Laboratorio 1 — Exploración de frameworks de recomendación con *Surprise* (MovieLens 100K)

Alvaro Andres Castiblanco Lopez[1], Bryan Mauricio Guzmán García[2], Juan Carlos Tovar Orjuela[3] y Lorraine Jazlady Rojas Parra[4]

[1]Maestría en Ingeniería de la Información, E-mail: a.castiblancol@uniandes.edu.co.
[2]Maestría en Arquitectura de Tecnologías de Información, E-mail: b.guzmang@uniandes.edu.co.
[3]Maestría en Ingeniería de la Información, E-mail: jc.tovaro1@uniandes.edu.co.
[4]Maestría en Ingeniería de la Información, E-mail: lj.rojasp1@uniandes.edu.co.

1. Introducción

Este informe documenta el desarrollo del **Laboratorio 1** del curso de Sistemas de Recomendación. El objetivo fue (i) familiarizarse con la estructura de archivos que representan una matriz de utilidad (MovieLens 100K), (ii) construir un recomendador simple como *baseline*, (iii) entrenar y evaluar un modelo de filtrado colaborativo usando el framework *Surprise*, y (iv) realizar una validación offline estadística (principalmente RMSE) y una evaluación adicional con listas Top-N.

1.1. Flujo general del laboratorio

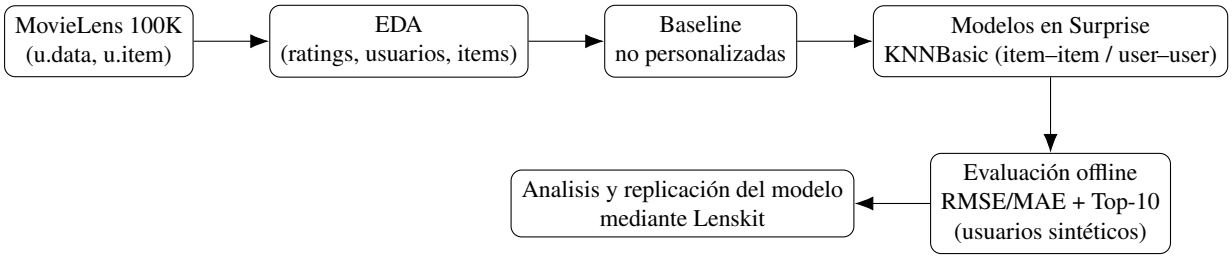

Figure 1: Pipeline seguido en el laboratorio.

2. Carga del dataset y estructura de la matriz de utilidad

2.1. Estructura de archivos: `u.data` y `u.item`

MovieLens 100K separa la información en:

- **`u.data`**: eventos de calificación (`user_id`, `item_id`, `rating`, `timestamp`).
- **`u.item`**: metadatos del ítem (título, fecha, y variables binarias por género).

2.2. Vista rápida de los datos (head)

Tabla 1: Muestra de `ratings.head()` (`u.data`).

user_id	item_id	rating	timestamp
196	242	3	881250949
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923
166	346	1	886397596

Tabla 2: Muestra de `items.head()` (`u.item`) con columnas iniciales y géneros (truncado).

movie_id	movie_title	release_date	unknown	action	adventure	animation	children	...
1	Toy Story (1995)	01-Jan-1995	0	0	0	0	1	...
2	GoldenEye (1995)	01-Jan-1995	0	1	1	0	0	...
3	Four Rooms (1995)	01-Jan-1995	0	0	0	0	0	...
4	Get Shorty (1995)	01-Jan-1995	0	1	0	0	0	...
5	Copycat (1995)	01-Jan-1995	0	0	0	0	0	...

3. Exploración y análisis del dataset (EDA)

3.1. Visualice la distribución de ratings, ¿Qué puede decir al respecto?

La distribución de ratings no es uniforme. En la ejecución del laboratorio se observó una clara concentración en valores medios y altos: el rating 4 es el más frecuente (34.2%), seguido por 3 (27.1%) y 5 (21.5%). En contraste, los valores bajos son menos comunes: 2 representa 11.4% y 1 apenas 6.1%.

Este sesgo positivo sugiere que el dataset contiene más evidencia sobre *qué gusta* que sobre *qué no gusta*. En términos de aprendizaje, puede empujar las predicciones hacia el promedio global y reducir variabilidad en los puntajes estimados, especialmente en escenarios de alta dispersión.

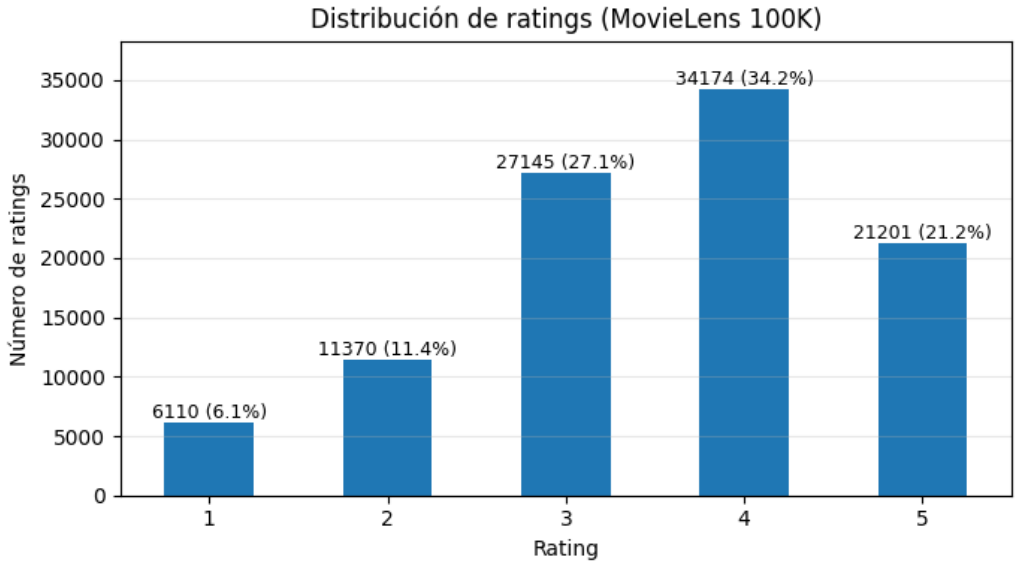

Figure 2: Distribución de ratings en MovieLens 100K (exportada del notebook).

3.2. ¿Cómo es la distribución del número de ratings por usuario?

La actividad por usuario muestra un patrón típico de *cola larga*: la mayoría califica pocas películas y un número reducido de usuarios es altamente activo. En nuestra ejecución, el dataset está filtrado para incluir usuarios con al menos 20 calificaciones (mínimo observado). Un resumen estadístico de `count` por usuario fue:

Tabla 3: Resumen de número de ratings por usuario (celda de resumen del notebook).

	min	p50	p75	p90	p95	max	media	desv.
# ratings/usuario	20	65	148	244.4	310.6	737	106.0	100.9

La mediana (65) es muy inferior al máximo (737), lo que confirma asimetría y presencia de pocos “power users”. Este fenómeno es relevante para filtrado colaborativo porque el solapamiento entre usuarios puede ser bajo y afectar la calidad del vecindario.

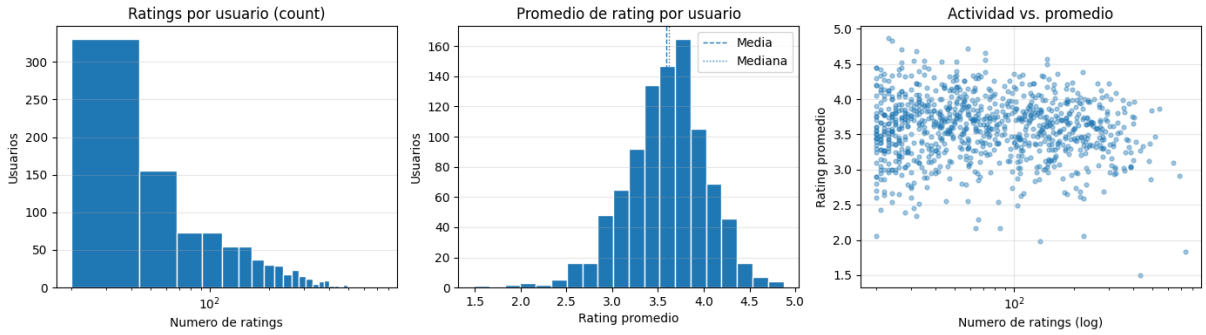

Figure 3: Distribución del número de ratings por usuario (exportada del notebook).

3.3. ¿Cómo es la distribución del promedio de calificación por usuario?

El promedio de rating por usuario fue aproximadamente normal y centrado cerca de 3.59. La desviación estándar fue moderada (0.45), lo que indica diferencias individuales, pero con cierta consistencia global.

Tabla 4: Resumen del promedio de rating por usuario (celda de resumen del notebook).

	min	p25	p50	p75	p95	max	media	desv.
promedio rating/usuario	1.49	3.32	3.62	3.87	4.30	4.87	3.59	0.45

3.4. ¿Cuáles son los items con más calificaciones?

Para identificar popularidad, se agruparon calificaciones por película y se ordenó por `count`. El Top-10 de películas más calificadas se presentó como una gráfica de barras horizontales (con etiquetas de media y varianza). En general, estos ítems corresponden a películas de alta exposición que concentran gran parte del feedback del dataset.

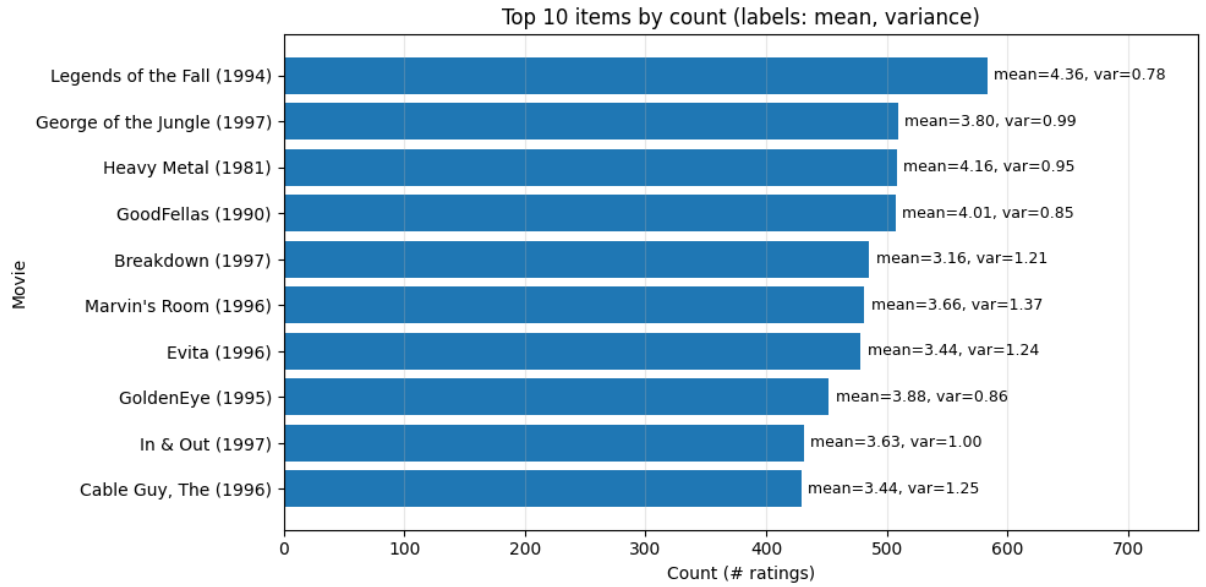

Figure 4: Top-10 ítems por número de ratings; etiquetas muestran media y varianza (exportada del notebook).

3.5. ¿Se puede observar el fenómeno de cola larga en este dataset?

Sí. Se cuantificó el fenómeno comparando masa de ratings concentrada en los ítems más populares:

- % de ratings en el **Top-10** ítems: **4.9%**.
- % de ratings en el **Top-10%** de ítems: **42.7%**.

Esto confirma una distribución sesgada: una fracción pequeña de películas concentra gran parte de las interacciones. Para recomendación, esto suele traducirse en *popularidad* como señal dominante y riesgos de baja cobertura de nichos.

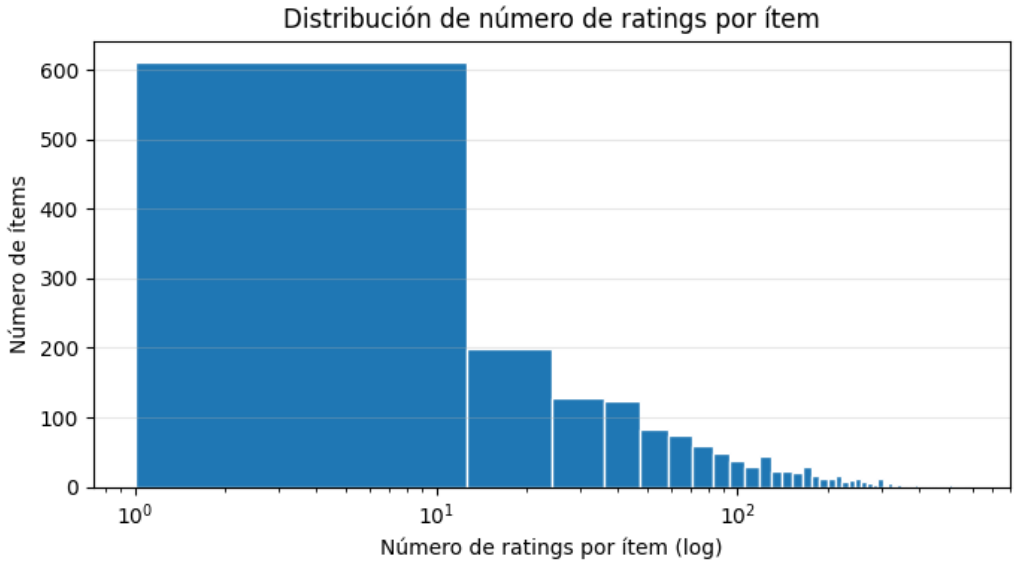

Figure 5: Evidencia visual del fenómeno de cola larga (exportada del notebook).

4. Listas de recomendación no personalizadas (baselines)

4.1. ¿Qué problemas tiene generar una lista no personalizada solamente con el promedio?

Ordenar solo por promedio es un baseline útil pero engañoso. En el laboratorio se identificaron, al menos, cuatro problemas:

1. **Sesgo por bajo volumen (incertidumbre).** Un ítem con un único rating de 5.0 puede quedar por encima de un clásico con 4.8 basado en cientos de votos.
2. **Inestabilidad.** Con pocos ratings, el promedio cambia drásticamente ante una nueva calificación.
3. **Ignora popularidad y cobertura.** Tiende a recomendar ítems "raros" con evidencia mínima, afectando confianza del ranking.
4. **No modela heterogeneidad.** Un promedio alto puede reflejar un nicho extremo en vez de calidad general.

Como evidencia, el Top-10 por promedio (sin corrección por conteo) queda dominado por ítems con muy pocas observaciones.

Tabla 5: Top-10 no personalizado ordenado solo por promedio (ejemplo de sesgo por baja evidencia).

#	Película	Promedio	N ratings
1	One Fine Day (1996)	5.00	1
2	Guantanamera (1994)	5.00	1
3	Maybe, Maybe Not (Bewegte Mann, Der) (1994)	5.00	1
4	Last Time I Saw Paris, The (1954)	5.00	1
5	Chairman of the Board (1998)	5.00	1
6	Ayn Rand: A Sense of Life (1997)	5.00	3
7	Saint of Fort Washington, The (1993)	5.00	2
8	Someone Else's America (1995)	5.00	1
9	Prefontaine (1997)	5.00	3
10	Star Kid (1997)	5.00	3

4.2. ¿Es posible generar una mejor lista no personalizada teniendo en cuenta el promedio y el conteo?

Sí. La estrategia general es combinar **señal (promedio)** con **confianza (número de ratings)** para penalizar la incertidumbre. En el notebook se exploraron cuatro enfoques:

4.2.1. (1) How Not To Sort By Average Rating (Wilson LCB)

Se transforma cada rating en un *like* (1 si rating ≥ umbral, 0 si no) y se calcula el **límite inferior** del intervalo de Wilson sobre la proporción de likes. Así, los ítems con pocos votos quedan penalizados [3].

Tabla 6: Top-10 por límite inferior de Wilson (likes definidos como rating ≥ 4).

#	Película	N ratings	Wilson LCB
1	Schindler's List (1993)	298	0.850
2	Casablanca (1942)	243	0.845
3	Rear Window (1954)	209	0.843
4	Star Wars (1977)	583	0.840
5	Shawshank Redemption, The (1994)	283	0.839
6	Dr. Strangelove or: How I Learned to Stop Worr...	194	0.836
7	Usual Suspects, The (1995)	267	0.835
8	Wrong Trousers, The (1993)	118	0.835
9	Raiders of the Lost Ark (1981)	420	0.833
10	Godfather, The (1972)	413	0.832

4.2.2. (2) Límite inferior del intervalo de confianza (IC) de la media

Se usa una estimación conservadora del promedio:

$$LCB = \bar{x} - z \cdot \frac{s}{\sqrt{n}},$$

donde $\bar{x}$ es la media del ítem, s su desviación estándar y n el número de ratings. Con esto, promedios altos con alta varianza o poco n bajan en el ranking [4].

Tabla 7: Top-10 por límite inferior del IC de la media (con filtro de mínimo número de ratings).

#	Película	N ratings	Promedio	LCB media
1	Star Wars (1977)	583	4.36	4.27
2	Raiders of the Lost Ark (1981)	420	4.25	4.16
3	Godfather, The (1972)	413	4.28	4.16
4	Fargo (1996)	508	4.16	4.08
5	Pulp Fiction (1994)	394	4.06	3.96
6	Return of the Jedi (1983)	507	4.01	3.93
7	Empire Strikes Back, The (1980)	368	4.21	4.10
8	Toy Story (1995)	452	3.88	3.79
9	Silence of the Lambs, The (1991)	390	4.16	4.05
10	Princess Bride, The (1987)	324	4.17	4.05

4.2.3. (3) Promedio bayesiano (shrinkage hacia la media global)

Se usa un promedio ponderado:

$$score = \frac{v}{v+m}R + \frac{m}{v+m}C,$$

donde v es el conteo del ítem, R su promedio, C el promedio global y m controla el *suavizamiento*. Esto evita que ítems con pocos ratings dominen el Top-N [5].

Tabla 8: Top-10 por promedio bayesiano (shrinkage hacia la media global).

#	Película	N ratings	Promedio	Score bayesiano
1	Star Wars (1977)	583	4.359	4.239
2	Fargo (1996)	508	4.156	4.096
3	Return of the Jedi (1983)	507	4.007	3.958
4	Contact (1997)	509	3.804	3.786
5	Toy Story (1995)	452	3.878	3.851
6	Empire Strikes Back, The (1980)	368	4.212	4.118
7	Raiders of the Lost Ark (1981)	420	4.252	4.140
8	Godfather, The (1972)	413	4.283	4.159
9	Princess Bride, The (1987)	324	4.173	4.077
10	Silence of the Lambs, The (1991)	390	4.161	4.094

4.2.4. (4) Evan Miller Star Ratings (Dirichlet + LCB)

Para ratings en estrellas (1–K), se aplica un prior Dirichlet para evitar extremos con pocos datos, se estima una media bayesiana μ y se penaliza por incertidumbre [6]:

$$score = \mu - z \cdot \sqrt{\text{var}}.$$

Tabla 9: Top-10 por Evan Miller star ratings (Dirichlet + penalización por incertidumbre).

#	Película	N ratings	Score
1	Star Wars (1977)	583	4.165
2	Raiders of the Lost Ark (1981)	420	4.037
3	Godfather, The (1972)	413	4.042
4	Empire Strikes Back, The (1980)	368	4.023
5	Silence of the Lambs, The (1991)	390	4.010
6	Fargo (1996)	508	3.998
7	Princess Bride, The (1987)	324	3.966
8	Return of the Jedi (1983)	507	3.871
9	Toy Story (1995)	452	3.818
10	Pulp Fiction (1994)	394	3.810

4.2.5. Comparación entre rankings

Para comparar consistencia entre métodos, se calculó la similitud de Jaccard entre conjuntos Top-10 y se construyó un *bump chart*. El resultado mostró solapamientos moderados (no hay consenso total), lo cual evidencia que la definición de "mejor" ranking depende de la noción elegida de confianza/penalización.

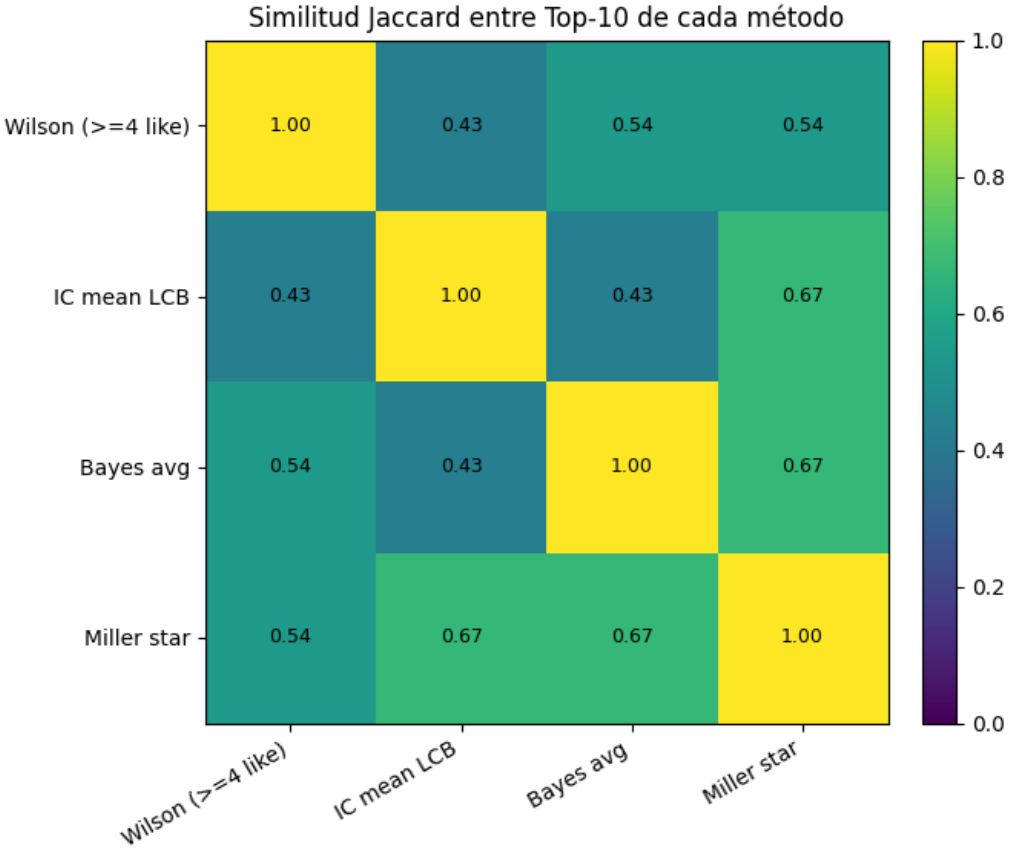

Figure 6: Similitud Jaccard entre conjuntos Top-10 por método (exportada del notebook).

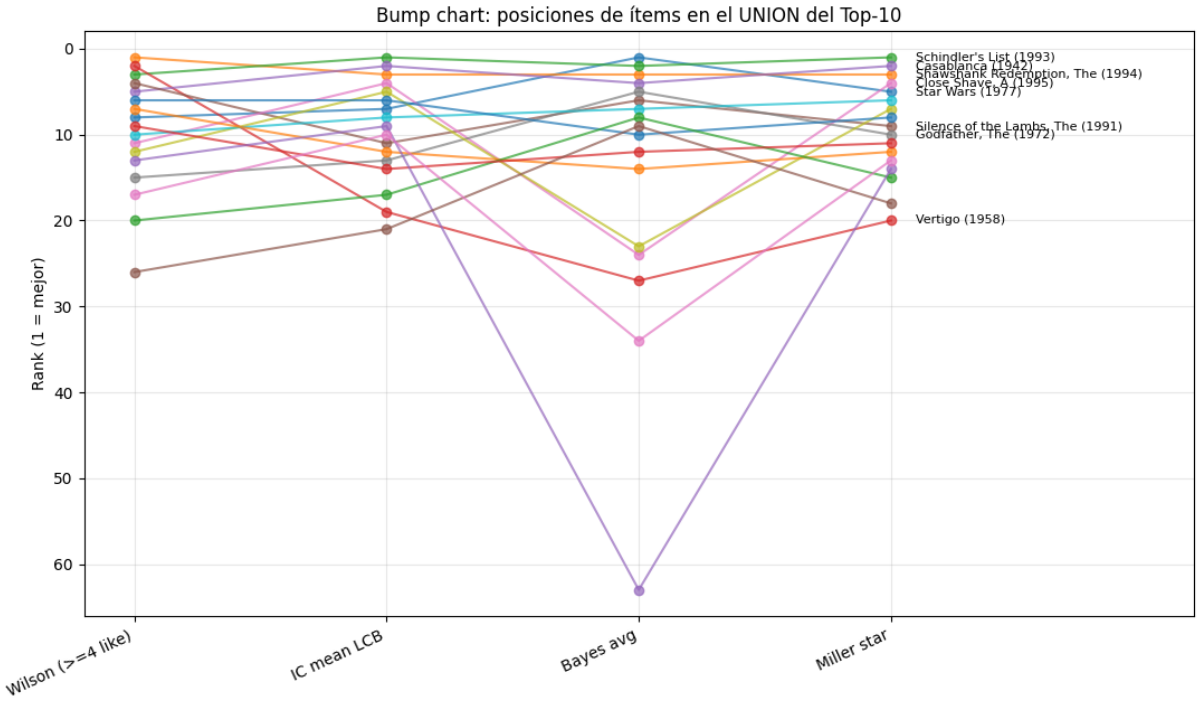

Figure 7: Comparación de posiciones (bump chart) para la unión de ítems recomendados por diferentes métodos.

5. Filtrado colaborativo con el framework *Surprise*

5.1. Construcción del dataset en Surprise

Se cargaron los ratings a un objeto `Dataset` de Surprise usando un `Reader` con escala (1,5). Luego se hizo un particionamiento train/test 80/20.

5.2. Modelo KNNBasic item–item con similitud coseno

Se entrenó un KNN item–item (`user_based=False`) con similitud coseno y parámetros $k = 20$, $min\_k = 2$. Como verificación, se inspeccionó una predicción individual (ej.: 4.25 estimado vs 4.0 real).

Para evaluación offline se usó RMSE en el conjunto de prueba:

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

El modelo item–item obtuvo **RMSE** ≈ 1.0453.

5.3. ¿Cuál es el RMSE de un modelo usuario–usuario con los mismos parámetros de similitud?

Entrenando el mismo algoritmo KNNBasic en modo usuario–usuario (`user_based=True`), con $k = 20$ y similitud coseno, se obtuvo **RMSE** ≈ 1.0166 en nuestra ejecución.
Este valor indica una precisión razonable, pero refleja limitaciones típicas del escenario:

- **Dispersión usuario–ítem**: el solapamiento entre usuarios puede ser bajo, afectando la calidad del vecindario.
- **Sesgo de popularidad**: cuando faltan vecinos "fuertes", el modelo tiende a apoyarse en promedios/ítems populares.
- **Dependencia de partición**: cambios leves en el split (semilla) pueden variar el RMSE en el segundo decimal.

5.4. ¿Cuál es el efecto de cambiar el número de vecinos en la calidad del modelo usuario–usuario?

Se evaluó $k \in \{5, 10, 20, 30, 40, 60, 80, 100\}$ midiendo RMSE y MAE. La tendencia observada fue:

- Con **pocos vecinos** (p.ej. $k = 5$) el error es mayor por varianza alta (poca evidencia).
- Al **aumentar** k, RMSE y MAE mejoran hasta estabilizarse alrededor de un rango medio ($k \approx 30$–60).
- Con k muy alto, las mejoras son marginales: se mezclan vecinos menos similares (más sesgo).

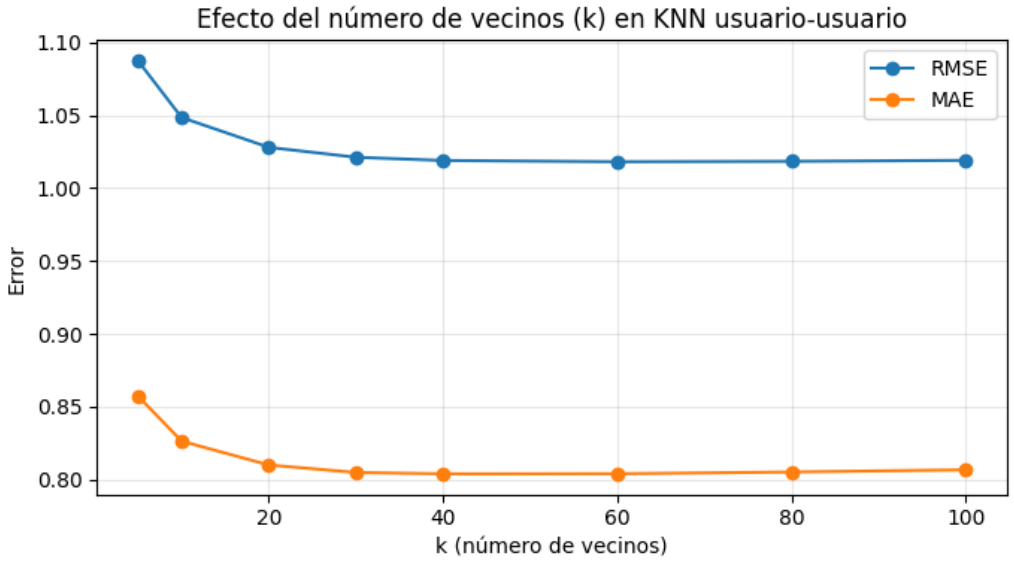

Figure 8: Efecto del número de vecinos (k) en KNN usuario–usuario (exportada del notebook).

6. Usuarios sintéticos y evaluación Top-N

6.1. Cree al menos 2 usuarios (944 mainstream y otro de nicho) y asigne ratings nuevos

Se construyeron usuarios sintéticos agregando nuevas filas de ratings (sin duplicados usuario–ítem) para simular perfiles:

- **944 (mainstream)**: likes a ítems populares y dislikes a algunos ítems de nicho.
- **945 (nicho)** y perfiles adicionales (946–950) con combinaciones de géneros específicos (p.ej., Film-Noir, Sci-Fi, Horror).

Luego se re-entrenó el modelo item–item con el dataset aumentado y se generaron recomendaciones Top-10 usando el `anti_testset` (ítems no vistos por cada usuario).

6.2. ¿Qué tan bien cree que el sistema está respondiendo a los gustos del usuario?

Se evaluó (i) un **hit-rate por géneros** en el Top-10 y (ii) la **popularidad promedio** de lo recomendado. La Tabla 10 resume los resultados.

Tabla 10: Evaluación offline sobre usuarios sintéticos: afinidad por géneros y popularidad promedio de las recomendaciones Top-10.

#	user_id	Géneros preferidos (likes)	Hit-rate géneros (Top-10)	Hit-rate Film-Noir	Popularidad promedio Top-10
1	944	Action, Adventure, Animation, Children, Comedy...	1.00		169.40
2	945	Crime, Documentary, Drama, Film-Noir, Thriller...	0.50		146.00
3	946	Crime, Film-Noir, Mystery, Sci-Fi, Thriller	0.70	0.00	405.00
4	947	Action, Adventure, Animation, Children, Comedy...	1.00		8.20
5	948	Action, Adventure, Comedy, Drama, Romance, Sci...	0.90		6.80
6	949	Comedy, Drama, Horror, Thriller	0.80		146.00
7	950	Action, Adventure, Comedy, Crime, Documentary,...	1.00		13.30

El sistema tiende a responder mejor para perfiles amplios/mainstream (hit-rate alto). Sin embargo, cuando el perfil es más de nicho, el hit-rate cae y el sistema se refugia en ítems populares. Además, el hit-rate puede sobreestimar desempeño si el conjunto de géneros preferidos es demasiado amplio (fácil de acertar).

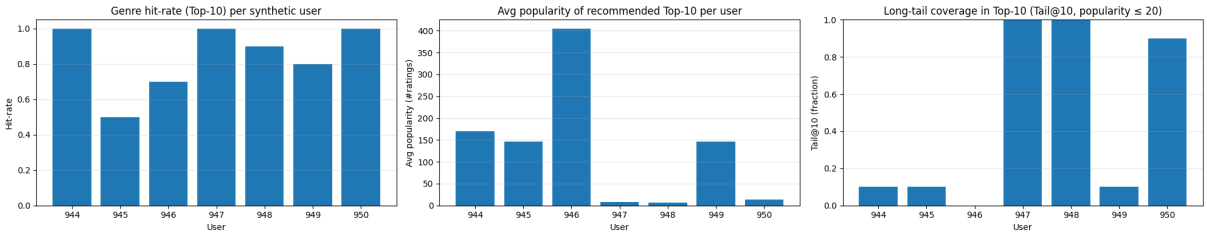

Figure 9: Evaluación sobre usuarios sintéticos: hit-rate por géneros, popularidad promedio Top-10 y cobertura de cola larga (exportada del notebook).

6.3. ¿Qué tan bien cree que el sistema responde al objetivo de buscar items para el usuario dentro de la cola larga?

La evidencia sugiere que el sistema **no** explora consistentemente la cola larga. Para algunos usuarios (p.ej. 947, 948, 950) la popularidad promedio de sus recomendaciones Top-10 es baja (explora long tail), mientras que para otros (p.ej. 944, 945, 946, 949) la popularidad promedio es alta (se queda en el *head*). En otras palabras: la exploración de long tail depende fuertemente del perfil y no es estable.

7. Hallazgos del framework *Surprise*: alcance y limitaciones

7.1. Alcance

- **Prototipado rápido**: cargar datos, entrenar KNN y evaluar RMSE requirió pocas líneas.
- **Implementaciones listas**: métricas (`accuracy.rmse`) y estructuras de train/test simplifican evaluación offline.
- **Configurabilidad**: permite alternar item–item vs user–user y cambiar métricas de similitud mediante `sim_options`.

7.2. Limitaciones observadas

- **Sparsity sensitivity**: KNN requiere solapamiento; en matrices dispersas la calidad cae o se apoya en promedios.
- **Popularidad como atajo**: al no incorporar señales de contenido o contexto, el modelo tiende a recomendar head en perfiles difíciles.
- **Evaluación limitada**: RMSE mide calidad de predicción de rating, pero no necesariamente calidad de ranking Top-N; por eso se complementó con métricas de listas (hit-rate, popularidad, long tail).
- **Producción vs laboratorio**: Surprise es excelente para *baseline* offline, pero no es una solución completa de producción (serving, features, retraining, etc.).

8. ¿Por qué LensKit mejora el RMSE frente a Surprise?

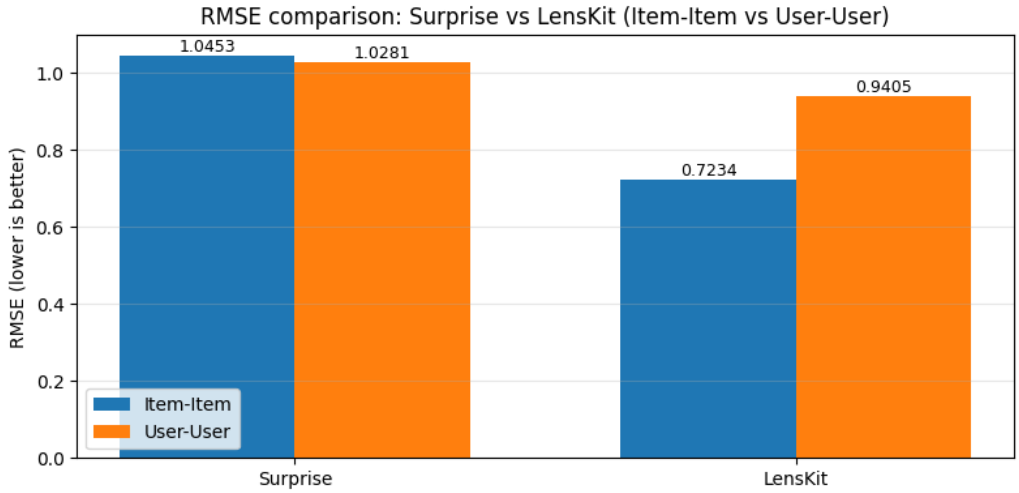

Figure 10: Comparación RMSE Usuario-Usuario e Item-Item entre Surprise y Lenskit (exportada del notebook).

En los experimentos se observa un menor RMSE en LensKit que en Surprise para modelos KNN equivalentes (Item-Item y User-User). Esta diferencia no necesariamente implica que "LensKit sea mejor" en abstracto, sino que **las decisiones de modelado y evaluación no son idénticas** entre frameworks. Las principales razones son:

- **LensKit incorpora (o recomienda incorporar) un modelo de sesgos (baseline) como respaldo.** LensKit provee `BiasScorer`, que modela el puntaje como una combinación del promedio global más sesgos por usuario e ítem ($b_{ui} = b_g + b_i + b_u$). Este baseline captura tendencias sistemáticas (usuarios que califican alto/bajo e ítems generalmente bien/mal calificados), lo cual suele reducir error en predicción de ratings. *En cambio, en Surprise se usó `KNNBasic` con coseno, que es un vecindario "puro" y predice como un promedio ponderado de ratings de vecinos, sin una corrección explícita por sesgos.* [7]
- **LensKit enfatiza la cobertura completa en predicción de ratings mediante `fallback`.** En su guía de evaluación de predicciones, LensKit recomienda usar un `FallbackScorer` con `BiasScorer` para asegurar que **todos los pares (usuario, ítem) del test queden puntuados**. De lo contrario, si un modelo sólo predice para una fracción "fácil" de casos y se ignoran los faltantes, el RMSE puede quedar sesgado y las comparaciones entre modelos se vuelven injustas. Esta práctica de "siempre puntuar" tiende a estabilizar y, en muchos casos, mejorar el RMSE observado. [8]

- **Pequeñas diferencias en partición, filtrado o parámetros cambian RMSE.** Aun con el mismo dataset, variaciones en el *split* (semilla), filtros de usuarios/ítems, tratamiento de ítems sin vecinos, y definición exacta de candidatos evaluados pueden mover el RMSE. Por eso, para una comparación estricta, conviene controlar: misma partición, mismos usuarios/ítems, mismo criterio de "predicción faltante", y (si aplica) incluir o excluir baseline de forma consistente.

9. Conclusiones

- MovieLens 100K evidencia sesgo hacia ratings altos y un fuerte fenómeno de cola larga (en usuarios e ítems).
- Un ranking no-personalizado por promedio es inadecuado sin correcciones por incertidumbre; métodos como Wilson LCB, LCB de la media y promedio bayesiano producen listas más robustas.
- Con Surprise, el KNN usuario–usuario mostró RMSE ligeramente mejor que item–item en nuestra ejecución, y el ajuste de k evidencia el trade-off sesgo–varianza.
- En evaluación Top-N, el modelo se comporta razonablemente para perfiles mainstream, pero es inconsistente para nichos y para el objetivo de long tail.

A. Apéndice: Unión de ítems Top-10 por método

Tabla 11: Ítems presentes en el Top-10 de uno o más métodos (unión) y número de métodos que lo incluyen.

#	Película	# métodos	Métodos
1	Casablanca (1942)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
2	Rear Window (1954)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
3	Schindler's List (1993)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
4	Shawshank Redemption, The (1994)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
5	Star Wars (1977)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
6	Raiders of the Lost Ark (1981)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
7	Godfather, The (1972)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
8	Empire Strikes Back, The (1980)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
9	Silence of the Lambs, The (1991)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
10	Fargo (1996)	4	Wilson (>=4 like), IC mean LCB, Bayes avg, Mil...
11	Princess Bride, The (1987)	3	IC mean LCB, Bayes avg, Miller star
12	Toy Story (1995)	3	IC mean LCB, Bayes avg, Miller star
13	Pulp Fiction (1994)	2	IC mean LCB, Miller star
14	Return of the Jedi (1983)	3	IC mean LCB, Bayes avg, Miller star
15	Contact (1997)	1	Bayes avg
16	Usual Suspects, The (1995)	1	Wilson (>=4 like)
17	Dr. Strangelove or: How I Learned to Stop Worr...	1	Wilson (>=4 like)

Referencias

[1] N. Hug, *Surprise: A Python scikit for building and analyzing recommender systems*. Documentación oficial.
[2] F. Maxwell Harper and J. A. Konstan, "The MovieLens Datasets: History and Context", *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 2015.
[3] Evan Miller. *How Not To Sort By Average Rating*. 2009. Disponible en: https://www.evanmiller.org/how-not-to-sort-by-average-rating.html
[4] Wikipedia. *Intervalo de confianza*. Consultado en 2026. Disponible en: https://es.wikipedia.org/wiki/Intervalo_de_confianza
[5] Arpit Bhayani. *Bayesian Average*. Disponible en: https://arpitbhayani-me.translate.goog/blogs/bayesian-average/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc
[6] Evan Miller. *Ranking Items with Star Ratings*. 2009. Disponible en: https://www.evanmiller.org/ranking-items-with-star-ratings.html
[7] LensKit. *API Reference:* `lenskit.basic.bias`. Modelo de sesgos $b_{ui} = b_g + b_i + b_u$. Disponible en la documentación oficial de LensKit. [**?**]
[8] LensKit. *Evaluating Rating Predictions*. Recomendación de usar `FallbackScorer` con `BiasScorer` para puntuar todos los ítems y evitar sesgos por predicciones faltantes. [**?**]