Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

Dataset Analysis

1 Training Set

1.1 Contenido

El conjunto de datos de entrenamiento consta de las columnas userID, itemID, styleID y rating, que contienen las siguientes especificaciones:

Column	Non-Null Count	Dtype	min	max
userID	35534	int64	1	8320
itemID	35534	int64	175	77207
styleID	35534	int64	1	27797
rating	35534	float64	0	5

Table 1: Contenido del Training Set

Para cada columna, realizamos un gráfico que muestra la distribución frecuencial de los datos:

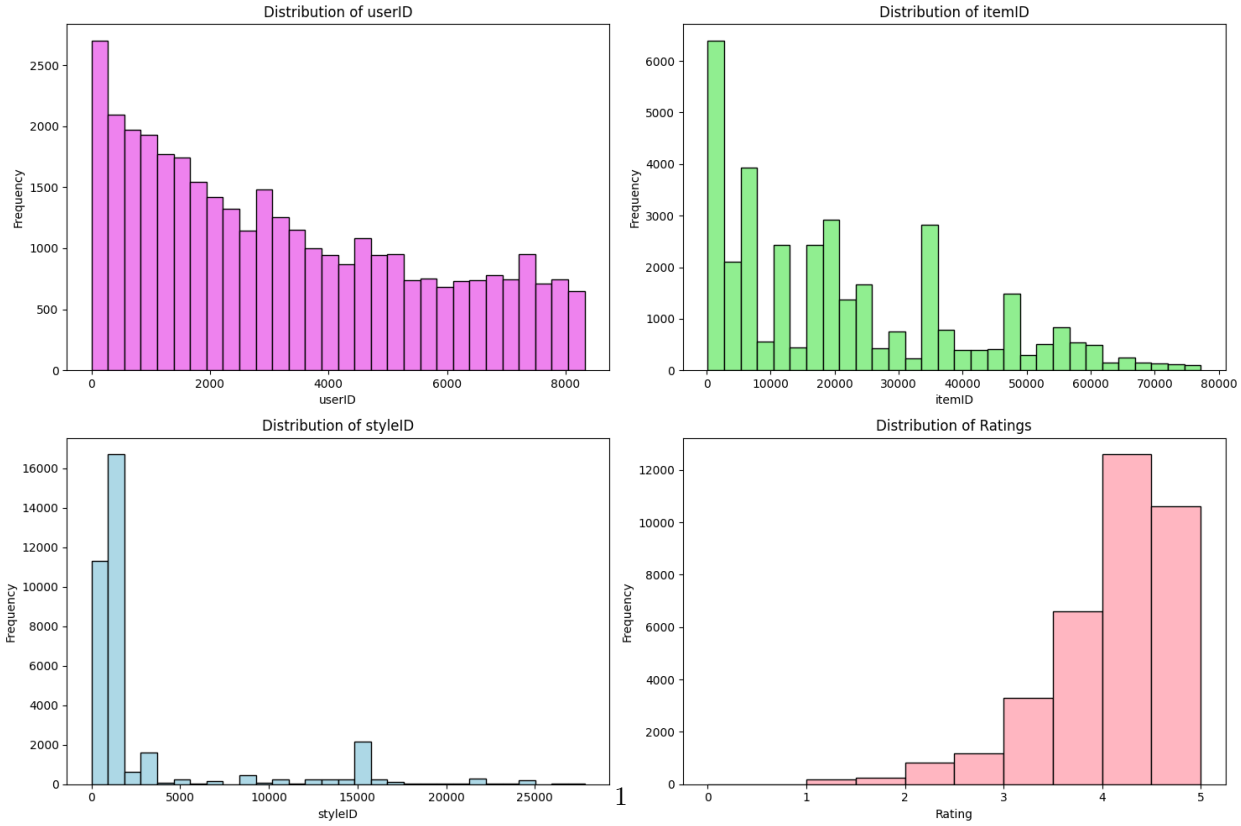

Figure 1: Distribución de las columnas del training set

La distribución de los IDs de usuario está bastante dispersa, con una concentración de usuarios que tienen IDs en el rango más bajo (más cercanos a 1). En cambio, hay una disminución notable en la frecuencia a medida que aumenta el ID de los ítems, aunque ciertos rangos más altos también tienen barras altas, lo que indica ítems populares.

Los IDs de estilo están fuertemente sesgados hacia el extremo inferior, con una gran cantidad de ítems que tienen IDs de estilo por debajo de 5,000. Más allá de ese valor, hay menos ítems.

Las calificaciones se concentran principalmente en los valores más altos, siendo la mayoría de las calificaciones de 3.5, 4.0 y 4.5. Hay muy pocas calificaciones bajas (de 0.0 a 2.0), lo que indica que la mayoría de los usuarios evalúa los ítems de manera positiva.

1.2 Correlaciones

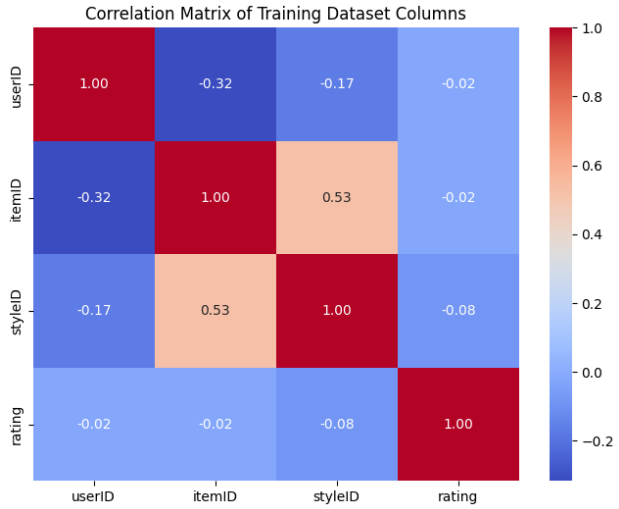

Figure 2: Correlation Matrix

La correlación más fuerte es entre itemID y styleID (0.53), lo que sugiere que los ítems y los estilos están algo relacionados, posiblemente porque ciertos estilos son más populares entre ítems específicos. Aparte de este análisis, podemos concluir que las demás columnas tienen correlaciones muy débiles entre sí, lo que indica que los ratings, usuarios, ítems y los identificadores de estilo son en gran medida independientes.

Por otro lado, podemos analizar la cantidad de data y las relaciones entre las columnas:

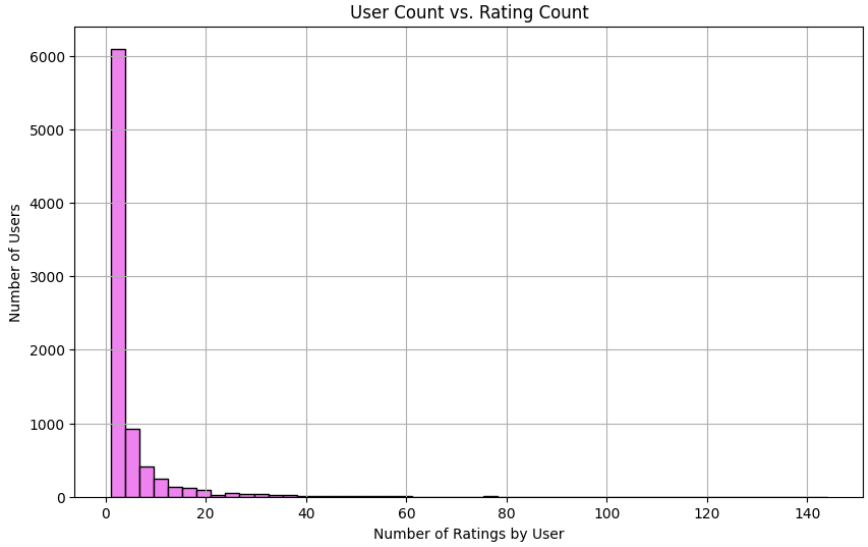

Figure 3: User count vs rating count graph

La mayoría de los usuarios han hecho solo una pequeña cantidad de ratings, con muchos de ellos contribuyendo con menos de 10 ratings. Un pequeño número de usuarios ha realizado una gran cantidad de ratings, pero estos usuarios son poco comunes.

Tenemos un gráfico similar para la cantidad de items y ratings

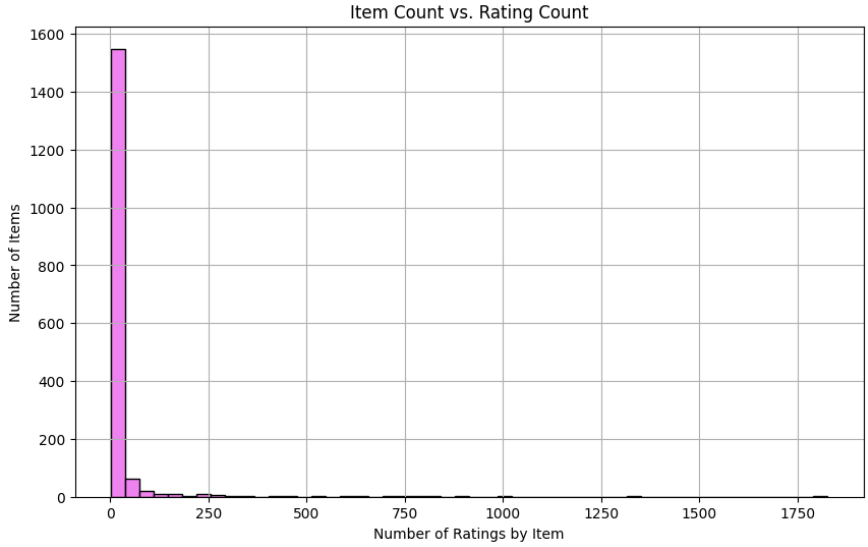

Figure 4: Item count vs rating count graph

Esto significa que la mayoría de los ítems tienen una pequeña cantidad de calificaciones, pero en este caso, "pequeña" aún se refiere a menos de 250 calificaciones, lo cual en realidad es una cantidad significativa.

2 Validation Set

2.1 Contenido

El conjunto de datos de validación consta de las columnas userID, itemID, styleID y rating, que contienen las siguientes especificaciones:

Column	Non-Null Count	Dtype	min	max
userID	8845	int64	1	8300
itemID	8845	int64	175	76999
styleID	8845	int64	1	26990
rating	8845	float64	0	5

Table 2: Validation Set Content

Para cada columna, realizamos un gráfico que muestra la distribución frecuencial de los datos:

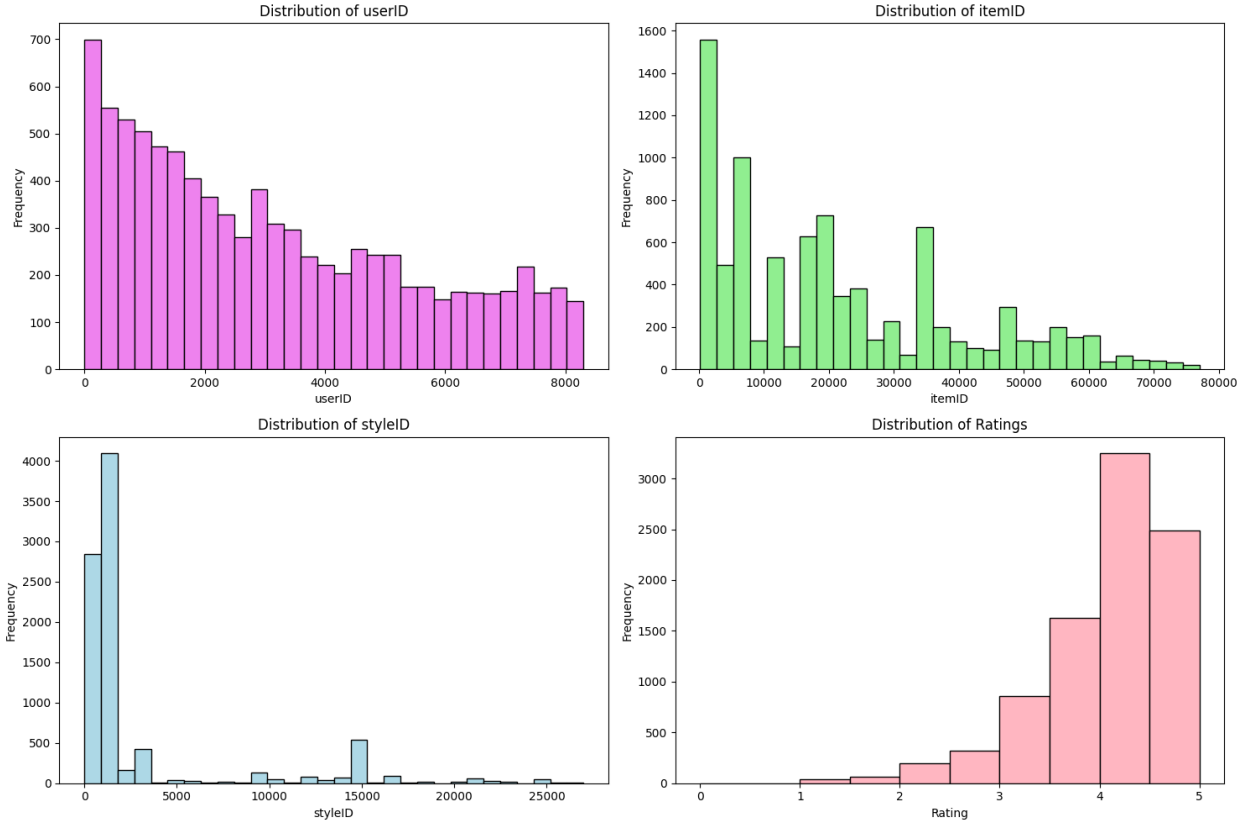

Figure 5: Distribution of the columns of the validation set

A primera vista, podemos decir que la diferencia entre el conjunto de entrenamiento y el conjunto de validación es mínima, al menos en términos de la distribución de las columnas. Algo a destacar es que las calificaciones en el conjunto de validación son ligeramente más positivas.

2.2 Correlaciones

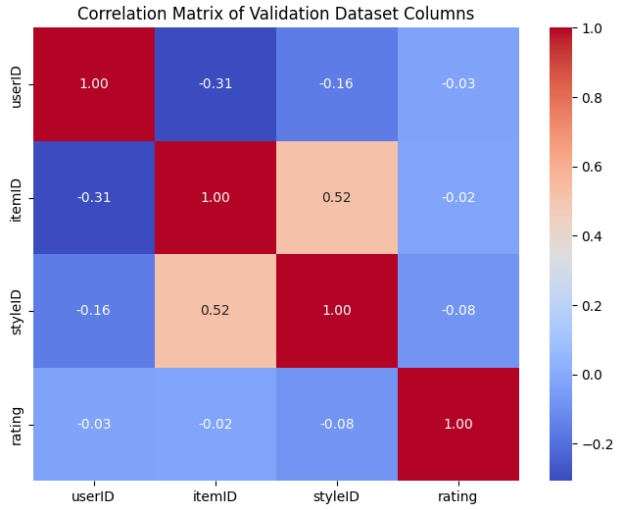

Figure 6: Correlation Matrix Validation

La correlación más notable es entre itemID y styleID (0.52), lo que indica que ciertos estilos tienden a asociarse con ítems específicos. Además, hay una correlación negativa moderada entre userID e itemID (-0.31), lo que podría reflejar patrones de preferencia divergentes entre usuarios y los ítems que califican.

Con respecto a la cantidad de datos y realaciones con el rating, no difiere mucho del training set:

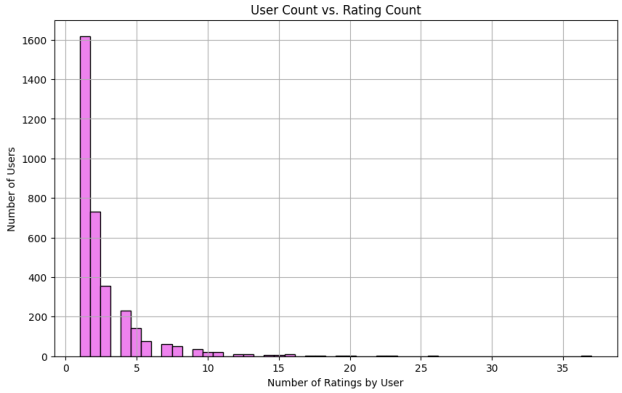

Figure 7: User count vs rating count validation graph

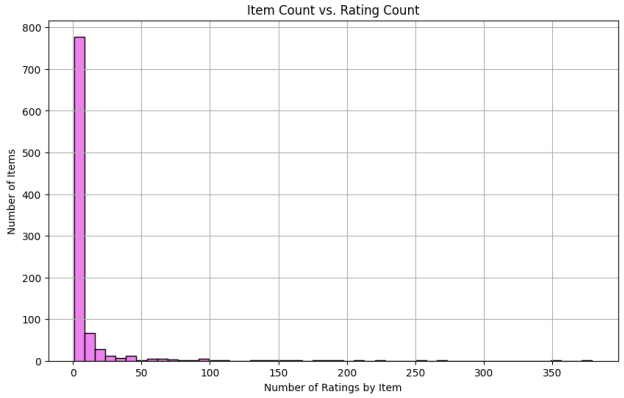

Figure 8: Item count vs rating count validation graph

3 Training y Validation, diferencias y similitudes

Statistic	Training	Validation
Number of Users	8320	3410
Number of Items	1692	942
Total Ratings	35534	8845
Average Number of Ratings per User	4.2709	2.5938
Average Number of Ratings per Item	21.0011	9.3895
Average Rating	3.868971	3.849576
Rating Standard Deviation	0.714726	0.7039
Highest Number of Ratings by a User	144	37
Highest Number of Ratings for an Item	1827	379
Density (%)	0.2524	0.2753

Table 3: Training and Validation

La tabla compara el conjunto de training y validation en términos de varias estadísticas clave. El conjunto de entrenamiento incluye más usuarios (8320 frente a 3410) y más ítems (1692 frente a 942), lo que resulta en un mayor número de ratings totales (35534 frente a 8845). El número promedio de calificaciones por usuario y por ítem también es mayor en el conjunto de entrenamiento, lo que refleja un mayor volumen de interacciones. Sin embargo, tanto la media de ratings como la desviación estándar de los ratings son muy similares entre ambos conjuntos, lo que indica que las distribuciones de las calificaciones en training y validation son comparables. Finalmente, la densidad del conjunto de validación (0.2753%) es ligeramente mayor que la del conjunto de entrenamiento (0.2524%), lo que sugiere una mayor proporción de interacciones observadas en el conjunto de validación respecto al total de usuarios e ítems.

Análisis de Modelos

1 Modelos para predicción de Ratings

1.1 User-based collaborative filtering

Análisis de sensibilidad en base a cantidad K de vecinos, con métrica de error de predicción RMSE:

- Utilizando Pearson

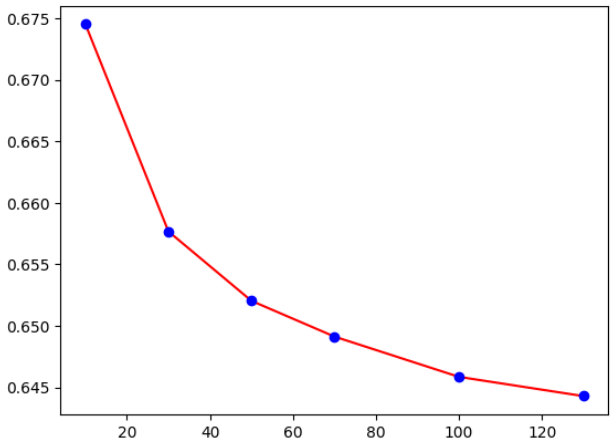

Figure 9: RMSE vs Valores K, con similaridad Pearson

- Utilizando Coseno

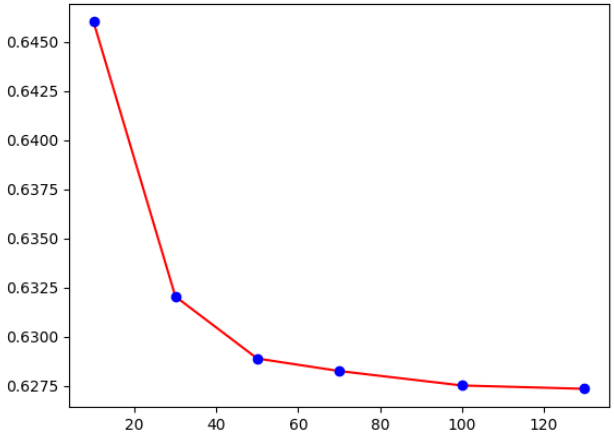

Figure 10: RMSE vs Valores K, con similaridad Coseno

Los gráficos muestran cómo el error de predicción RMSE disminuye conforme se incrementa el número de vecinos K, utilizando las similitudes Pearson y Coseno. En ambos casos, el RMSE desciende de manera notable entre K=10 y K=70, pero se estabiliza con mejoras menores para valores mayores de K. El método de similitud Coseno comienza con un RMSE más bajo en K=10 (0.645 frente a 0.675 para Pearson) y, en general, logra un mejor rendimiento con un RMSE mínimo de 0.625 en K=130, mientras que Pearson alcanza

0.645. Aunque ambos métodos se benefician al aumentar K, Coseno ofrece una ligera ventaja en términos de precisión.

1.2 Item-based collaborative filtering

Análisis de sensibilidad en base a cantidad K de vecinos, con métrica de error de predicción RMSE:

- Utilizando Pearson

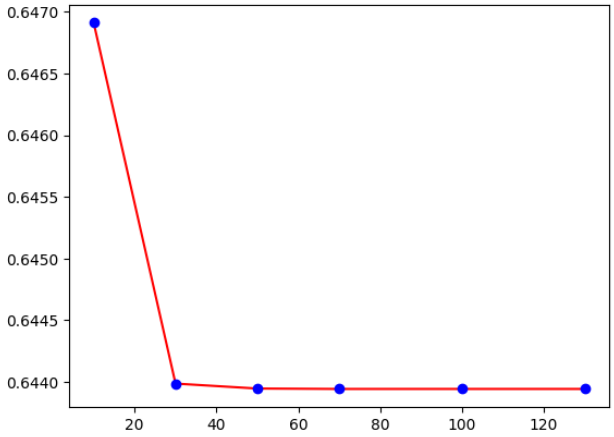

Figure 11: RMSE vs Valores K, con similaridad Pearson

- Utilizando Coseno

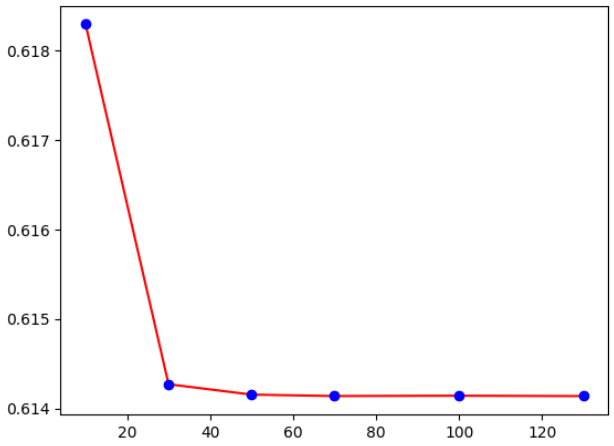

Figure 12: RMSE vs Valores K, con similaridad Coseno

En estos gráficos, que muestran el comportamiento del RMSE en función de la cantidad de vecinos K utilizando similitudes Pearson y Coseno, se observa una notable disminución del RMSE entre K=10 y K=20 en ambos métodos. En el caso de Pearson, el RMSE baja rápidamente desde 0.647 hasta estabilizarse en torno a 0.644 a partir de K=20, sin mostrar mejoras adicionales con más vecinos. De manera similar, con Coseno, el RMSE desciende de 0.618 a 0.614 en el mismo rango de K, manteniéndose casi constante después de K=20. Esto sugiere que, en el filtrado basado en ítems, incluir más de 20 vecinos no proporciona mejoras significativas en la precisión, ya que el error se estabiliza rápidamente.

1.3 FunkSVD

Análisis de sensibilidad en base a cantidad de factores latentes, con métrica de error de predicción RMSE:

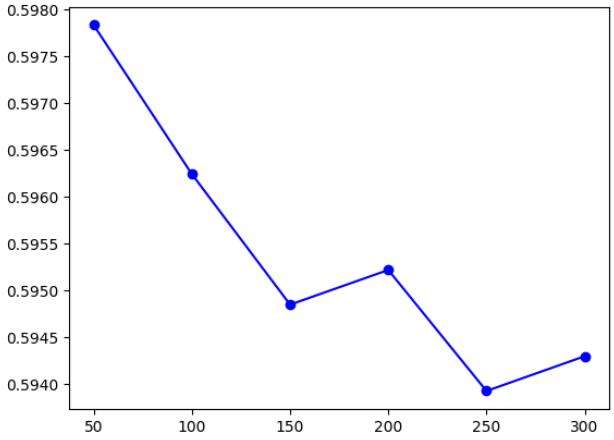

Figure 13: RMSE vs factores latentes

El gráfico muestra que al aumentar los factores latentes en el modelo FunkSVD, el RMSE disminuye, lo que indica una mejora en la precisión de las predicciones. Inicialmente, el RMSE baja de aproximadamente 0.598 con 50 factores a 0.594 con 250 factores, aunque la mejora se vuelve marginal después de los 300 factores. Esto sugiere que, si bien más factores ayudan al modelo a capturar mejor las relaciones, después de cierto punto el beneficio es pequeño y puede no justificar el mayor costo computacional.

Análisis de sensibilidad con RMSE, en base a hiperparámetros, learning rate y regularizacion (λ):

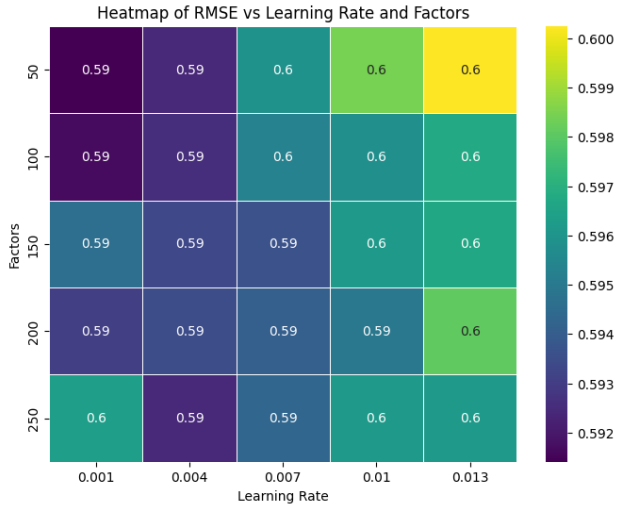

Figure 14: Heapmap RMSE vs Learning Rate and Factors

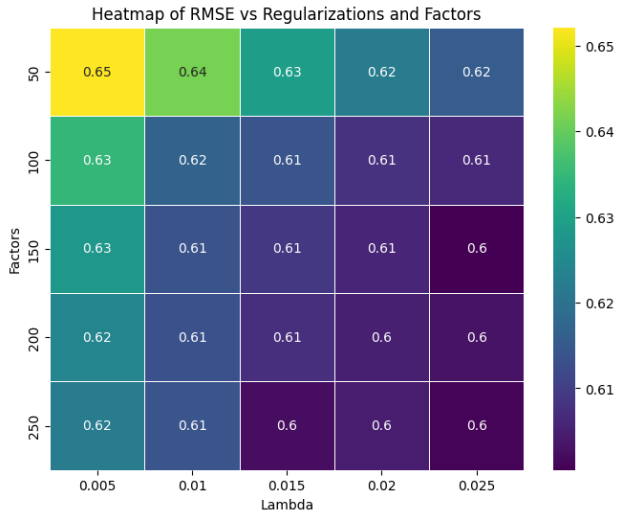

Figure 15: Heapmap RMSE vs Regularizations and Factors

En el primer heatmap, que relaciona el learning rate y los factores latentes con el RMSE, se observa que los valores de RMSE tienden a mejorar (disminuir) conforme aumentan los factores latentes, especialmente cuando el learning rate está entre 0.007 y 0.01. Sin embargo, para valores más altos de learning rate (mayor a 0.01), el RMSE empeora ligeramente. Esto sugiere que un learning rate moderado y un mayor número de factores latentes tienden a producir un mejor rendimiento.

En el segundo heatmap, que analiza la relación entre la regularización (lambda) y los factores latentes con respecto al RMSE, se observa que al aumentar la regularización hasta aproximadamente 0.015, el RMSE disminuye, pero incrementos mayores en la regularización no ofrecen mejoras significativas. Además, el número de factores latentes tiene un impacto constante en la reducción del RMSE, donde el rendimiento mejora con más factores. El valor óptimo parece darse con regularización alrededor de 0.02 y con más de 200 factores.

A pesar del análisis de los heatmap, la relación del RMSE entre los 3 hiperparámetros, evaluando en sus distintas combinaciones, se encontró que la mejor combinación de hiperparámetros es: 'Learning Rate': 0.001, 'Regularization': 0.025, 'Factors': 50, 'RMSE': 0.5932867505149496

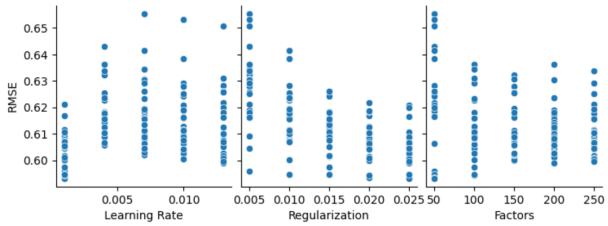

Figure 16: Enter Caption

1.4 SVD++

Análisis de sensibilidad en base a cantidad de factores latentes, con métrica de error de predicción RMSE:

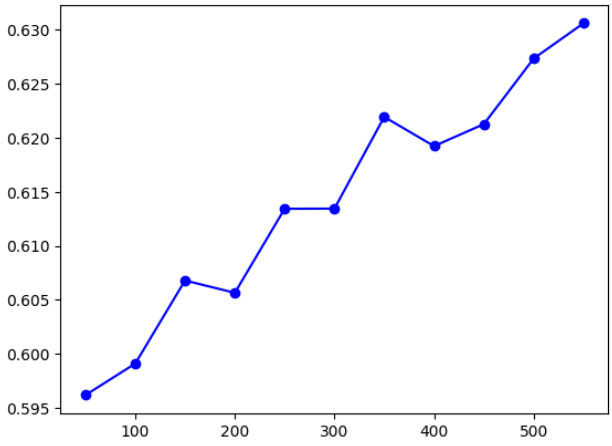

Figure 17: Enter Caption

1.5 Comparación de métricas entre modelos para rating

Considerando el menor RMSE para cada modelo, se obtuvieron los siguientes datos:

Modelo	RMSE	Variable K	Similaridad	Factor Latente	Learning Rate	λ
uKNN	0.6273	130	Coseno	X	X	X
iKNN	0.6141	100	Coseno	X	X	X
FunkSVD	0.5932	X	X	50	0.001	0.025
SVD++	0.594	X	X	50	0.001	0.025

Table 4: Training Set Content

A partir de la métrica RMSE de cada modelo, se determina utilizar el modelo FunkSVD con 50 factores latentes, learning rate 0.001 y λ 0.025, para la predicción de ratings del archivo, debido a que es el que posee el RMSE más bajo, es decir, el que posee menor error y mayor precisión.

2 Modelos para listas de recomendación

2.1 Item-based collaborative filtering

Para este modelo se realizó un análisis en torno a su variable K, cantidad de vecinos más cercanos. Bajo esta base, se analisó la sensibilidad de las métricas MAP@10 y nDCG@10 ante la variabilidad de K. Lamentablemente, y por temas de la duración del modelo, no se logró correr el código el analisis de sensibilidad.

2.2 FunkSVD

Para este modelo se probó el efecto en las métricas MAP y nDCG para tres hiperparámetros: Factores, Learning Rate y Regularización (λ). Para visualizar este efecto, se realizaron varias pruebas cambiando dos de estos valores y manteniendo constante el tercero.

En primer lugar, se probó la sensibilidad para un cambio de factores y learning rate, obteniendo los resultados presentados en la Figura 18 y 19. Se puede observar de estas figuras que el mejor valor de MAP@10 y nDCG@10 se da para un valor de 30 factores y 0.001 de learning rate.

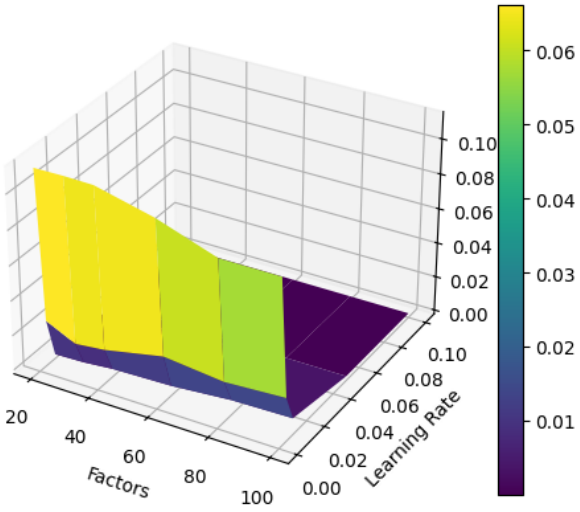

Figure 18: MAP@10 en FunkSVD con variación de factores y learning rate

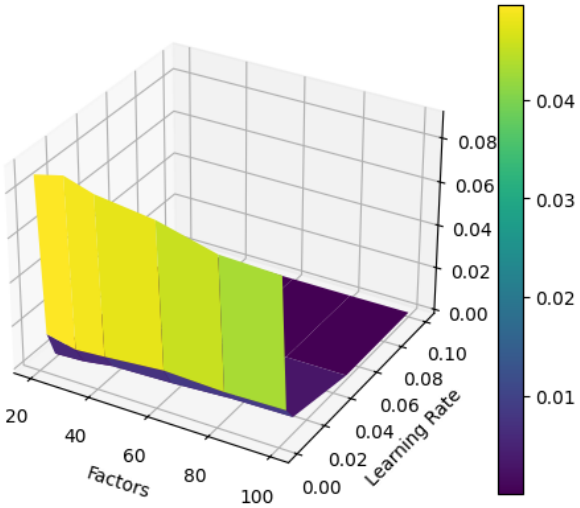

Figure 19: nDCG@10 en FunkSVD con variación de factores y learning rate

En segundo lugar, se probó la sensibilidad para un cambio de factores y regularización, obteniendo los resultados presentados en la Figura 20 y 21. De estas figuras, se aprecia que el mejor desempeño se obtiene para un modelo con 80 factores y 0.001 de reagularización.

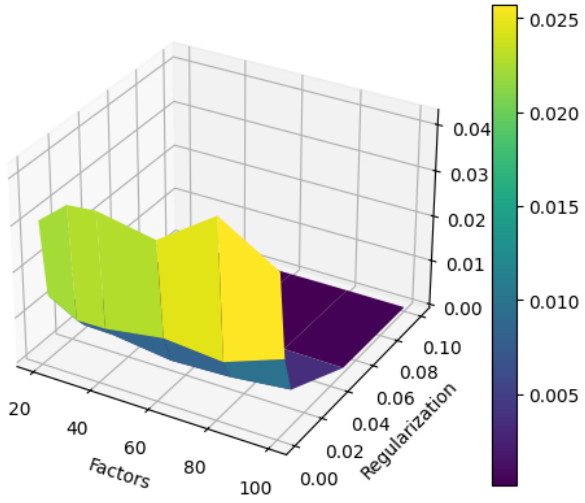

Figure 20: MAP@10 en FunkSVD con variación de factores y regularización

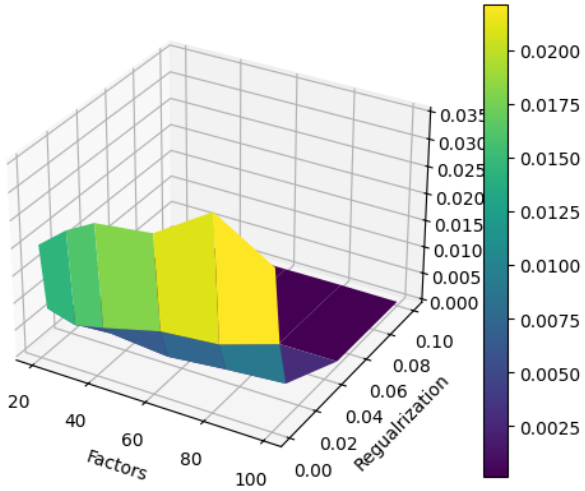

Figure 21: nDCG@10 en FunkSVD con variación de factores y regularización

Finalmente, se probó la sensibilidad para un cambio en learning rate y regularización, obteniendo los resultados presentados en la Figura 18 y 19. De estas, en ambas se aprecia que el mejor desempeño se alcanza cuando el learning rate y la regularización tienen un valor de 0.001.

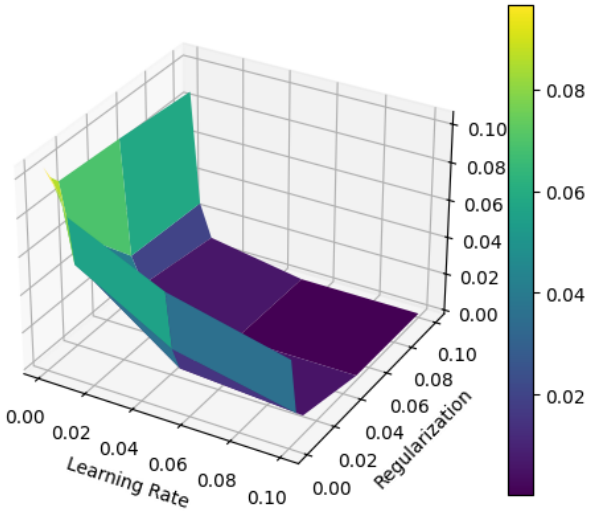

Figure 22: MAP@10 en FunkSVD con variación de learning rate y regularización

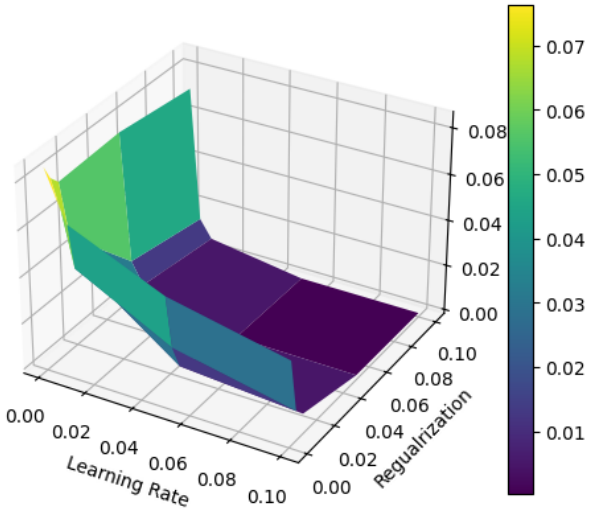

Figure 23: MAP@10 en FunkSVD con variación de learning rate y regularización

En base a lo mencionado anteriormente, se decidió entrenar un nuevo modelo con 30 factores, learning rate de 0.001 y regularización de 0.001, el cual es analizado en la sección 2.6 y en la sección 3.

2.3 BPR

Se realizó un análisis de sensibilidad en base a los hiperparámetros factors e iterations. Los resultados son presentados en las Figuras 24 y 25.

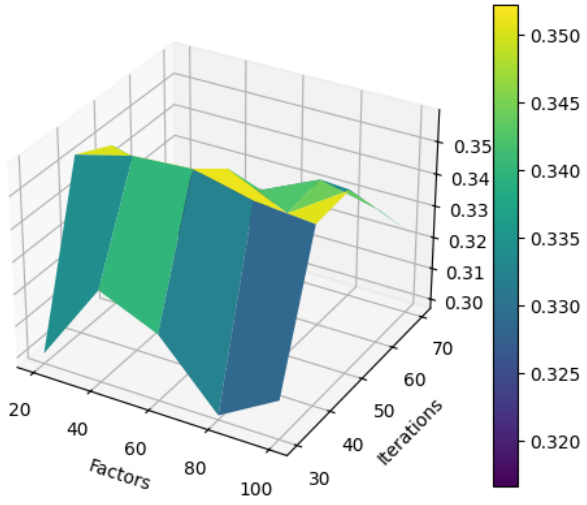

Figure 24: MAP@10 en BPR con variación de factores e iteraciones

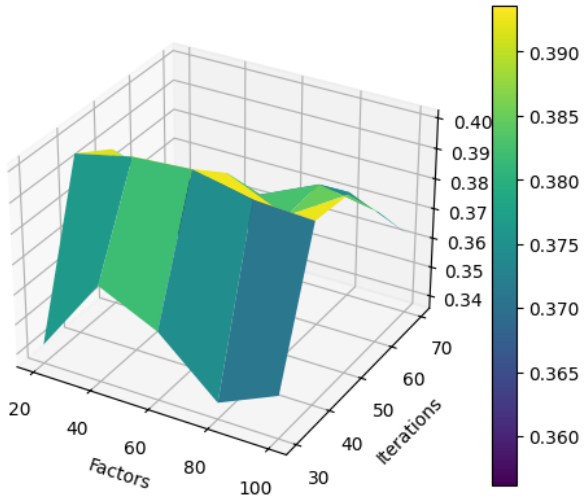

Figure 25: nDCG@10 en BPR con variación de factores e iteraciones

De estos resultados se puede apreciar que de los dos hiperparámetros, el que más efecto tiene en el desempeño es el número de iteraciones. Esto ya que, si bien los factores producen variabilidad en las métricas presentadas, el efecto no es tan grande al mantener un número de iteraciones constante. Por el contrario, al mantener constante el número de factores y variar el número de iteraciones se puede apreciar un mayor rango entre el peor y el mejor valor de la curva.

Asimismo, de las figuras se puede apreciar que el mejor desempeño se obtiene cuando se utilizan 100 factores latentes y 60 iteraciones, lo que se puede corroborar por el color de la superficie en esa sección. Es por esta razón que se decidió entrenar un modelo con estos hiperparámetros para realizar el reporte de la sección 2.3 y de la sección 3.

2.4 ALS

Al igual que en la sección de BPR, se realizó un análisis de sensibilidad en base a los hiperparámetros factors e iterations. Los resultados son presentados en las Figuras 26 y 27.

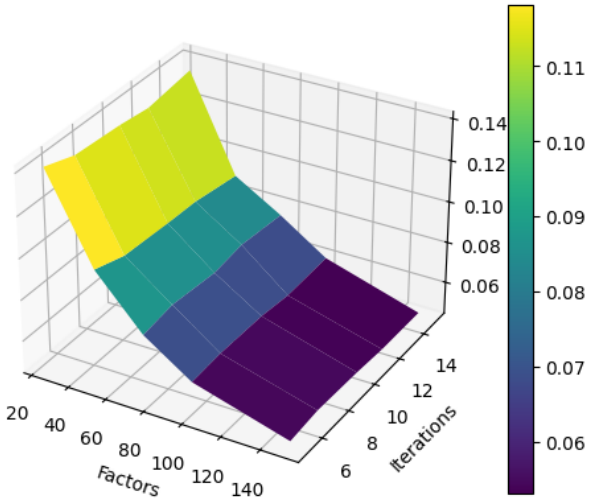

Figure 26: MAP@10 en ALS con variación de factores e iteraciones

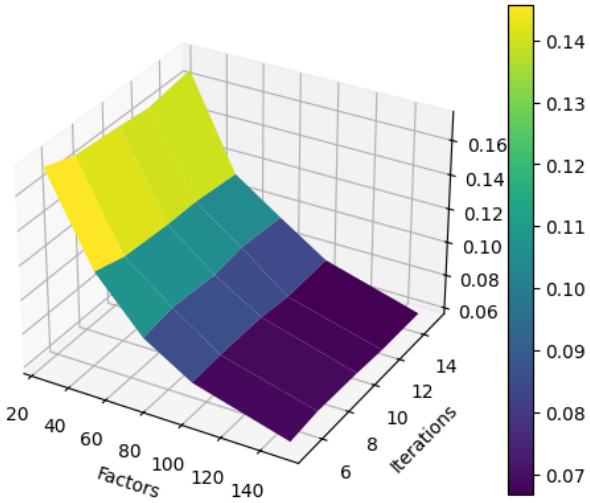

Figure 27: nDCG@10 en ALS con variación de factores e iteraciones

Se puede apreciar en ambas figuras que el número de iteraciones no tiene un gran efecto en el desempeño del modelo. Por el contrario, el número de factores latentes sí es clave a la hora de entrenar el modelo ALS, puesto que a menor número de factores mayor son los valores de MAP@10 y nDCG@10.

En base al comportamiento descrito anteriormente, se decidió entrenar un modelo con 20 factores latentes y con 5 iteraciones, puesto que es el que mejor desempeño muestra en las Figuras 26 y 27. Este es el modelo utilizado para describir las métricas en las secciones 2.6 y 3 del presente informe.

2.5 Factorization Machines

Se realizó un análisis de sensibilidad en base a los hiperparámetros iterations (n_iter) y regularization (l2_reg_w). A diferencia de los otros modelos de ranking, se utilizaron 3 valores en vez de 5 para cada hiperparámetro. Esto se debe a que la implementación de este modelo era bastante más costosa computacionalmente que las demás, por lo que se decidió simplificar el análisis. Los resultados son presentados en las Figuras 28 y 29.

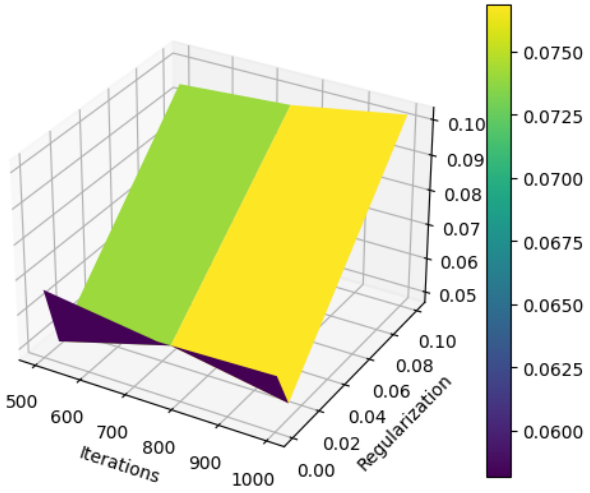

Figure 28: MAP@10 en Factorization Machines con variación de iteraciones y regularización

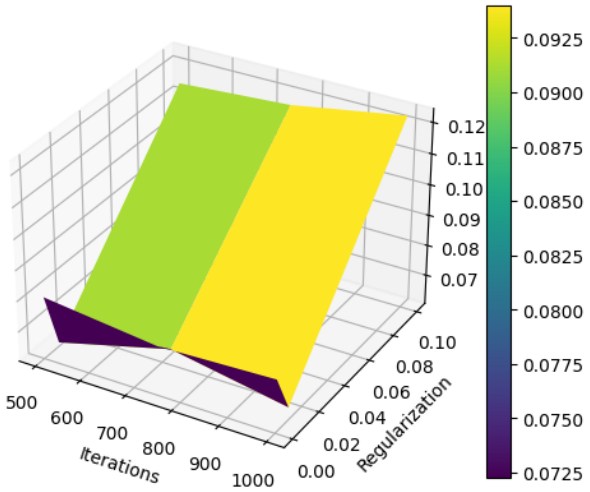

Figure 29: nDCG@10 en Factorization Machines con variación de iteraciones y regularización

Como se puede observar, el número de iteraciones no tiene un gran efecto en el desempeño del modelo, a diferencia de la regularización que juega un papel clave en este. A partir de las figuras, se aprecia que el mejor desempeño se alcanza con 1000 iteraciones y con una regularización de 0.1. De este modo, se escogen estos valores para entrenar el modelo utilizado para calcular las métricas de las secciones 2.6 y 3.

2.6 Desempeño con set de validación

Como se mencionó en las secciones anteriores, para cada modelo de ranking se ajustaron los hiperparámetros para obtener un mayor desempeño en la generación de top 10 recomendaciones. Posteriormente, para cada uno de estos modelos se les calcularon las métricas de desempeño en ranking en base al set de validación proporcionado. Estos resultados presentan en la siguiente tabla:

Modelo	Recall@10	nDCG@10	MAP@10	Diversity	Novelty
iKNN	0.0	0.0011	0.0012	0.5	7.4689
FunkSVD	0.1391	0.0902	0.1078	0.4326	6.1780
ALS	0.2118	0.2138	0.1774	0.5069	4.7766
BPR	0.3889	0.3916	0.3509	0.4349	4.5173
FM	0.1223	0.1231	0.1023	0.7248	5.8044

Table 5: Métricas de ranking para modelos ajustados

3 Análisis de métricas de entrenamiento

Modelo	Tiempo (s)	Memoria Utilizada (MB)	CPU
uKNN	32.44	1320.64	6.0%
iKNN	0.6591	12.05	4.7%
FunkSVD	2.7732	0.2304	6.4%
SVDpp	1.3296	0.0	6.7%
ALS	0.2529	0.2416	82.7%
BPR	0.3287	3.7478	94.90 %
FM	18.9254	4.5793	14.90 %

Table 6: Recursos utilizados durante entrenamiento

Finalmente, la última tabla compara los modelos de recomendación estudiados en términos de tiempo, memoria y uso de CPU. Podemos rescatar que uKNN es el más costoso, con 32.44 segundos de tiempo de ejecución y 1320.64 MB de memoria, mientras que iKNN es mucho más eficiente con solo 0.6591 segundos y 12.05 MB. FunkSVD y SVDpp tienen tiempos cortos y bajo uso de memoria, pero un consumo de CPU similar. ALS destaca por su rapidez y baja memoria, aunque usa bastante CPU, al igual que BPR consume mucho CPU, mientras que FM tiene un tiempo alto (18.93 segundos) pero bajo consumo de memoria.