentro del conjunto de ítems previamente consumidos por el usuario:

$$N_k(i) = Top_k(\{\text{sim}(i,j) | j \in I_u\})$$

La puntuación estimada para un ítem candidato i se calcula como una combinación ponderada de las valoraciones del usuario sobre los ítems similares:

$$f(u,i) = \sum_{j \in N_k(i)} \text{sim}(i,j) \cdot r_{u,j}$$

Finalmente, las recomendaciones se generan seleccionando los N ítems con mayor puntuación dentro del conjunto de candidatos:

$$\text{Top-N}(u) = Top_N(\{f(u,i) | i \in C_u\})$$

La formulación clásica suele incorporar normalizaciones adicionales, como el centrado respecto a la media de usuario, con el objetivo de predecir valoraciones explícitas en un rango determinado. Sin embargo, en este trabajo se emplea la versión no normalizada, ya que el objetivo no es estimar una puntuación exacta, sino generar un ranking de ítems. En escenarios de recomendación Top-N, únicamente interesa el orden relativo de las puntuaciones y no su valor absoluto, por lo que no resulta necesarios que estas estén acotadas en un rango específico. De este modo, se favorece una formulación más simple que resulta adecuada para tareas de recomendación basadas en ranking.

Dentro del filtrado colaborativo, además de los métodos basados en vecinos, existen enfoques basados en modelos que aprenden representaciones latentes de usuarios e ítems a partir de interacciones observadas. Entre estos métodos destacan las técnicas de factorización de matrices y los modelos basados en aprendizaje profundo, que permiten capturar relaciones más complejas en los datos.

- Optimización mediante Bayesian Personalized Ranking (BPR)

Bayesian Personalized Ranking (BPR) es un método de aprendizaje para sistemas de recomendación en escenarios de feedback implícito, donde únicamente se dispone de información sobre interacciones observadas, sin valoraciones explícitas.

Sea U el conjunto de usuarios, I el conjunto de ítems e $I_u \subseteq I$ el conjunto de ítems con los que el usuario $u \in U$ ha interactuado. En este contexto, BPR no define un modelo en sí mismo, sino un criterio de optimización cuyo objetivo es aprender una función de puntuación $f(u,i)$ tal que, para cada usuario, los ítems observados obtengan una puntuación mayor que los no observados [12].

Para ello, BPR trabaja con tripletas (u,i,j), donde:

- $i \in I_u$ es un ítem positivo, es decir, un ítem con el que el usuario ha interactuado
- $j \notin I_u$ es un ítem negativo, es decir, un ítem no observado para dicho usuario

El modelo trata de satisfacer la relación de preferencia: $f(u,i) > f(u,j)$

Cada usuario u y cada ítem i se representan mediante vectores latentes $p_u \in \mathbb{R}^d$ y $q_i \in \mathbb{R}^d$, respectivamente, donde d es la dimensión del espacio latente. Esta formulación corresponde al paradigma de factorización de matrices ampliamente utilizado en sistemas de recomendación [4].

La función de puntación se define mediane el producto escalar entre dichos vectores:

$$f(u,i) = p_u^\top q_i$$

De este modo, para una tripleta (u,i,j), se define la diferencia de puntuaciones:

$$x_{uij} = f(u,i) - f(u,j)$$

La probabilidad de que el usuario u prefiera el ítem positivo i frente al ítem negativo j se modela mediante una función sigmoide:

$$P(i \succ_u j) = \sigma(x_{uij}) = \frac{1}{1+e^{-x_{uij}}}$$

El objetivo de aprendizaje consiste en maximizar la probabilidad de observar estas preferencias, lo que conduce a la siguiente función objetivo regularizada propuesta en el trabajo original de BPR [12]:

$$\max_{(u,i,j)} \sum ln\sigma(x_{uij}) - \lambda\|\Theta\|^2$$

Donde Θ representa el conjunto de parámetros del modelo y λ es el parámetro de regularización.

Durante el entrenamiento, el modelo selecciona iterativamente tripletas (u, i, j), y actualiza los vectores latentes mediante descenso gradiente estocástico, con el fin de incrementar la puntuación de los ítems positivos y reducir la de los negativos.

Una vez entrenado el modelo, para cada usuario u se calcula la puntuación de todos los ítems candidatos no consumidos:

$$C_u = I \setminus I_u$$

$$\text{Top-N}(u) = Top_N(\{f(u,i) | i \in C_u\})$$

Este enfoque resulta especialmente adecuado para escenarios de recomendación Top-N ya que optimiza directamente el orden relativo entre ítems observados y no observados, en lugar de centrarse en la predicción exacta de valoraciones [12].

Además de los modelos de factorización de matrices, existen enfoques basados en aprendizaje profundo que utilizan redes neuronales para modelar las interacciones entre usuarios e ítems. Estos modelos permiten capturar relaciones no lineales y patrones complejos en los datos, aunque requieren un mayor volumen de información y capacidad computacional.

- Redes Neuronales

Los modelos basados en redes neuronales dentro de sistemas de recomendación permiten capturar relaciones complejas y no lineales entre usuarios e ítems a partir de las interacciones

observadas. Estos enfoques no se basan en medidas de similitud explícitas, sino que aprenden representaciones latentes de los usuarios y los ítems optimizadas directamente para la tarea de recomendación [5], [2].

En este trabajo se ha implementado un modelo basado en una arquitectura tipo Multi-Layer Perceptron (MLP), el cual aprende una función de puntuación que estima la afinidad entre un usuario y un ítem.

Cada usuario u y cada ítem i se representan mediante embeddings densos aprendidos durante el entrenamiento, $p_u \in \mathbb{R}^d$ y $q_i \in \mathbb{R}^d$, respectivamente, donde d es la dimensión del espacio latente. Este tipo de representación se basa en técnicas de factorización de matrices extendidas mediante aprendizaje profundo [4], [5].

Adicionalmente, el modelo puede incorporar variables auxiliares tanto de usuarios como de ítems, derivadas de estadísticas agregadas del conjunto de entrenamiento, lo que permite enriquecer la representación más allá de las interacciones explícitas.

La entrada del modelo se construye mediante la concatenación de las representaciones del usuario y del ítem, junto con las posibles características adicionales:

$$x_{u,i}=[p_u; q_i; f_u; f_i]$$

donde f_u **y** f_i representan las features adicionales de usuario e ítem respectivamente.

Esta representación es introducida en una red neuronal feed-forward compuesta por varias capas totalmente conectadas:

$$h_1=\phi\left(W_1 x_{u,i}+b_1\right)$$

$$h_2=\phi(W_2 h_1+b_2)$$

...

$$f(u,i)=W_k h_{k\text{-}1}+b_k$$

Donde W_k representa la matriz de pesos aprendida en la capa k, b_k el vector de sesgos asociado a dicha capa y h_k la representación intermedia obtenida tras aplicar la función de activación no lineal. La función $\phi(\cdot)$ corresponde a la función de activación no lineal ReLU. La salida del modelo es una única puntuación escalar $f(u,i)$, que representa la relevancia estimada del ítem i para el usuario u. El modelo puede entrenarse bajo dos configuraciones dependiendo del tipo de feedback:

En el caso de disponer de feedback explícito, el modelo se optimiza minimizando el error cuadrático medio, siguiendo enfoques clásicos de predicción de ratings en sistemas de recomendación [4]:

$$L=\frac{1}{N}\sum_{(u,i)}\left(r_{u,i}-f(u,i)\right)^2$$

Donde $r_{u,i}$ representa la valoración real del usuario sobre el ítem.

En el caso de feedback implícito, el modelo se entrena mediante el criterio de Bayesian Personalized Ranking (BPR), que optimiza directamente el orden relativo entre ítems positivos.y negativos [12].

Para una tripleta (u,i,j) donde i es un ítem observado y j no observado, se define:

$$x_{uij}=f(u,i)-f(u,j)$$

Y la función objetivo:

$$L=-\sum log\sigma\left(x_{uij}\right)+\lambda\|\Theta\|^2$$

Donde $\sigma(\cdot)$ es la función sigmoide y Θ representa los parámetros del modelo.

Una vez entrenado el modelo, se calcula la puntuación para todos los ítems candidatos no consumidos por un usuario:

$$C_u=I\setminus I_u$$

Y se genera un ranking ordenando los ítems según:

$$\text{Top-N}(u)=Top_N(\{f(u,i)|i \in C_u\})$$

Este tipo de modelos permite capturar interacciones complejas entre usuarios e ítems, combinando representaciones latentes con información adicional cuando está disponible. Además, su formulación flexible permite adaptarse tanto a escenarios de feedback explícito como implícito, lo que los convierte en una alternativa más expresiva frente a los métodos clásicos de filtrado colaborativo [2], [5].

2.4.3 SISTEMAS HÍBRIDOS

Los sistemas híbridos combinan varias técnicas de recomendación con el objetivo de aprovechar las ventajas de cada una y reducir sus limitaciones [2], [5]. Habitualmente, integran métodos basados en contenido y filtrado colaborativo, aunque también pueden incorporar otros enfoques.

En este trabajo, se ha implementado un sistema híbrido basado en la combinación ponderada de modelos de recomendación previamente entrenados. En concreto, las recomendaciones finales se obtienen mediante la agregación lineal de las puntuaciones generadas por distintos algoritmos, asignando a cada uno un peso específico que determina su contribución relativa al resultado final.

De forma general, la puntuación final de un ítem i para un usuario u se define como:

$$f(u,i)=\sum_k w_k \cdot f_k(u,i)$$

Donde $f_k(u,i)$ representa la puntuación proporcionada por cada modelo base y w_k los pesos asociados a cada uno de ellos.

Los pesos se han ajustado mediante un proceso de búsqueda en rejilla (grid search), evaluando múltiples combinaciones con el objetivo de maximizar métricas de ranking como NDCG [6]. Este procedimiento permite identificar la combinación óptima de modelos para

cada dataset y estrategia de partición, adaptando el comportamiento del sistema híbrido a diferentes escenarios de evaluación.

Este tipo de sistemas permite mejorar tanto la robustez de las recomendaciones como su calidad global, al combinar modelos con comportamientos complementarios. Por este motivo, constituyen la base de muchos sistemas de recomendación utilizados en aplicaciones reales.

2.4.4 OTROS ENFOQUES

Además de las familias anteriores, existen otros métodos como los sistemas basados en conocimiento, que emplean reglas explícitas o información estructurada sobre las preferencias de usuario, y los sistemas demográficos, que generan recomendaciones a partir de características generales como edad, género o ubicación.

Aunque estos enfoques pueden resultar útiles en determinados contextos, su uso es más limitado en comparación con las técnicas basadas en contenido y filtrado colaborativo, por lo que no constituyen el foco principal de este trabajo.

Como referencia para la evaluación de los algoritmos descritos, es habitual emplear modelos base como el aleatorio (Random) y el basado en popularidad (Popularity). El modelo aleatorio genera recomendaciones sin tener en cuenta ninguna información del usuario, lo que da lugar a un bajo rendimiento en términos de relevancia, pero permite establecer un límite superior en diversidad y novedad. Por su parte, el modelo de popularidad recomienda los ítems más consumidos globalmente, constituyendo un baseline competitivo en términos de precisión. Estos enfoques sirven como punto de comparación para evaluar si los modelos más avanzados aportan mejoras reales.

2.5 EVALUACIÓN DE SISTEMAS DE RECOMENDACIÓN

La evaluación de los sistemas de recomendación en este trabajo se realiza mediante un conjunto de métricas orientadas a escenarios de recomendación Top-N. Estas métricas

permiten analizar no solo la precisión de las recomendaciones, sino también su cobertura, diversidad y novedad.

En el contexto de este trabajo, un ítem se considera relevante para un usuario si aparece en el conjunto de prueba y presenta una interacción positiva suficiente según el criterio definido para cada conjunto de datos. En escenarios de feedback implícito, la relevancia se infiere a partir de la existencia de interacción observada.

En particular, las métricas Precision, Recall, NDCG y Novelty se calculan individualmente para cada usuario y posteriormente se agregan mediante la media aritmética sobre el conjunto de usuarios evaluados U'. Por el contrario, las métricas Aggregated Diversity y User Coverage se definen a nivel global del sistema, por lo que no admiten una agregación mediante media entre usuarios.

$$\text{Metric@K}=\frac{1}{|U'|}\sum_{u\in U'}\text{Metric@K}(u)$$

2.5.1 PRECISIÓN (PRECISION)

La precisión mide la proporción de ítems recomendados que resultan relevantes para el usuario. Para un usuario u, la precisión en el corte K se define como:

$$\text{Precision@K}(u)=\frac{|\text{Rec}_u^K \cap \text{Rel}_u|}{K}$$

Donde Rec_u^K es el conjunto de los K ítems recomendados al usuario u, y Rel_u es el conjunto de ítems relevantes para dicho usuario.

2.5.2 EXHAUSTIVIDAD (RECALL)

El recall mide la capacidad del sistema para recuperar los ítems relevantes de cada usuario. Para un usuario u, se define como:

$$\text{Recall@K}(u)=\frac{|\text{Rec}_u^K \cap \text{Rel}_u|}{|\text{Rel}_u|}$$

Mientras que la precisión evalúa la exactitud de las recomendaciones, el recall mide el grado en que el sistema es capaz de recuperar los ítems relevantes existentes.

2.5.3 NORMALIZED DISCOUNTED CUMULATIVE GAIN (NDCG)

Además de las métricas anteriores, en este trabajo se utiliza la métrica NDCG como criterio de optimización en la selección de los pesos del modelo híbrido.

Esta métrica permite evaluar la calidad del ordenamiento de las recomendaciones, dando mayor importancia a los ítems relevantes que aparecen en posiciones más altas del ranking. Por ese motivo, resulta especialmente adecuada para escenarios de recomendación Top-N.

Para un usuario u, el DCG en el corte K se define como:

$$\text{DCG@K}(u)=\sum_{i=1}^{K}\frac{\text{Rel}_i}{log_2(i+1)}$$

Donde Rel_i indica la relevancia del ítem en la posición i.

El NDCG se obtiene normalizando el DCG por el valor óptimo posible (IDCG):

$$\text{NDCG@K}(u)=\frac{\text{DCG@K}(u)}{\text{IDCG@K}(u)}$$

Donde $\text{IDCG@K}(u)$ representa el valor máximo posible del DCG para el usuario u, obtenido al ordenar los ítems más relevantes de forma óptima en las primeras posiciones del ranking.

En este trabajo, NDCG no se utiliza como métrica principal de comparación entre modelos, sino como función objetivo en el proceso de búsqueda del sistema híbrido.

2.5.4 NOVEDAD (NOVELTY)

La novedad mide hasta qué punto los ítems recomendados son poco populares o menos conocidos. En este trabajo, la novedad de un ítem i se define como:

$$\text{Novelty}(i)=1-\frac{|u \in U: r_{u,i}>0|}{|U|}$$

Donde el numerador representa el número de usuarios que han interactuado con el ítem i, y $|U|$ es el número total de usuarios en el conjunto de entrenamiento.

La novedad para un usuario en el corte K se calcula como:

$$\text{Novelty@K}(u)=\frac{1}{K}\sum_{i\in \text{Rec}_u^K} \text{Novelty}(i)$$

Esta métrica permite evaluar la capacidad del sistema para recomendar ítems menos populares y favorecer el descubrimiento de nuevos elementos.

2.5.5 COBERTURA DE USUARIOS (USER COVERAGE)

La cobertura de usuarios mide la proporción de usuarios para los que el sistema es capaz de generar recomendaciones. Se define como:

$$\text{UserCoverage}=\frac{|U_{rec}|}{|U_{test}|}$$

Donde U_{rec} es el conjunto de usuarios que reciben recomendaciones y U_{test} es el conjunto de usuarios presentes en el conjunto de prueba.

Esta métrica permite complementar a precisión y recall, ya que un modelo puede obtener buenos resultados sobre los usuarios cubiertos, pero no ser capaz de recomendar para todos ellos.

2.5.6 DIVERSIDAD AGREGADA (AGGREGATED DIVERSITY)

La diversidad agregada mide el grado de variedad global de los ítems recomendados en todo el sistema. Se define como la proporción de ítems distintos que aparecen en las recomendaciones Top-K de todos los usuarios respecto al total de ítems conocidos por el

sistema. A diferencia de otras métricas, no se define a nivel de usuario, sino directamente sobre el conjunto completo de recomendaciones generadas:

$$\text{AggDiv@K}=\frac{|u \in U \text{Rec}_u^K|}{|I_{train}|}$$

Donde I_{train} es el conjunto de ítems presentes en el conjunto de entrenamiento.

Valores altos de esta métrica indican que el sistema reparte sus recomendaciones entre un mayor número de ítems, mientras que valores bajos reflejan una tendencia a concentrarlas en pocos elementos populares.

2.6 TRABAJOS RELACIONADOS

En la literatura de sistemas de recomendación existe una gran cantidad de trabajos que abordan tanto el desarrollo de modelos como su evaluación en distintos escenarios [1], [2]. Tradicionalmente, la investigación se ha centrado en la mejora de algoritmos de filtrado colaborativo y en modelos capaces de capturar relaciones latentes entre usuarios e ítems.

Entre los enfoques clásicos, destacan los basados en vecinos más cercanos (KNN), ampliamente utilizados por su simplicidad e interpretabilidad [5]. Posteriormente, los métodos basados en factorización de matrices, como los propuestos en el contexto del problema *Netflix Prize*, demostraron mejoras significativas en términos de precisión al modelar factores latentes [4].

En escenarios de feedback implícito, métodos como Bayesian Personalized Ranking (BPR) han sido ampliamente adoptados, ya que permiten optimizar directamente el ranking de ítems en lugar de predecir valoraciones explícitas [12]. Este enfoque ha dado lugar a múltiples extensiones, incluyendo extensiones basadas en aprendizaje profundo.

Más recientemente, los modelos neuronales han ganado relevancia al capturar relaciones no lineales y combinar múltiples fuentes de información [1], aunque con mayor coste computacional.

En evaluación, diversos trabajos han analizado el impacto de las métricas utilizadas y la necesidad de considerar criterios adicionales como la precisión, la diversidad o la novedad [3], [9]. Asimismo, algunos estudios han señalado la relevancia de utilizar particiones temporales para simular escenarios más realistas de recomendación [7], [11]. Sin embargo, la mayoría de ellos se centran en el diseño del modelo, prestando menor atención al impacto de las estrategias de partición de los datos sobre los resultados obtenidos [6].

2.7 LIMITACIONES DE LAS SOLUCIONES ACTUALES

A pesar del amplio desarrollo de técnicas de recomendación, existen diversas limitaciones en la literatura actual que dificultan la comparación objetiva entre modelos y la extrapolación de resultados a escenarios reales.

Muchos trabajos utilizan configuraciones experimentales distintas que no siempre son comparables entre sí. Diferencias en los conjuntos de datos, las métricas o las técnicas de evaluación dificultan determinar qué modelo ofrece mejor rendimiento global [6].

Además, gran parte de la literatura se centra en mejorar métricas como la precisión, mientras que otros aspectos relevantes, como la diversidad, la novedad o la cobertura, reciben menor atención o se analizan de forma secundaria [9].

Otra limitación relevante es el tratamiento de la partición de los datos. Diferentes estrategias de división en entrenamiento y prueba pueden generar escenarios de evaluación significativamente distintos, afectando tanto a la dificultad del problema como a la validez de las conclusiones obtenidas [10]. En particular, las particiones aleatorias pueden introducir *data leakage,* mientras que las particiones temporales, aunque más representativas de escenarios reales, no siempre se emplean de forma sistemática [7], [11].

En consecuencia, resulta complejo comparar resultados entre trabajos y extraer conclusiones generales sobre el comportamiento de los distintos algoritmos. Esta falta de estandarización en los procedimientos de evaluación pone de manifiesto la necesidad de analizar de forma más detallada el impacto de las estrategias de partición en los sistemas de recomendación.

Capítulo 3. SISTEMA/MODELO DESARROLLADO

3.1 PLANTEAMIENTO DEL PROBLEMA

3.1.1 DEFINICIÓN FORMAL DEL PROBLEMA

En el ámbito de los sistemas de recomendación existe un amplio número de modelos y propuestas en la literatura, que reportan resultados muy diversos en función del enfoque utilizado. Sin embargo, la comparación entre estos métodos no siempre resulta directa, ya que cada trabajo emplea configuraciones experimentales distintas, incluyendo diferentes conjuntos de datos, métricas y estrategias de evaluación. Esta falta de estandarización dificulta determinar de manera objetiva qué modelos ofrecen un mejor rendimiento en escenarios reales [6].

En este contexto, la evaluación de los sistemas de recomendación adquiere un papel fundamental. En particular, la forma en que se particiona un conjunto de datos en subconjuntos de entrenamiento, validación y prueba puede influir de manera significativa en los resultados obtenidos [1]. Distintas estrategias de partición, como las basadas en criterios aleatorios o temporales, así como su aplicación a nivel global o por usuario, generan escenarios experimentales con propiedades diferentes y pueden favorecer o penalizar determinados modelos.

Por lo tanto, el problema abordado en este trabajo no se centra únicamente al desarrollo de modelos de recomendación, sino en el análisis de su evaluación. En particular, se pretende analizar cómo varía el rendimiento de distintos algoritmos de recomendación al aplicar diferentes estrategias de partición de los datos, así como las implicaciones que estas variaciones tienen en la interpretación de los resultados.

3.1.2 REQUISITOS

Para abordar el problema planteado, se han definido los siguientes requisitos del sistema:

Requisitos funcionales:

- Implementar distintos algoritmos de recomendación representativos, incluyendo enfoques básicos como popularidad y modelos aleatorios, así como métodos más avanzados basados en filtrado colaborativo, modelos de ranking y redes neuronales, junto con un sistema híbrido basado en la combinación ponderada de dichos modelos.
- Permitir la generación de recomendaciones personalizadas para cada usuario.
- Incorporar un proceso de preprocesamiento de datos que permita unificar la estructura de los distintos conjuntos de datos utilizados, así como aplicar transformaciones necesarias para facilitar su tratamiento y mejorar la eficiencia computacional del sistema.
- Implementar diferentes estrategias de partición de los datos, incluyendo: partición aleatoria global, partición aleatoria por usuario, partición temporal global y partición temporal por usuario.
- Evaluar los modelos bajo cada una de estas estrategias de partición.
- Incorporar múltiples métricas de evaluación que permitan analizar el rendimiento desde distintas perspectivas, tales como precisión, recall, cobertura de usuarios, diversidad agregada y novedad.
- Facilitar la comparación sistemática de resultados entre algoritmos y estrategias de partición.

Requisitos no funcionales:

- Garantizar la reproducibilidad de los experimentos mediante la fijación de semillas y la correcta documentación del proceso.
- Asegurar una ejecución eficiente de los experimentos, dado el elevado número de combinaciones entre algoritmos y particiones.

- Mantener una estructura modular del sistema que permita añadir de forma sencilla nuevos algoritmos de recomendación y métricas de evaluación, facilitando su extensión y adaptación a distintos escenarios experimentales.
- Proporcionar resultados claros e interpretables que faciliten el análisis comparativo.

El código fuente del proyecto se encuentra disponible en el siguiente repositorio GitHub:

https://github.com/Mariacarrenonindecardona/TFG_MARIA_CARRENO.git

3.1.3 PRUEBAS Y MÉTRICAS DE EVALUACIÓN PARA LA VALIDACIÓN DEL SISTEMA

La validación del sistema se basa en el análisis del rendimiento de los algoritmos de recomendación bajo distintas estrategias de partición de los datos. Para ello, se emplean métricas de evaluación orientadas a escenarios de recomendación Top-N, donde el objetivo es medir la calidad de los ítems recomendados.

En las métricas utilizadas se incluyen:

- Precisión (Precision): proporción de ítems relevantes entre los recomendados.
- Recall: capacidad del sistema para recuperar los ítems relevantes.
- Cobertura de usuarios (User Coverage): porcentaje de usuarios para los que el sistema es capaz de generar recomendaciones.
- Diversidad agregada (Aggregated Diversity): grado de variedad global de los ítems recomendados.
- Novedad (Novelty): medida en que los ítems recomendados son poco conocidos o menos populares.

Estas métricas permiten analizar el rendimiento de los sistemas de recomendación desde distintas perspectivas, no limitándose únicamente a la precisión, sino también considerando aspectos como la diversidad, la cobertura y la capacidad de descubrimiento de nuevos ítems.

Adicionalmente, el diseño del framework de evaluación se ha realizado de forma modular, lo que permite incorporar de manera sencilla nuevas métricas sin modificar la lógica principal del sistema. En concreto, cada métrica se implementa como una función independente que opera sobre las listas de recomendación generadas para cada usuario, lo que facilita la extensión del conjunto de métricas simplemente añadiendo nuevas funciones de evaluación al pipeline existente.

En cuanto a las estrategias de partición de los datos, se consideran los siguientes enfoques:

- Random global: división aleatoria del conjunto completo de interacciones.
- Random user: división independiente de las interacciones de cada usuario.
- Temporal global: partición cronológica considerando todas las interacciones conjuntamente.
- Temporal user: partición cronológica aplicada individualmente a cada usuario.

La combinación de estas métricas y estrategias de partición permiten analizar cómo varía la evaluación de los modelos en función del protocolo experimental empleado. Este análisis resulta fundamental para identificar posibles sesgos y limitaciones en la evaluación de sistemas de recomendación, así como para obtener conclusiones más robustas y representativas sobre su rendimiento.

3.2 DISEÑO DE LA SOLUCIÓN

3.2.1 ENFOQUE PROPUESTO

El enfoque adoptado en este trabajo consiste en el desarrollo de un entorno experimental unificado que permita evaluar distintos algoritmos de recomendación bajo múltiples estrategias de partición de los datos. A diferencia de otros enfoques centrados exclusivamente en la mejora del rendimiento de los modelos, este trabajo pone el foco en analizar cómo el protocolo de evaluación influye en los resultados obtenidos.

Para ello, se ha diseñado un sistema modular que permite aplicar diferentes algoritmos sobre los mismos conjuntos de datos, variando únicamente la estrategia de partición. De este modo, se garantiza que las comparaciones realizadas sean consistentes y que las diferencias observadas en el rendimiento se deban principalmente al esquema de evaluación empleado.

3.2.2 ARQUITECTURA DEL SISTEMA

La Figura 6 muestra la arquitectura general del sistema desarrollado, estructurado como un pipeline que abarca desde la carga y preprocesamiento de los datos hasta la generación y evaluación de las recomendaciones.

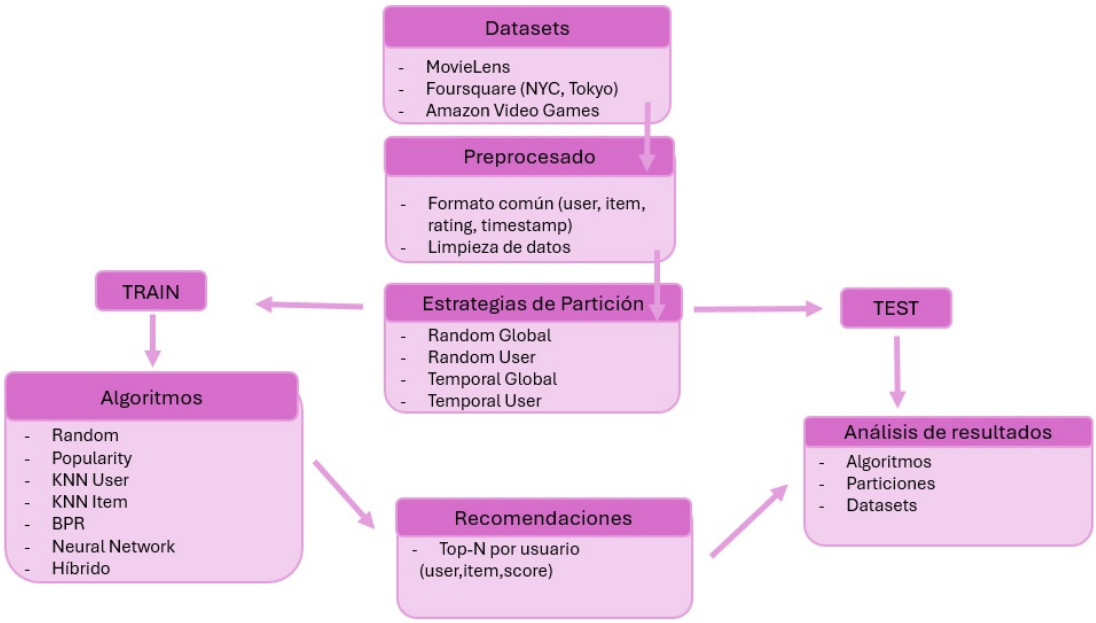


Figura 6. Esquema del sistema propuesto

El sistema comienza con la incorporación de distintos conjuntos de datos, que posteriormente son sometidos a un proceso de preprocesamiento con el objetivo de unificar su formato y facilitar su tratamiento. A continuación, se aplican diversas estrategias de partición de los datos, incluyendo enfoques aleatorios y temporales, tanto a nivel global como por usuario.

Una vez definidos los conjuntos de entrenamiento y prueba, se procede al entrenamiento de los distintos algoritmos de recomendación implementados, que incluyen tanto modelos base como enfoques más avanzados. Estos modelos generan listas de recomendaciones para cada usuario en formato Top-N.

Finalmente, el sistema evalúa las recomendaciones generadas mediante las métricas definidas, permitiendo analizar y comparar el rendimiento de los algoritmos en función de la estrategia de partición empleada. Este enfoque facilita el estudio del impacto del protocolo de evaluación en los resultados obtenidos.

Cada uno de los bloques del pipeline se ha diseñado de forma modular, permitiendo la reutilización de componentes y la incorporación de nuevas estrategias o algoritmos.

3.2.3 ALGORITMOS/MODELOS UTILIZADOS

En el sistema desarrollado se han implementado diversos algoritmos de recomendación con el objetivo de comparar el impacto de las estrategias de partición de los datos sobre modelos de diferente complejidad.

Como baselines, se utilizaron los modelos aleatorios y de popularidad. Además. Se implementaron algoritmos de filtrado colaborativo basados en vecinos próximos, tanto a nivel de usuario (KNN user-based) como a nivel de ítem (KNN item-based), siguiendo las formulaciones descritas en la Sección 2.4.2.

También se incorporaron modelos basados en factorización y aprendizaje profundo, incluyendo Bayesian Personalized Ranking (BPR) [12] y un modelo basado en redes neuronales. Finalmente, se implementó un sistema híbrido basado en la combinación ponderada de los modelos anteriores.

El sistema se diseñó de forma modular, permitiendo incorporar nuevos algoritmos mediante una interfaz común de entrenamiento y predicción.

3.2.4 CONJUNTOS DE DATOS UTILIZADOS

Para la validación del sistema se han empleado distintos conjuntos de datos pertenecientes a diferentes dominios, con el objetivo de analizar el comportamiento de los algoritmos bajo escenarios variados. En primer lugar, se ha utilizado el dataset MovieLens, ampliamente empleado en la investigación en sistemas de recomendación. Este conjunto de datos contiene valoraciones explícitas de usuarios sobre películas. Asimismo, se utilizaron los subconjuntos de Foursquare correspondientes a las ciudades de Nueva York y Tokio, que contienen interacciones implícitas en forma de check-ins en puntos de interés. Por último, se ha utilizado el dataset Amazon Video Games, que contiene información sobre interacciones de usuarios con productos de la categoría videojuegos. Debido a su gran tamaño y elevada dispersión, se ha utilizado su versión reducida denominada Amazon Video Games 5-core, que filtra las interacciones manteniendo únicamente aquellos usuarios e ítems con al menos cinco interacciones. Esta reducción permite disminuir la dispersión del conjunto de datos y facilita la ejecución de los algoritmos manteniendo al mismo tiempo un volumen de información representativo del dominio. A partir de este punto, cualquier referencia al conjunto de datos Amazon Video Games hará alusión a esta versión reducida.

Dataset	**Dominio**	**Tipo de Feedback**	**Usuarios**	**Ítems**	**Interacciones**
MovieLens	Películas	Explícito	610	9.724	100.836
Foursquare Nueva York	Puntos de interés	Implícito	1.083	38.333	91.024
Foursquare Tokyo	Puntos de interés	Implícito	2.293	61.858	211.955
Amazon Video Games	Videojuegos	Explícito	2.766.656	167.645	4.624.615
Amazon Video Games 5-core	Videojuegos	Explícito	55.223	17.408	497.577

Tabla 1. Características de los datasets utilizados

La Tabla 1 resume las principales características de los conjuntos de datos empleados. Como puede observarse, existen diferencias significativas en tamaño, densidad y tipo de interacción entre los distintos conjuntos de datos. En particular, el conjunto de datos Amazon Video Games presenta una escala considerablemente mayor, mientras que su versión 5-core reduce la dispersión al conservar únicamente usuarios e ítems con al menos cinco interacciones.

MovieLens presenta un tamaño moderado, lo que lo convierte en un entorno adecuado para validar algoritmos en condiciones controladas. Por su parte, los subconjuntos de Foursquare contienen un mayor número de ítems en relación con el número de usuarios, además de interacciones implícitas basadas en check-ins, lo que introduce un escenario más disperso y cercano a aplicaciones reales de recomendación contextual.

Todos los datasets fueron preprocesados para unificar su estructura y adaptarlos a un formato común dentro del pipeline experimental. En conjunto, estas diferencias en tamaño, dominio y tipo de interacción permiten evaluar los modelos bajo condiciones dispersas, aportando una visión más completa de su rendimiento y capacidad de generalización.

La utilización de datasets pertenecientes a distintos dominios y con características heterogéneas permite evaluar los modelos en escenarios variados y analizar su capacidad de generalización.

3.2.5 JUSTIFICACIÓN TÉCNICA

El diseño del sistema responde a la necesidad de analizar de forma rigurosa el impacto de las estrategias de partición de los datos en la evaluación de los sistemas de recomendación. Para ello, se ha utilizado un enfoque modular basado en un pipeline que desacopla las distintas fases del proceso, garantizando la reproducibilidad de los experimentos.

Los algoritmos de recomendación se aplican sobre los mismos conjuntos de datos, variando únicamente la estrategia de partición. Esto permite aislar el efecto del protocolo de evaluación y asegurar que las diferencias en el rendimiento se deban principalmente a la

forma de dividir los datos. Además, se incluyen múltiples algoritmos, desde modelos base hasta enfoques más avanzados, con el objetivo de analizar si el impacto de la partición es consistente entre distintos tipos de modelos. Del mismo modo, se utilizan datasets de distintos dominios para evaluar la robustez del sistema en contextos variados.

En la evaluación, se incorporan métricas más allá de la precisión, como cobertura, diversidad y novedad, lo que permite una visión más completa del rendimiento de los sistemas de recomendación. Finalmente, el preprocesamiento unifica la estructura de los datasets para integrarlos en un mismo pipeline, garantizando la coherencia de los experimentos y la comparabilidad de los resultados.

En conjunto, estas decisiones permiten construir un entorno experimental sólido y flexible, para analizar el impacto de las estrategias de partición en la evaluación de los sistemas.

3.3 IMPLEMENTACIÓN

3.3.1 TECNOLOGÍAS Y RECURSOS HW/SW EMPLEADOS

El sistema desarrollado ha sido implementado íntegramente en Python, utilizando diversas librerías para el procesamiento de datos y la implementación de los algoritmos de recomendación. Entre las principales herramientas empleadas destacan:

- Pandas y NumPy: Utilizadas para la manipulación y el procesamiento de los datos.
- Scikit-learn: Utilizada para la partición de los datos y el cálculo de similitudes en los algoritmos basados en KNN.
- SciPy: Utilizada para la representación de matrices dispersas (csr_matrix), fundamentales en sistemas de recomendación.
- PyTorch: Utilizado para la implementación de modelos de aprendizaje profundo.
- Matplotlib: Utilizado para la generación de gráficos y análisis visual de resultados.
- Librerías estándar de Python para la gestión de archivos y estructuras del sistema: os, sys, json, glob, datetime, re, abc y collections.

Con el objetivo de garantizar la reproductibilidad del trabajo, se ha hecho uso de la plataforma GitHub para el almacenamiento y la gestión de código desarrollado. En el repositorio se incluyen todos los archivos necesarios para la ejecución del trabajo, así como los conjuntos de datos utilizados y un archivo con las instrucciones que detallan el procedimiento a seguir para su correcta utilización. Repositorio del proyecto:

https://github.com/Mariacarrenonindecardona/TFG_MARIA_CARRENO.git

Para la ejecución de experimentos sobre datasets de mayor tamaño, como el dataset de videojuegos de Amazon, se ha utilizado el servidor de alto rendimiento de la universidad (DGX), equipado con múltiples GPUs NVIDIA de alta capacidad, Este tipo de infraestructura está específicamente diseñado para el entrenamiento de modelos de inteligencia artificial, lo que ha permitido reducir significativamente los tiempos de entrenamiento, especialmente en modelos más complejos como BPR.

3.3.2 ESTRATEGIAS DE PARTICIÓN DE DATOS

Las estrategias de partición descritas en la Sección 2.3 se implementan dentro del módulo DataPartition. Todas ellas generan los mismos subconjuntos de entrenamiento (training), validación (validation) y prueba (test) utilizados por los distintos algoritmos.

De forma general, cualquier conjunto de interacciones D se divide en tres subconjuntos disjuntos:

$$D=D_{train} \cup D_{val} \cup D_{test}$$

Donde:

$$D_{train} \cap D_{val}=D_{train} \cap D_{test}=D_{val} \cap D_{test}=\emptyset$$

Y cumpliendo aproximadamente las proporciones:

$$|D_{train}| \approx 0.7|D|, |D_{val}| \approx 0.1|D|, |D_{test}| \approx 0.2|D|$$

A partir de esta definición común, se describen las distintas estrategias de partición.

- Random Global

En la partición aleatoria global, todas las interacciones del conjunto de datos se consideran conjuntamente y se distribuyen aleatoriamente entre los subconjuntos de entrenamiento, validación y prueba, sin distinguir entre usuarios ni considerar el orden temporal. Aunque es un enfoque sencillo, puede introducir problemas de data leakage o fuga de información, especialmente en conjuntos de datos con componente temporal. Al asignar interacciones de forma aleatoria, es posible que el conjunto de entrenamiento contenga eventos posteriores a algunos ejemplos del conjunto de prueba, proporcionando al modelo información futura que no estaría disponible en un escenario real [10].

- Random User

En la partición aleatoria por usuario, la división se realiza de forma independiente para cada usuario, manteniendo la misma proporción de datos en cada subconjunto. Esto garantiza la presencia de todos los usuarios durante la evaluación, aunque no preserva información temporal.

- Temporal Global

En la partición temporal global, todas las interacciones se ordenan cronológicamente según su marca temporal t. Posteriormente, se dividen secuencialmente en entrenamiento, validación y prueba. Este enfoque permite simular escenarios más realistas, donde el sistema de recomendación debe predecir interacciones futuras a partir de datos pasados. Sin embargo, al realizar la división de forma global, algunos usuarios pueden quedar desigualmente representados en cada subconjunto

- Temporal User

En la partición temporal por usuario, las interacciones de cada usuario se ordenan cronológicamente de forma independiente. Posteriormente, cada secuencia individual se divide en entrenamiento, validación y prueba, donde los conjuntos globales se obtienen

como la unión de las particiones por usuario. De este modo, se preserva tanto la evolución temporal como la consistencia de las interacciones de cada usuario.

Los porcentajes de partición pueden modificarse fácilmente en el módulo DataPartition mediante la función train_test_split de Scikit-learn. Esto permite definir distintas configuraciones de entrenamiento, validación y prueba sin alterar la lógica general del sistema.

3.3.3 SELECCIÓN DE HIPERPARÁMETROS Y CONFIGURACIÓN EXPERIMENTAL

La selección de los hiperparámetros se ha realizado con el objetivo de equilibrar rendimiento predictivo y eficiencia computacional, manteniendo coherencia entre experimentos. Se ha definido un espacio de búsqueda para cada modelo, resumido en la Tabla 2.

Modelo	**Hiperparámetros evaluados**
Random	No presenta hiperparámetros
Popularity	No presenta hiperparámetros
KNN (User / Item)	Número de vecinos $k \in [1,25]$; métrica de similitud (coseno, Pearson)
BPR	Dimensión de factores latentes {32, 64, 128}; tasa de aprendizaje [10^{-4}, 10^{-2}]; regulación [10^{-4}, 10^{-2}]; número de muestras negativas {1, 4, 5, 10}
NeuralCF	Dimensión de embeddings {32, 64, 128}; arquitectura de capas ocultas {1, 2}; tasa de aprendizaje [10^{-4}, 10^{-2}]; tamaño de batch {32, 64, 128}; número de épocas {2, 5, 10, 20, 30}; dropout {0.1, 0.2, 0.5}; muestreo negativo {1, 4, 5, 10}

Híbrido	Modelo A ∈ {KNN User, KNN Item, Popularity, BPR, NeuralCF}; Modelo B ∈ {KNN User, KNN Item, Popularity, BPR, NeuralCF}; pesos de combinación [0.5, 0.5], [0.7, 0.3], [0.4, 0.6], [0.3, 0.7], [0.6, 0.4]

Tabla 2. Espacio de búsqueda de hiperparámetros por modelo

A partir del espacio de búsqueda definido, los hiperparámetros se seleccionan maximizando NDCG@10 en validación y se evalúan posteriormente en test.

En los modelos KNN, se explora $k \in [1,25]$, seleccionando finalmente k =10 con similitud coseno por su rendimiento más estable.

En los modelos neuronales se emplea una configuración de complejidad moderada: embeddings de 64 dimensiones, dos capas ocultas (64, 32), batch size de 512, tasa de aprendizaje de 10^{-3}, 10 épocas, dropout de 0.1 y 4 muestras negativas por interacción positiva.

El modelo híbrido se ajusta de forma independiente para cada dataset y estrategia de partición. Las configuraciones óptimas se recogen en detalle en la Tabla 3 (Anexo II).

Finalmente, todos los experimentos se ejecutan con semilla fija (seed=42) para garantizar la reproducibilidad.

3.3.4 ESTRUCTURA DEL CÓDIGO

El sistema ha sido diseñado siguiendo una estructura modular que separa claramente las distintas fases del pipeline. Esta organización facilita la comprensión del sistema, su mantenimiento y su extensibilidad.

La Figura 7 muestra la estructura general del proyecto y el flujo de interacción entre distintos módulos. El diagrama refleja cómo los componentes se comunican entre sí y cómo el usuario interactúa con el sistema a través del módulo principal de ejecución.

Como puede observarse, el sistema sigue un flujo secuencial que parte de los datos, atraviesa las fases de preparación y partición, y culmina en la generación y evaluación de recomendaciones. Además, el módulo "Main" actúa como coordinador de la ejecución de los distintos componentes y gestionando la configuración de los experimentos.

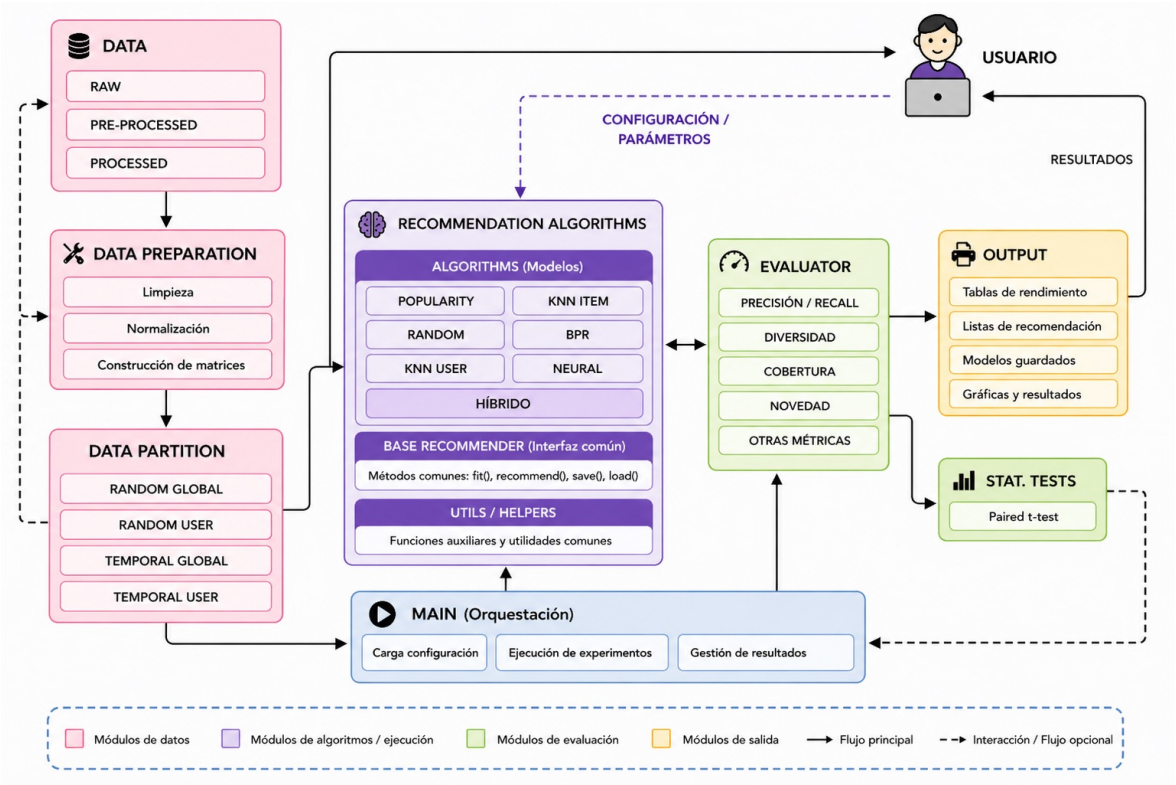


Figura 7. Arquitectura del sistema y flujo de interacción entre los módulos

La estructura del código se compone de los siguientes elementos principales:

- Data/RAW: Contiene los datasets originales descargados, sin ningún tipo de preprocesamiento.
- Data/PRE-PROCESSED: Incluye los datos tras una primera fase de limpieza y transformación, adaptados a un formato común.
- Data/PROCESSED: Contiene los datasets finales, organizados según las distintas estrategias de partición (random global, random user, temporal global, temporal user), e incluyendo los conjuntos de entrenamiento, validación y test.

- DataPreparation: Incluye los scripts encargados de preparar los datos de cada dataset específico. En estos scripts se realizan transformaciones como el cambio de formato, limpieza de datos y adaptación de identificadores.
- DataPartition: Contiene los módulos responsables de generar las distintas particiones de los datos. Cada estrategia de partición se implementa en un archivo independiente, lo que facilita su reutilización y comparación.
- RecommendationAlgorithms: Incluye la implementación de los distintos algoritmos de recomendación, estructurados mediante una clase base común que define la interfaz general del sistema. Esta clase permite cargar automáticamente los datos de entrenamiento en función del dataset y la partición seleccionada.
- RecommendationAlgorithms/Evaluator: Módulo encargado de calcular las métricas de evaluación a partir de las recomendaciones generadas.
- RecommendationAlgorithms/Main: Script principal que organiza la ejecución del sistema, permitiendo seleccionar el algoritmo, el dataset y la estrategia de partición, así como generar recomendaciones y evaluar los resultados.

Esta organización modular permite una clara separación de responsabilidades y facilita la extensibilidad del sistema, posibilitando la incorporación de nuevos algoritmos, datasets o estrategias de evaluación de manera sencilla.

3.3.5 PIPELINE (PREPROCESADO, ENTRENAMIENTO, EVALUACIÓN)

El sistema desarrollado sigue un pipeline estructurado que abarca todas las fases necesarias para la generación y evaluación de recomendaciones, desde el tratamiento inicial de los datos hasta el análisis final de los resultados. En primer lugar, los datasets originales, almacenados en la carpeta Data/RAW, son sometidos a un proceso de preprocesamiento mediante los módulos incluidos en DataPreparation. En esta fase se unifica el formato de los datos, adaptándolos a una estructura común compuesta por las columnas user_id, item_id, rating y timestamp. Además, se realizan transformaciones adicionales en cada conjunto de datos, ya que presentaban particularidades propias que requerían tratamientos específicos.

MovieLens

El dataset MovieLens ya dispone de una estructura muy próxima al formato requerido por el sistema, incluyendo el identificador de usuario, identificador de ítem, valoración explícita y marca temporal. Por ello, el preprocesado consistió principalmente en el renombrado de columnas (userId a user_id, movieId a item_id) y su almacenamiento en el formato estándar utilizado por el resto del pipeline.

Foursquare (Tokyo y Nueva York)

Los datasets de Foursquare correspondientes a Tokyo y Nueva York no contienen valoraciones explícitas, sino registros de visitas (check-ins) realizados por los usuarios en diferentes localizaciones. Para adaptar estos datos a un sistema de recomendación basado en ratings, se definió la valoración implícita como el número de visitas realizadas por cada usuario a cada localización. De este modo, para cada par usuario-ítem se contabilizó el número total de interacciones, utilizándose dicho valor como rating. Además, cuando existían múltiples visitas al mismo lugar, se conservó únicamente la marca temporal más reciente, permitiendo así mantener coherencia temporal en posteriores particiones cronológicas. Este enfoque permite interpretar una mayor frecuencia de visitas como una señal de mayor preferencia del usuario hacia dicho establecimiento.

Amazon Video Games 5-core

En el caso del dataset Amazon Video Games 5-core, los datos originales se encontraban en formato JSON, por lo que fue necesario realizar una transformación previa para adaptarlos a la estructura común empleada en todo el sistema. Para ello, se extrajeron los campos relevantes asociados al usuario, producto, valoración y marca temporal, renombrándolos respectivamente como user_id, item_id, rating y timestamp, para seguir la notación establecida. Posteriormente, los identificadores originales de usuario, representados mediante cadenas alfanuméricas, fueron convertidos a identificadores numéricos consecutivos. Esta transformación permite reducir el consumo de memoria y mejorar la eficiencia computacional durante el entrenamiento de los algoritmos de recomendación.

Cabe destacar que la versión 5-core del dataset ya incorpora un filtrado previo habitual en sistemas de recomendación, garantizando que cada usuario y cada producto dispongan de al menos cinco interacciones registradas. Esto contribuye a reducir la dispersión de la matriz usuario-ítem y facilita la obtención de resultados más estables durante la evaluación. Finalmente, el conjunto de datos fue almacenado en formato CSV siguiendo el esquema utilizado por el resto de datasets del trabajo.

Gracias a este proceso, todos los datasets quedaron representados bajo un esquema homogéneo, permitiendo reutilizar el mismo pipeline experimental independientemente del dominio de los datos. Una vez preprocesados, los datos son divididos en conjuntos de entrenamiento, validación y test mediante los módulos definidos en DataPartition. Se implementan diferentes estrategias de partición, incluyendo enfoques aleatorios y temporales, tanto a nivel global como por usuario. Estas particiones se almacenan en la carpeta Data/PROCESSED, permitiendo reutilizar los mismos conjuntos de datos en distintos experimentos.

A continuación, los algoritmos de recomendación son entrenados utilizando exclusivamente los datos de entrenamiento. Dependiendo del modelo, este proceso puede implicar el cálculo de similitudes entre usuarios o ítems, como en los algoritmos basados en KNN, o el aprendizaje de parámetros mediante optimización, como en el caso del modelo BPR o en la red neuronal propuesta. Una vez entrenados los modelos, se generan recomendaciones en formato Top-N para cada usuario del conjunto de test. Estas recomendaciones se representan como tripletas (user_id, item_id, score), que indican la relevancia estimada de cada ítem para cada usuario.

Finalmente, el módulo Evaluator calcula las métricas de evaluación a partir de las recomendaciones generadas, comparándolas con las interacciones reales del conjunto de test. Entre las métricas consideradas se incluyen precisión, recall, cobertura, diversidad y novedad, lo que permite obtener una evaluación completa del rendimiento del sistema.

Este pipeline permite desacoplar las distintas fases del proceso y garantiza la coherencia de los experimentos, facilitando la comparación de resultados entre diferentes algoritmos y estrategias de partición.

3.3.6 REPRODUCIBILIDAD

Con el objetivo de garantizar la reproducibilidad de los experimentos, el sistema ha sido diseñado de manera que permita replicar fácilmente los resultados obtenidos bajo las mismas condiciones. En este sentido, las estrategias de partición utilizan semillas aleatorias fijas, lo asegurando que las divisiones en conjuntos de entrenamiento, validación y test sean consistentes entre distintas ejecuciones, lo que resulta fundamental para comparar de forma justa el rendimiento de los algoritmos.

Asimismo, la estructura modular del sistema permite ejecutar experimentos de forma controlada mediante el script principal (main.py), en el que se especifican parámetros clave como el algoritmo utilizado, el dataset y la estrategia de partición. Esta configuración centralizada facilita la repetición de experimentos y la comparación sistemática de resultados. Además, tanto las recomendaciones generadas como las métricas de evaluación se almacenan automáticamente en archivos con marcas temporales, lo que permite identificar de manera precisa cada ejecución y mantener un registro organizado de los resultados obtenidos.

Por último, la organización de los datos en las carpetas permite reconstruir todo el pipeline desde el origen de los datos hasta la evaluación final, garantizando la trazabilidad completa del proceso. En conjunto, estas decisiones permiten asegurar que los experimentos realizados puedan ser reproducidos y verificados, contribuyendo a la fiabilidad de los resultados presentados.

Capítulo 4. RESULTADOS

El objetivo principal de este capítulo es analizar el comportamiento de cada algoritmo en diferentes particiones de los datasets, comparando su capacidad para generar recomendaciones relevantes, novedosas y diversas.

4.1 METODOLOGÍA DE VALIDACIÓN Y CRITERIOS DE EVALUACIÓN

Una vez implementados los algoritmos de recomendación y definidos los distintos esquemas de partición de los datos, se ha procedido a la evaluación experimental de los modelos desarrollados.

Con el fin de garantizar la reproducibilidad de los resultados, todas las pruebas se han ejecutado bajo una metodología homogénea, manteniendo configuraciones consistentes entre experimentos y fijando una semilla aleatoria común en aquellos algoritmos o particiones que incorporan componentes estocásticos. Cada algoritmo ha sido evaluado sobre los cuatro conjuntos de datos empleados en este trabajo:

- MovieLens Latest Small, centrado en valoraciones de películas.
- Foursquare Nueva York, basado en registros de localización en la ciudad de Nueva York.
- Foursquare Tokyo, compuesto por interacciones localizadas en Tokio.
- Amazon Video Games, correspondiente a reseñas e interacciones sobre videojuegos.

Sobre cada dataset se han aplicado las cuatro estrategias de partición de datos: aleatoria por usuario, aleatoria global, temporal por usuario y temporal global. Estas particiones permiten simular los distintos escenarios de recomendación, desde divisiones aleatorias tradicionales hasta entornos más realistas donde se respeta el orden temporal de las interacciones.

Los algoritmos evaluados en este estudio corresponden a los modelos ya descritos en las secciones anteriores del trabajo, incluyendo tanto modelos base (KNN, Popularity, BPR,

Neural Collaborative Filtering) como combinaciones híbridas de dichos modelos, ajustadas mediante ponderaciones específicas por partición.

Con el objetivo de analizar la sensibilidad de los resultados respecto al número de elementos evaluados, las métricas se calcularon utilizando dos puntos de corte, K=5 (tonos más claros en las gráficas) y K=10 (tonos más oscuros). Así, cada indicador refleja la calidad de las primeras K posiciones del ranking recomendado para cada usuario.

La calidad de las recomendaciones se ha medido mediante las siguientes métricas:

- Precision@K, que cuantifica la proporción de recomendaciones relevantes.
- Recall@K, que mide la capacidad de recuperar elementos relevantes.
- Novelty@K, asociada al grado de novedad de los ítems recomendados.
- Aggregated diversity@K, que evalúa la variedad global del catálogo recomendado.

Los resultados obtenidos se presentan en los apartados siguientes, organizados por conjunto de datos, junto con un análisis comparativo e interpretación crítica de los mismos. Estos resultados se representan mediante gráficos estructurados de forma homogénea para todos los datasets. En cada figura, los algoritmos se identifican mediante un código de color consistente a lo largo de todo el análisis, lo que permite su comparación directa entre experimentos. Cada fila de las gráficas corresponde a una estrategia de partición de los datos, mientras que cada columna representa una de las métricas de evaluación consideradas. En las figuras principales se incluyen todas las métricas mencionadas anteriormente, excepto recall@K, cuyos resultados se recogen en el Anexo I para facilitar la visualización e interpretación de los resultados.

4.2 ANÁLISIS GENERAL DE RESULTADOS

De forma global, los resultados muestran que el rendimiento de los algoritmos depende tanto del conjunto de datos como de la estrategia de partición empleada, sin que exista un modelo claramente superior en todos los escenarios.

En particular, los mejores resultados en precisión no se concentran en los modelos más complejos. En los datasets de Foursquare, el algoritmo Popularity destaca frente al resto, mientras que en MovieLens los modelos basados en KNN alcanzan los valores más altos en varias particiones. En cambio, modelos más complejos como BPR y Neural Collaborative Filtering no muestran una ventaja sistemática.

De la misma forma, el efecto de la estrategia de partición tampoco es uniforme. Aunque las particiones aleatorias pueden favorecer el rendimiento en determinados casos, especialmente en datasets más densos, los resultados observados indican que la diferencia entre particiones aleatorias y temporales depende en gran medida del dominio analizado y del algoritmo utilizado.

Respecto a la novedad y la diversidad, se identifica un compromiso claro con la precisión, los modelos que maximizan la relevancia tienden a recomendar ítems más populares, reduciendo así la novedad, mientras que aquellos que introducen mayor novedad suelen sacrificar parcialmente la precisión.

Por último, cabe destacar que los datasets basados en puntos de interés, como Foursquare, presentan en general un rendimiento inferior en comparación con MovieLens. Este comportamiento puede explicarse por su mayor grado de dispersión (sparsity), definido como la proporción de interacciones observadas respecto al total posible de combinaciones usuario-ítem. En estos datasets, la densidad es considerablemente menor, lo que dificulta el aprendizaje de patrones robustos por parte de los algoritmos.

4.2.1 LIMITACIONES DE GENERALIZACIÓN DE LOS RESULTADOS

A pesar de las tendencias observadas en los experimentos, es importante señalar que no todos los patrones identificados pueden considerarse generalizables a cualquier escenario de recomendación.

En primer lugar, se observa que el rendimiento relativo de los algoritmos depende fuertemente del nivel de dispersión del dataset. En entornos densos, como MovieLens, los modelos colaborativos basados en KNN presentan un comportamiento competitivo, mientras

que en datasets altamente dispersos como Foursquare, los modelos más simples basados en popularidad tienden a dominar. Esto indica que no existe un algoritmo universalmente superior, sino que su rendimiento está condicionado por las propiedades del dominio.

En segundo lugar, se ha observado que el compromiso entre precisión, novedad y diversidad es estable en términos cualitativos, pero su intensidad varía entre datasets. En particular, la penalización de la diversidad asociada a la optimización de la precisión es más marcada en datasets dispersos, mientras que en datasets densos es posible alcanzar mejores equilibrios entre ambas dimensiones.

Finalmente, otro aspecto no generalizable es el comportamiento relativo de modelos más complejos como BPR o Neural Collaborative Filtering. Aunque en teoría estos modelos deberían capturar patrones más complejos, en la práctica no siempre superan a modelos más simples, especialmente cuando la cantidad de datos por usuarios es limitada.

En conjunto, estos resultados sugieren que el rendimiento de los algoritmos de recomendación está fuertemente condicionado por las características del dataset y del protocolo de evaluación, por lo que los resultados deben interpretarse dentro del contexto experimental específico.

4.3 RESULTADOS EN MOVIELENS LATEST SMALL

El análisis de resultados en el dataset MovieLens Latest Small (Figura 8) confirma que no existe un único algoritmo claramente superior en todos los escenarios de evaluación, sino que el rendimiento depende de forma significativa de las estrategias de partición empleada.

En términos de precisión, los modelos basados en filtrado colaborativo, en particular KNN User-Based y KNN Item-Based, muestran un comportamiento consistente y competitivo en la mayoría de las particiones, destacando especialmente en escenarios aleatorios. En la partición random user, estos modelos alcanzan los valores más elevados de precisión, lo que sugiere que, cuando se preserva la distribución de interacciones por usuario, los métodos colaborativos son capaces de explotar eficazmente la información disponible.

Sin embargo, este comportamiento cambia en escenarios temporales. En la partición temporal global, el modelo basado en popularidad presenta un rendimiento notablemente superior en precisión, lo que indica que, en contextos donde se respeta la evolución temporal de las interacciones, los patrones globales de consumo recientes adquieren mayor relevancia que las similitudes entre usuarios o ítems. Este resultado pone de manifiesto la sensibilidad de los modelos colaborativos a la disponibilidad de información histórica y su posible degradación en escenarios más realistas.

En cuanto al modelo híbrido, su comportamiento resulta especialmente interesante. En general, presenta un rendimiento competitivo en todas las particiones, situándose frecuentemente entre los mejores modelos en términos de precisión. Aunque no siempre alcanza el valor máximo, sí muestra una mayor estabilidad entre particiones en comparación con los modelos individuales, lo que sugiere que la combinación de algoritmos permite mitigar parcialmente las debilidades de cada enfoque por separado. Este efecto es particularmente visible en escenarios donde ningún modelo base domina claramente.

Respecto a la novedad, los resultados siguen el patrón esperado. El modelo aleatorio alcanza sistemáticamente los valores más altos, debido a la ausencia de sesgo hacia ítems populares. A continuación, los modelos colaborativos y neuronales presentan niveles intermedios de

novedad, mientras que el algoritmo de popularidad obtiene los valores más bajos, al concentrar sus recomendaciones en los ítems más frecuentes del dataset. El modelo híbrido se sitúa generalmente en una posición intermedia, lo que refleja el equilibrio entre relevancia y exploración derivado de la combinación de algoritmos.

En relación con la diversidad agregada, se observa nuevamente que el modelo aleatorio maximiza la cobertura del catálogo, recomendado un conjunto muy amplio de ítems. En el extremo opuesto, el modelo de popularidad presenta una diversidad muy reducida, al concentrar sus recomendaciones en un número limitado de elementos. Los modelos colaborativos, especialmente los basados en KNN, ofrecen un compromiso más equilibrado entere diversidad y precisión. El modelo híbrido, por su parte, mantiene niveles de diversidad superiores a los de popularidad, aunque sin alcanzar los valores del modelo aleatorio, lo que refuerza su carácter intermedio.

De forma global, los resultados de este dataset evidencian el clásico compromiso entre precisión, novedad y diversidad. Los modelos que optimizan la relevancia tienden a reducir la exploración del espacio de ítems, mientras que aquellos que introducen mayor diversidad sacrifican parcialmente la precisión. En este contexto, los modelos híbridos emergen como una alternativa interesante, al proporcionar soluciones más equilibradas y robustas frente a cambios en la estrategia de partición.

Finalmente, cabe destacar que las particiones aleatorias, y en particular random user, tienden a ofrecer resultados más favorables para la mayoría de los algoritmos, mientras que las particiones temporales introducen un escenario más exigente y cercano a situaciones reales. Esta diferencia refuerza la importancia de considerar múltiples esquemas de evaluación a la hora de analizar el rendimiento de los sistemas de recomendación.

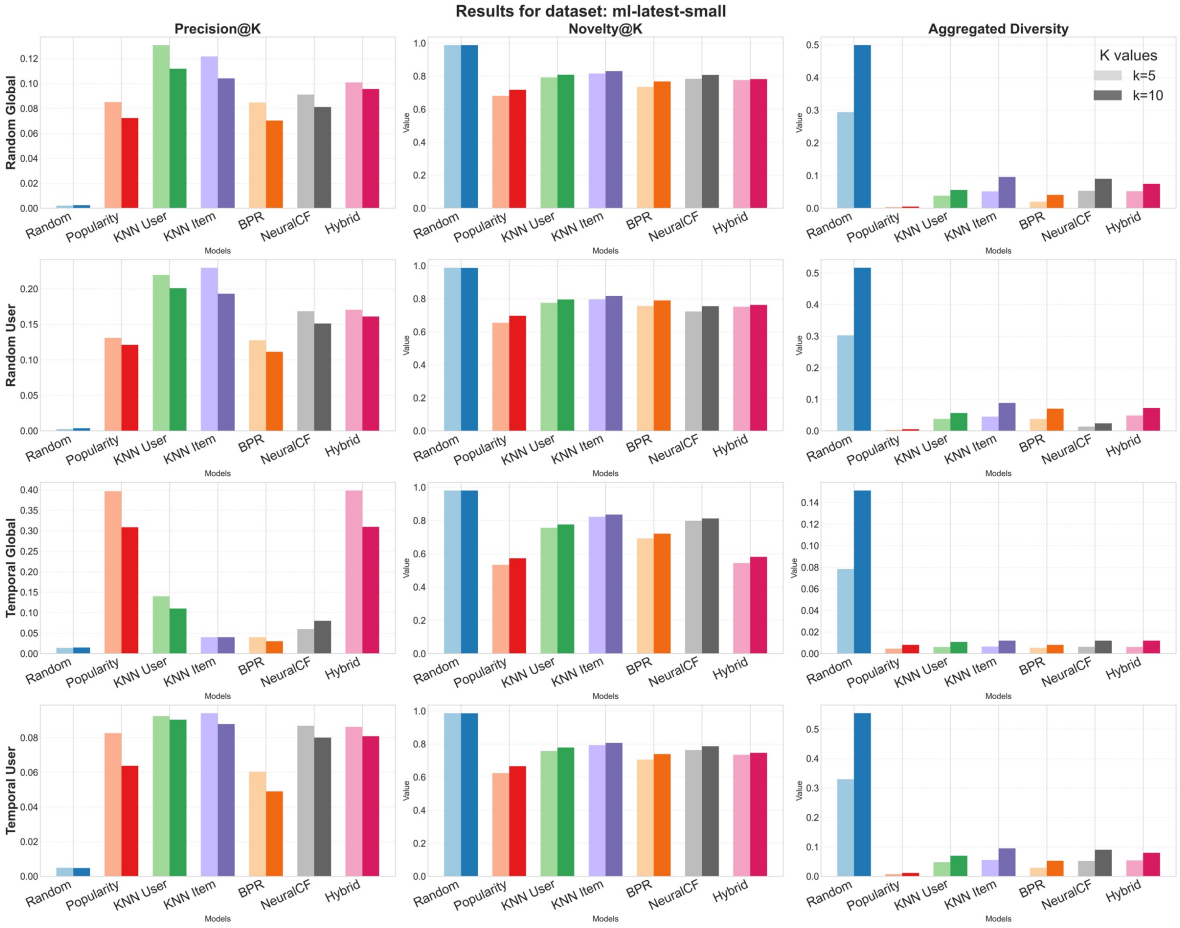


Figura 8. Resultados dataset MovieLens

4.4 RESULTADOS EN FOURSQUARE NUEVA YORK

En los resultados del dataset Foursquare Nueva York (Figura 9) se observa un comportamiento significativamente distinto al de MovieLens, debido principalmente al mayor grado de dispersión de las interacciones. En este caso, el algoritmo Popularity obtiene de forma consistente los mejores resultados en términos de precisión en todas las estrategias de partición consideradas. Este resultado sugiere que, en entornos altamente dispersos, la recomendación basada en popularidad resulta más efectiva que los modelos colaborativos o basados en aprendizaje.

Los modelos KNN Item y BPR presentan un rendimiento notablemente inferior en precisión, lo que puede explicarse por la limitada cantidad de información disponible por usuario. Por su parte, Neural Collaborative Fitering muestra un comportamiento intermedio, situándose generalmente como la segunda mejor alternativa tras Popularity. En tercera posición se encontraría el modelo KNN User. El modelo híbrido, por su parte, mejora el rendimiento de varios modelos base, pero no logra superar al enfoque basado exclusivamente en popularidad, lo que refleja la fuerte dependencia del dataset respecto a este tipo de información global. A diferencia de lo observado en MovieLens, el impacto de la estrategia de partición es menos significativo, manteniéndose una jerarquía de rendimiento similar entre algoritmos en todos los splits.

Respecto a la novedad, se mantiene el patrón esperado. El modelo aleatorio alcanza los valores más elevados, mientras que Popularity presenta los más bajos, al centrarse en los ítems más frecuentes. El resto de los algoritmos, incluidos los colaborativos y el modelo híbrido, ofrecen valores elevados de novedad, lo que indica una mayor capacidad de exploración de catálogo.

Por último, en términos de diversidad agregada, el modelo aleatorio vuelve a destacar claramente, alcanzando los valores más altos en todas las particiones. En el extremo opuesto, el modelo de popularidad presenta valores prácticamente nulos, al concentrar sus recomendaciones en un conjunto muy reducido de ítems. Entre los modelos con capacidad

predictiva, KNN Item y NeuralCF muestran los mejores valores de diversidad, seguidos por KNN User y BPR. El modelo híbrido presenta valores intermedios, superiores a los de popularidad, pero inferiores a los modelos más exploratorios, lo que refuerza su carácter equilibrado.

En conjunto, los resultados ponen de manifiesto que, en entornos altamente dispersos, los modelos simples basados en popularidad pueden superar a enfoques más complejos. Asimismo, se evidencia que el compromiso entre precisión, novedad y diversidad es más pronunciado en este tipo de datasets, y que la capacidad de los modelos híbridos para mejorar el rendimiento está limitada por la calidad y cantidad de los datos disponibles.

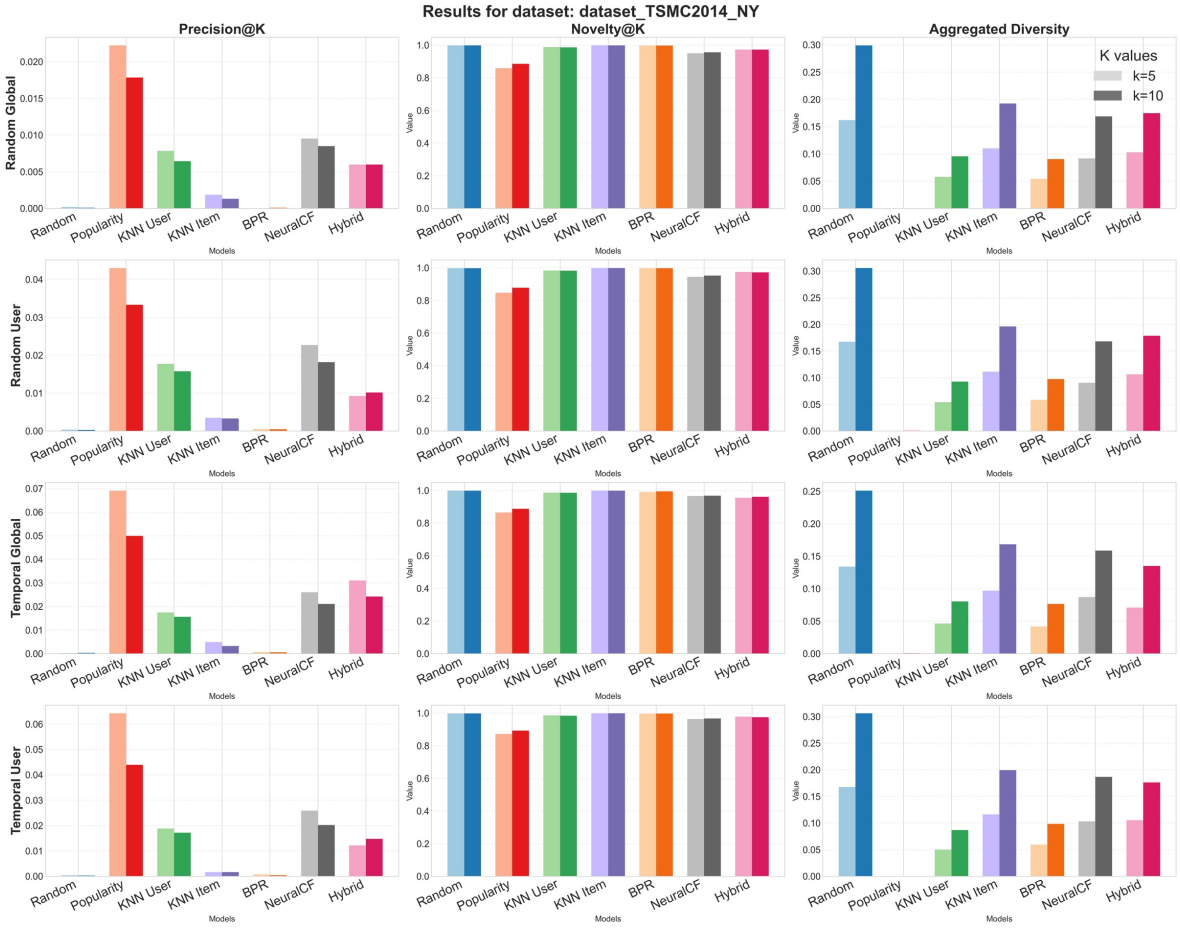


Figura 9. Resultados dataset Foursquare Nueva York

4.5 RESULTADOS EN FOURSQUARE TOKYO

En los resultados del dataset Foursquare Tokyo (Figura 10) se observa un comportamiento muy similar al observado en el caso de Nueva York, debido a su elevada dispersión y a la naturaleza implícita de las interacciones. En este contexto, el algoritmo Popularity vuelve a posicionarse como el modelo con mejor rendimiento en términos de precisión en todas las estrategias de partición analizadas. Este resultado refuerza la idea de que, en entornos con escasa información por usuario, los modelos basados en popularidad resultan más efectivos que los enfoques colaborativos.

El modelo KNN User presenta un rendimiento inferior en comparación con Popularity, mientras que BPR muestra resultados muy limitados en este dataset. Por su parte, NeuralCF alcanza un rendimiento intermedio, situándose en muchos casos como la segunda mejor alternativa y mostrando un comportamiento algo más competitivo que en el dataset de Nueva York. Al igual que en el caso anterior, el modelo híbrido mejora el rendimiento de varios modelos base, pero obtiene mejores resultados que el algoritmo de popularidad. Esto refleja la fuerte dependencia del dataset respecto a este tipo de información global.

En relación con las estrategias de partición, no se observan diferencias significativas en el orden relativo de los algoritmos, manteniéndose un comportamiento consistente entre particiones aleatorias y temporales. No obstante, en algunos casos, se aprecian ligeras variaciones en los valores de precisión, especialmente en los escenarios temporales.

En cuanto a la novedad, el modelo Random presenta los valores más elevados, mientras que Popularity obtiene los más bajos, debido a su tendencia a recomendar los ítems más frecuentes. El resto de los algoritmos muestran valores elevados de novedad, reflejando una mayor capacidad de exploración del catálogo.

Finalmente, en términos de diversidad agregada, Random vuelve a destacar claramente, mientras que Popularity presenta valores prácticamente nulos. Entre los modelos con capacidad predictiva, KNN Item y el modelo híbrido muestran los mejores niveles de diversidad, seguidos por NeuralCF, KNN User y BPR. Estos resultados confirman el

compromiso existente entre precisión y diversidad, especialmente en datasets altamente dispersos como los basados en puntos de interés.

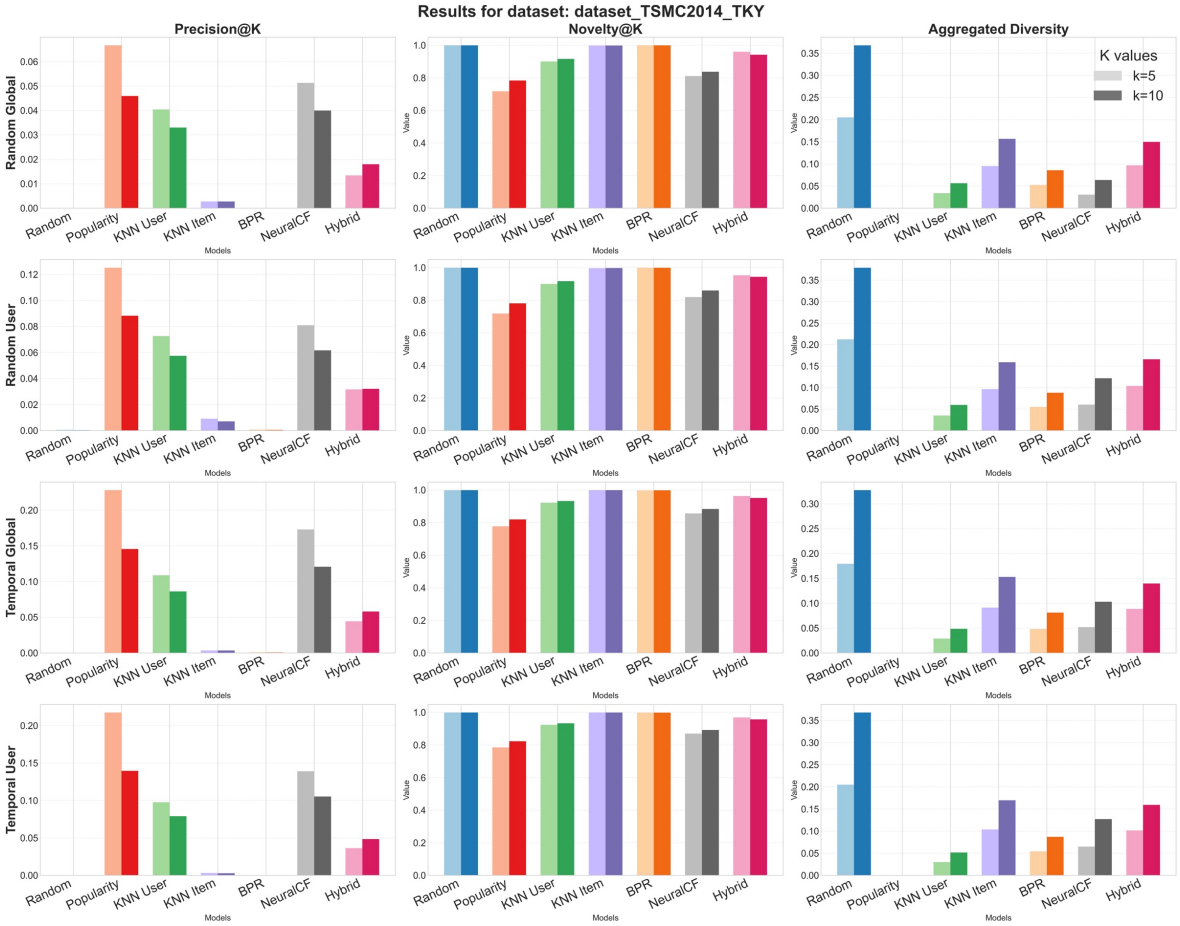


Figura 10. Resultados dataset Foursquare Tokyo

4.6 RESULTADOS EN AMAZON VIDEO GAMES

En los resultados del dataset Amazon Video Games 5core (Figura 11) se ha evaluado el comportamiento de los algoritmos Random, Popularity, KNN User y el modelo híbrido. Este conjunto de datos presenta un nivel de dispersión intermedio en comparación con los datasets anteriores, lo que permite analizar el rendimiento de modelos simples y colaborativos en un contexto más realista.

En términos de precisión, el modelo KNN User obtiene los mejores resultados en todas las particiones, lo que indica que la información disponible es suficiente para explotar similitudes entre usuarios. El modelo híbrido ofrece resultados muy similares al modelo KNN User, manteniéndose consistente en todas las particiones. El modelo Popularity presenta un rendimiento inferior, aunque sigue ofreciendo resultados competitivos en algunos escenarios. Por su parte, Random, obtiene valores de precisión prácticamente nulos.

En cuanto a las estrategias de partición, se observa que las particiones aleatorias tienen a ofrecer mejores resultados en precisión que las temporales, reflejando la mayor dificultad de estos últimos escenarios al requerir la predicción de interacciones futuras.

Respecto a la novedad, el modelo Random alcanza los valores más elevados, aunque el resto de los algoritmos obtienen valores muy cercanos.

Finalmente, en términos de diversidad agregada, Random vuelve a destacar claramente, mientras que Popularity presenta valores muy reducidos. El modelo KNN User se sitúa en una posición intermedia, ofreciendo un mayor grado de diversidad que Popularity sin renunciar completamente a la precisión. El modelo híbrido obtiene unos valores muy similares a los de KNN User, aunque ligeramente inferiores en todas las particiones.

La selección de únicamente los algoritmos Random, Popularity, KNN User y el modelo híbrido en este conjunto de experimentos se debe principalmente a restricciones computacionales asociadas al tamaño del dataset. Amazon Video Games 5core presenta un volumen de datos significativamente mayor en comparación con los datasets anteriores, lo

que incrementa de forma notable el coste de entrenamiento y evaluación de los modelos. En particular, los algoritmos como BPR o Neural Collaborative Filtering, requieren tiempos de ejecución considerablemente más elevados, tanto en la fase de entrenamiento como en la generación de recomendaciones. Dado el carácter exploratorio del análisis y la necesidad de mantener un marco experimental homogéneo, se optó por utilizar únicamente modelos representativos de distinta complejidad, que permiten cubrir distintos enfoques de recomendación sin comprometer la viabilidad computacional del estudio. Esta decisión permite, además, centrarse en el análisis comparativo de comportamientos fundamentales sin que el coste de cómputo limite la reproductibilidad de los experimentos.

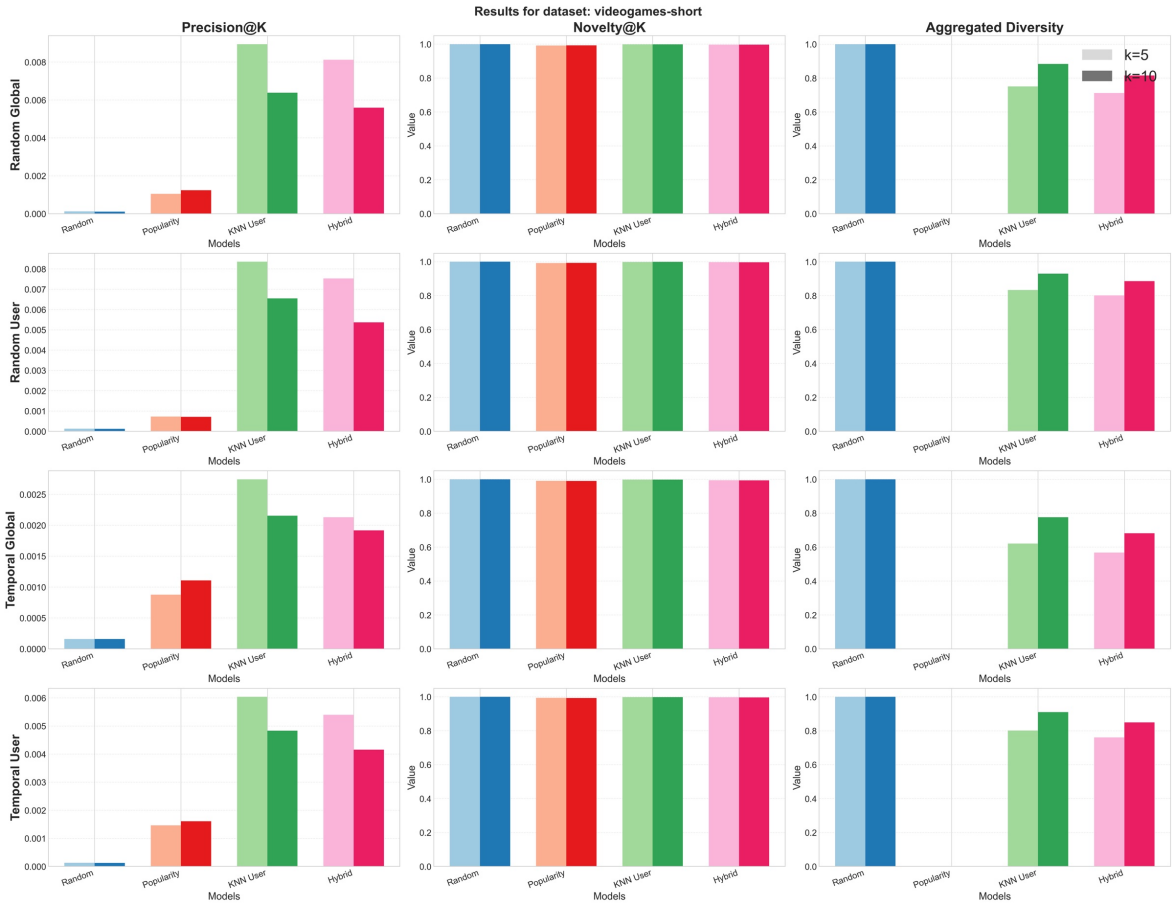


Figura 11. Resultados dataset Amazon Video Games

Capítulo 5. CONCLUSIONES Y TRABAJOS FUTUROS

En este trabajo se ha abordado el problema de la recomendación desde una perspectiva experimental, analizando el comportamiento de distintos algoritmos en escenarios de validación con diferentes estrategias de partición de datos. Se han implementado y evaluado modelos de recomendación basados en enfoques simples, colaborativos y basados en aprendizaje, con el objetivo de estudiar su rendimiento en términos de precisión, novedad y diversidad.

A lo largo del proyecto se han cumplido los objetivos planteados inicialmente, en particular la implementación de un framework experimental que permite evaluar de forma homogénea distintos algoritmos de recomendación bajo múltiples configuraciones de evaluación. Asimismo, se ha conseguido analizar de forma comparativa el impacto del tipo de partición de los datos y del dominio del dataset en el rendimiento de los modelos identificando patrones relevantes como la influencia de la dispersión de los datos o el compromiso entre precisión y diversidad.

Los resultados obtenidos muestran que no existe un único algoritmo óptimo para todos los escenarios, sino que el rendimiento depende fuertemente de las características del dataset y del contexto de evaluación. En particular, los modelos simples como Popularity o KNN pueden ofrecer resultados competitivos frente a modelos más complejos en determinados entornos, especialmente cuando los datos presentan alta dispersión o limitada información por usuario.

Asimismo, la incorporación de modelos híbridos ha permitido observar que la combinación de algoritmos puede mejorar la robustez del sistema, ofreciendo un rendimiento más estable en distintos escenarios. Sin embargo, los resultados también evidencian que no existe una configuración híbrida universalmente óptima, sino que las combinaciones de algoritmos y sus pesos dependen de forma significativa del dataset y de la estrategia de partición empleada.

En cuanto a las principales aportaciones del trabajo, destaca la comparación sistemática de distintos algoritmos de recomendación bajo un marco experimental unificado, así como el análisis del impacto de distintas estrategias de partición en varios dominios de datos reales. Además, se ha evidenciado la importancia de considerar no solo la precisión, sino también métricas relacionadas con la novedad y la diversidad para obtener una evaluación más completa de los sistemas de recomendación.

En relación con los trabajos futuros, una posible extensión de este estudio consistiría en ampliar significativamente el número de datasets utilizados, incorporando dominios más variados y de mayor escala, lo que permitiría validar con mayor robustez la generalización de los resultados obtenidos. Asimismo, sería interesante incluir un mayor número de algoritmos, especialmente modelos basados en Deep learning, optimizando su eficiencia computacional para su aplicación en datasets de gran tamaño.

Otra línea de trabajo futura relevantes sería la incorporación de información contextual en el proceso de recomendación. En muchos escenarios reales, las preferencias de usuarios no dependen únicamente de interacciones pasadas, sino también de factores como el momento temporal, la secuencialidad de las acciones o el contexto en el que se produce la interacción. En particular, en dominios como la recomendación de puntos de interés, resulta especialmente relevante considerar la componente geográfica, incorporando información sobre la localización del usuario y la proximidad entre ítems. La integración de este tipo de información permitiría desarrollar sistemas de recomendación más precisos y adaptados al contexto real de uso,

Finalmente, otra posible línea de trabajo consistiría en la incorporación de evaluación online o simulaciones más realistas de interacción usuario-sistema, con el objetivo de complementar las métricas offline utilizadas en este estudio.

Capítulo 6. BIBLIOGRAFÍA

[1] F. Ricci, L. Rokach y B.Shapira (eds.), *Recommender Systems Handbook*, 3rd ed., Springer, 2022. https://doi.org/10.1007/978-1-0716-2197-4

[2] G. Adomavicius y A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005. https://doi.org/10.1109/TKDE.2005.99

[3] J. L. Herlocker, J. A. Konstan, L. G. Terveen y J. T. Riedl, "Evaluating Collaborative Filtering Recommender Systems," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5–53, 2004. https://doi.org/10.1145/963770.963772

[4] Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *IEEE Computer*, vol. 42, no. 8, pp. 42–49, 2009. https://doi.org/10.1109/MC.2009.263

[5] M. D. Ekstrand, J. T. Riedl and J. A. Konstan, "Collaborative Filtering Recommender Systems", *Foundations and Trends in Human-Computer Interaction,* vol. 4, no. 2, pp. 81-173, 2010 https://doi.org/10.1561/1100000009

[6] S. Rendle, L. Zhang and Y. Koren, "On the Difficulty of Evaluating Baselines: A Study on Recommender Systems", arXiv preprint arXiv:1905.01395, 2019. https://arxiv.org/abs/1905.01395

[7] N. Lathia, S. Hailes, L. Capra and X. Amatriain, "Temporal Diversity in Recommender Systems," in *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Geneva, Switzerland, 2010, pp. 210–217. https://doi.org/10.1145/1835449.1835486

[8] E. Pariser, *The Filter Bubble: What the Internet Is Hiding from You*, Penguin Press, 2011. https://hci.stanford.edu/courses/cs047n/readings/The_Filter_Bubble.pdf

[9] S. M. McNee, J. Riedl, and J. A. Konstan, "Being accurate is not enough: How accuracy metrics have hurt recommender systems," in *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, Montréal, Québec, Canada, Apr. 2006, pp. 1097–1101. https://doi.org/10.1145/1125451.1125659

[10] B. M. Marlin and R. S. Zemel, "Collaborative prediction and ranking with non-random missing data," in *Proc. ACM Conf. Recommender Systems (RecSys)*, 2009. https://www.cs.toronto.edu/~zemel/documents/acmrec2009-MarlinZemel.pdf

[11] N. Lathia, S. Hailes, and L. Capra, "Temporal collaborative filtering with adaptive neighbourhoods," *Proceedings of the 2008 ACM Conference on Recommender Systems (RecSys '08)*, 2008. https://doi.org/10.1145/1571941.1572133

[12] S. Rendle, C. Freudenthaler, Z. Gantner and L. Schmidt-Thieme, "BPR: Bayesian Personalized Ranking from Implicit Feedback," in Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI), 2009. https://arxiv.org/abs/1205.2618

[13] D. Billsus and M. J. Pazzani, "A Hybrid User Model for News Story Classification," Department of Information and Computer Science, University of California, Irvine, CA, USA, 1999. https://doi.org/10.1007/978-3-7091-2490-1_10

[14] O. Barkan and N. Koenigstein, "Item2Vec: Neural Item Embedding for Collaborative Filtering," Microsoft Research and Tel Aviv University, 2016. https://arxiv.org/abs/1603.04259

ANEXO I

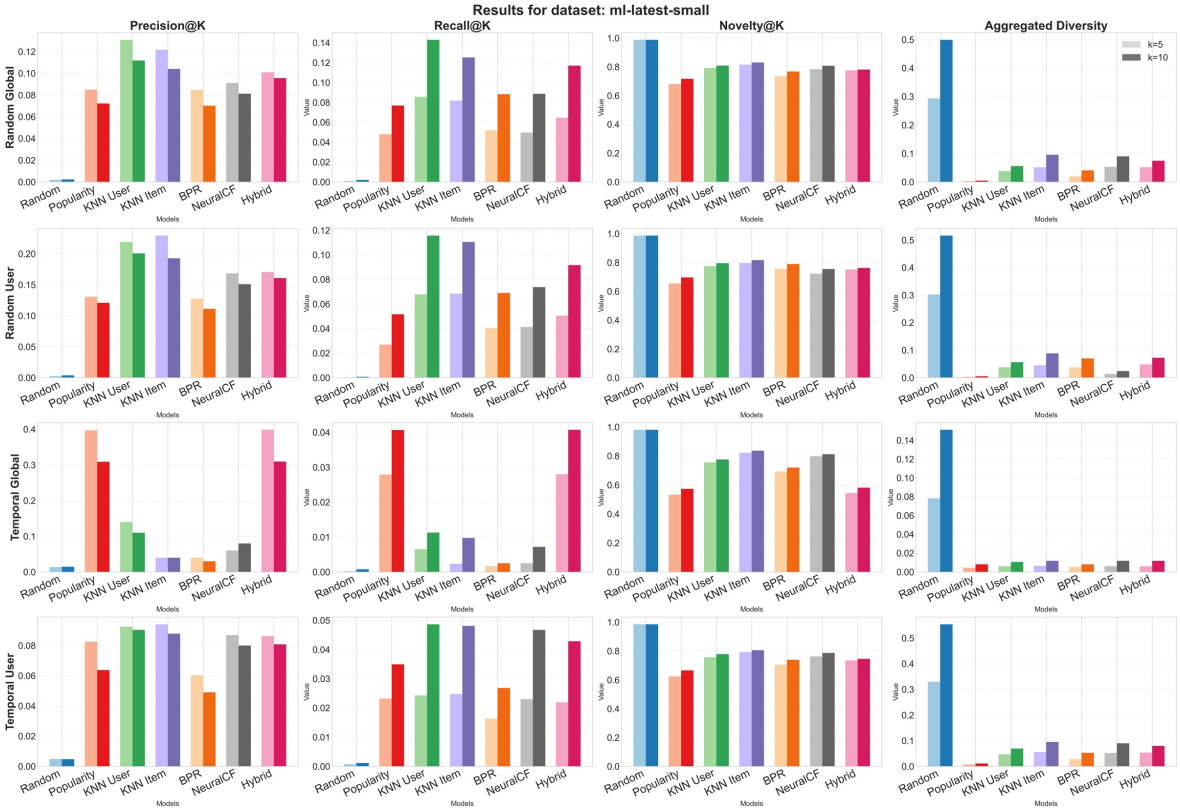


Figura 12. Resultados con Recall@K dataset MovieLens

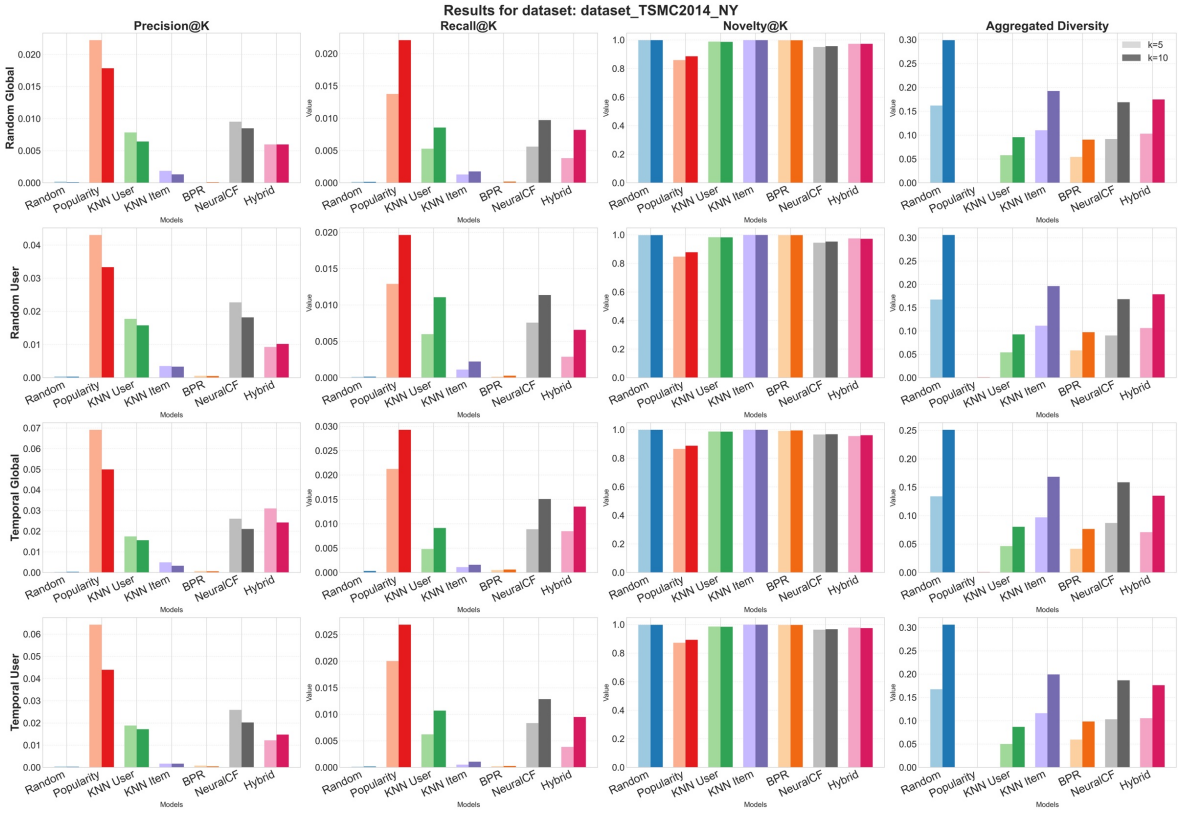


Figura 13. Resultados con Recall@K dataset Foursquare Nueva York

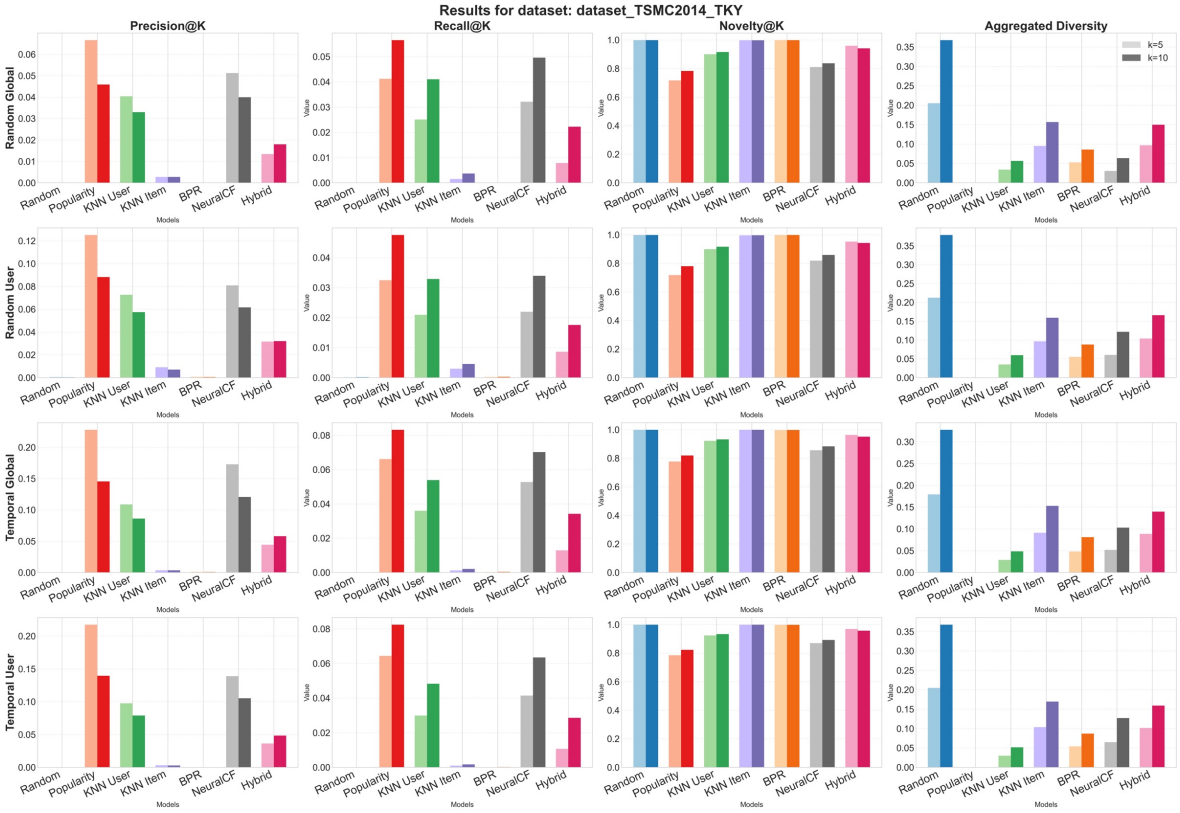


Figura 14. Resultados con Recall@K dataset Foursquare Tokyo

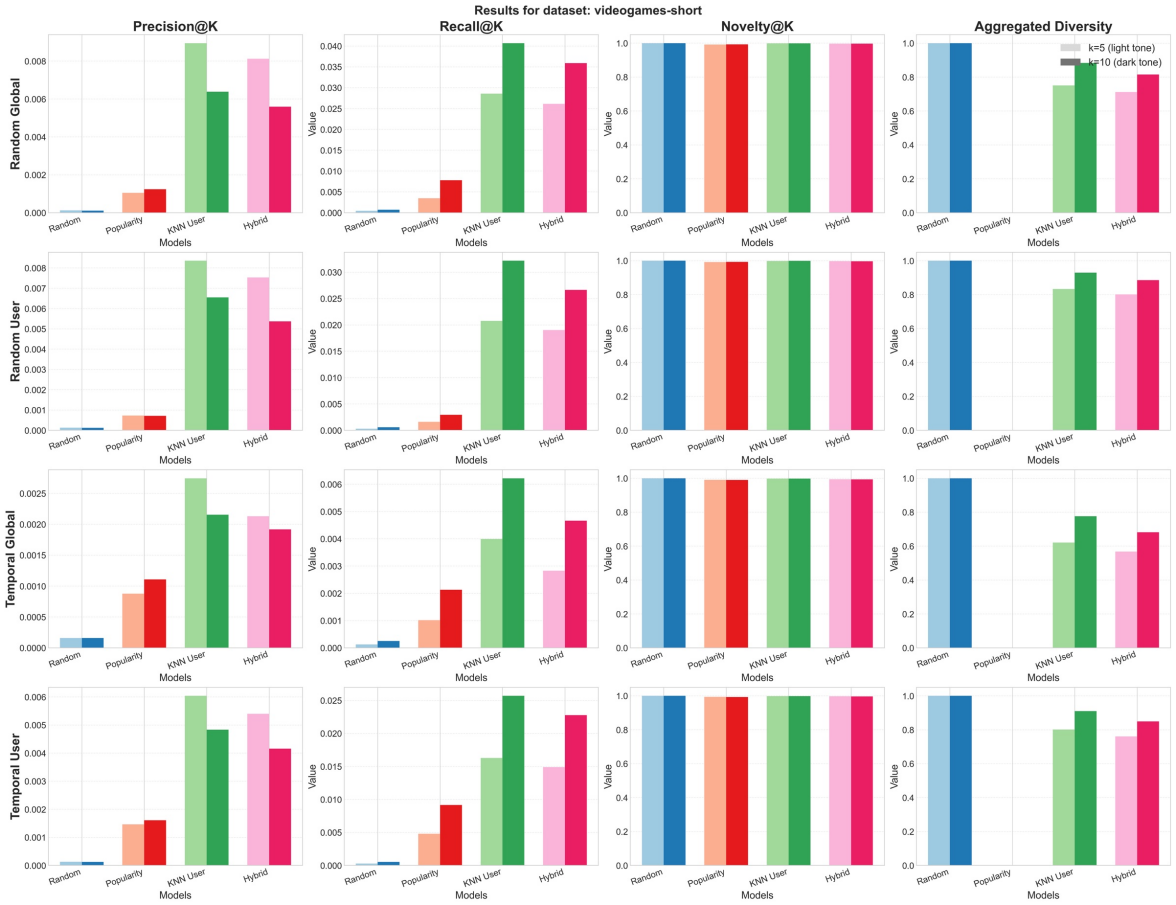


Figura 15. Resultados con Recall@K dataset Amazon Video Games

ANEXO II

Dataset	Partición	Modelo A	Modelo B	Pesos
MovieLens	Random Global	KNN User	KKN Item	[0.5, 0.5]
MovieLens	Random User	KNN User	KKN Item	[0.4, 0.6]
MovieLens	Temporal Global	KNN User	Popularity	[0.5, 0.5]
MovieLens	Temporal User	KNN User	KKN Item	[0.3, 0.7]
Foursquare Nueva York	Random Global	KNN User	Popularity	[0.3, 0.7]
Foursquare Nueva York	Random User	KNN User	Popularity	[0.3, 0.7]
Foursquare Nueva York	Temporal Global	KNN User	Popularity	[0.3, 0.7]
Foursquare Nueva York	Temporal User	KNN User	Popularity	[0.3, 0.7]
Foursquare Tokyo	Random Global	Popularity	NeuralCF	[0.7, 0.3]
Foursquare Tokyo	Random User	KNN User	Popularity	[0.3, 0.7]
Foursquare Tokyo	Temporal Global	Popularity	NeuralCF	[0.7, 0.3]
Foursquare Tokyo	Temporal User	Popularity	NeuralCF	[0.7, 0.3]
Amazon Video Games	Random Global	KNN User	Popularity	[0.7, 0.3]
Amazon Video Games	Random User	KNN User	Popularity	[0.7, 0.3]
Amazon Video Games	Temporal Global	KNN User	Popularity	[0.7, 0.3]
Amazon Video Games	Temporal User	KNN User	Popularity	[0.7, 0.3]

Tabla 3. Configuración óptima del modelo híbrido para cada dataset y partición