Sistemas de Recuperación de Información y de Recomendación

Máster en Ciencia de Datos e Ingeniería de Computadores

Universidad de Granada

CURSO 2018/2019

Implementación y evaluación de un sistema de recomendación de películas

Autores:
Javier Poyatos Amador
Juan Luis Suárez Díaz
jpoyatosamador@correo.ugr.es
jlsuarezdiaz@correo.ugr.es

2 de junio de 2019

Índice

1. Introducción

En este trabajo se ha desarrollado un sistema de recomendación sobre una base de datos de películas, que permite obtener recomendaciones siguiendo distintos criterios y evaluar mediante diferentes medidas la calidad de dichas recomendaciones. El sistema de recomendación se ha desarrollado en el lenguaje Python. Esta documentación muestra las principales características del sistema, cómo se han desarrollado los distintos métodos de recomendación, los resultados de la evaluación de dichos métodos y un breve manual de usuario que explica los principales comandos para trabajar con el sistema de recomendación.

1.1. Sistemas desarrollados

Se han llevado a cabo varias propuestas de sistemas de recomendación, las cuales se muestran a continuación.

- **Recomendación basada en popularidad.** Este esquema de recomendación es uno de los más simples, y consiste en recomendar películas a los usuarios en función de su valoración. También se pondera el número de votos para evitar recomendar películas con valoraciones muy altas pero muy pocos votos.
- **Recomendación basada en contenido.** Este sistema sugiere películas a un determinado usuario en función de aquellas que el usuario haya visto y votado positivamente. La búsqueda de las películas se realiza analizando las semejanzas de contenido entre los distintos metadatos que ofrece la base de datos.
- **Recomendación colaborativa.** Este sistema recomienda películas a un usuario en función a otros usuarios que han visto películas similares a las que el primer usuario también ha visualizado.
- **Recomendación híbrida.** Combina los enfoques colaborativos y basados en contenido para intentar obtener recomendaciones más adecuadas para los usuarios.

1.2. Conjunto de datos utilizado

La base de datos utilizada, denominada *The Movies Dataset*, está disponible en la plataforma Kaggle [1]. Consta de un conjunto de ficheros con diferente información sobre películas y valoraciones de usuarios. El principal fichero, `movies_metadata.csv`, contiene una amplia variedad de información sobre 45.000 películas del dataset MovieLens. Entre esta información podemos destacar título, resumen del argumento, géneros, presupuesto o valoración. Los ficheros `keywords.csv` y `credits.csv` están muy ligados a dichos metadatos, conteniendo información sobre las palabras clave y el elenco y equipo de rodaje de la película, respectivamente. El fichero `ratings.csv` contiene información sobre usuarios valorando (del 0 al 5) distintas películas. En total hay 26 millones de valoraciones, lo cual no es tratable computacionalmente desde ordenadores personales, por lo que la base de datos proporciona también un fichero `ratings.csv` con un subconjunto de 100.000 valoraciones de 671 usuarios. Finalmente, el fichero `links.csv` proporciona una conexión entre los índices de películas en el conjunto de metadatos y los índices de las películas valoradas en el conjunto de ratings.

2. Descripción de los sistemas elaborados

2.1. Recomendación basada en popularidad

El objetivo de esta primera aproximación es simple: recomendar películas en base a una determinada métrica o sistema de puntuación. En este caso se muestran las n mejores películas en función del voto medio, votos totales y de la puntuación de cada película.

Podría haberse optado por escoger simplemente las mejores películas en base a su puntuación, pero en este caso las recomendaciones podrían verse muy penalizadas por películas votadas positivamente pero con un número de votos muy bajo, para las cuales su valoración real podría estar muy lejos de su valoración empírica. Cuando el número de votos aumenta, las valoraciones empíricas se aproximan mucho más a las valoraciones reales.

Para realizar esta tarea necesitamos saber los votos de cada película, el número de votos medio y establecer un cuantil (en nuestro caso hemos elegido 0.9), que representa el mínimo número de votos que necesita una película para estar dentro de la lista de recomendaciones. Con estos dos coeficientes calculamos un peso ponderado para cada película que va a ser su puntuación,

$$WR = \frac{vR}{v+m} + \frac{mC}{v+m},$$

donde v es el número de votos para cada película, m es el mínimo número de votos que necesita una película para estar en las recomendaciones, obtenido a partir del cuantil establecido, R es la valoración media de la película y C la valoración media global de todas las películas en la base de datos. Las películas con mayor peso ponderado serán las que conformen la lista de recomendaciones por popularidad.

2.2. Recomendación basada en contenido

Uno de los problemas que tiene la aproximación anterior es que distintos usuarios van a obtener la misma recomendación cuando no tiene por qué darse que a ambos les gusten las mismas películas. El objetivo del sistema basado en contenido es personalizar la recomendación en base a las películas que el usuario haya visto y valorado positivamente. Para ello, el sistema calcula la similitud entre las películas en base a un determinado criterio y sugiere a cada usuario las películas que considere más similares de acuerdo con sus gustos.

En este sistema son fundamentales dos elementos: cómo extraer datos de las películas para poder medir distancias o similitudes entre ellos, y la métrica de distancia o similitud a usar. Para el segundo caso se ha utilizado la similitud del coseno, una métrica muy extendida en problemas de minería de texto. Para el primer caso, se han seguido dos procedimientos distintos, ambos basados en los datos textuales que proporciona la base de datos:

- Extracción de datos a partir de la descripción de la película. Para este procedimiento se ha utilizado el resumen del argumento de la película, disponible en los metadatos. Tras tokenizar los resúmenes y eliminar las palabras de parada, se ha calculado el score TF-IDF para cada película, premiando así las palabras frecuentes en un mismo resumen, pero penalizando aquellas que aparecen comúnmente en la mayoría de resúmenes. Los datos numéricos obtenidos tras calcular el score TF-IDF son los utilizados para obtener las similitudes entre películas
- Método basado en créditos, géneros y palabras clave: en este procedimiento, a partir de los metadatos especificados se extraen diversas características: el director de la película, los 3 actores principales, los 3 géneros principales y las 3 palabras clave principales. Estos elementos se preprocesan, transformándolos a minúsculas y eliminando espacios en blanco (para que un nombre y apellidos completo sea considerado una entidad propia). Después, todos los elementos se añaden a una bolsa de palabras. En este caso, en lugar del TF-IDF, la transformación numérica la realizamos contando las ocurrencias de cada término en la bolsa de palabras, pues todos los términos que la componen son relevantes en la película. A partir de las ocurrencias obtenemos las similitudes entre películas. A diferencia del primer procedimiento, este método permite detectar más fácilmente los gustos de un usuario por un determinado género, actor o director y recomendar otras películas de dicho género o en las que intervengan dicho actor o director.

Una vez calculadas las similitudes entre las películas, para recomendar películas a un usuario simplemente tenemos que recuperar las más similares a las películas valoradas positivamente por este.

2.3. Recomendación colaborativa

El problema que tiene el sistema anterior es que presenta dificultad para recomendar películas de diferentes géneros ya que su funcionamiento consiste en recomendar películas similares. El enfoque colaborativo (basado en usuarios) trata de solventar este problema. Para ello, tiene en cuenta el historial de valoraciones de películas de todos los usuarios. La premisa que sigue este sistema es que si a un usuario le han gustado determinadas películas, y otro usuario al que también le han gustado dichas películas ha valorado positivamente una película distinta, entonces al primer usuario también puede gustarle dicha película.

Existen diversos métodos para aprender modelos de recomendación basados en usuarios. En nuestro sistema de recomendación hemos utilizado los métodos SVD y KNN, disponibles en la librería `surprise` [2]. Una vez aprendidos los modelos, pueden utilizarse para estimar la valoración que daría un usuario a una película que no ha visto, en función de todas las valoraciones de usuarios entrenadas. Por tanto, las recomendaciones resultantes para un usuario podemos obtenerlas como aquellas películas que no ha visto para las que la estimación de valoración es mayor.

2.4. Recomendación híbrida

El sistema de recomendación híbrido combina las ideas de los dos sistemas anteriores, para intentar proporcionar una recomendación que se adapte mejor a cada usuarios. Existen diversos enfoques para combinar las recomendaciones de contenido y colaborativas. En nuestro caso, hemos realizado dos propuestas distintas.

La primera propuesta elaborada es la de un sistema híbrido en cascada, para el que se ha obtenido en primer lugar un listado amplio de películas similares, obtenidas mediante recomendación basada en contenido, y sobre dichas películas se ha estimado la valoración que proporcionaría el usuario. La recomendación final se obtiene como el listado de películas con mejor valoración estimada de entre las películas similares recuperadas en el primer paso.

La segunda propuesta elaborada es de un sistema híbrido ponderado, en el que se consideran los scores de similaridad proporcionados por el sistema basado en contenido y las valoraciones estimadas proporcionadas por el sistema colaborativo. Ambas valoraciones se normalizan al intervalo $[0, 1]$ para que tengan igual peso, y se ponderan según la cantidad de películas que haya visto el usuario. Cuando el usuario haya visto pocas películas tendrá más peso el sistema basado en contenido, mientras que cuando haya visto muchas películas tendrá mayor peso el sistema colaborativo. La fórmula utilizada para combinar los scores ha sido

$$s_{hybrid} = (1 - \lambda)s_{content} + \lambda s_{collaborative},$$

donde s_x representa el score de cada uno de los sistemas considerados y λ se ha tomado como

$$\lambda = (1 - 2\alpha)\frac{n}{N},$$

donde n es el número de usuarios que han visto un número de películas menor o igual que el usuario a evaluar, N es el total de usuarios, y α representa la fracción mínima que aportan al score híbrido cada uno de los dos scores, y se ha escogido como $\alpha = 0, 1$.

3. Evaluación y validación

Los distintos sistemas de recomendación implementados se han evaluado mediante distintos criterios, para tratar de ver cuáles son sus capacidades reales a la hora de realizar buenas recomendaciones.

Para evaluar estos sistemas, puesto que disponemos de valoraciones reales de usuarios, podemos seguir dos enfoques:

- **Enfoque no supervisado.** Mediante la evaluación no supervisada, podemos analizar la lista total de recomendaciones obtenidas para el conjunto de usuarios en la base de datos. Sobre dicha lista podemos medir, por ejemplo, qué cantidad total de películas es capaz de recomendar el sistema, cómo de diferentes son las recomendaciones para cada usuario, o la variedad o capacidad de sorprender que presentan las recomendaciones.
- **Enfoque supervisado.** Mediante la evaluación supervisada, podemos usar distintas técnicas de validación para separar los datos en entrenamiento y test, y evaluar sobre los datos de test cómo de buenas han sido las recomendaciones aprendidas usando el conjunto de entrenamiento.

3.1. Evaluación no supervisada

En primer lugar realizamos una evaluación no supervisada de los sistemas de recomendación implementados. Para ello, utilizamos las siguientes medidas clásicas para sistemas de recomendación [3, 4]:

- **Cobertura.** La cobertura mide el porcentaje de películas que el sistema es capaz de recomendar, dentro del catálogo completo. Se obtiene como

  $$C = \frac{len(unique(L))}{N},$$

  donde L es la lista de recomendaciones para todos los usuarios, y N el total de películas en la base de datos.

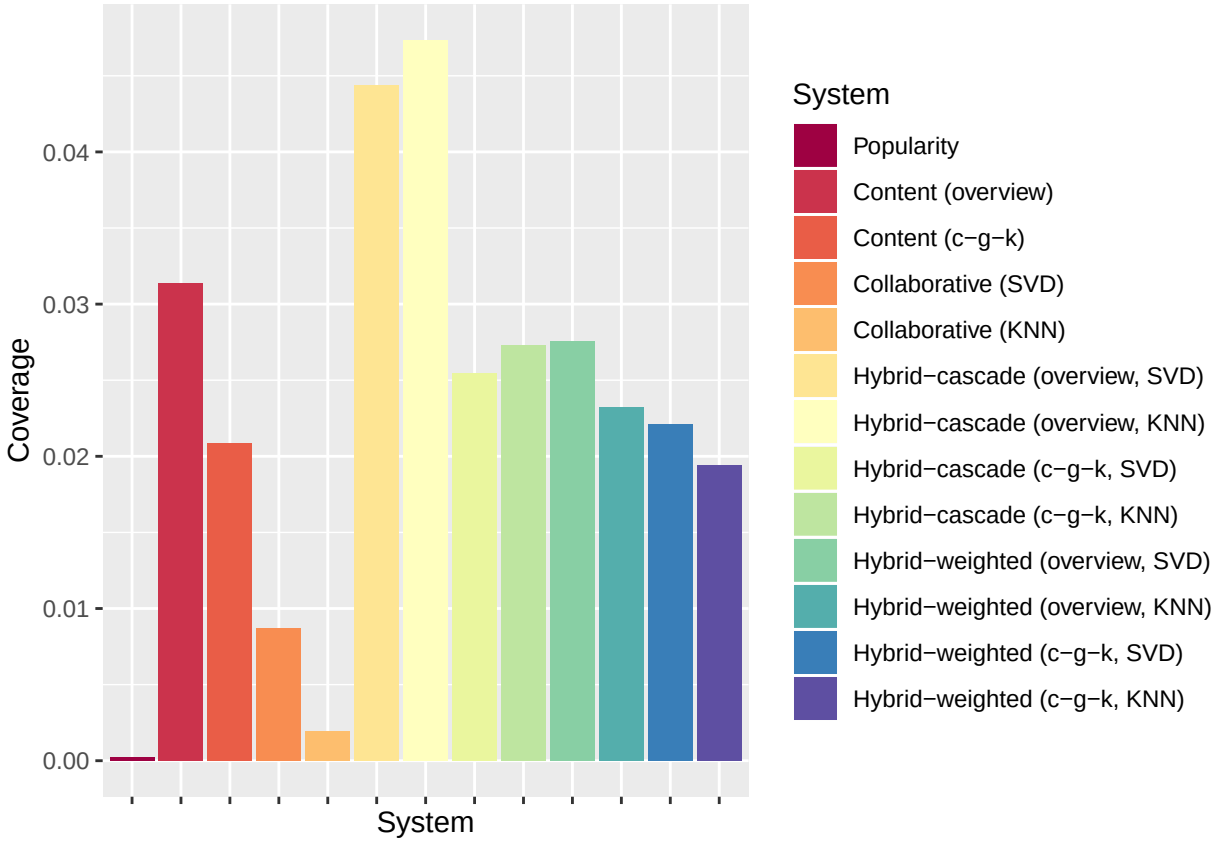

Figura 1: Cobertura obtenida en los distintos sistemas de recomendación.

- **Personalización.** La personalización mide la capacidad que tiene el sistema para recomendar películas diferentes a usuarios diferentes. Para calcularla, en primer lugar se obtiene una matriz binaria M donde $M_{ij} = 1$ si y solo si el usuario i ha visto la película j. Sobre esta matrix se calcula la similitud del coseno para todos los usuarios, y se obtiene su promedio, s. La personalización viene dada por $P = 1 - s$.
- **Similitud intra-lista.** La similitud intra-lista mide cómo de parecidas son las recomendaciones para un mismo usuario. Para ello, simplemente se calcula, para cada lista de recomendaciones, la similitud media entre las películas recomendadas. La similitud intra-lista final viene dada por el promedio de las similitudes en las listas de cada usuario. Como métrica de similitud se han utilizado las dos métricas de similitud implementadas en el sistema basado en contenido: la basada en descripción y la basada en créditos, géneros y palabras clave.
- **Novedad.** La novedad es una medida basada en la teoría de la información mide la cantidad media de capacidad de sorpresa que presentan las listas de recomendaciones. Para un usuario U, viene dada por

$$N_U = \sum_{i \in rec(U)} \frac{p_i \log p_i}{|rec(U)|},$$

donde $rec(U)$ es la lista de recomendaciones para el usuario U y p_i es la frecuencia de la película i en la lista total de recomendaciones de todos los usuarios. Por tanto, N_U mide la entropía o cantidad de incertidumbre que presentan las recomendaciones para el usuario U, con respecto a la distribución de frecuencias de los elementos recomendados, lo que puede entenderse como la capacidad de sorprender que tienen las recomendaciones. La novedad global se calcula como el promedio de los N_U para cada usuario U.

3.1.1. Resultados

Las Figuras 1, 2, 3, 4 y 5 comparan, respectivamente, las medidas de cobertura, personalización, similitud intra-lista de descripción, similitud intra-lista de créditos, géneros y autores, y novedad, obtenidas sobre los distintos sistemas de recomendación implementados. Se han utilizado las 10 primeras recomendaciones proporcionadas por cada sistema para cada usuario en el cálculo de estas medidas.

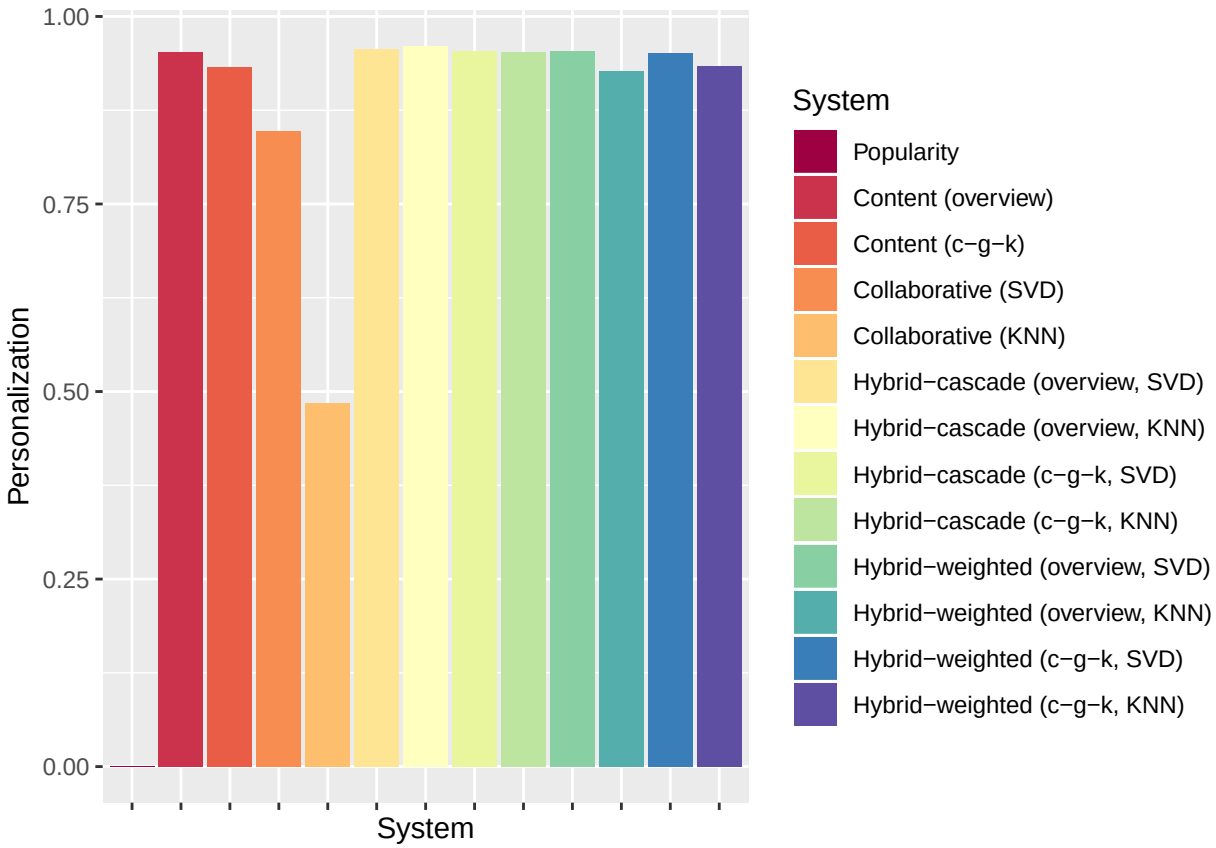

Figura 2: Personalización obtenida en los distintos sistemas de recomendación.

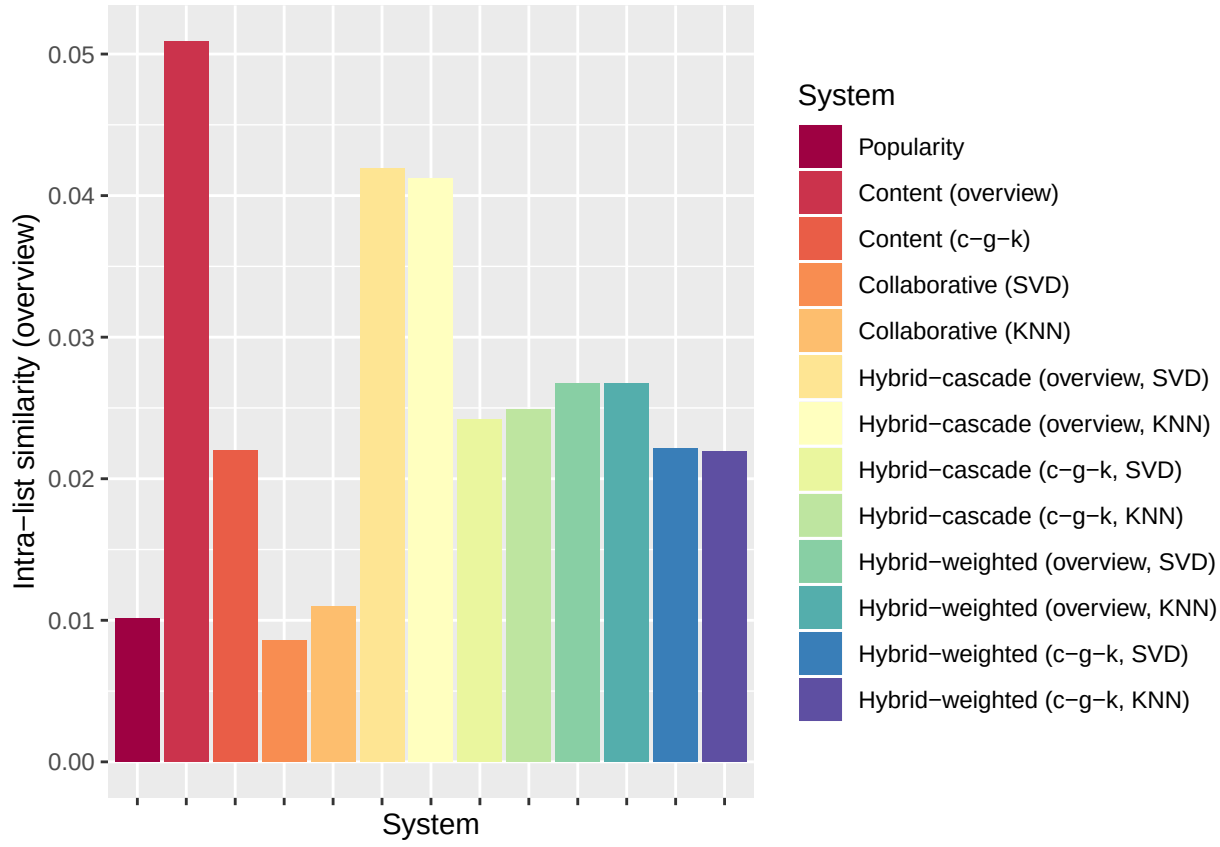

Figura 3: Similitud intra-lista con la similitud basada en argumento obtenida en los distintos sistemas de recomendación.

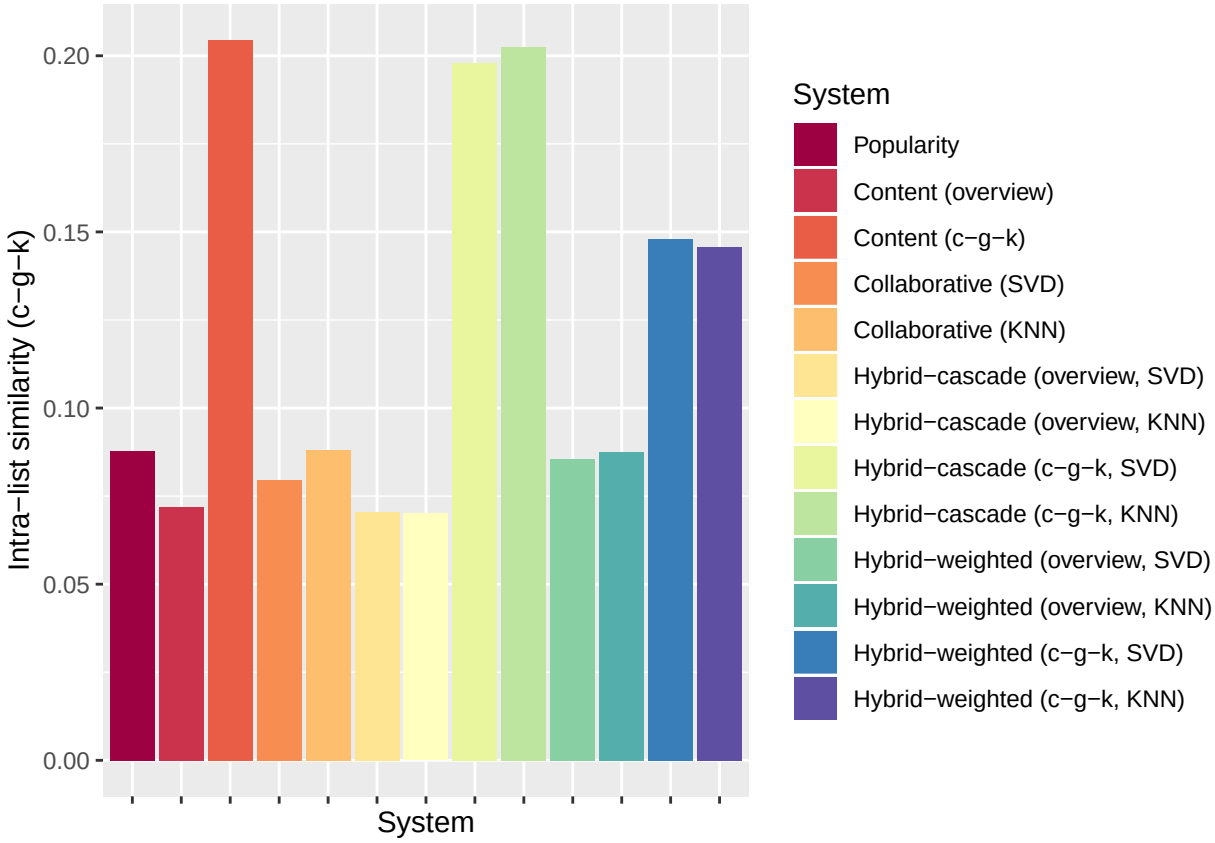

Figura 4: Similitud intra-lista con la similitud basada en créditos, géneros y palabras clave obtenida en los distintos sistemas de recomendación.

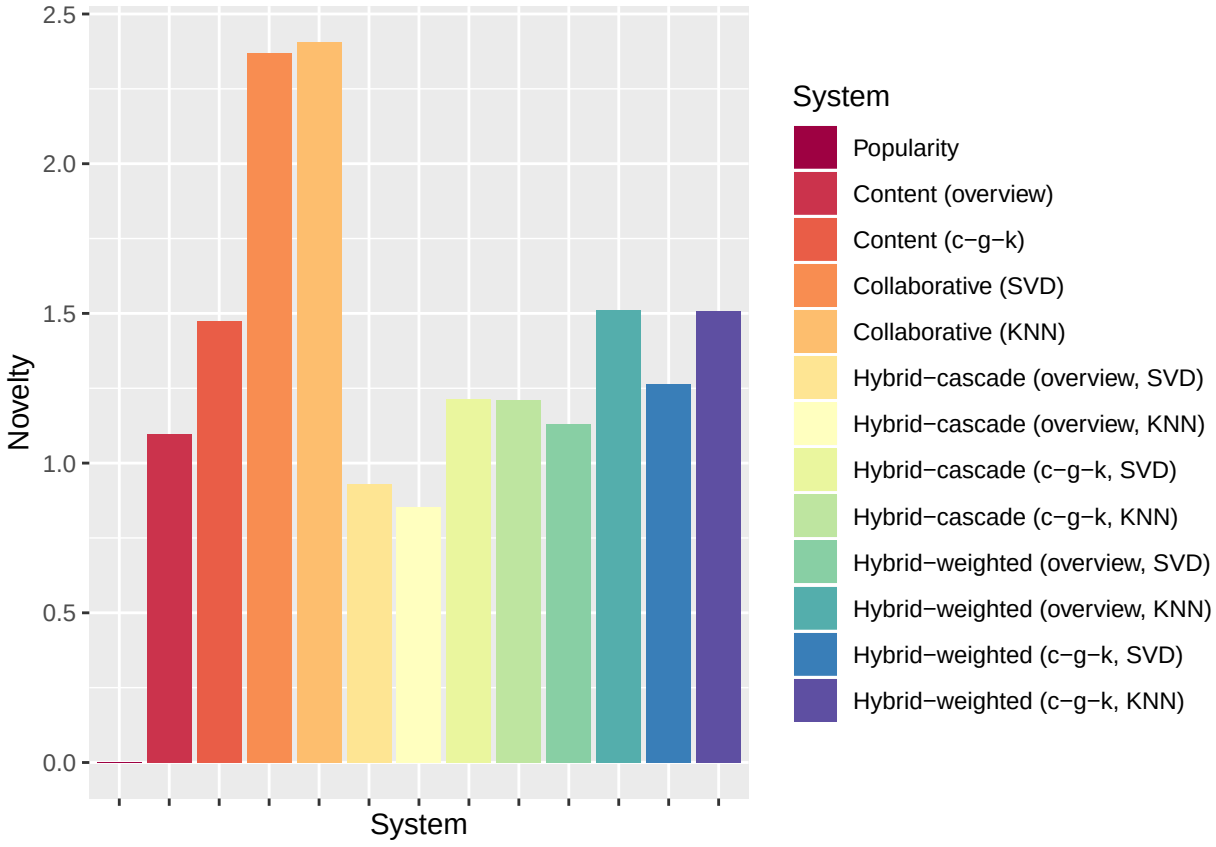

Figura 5: Novedad obtenida en los distintos sistemas de recomendación.

Si nos fijamos en la cobertura, vemos que en general los valores son muy bajos, estando el sistema con mayor cobertura en torno al 4.5 % del total de películas. Sin embargo, si tenemos en cuenta que el conjunto de datos de usuarios utilizado es reducido y está restringido a valoraciones sobre 9000 películas de la base de datos, la cobertura con respecto a esas 9000 películas sería 5 veces más a las mostradas, llegando el sistema con mayor cobertura al 22.5 % del total de películas. Podemos ver que, como era de esperar, el sistema basado en popularidad apenas tiene cobertura, pues recomenda siempre las mismas 10 películas. También vemos que la cobertura de los sistemas colaborativos es bastante baja, especialmente el que utiliza KNN, mientras que las coberturas más altas se alcanzan con los sistemas que utilizan la similitud de las descripciones de las películas, tanto el de contenido, como (especialmente) los dos híbridos en cascada. Esto muestra que la similitud basada en las descripciones de películas es capaz de expandir en mayor medida el rango de películas que se muestran en las recomendaciones.

Si observamos la personalización, podemos ver que los valores son en general bastante altos, con la excepción del basado en popularidad, cuya personalización es nula, al proporcionar siempre las mismas recomendaciones, y el colaborativo que utiliza KNN, que no llega a superar el valor de 0.5. Todos los sistemas basados en contenido e híbridos llegan a superar el valor de 0.9 de personalización.

En cuanto a las similitudes intra-lista, observamos unos valores bastante bajos en general. Esto muestra que la lista de recomendaciones proporcionada a cada usuario suele ser diversa, teniendo en general películas variadas y poco parecidas entre sí. También podemos ver que la similitud intra-lista se hace claramente superior en los sistemas que utilizan la misma medida de similitud para elaborar las recomendaciones, lo cual es razonable, pues las películas recomendadas por esos criterios van a ser más similares a la hora de ser evaluadas de nuevo por los correspondientes criterios. Aun así, los valores en estos casos también son bastante bajos.

Finalmente, si observamos la medida de novedad, podemos ver una vez más cómo es nula para el sistema basado en popularidad, como consecuencia de recomendar siempre lo mismo, pues estas recomendaciones no introducen ninguna capacidad de sorpresa. En este caso vemos cómo se destacan de forma clara los dos sistemas colaborativos. Esto puede deberse a que, mientras que los sistemas basados en contenido y los híbridos parten de una lista de películas similares para realizar las recomendaciones, los colaborativos no miran semejanzas entre películas, y por tanto en sus recomendaciones pueden aparecer películas con una mayor probabilidad de sorprender al usuario.

3.2. Evaluación supervisada

Finalmente realizamos la evaluación supervisada de los distintos sistemas implementados. Para ello se ha realizado una validación cruzada de 5 particiones, con respecto al conjunto de usuarios y valoraciones. En cada partición se utilizan 4 de las particiones de usuarios para entrenar (si es necesario, cuando el sistema dispone de una parte colaborativa) y la partición restante para evaluar los resultados. Dentro de la partición con usuarios de test, dividimos las películas en dos partes: películas vistas, que se usarán para realizar las recomendaciones sobre los usuarios test, y películas test, que se usarán para comparar con las recomendaciones. La división de las películas sobre los usuarios test se ha realizado al 50 % y mediante el timestamp presente en el conjunto de datos, de forma que la mitad de valoraciones más antiguas se utilizará para generar recomendaciones, mientras que la mitad más reciente se utilizará para validarlas. Puesto que las valoraciones de los usuarios se establecen mediante un ranking entre 0 y 5, se ha establecido un umbral de relevancia de 3.5, de forma que una película se considera relevante para un usuario si ha sido valorada por este con una puntuación mayor o igual a 3.5. Las medidas de evaluación empleadas son las medidas clásicas supervisadas sobre sistemas de recomendación:

- **Precisión.** Mide la cantidad de películas recomendadas relevantes entre el total de películas recomendadas al usuario.
- **Recall.** Mide la cantidad de películas recomendadas relevantes entre el total de películas que el usuario ha considerado relevantes.

Aunque estas dos medidas son de las más utilizadas sobre sistemas de recomendación, con nuestros datos pueden no resultar muy adecuadas, puesto que las películas usadas como test no han estado nunca sometidas al efecto del sistema de recomendación, y por tanto son independientes de este. Si las películas de test se hubieran recogido tras implantar el sistema, la precisión y el recall sí estarían reflejando de forma apropiada el comportamiento del sistema. Por ello, se ha optado por una tercera medida que puede resultar algo más adecuada en esta situación:

- **Precisión restringida.** Mide la cantidad de películas recomendadas relevantes entre el total de películas recomendadas que ha visto el usuario.

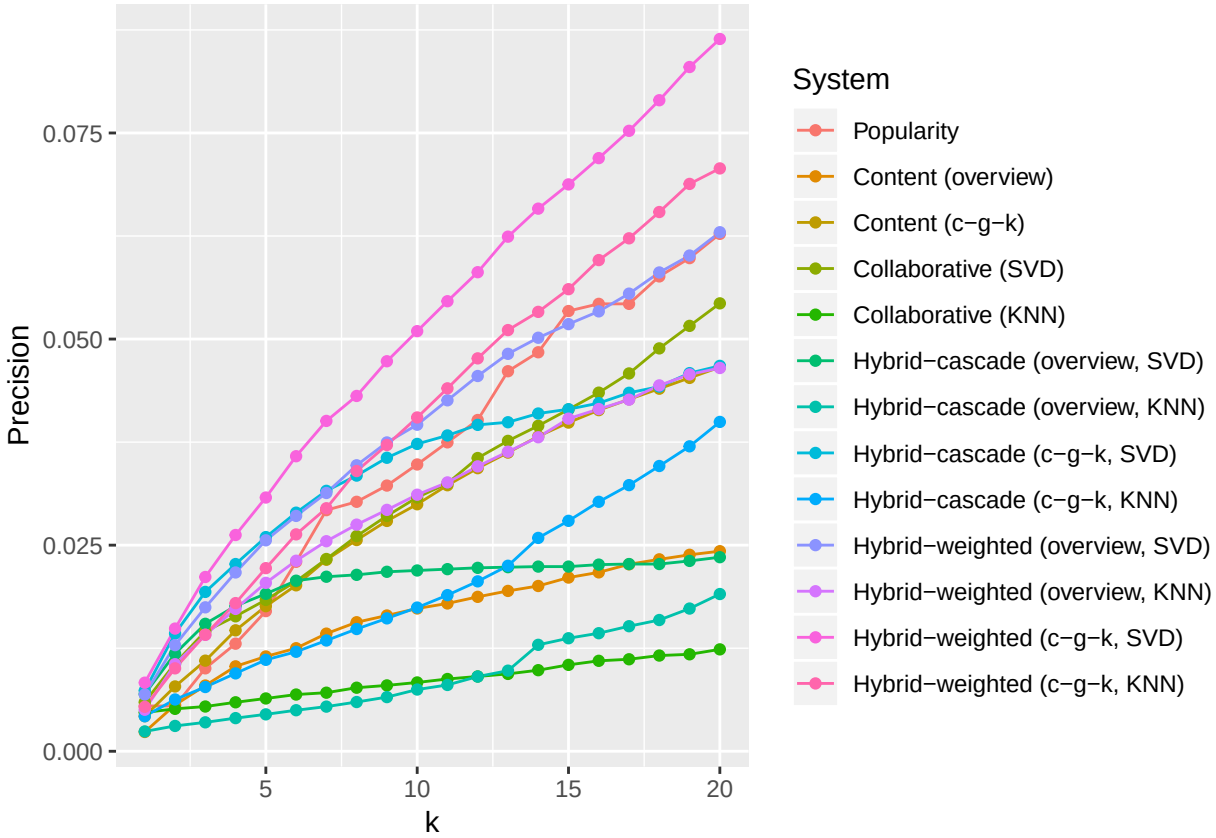

Figura 6: Precisión obtenida con las k-top recomendaciones con los distintos sistemas de recomendación.

Esta última medida nos permite observar cuáles de las recomendaciones dentro del conocimiento del usuario habrían resultado relevantes para este.

Es interesante también restringir estas medidas a las primeras k recomendaciones. Esto permite valorar la calidad de la ordenación del ranking de recomendaciones. Por ello mostraremos los resultados de precisión, recall y precisión restringida a las top k recomendaciones, para k desde 1 hasta 20.

3.2.1. Resultados

Las Figuras 6, 7 y 8 muestran, respectivamente, los valores de precisión, recall y precisión restringida a las películas vistas, obtenidos para tamaños de lista de recomendación entre 1 y 20.

Si observamos las gráficas de precisión y recall vemos que, como se comentó anteriormente, no tienen capacidad para reflejar las características de los sistemas de recomendación implementados. Los valores son muy bajos, pero esto se debe a que el sistema de recomendación no ha intervenido en la generación de los datos de test. Las películas vistas y valoradas por los usuarios son independientes de lo que hayan podido sugerir cualquiera de los sistemas. Esto lleva a que en general, la mayoría de películas recomendadas no han llegado a ser vistas pero esto no quiere decir que puedan ser irrelevantes para el usuario. Si el usuario llega a tener conocimiento de estas películas es bastante posible que las hubiera considerado relevantes. Como ejemplo podemos ver que el sistema basado en popularidad suele alcanzar los valores de precisión y recall más altos, superado solo ligeramente en precisión por los sistemas híbridos ponderados, pues las películas más populares suelen ser conocidas por los usuarios y llegan a ser valoradas (normalmente de forma positiva) más veces.

Por tanto, nos centramos en la precisión restringida. En este caso vemos, efectivamente, unos valores razonables, con la excepción del sistema basado en popularidad. Este sistema apenas supera el 30 % de precisión restringida, lo que muestra que falta calidad en sus recomendaciones. Los que alcanzan el valor más alto de precisión restringida son el sistema colaborativo y los híbridos ponderados con SVD, rozando el 0.85 desde la lista con $k = 10$ recomendaciones. Por tanto, de las películas recomendadas que ha visto el usuario a partir de la lista de 10 elementos, casi el 85 % suelen gustarle de media. El sistema híbrido ponderado con similitud basada en créditos, géneros y palabras clave y predicción con KNN sigue de cerca a los anteriores, rozando el 80 % de precisión

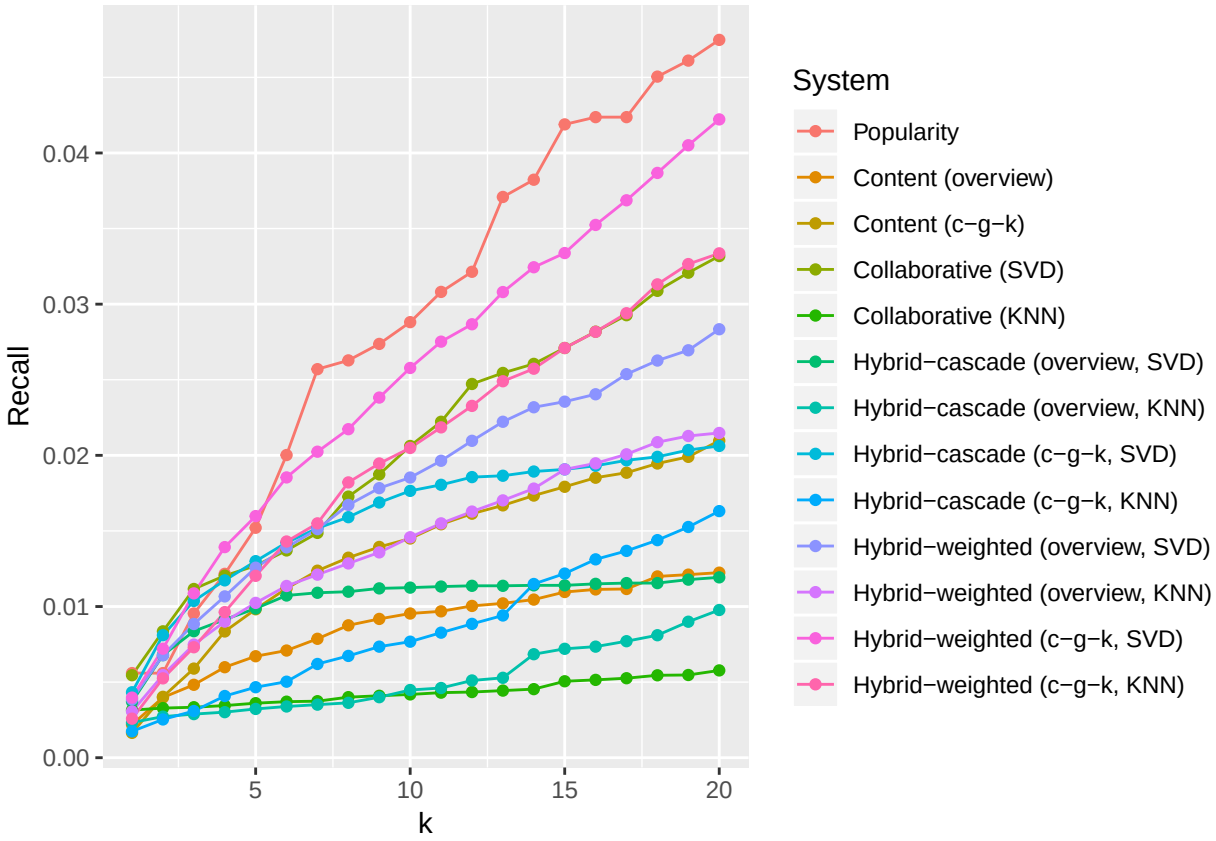

Figura 7: Recall obtenido con las k-top recomendaciones con los distintos sistemas de recomendación.

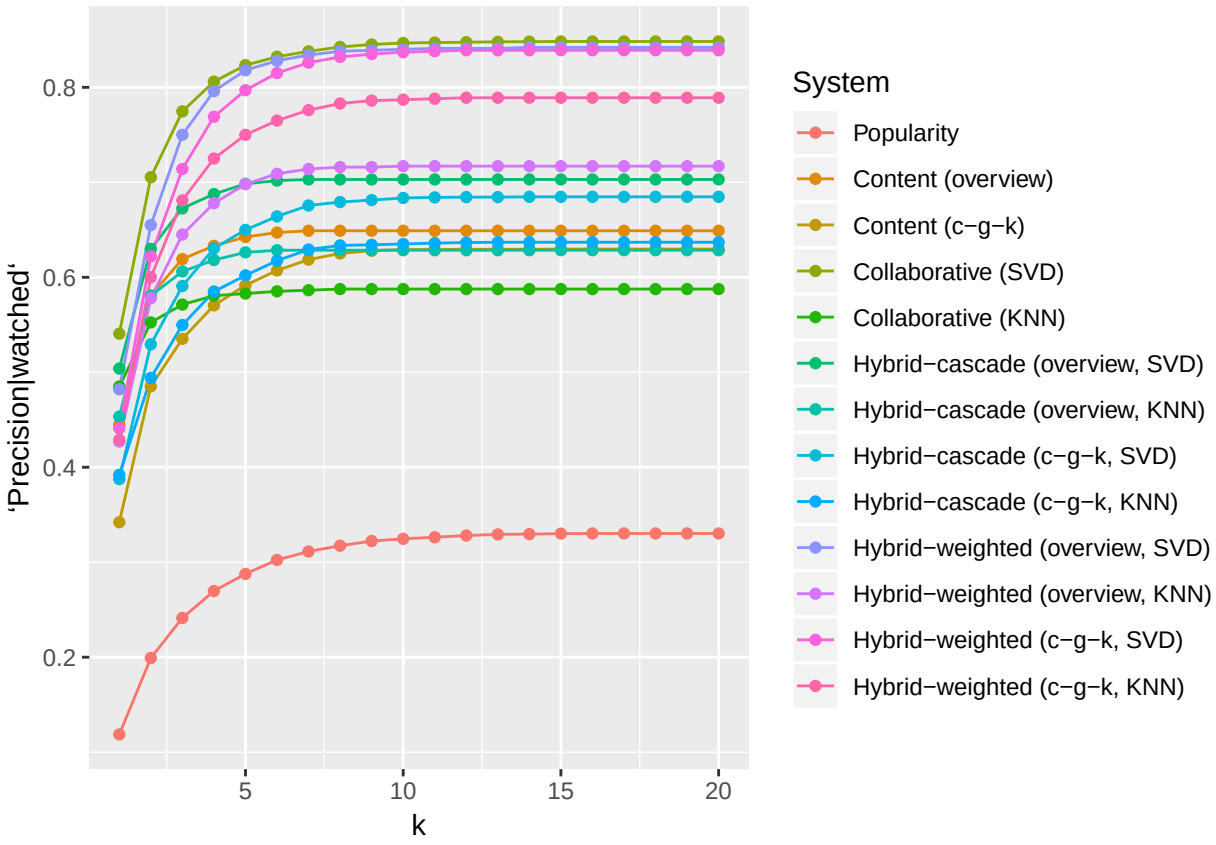

Figura 8: Precisión restringida obtenida con las k-top recomendaciones con los distintos sistemas de recomendación.

restringida. El resto de sistemas suelen alcanzar precisiones restringidas en torno al 60 % y 70 %, mostrando que tienen una tendencia algo mayor a equivocarse con sus recomendaciones. También hay que notar cómo evoluciona la precisión conforme aumenta la lista de recomendaciones. La recomendación top 1 toma valores de precisión entre el 30 % y el 55 % según el sistema, pero conforme aumenta la lista de recomendaciones la precisión crece de forma rápida hasta estabilizarse en un valor, normalmente antes de alcanzar el tamaño $k = 10$.

Por tanto, hemos visto cómo las distintas medidas utilizadas nos muestran que los distintos sistemas son capaces de destacar en cualidades concretas, dentro de las cualidades deseadas en un sistema de recomendación. Bajo el enfoque supervisado, teniendo en cuenta las limitaciones de los conjuntos de validación empleados, los que parecen proporcionar mejores resultados son los sistemas colaborativo e híbridos ponderados con SVD.

4. Manual de usuario

Para concluir, esta sección explica las principales características del software implementado y muestra los comandos y argumentos necesarios para trabajar con los distintos sistemas de recomendación.

El software elaborado está disponible en el fichero `recommender_system.py` adjunto, el cual contiene la clase `RecommenderSystem`, que implementa todas las funcionalidades descritas en este trabajo. Este fichero se apoya en el fichero `metrics.py`, que contiene las funciones que calculan las medidas de evaluación empleadas. Algunas de estas funciones son adaptaciones de las funciones que proporciona la librería `recmetrics` [3]. También hay disponible un script `test.py` que contiene las sentencias que se han utilizado para evaluar los distintos sistemas de recomendación.

El software puede utilizarse para la elaboración de nuevos scripts, importándolo en estos mediante la sentencia `from recommender_system import RecommenderSystem`, o puede utilizarse de forma interactiva en una terminal de Python. Para este último caso podemos abrir la sesión interactiva de Python escribiendo la sentencia `python -i recommender_system.py` en la línea de comandos. El software se ha elaborado y probado con las versiones de Python 3.6 y 3.7, y requiere las versiones más recientes de las librerías `numpy`, `pandas`, `scikit-learn` y `surprise`. También el dataset debe encontrarse en el mismo directorio, sobre una carpeta llamada `the-movies-dataset`, aunque puede especificarse otro nombre modificando las variables estáticas del script.

Una vez importada la librería o iniciada la sesión interactiva, el primer paso es crear un objeto `RecommenderSystem`. Durante la creación de este objeto, se cargarán los datos y se realizarán los preprocesamientos iniciales necesarios para el funcionamiento de los distintos sistemas de recomendación. Una vez cargados los datos, podemos empezar a realizar algunas consultas, como por ejemplo qué películas han visto y qué películas les han gustado a los distintos usuarios de la base de datos. Para ello, podemos utilizar los métodos `get_watched_movies(userId)` y `get_liked_movies(userId, positiveThresh)`, donde `userId` es el identificador del usuario (entre 1 y 671), y `positiveThresh` es el umbral con el que se considera que una valoración es positiva (entre 0 y 5). En la Figura 9 se ejemplifican la construcción y estas consultas iniciales.

Sin embargo, si queremos empezar a obtener recomendaciones, debemos especificar antes los métodos de obtención de similitudes (para basado en contenido e híbrido) o de entrenamiento (para basado en usuario e híbrido) a utilizar. Para el primer caso, los métodos `set_overview_similarity_metric()` y `set_cgk_similarity_metric()` permiten, respectivamente, establecer las similitudes basadas en las descripciones de las películas y en créditos, géneros y palabras clave. Para el segundo caso, los métodos `set_svd_user_training()` y `set_knn_user_training()` permiten usar los correspondientes métodos de entrenamiento. Todos estos métodos realizan la mayoría de cómputos necesarios para definir los sistemas, como el cálculo y almacenamiento de las matrices de semejanza o el aprendizaje de los modelos, de forma que a la hora de obtener las recuperaciones posteriores, el número de cálculos necesarios sea mínimo. En la Figura 10 se muestra el uso de estos métodos.

Una vez especificados los parámetros anteriores necesarios para cada sistema, podemos empezar a obtener recomendaciones. Para el sistema basado en popularidad, el método `get_popularity_recommendations(top)` nos devuelve el número de películas más populares especificado en `top`. El método `get_popularity_recommendations_for_user(userId, top)` es una envolvente que nos permite obtener las recomendaciones por popularidad para usuarios concretos, aunque en este caso podemos ver que las recomendaciones obtenidas son siempre las mismas, como se muestra en la Figura 11.

El método `get_content_recommendations(title, top)` nos permite obtener la lista de `top` películas más similares a las especificadas en `title`. El argumento `title` puede ser una cadena de texto con el título de una sola película o una lista de títulos. Para obtener recomendaciones por usuarios podemos utilizar el método

```
>>> recsys = RecommenderSystem()
Loading metadata...
Loading credits and keywords...
Preprocessing metadata...
Loading ratings...
Indexing...
>>>
>>> recsys.get_watched_movies(userId=1)
                              title  rating
0                   Dangerous Minds     2.5
1                             Dumbo     3.0
2                          Sleepers     3.0
3              Escape from New York     2.0
4                   Cinema Paradiso     4.0
5                   The Deer Hunter     2.0
6                           Ben-Hur     2.0
7                            Gandhi     2.0
8                           Dracula     3.5
9                         Cape Fear     2.0
10    Star Trek: The Motion Picture     2.5
11  Beavis and Butt-Head Do America     1.0
12            The French Connection     4.0
13                             Tron     4.0
14           The Gods Must Be Crazy     3.0
15                           Willow     2.0
16                             Antz     2.0
17                          The Fly     2.5
18                     Time Bandits     1.0
19                   Blazing Saddles     3.0
>>>
>>> recsys.get_liked_movies(userId=1, positiveThresh=3.5)
                        title  rating
0        Cinema Paradiso     4.0
1                 Dracula     3.5
2  The French Connection     4.0
3                    Tron     4.0
>>>
```

Figura 9: Inicialización del sistema de recomendación y consultas básicas.

```
>>> recsys.set_overview_similarity_metric()
Creating new similarity matrix...
>>>
>>> recsys.set_cgk_similarity_metric()
Creating new similarity matrix...
>>>
>>> recsys.set_svd_user_training()
Training SVD...
>>>
>>> recsys.set_knn_user_training()
Training KNN...
Computing the msd similarity matrix...
Done computing similarity matrix.
>>>
```

Figura 10: Configuración de las medidas de similitud y los métodos de entrenamiento a utilizar para los distintos sistemas.

`get_content_recommendations_for_user(userId, top, positiveThresh)`, el cual obtiene las películas más similares a las que el usuario ha visto y le han gustado, según el umbral dado por `positiveThresh`. Este método utilizará la similitud que se haya especificado previamente con el método `set_XXXX_similarity_metric()` correspondiente. La Figura 12 muestra ejemplos de recomendaciones basadas en contenido.

El método `get_collaborative_recommendations_for_user(userId, top)` muestra, para el usuario especificado, las recomendaciones obtenidas mediante el sistema colaborativo, es decir, aquellas para las que la estimación de valoración obtenida por el algoritmo de aprendizaje es mayor. De nuevo, este método utilizará como algoritmo de predicción aquel que se haya especificado previamente con el método `set_XXXX_user_training()` correspondiente. La Figura 13 muestra ejemplos de recomendaciones colaborativas.

En cuanto al sistema híbrido, el método `get_hybrid_cascade_recommendations_for_user(userId, top, content_top, positiveThresh)` permite obtener las recomendaciones combinando las de los sistemas colaborativo y basado en contenido en cascada, mientras que `get_hybrid_weighted_recommendations_for_user(userId, top, positiveThresh)` los combina por ponderaciones según el número de películas vistas. El parámetro `content_top` indica el número de recomendaciones a escoger del sistema basado en contenido, para el caso de la hibridación en cascada. Inicialmente se tomarán dicho número de recomendaciones de entre las películas más similares a las que le han gustado al usuario. Después el sistema colaborativo escogerá de entre esas películas el número indicado por `top` con mayor estimación de valoración. Estos métodos utilizarán la similitud que haya sido especificada con el método `set_XXXX_similarity_metric()` correspondiente y el algoritmo de predicción especificado con el método `set_XXXX_user_training()`. La Figura 14 muestra ejemplos de estas recomendaciones híbridas.

Finalmente, la clase `RecommenderSystem` proporciona los mecanismos necesarios para calcular las me-

```
>>> recsys.get_popularity_recommendations(top=5)
                          title  vote_count  vote_average     score
314       The Shawshank Redemption      8358.0           8.5  8.447079
10397  Dilwale Dulhania Le Jayenge       661.0           9.1  8.433941
841                The Godfather      6024.0           8.5  8.427092
12589              The Dark Knight     12269.0           8.3  8.266248
2870                    Fight Club      9678.0           8.3  8.257355
>>>
>>> recsys.get_popularity_recommendations_for_user(1, top=5)
                          title  vote_count  vote_average     score
314       The Shawshank Redemption      8358.0           8.5  8.447079
10397  Dilwale Dulhania Le Jayenge       661.0           9.1  8.433941
841                The Godfather      6024.0           8.5  8.427092
12589              The Dark Knight     12269.0           8.3  8.266248
2870                    Fight Club      9678.0           8.3  8.257355
>>> recsys.get_popularity_recommendations_for_user(2, top=5)
                          title  vote_count  vote_average     score
314       The Shawshank Redemption      8358.0           8.5  8.447079
10397  Dilwale Dulhania Le Jayenge       661.0           9.1  8.433941
841                The Godfather      6024.0           8.5  8.427092
12589              The Dark Knight     12269.0           8.3  8.266248
2870                    Fight Club      9678.0           8.3  8.257355
```

Figura 11: Recomendación basada en popularidad. Podemos observar cuáles son las películas mejor valoradas a nivel global de la base de datos. También vemos cómo las recomendaciones son siempre las mismas para todos los usuarios.

```
>>> recsys.set_overview_similarity_metric()
Creating new similarity matrix...
>>> recsys.get_content_recommendations([
...     'Pirates of the Caribbean: The Curse of the Black Pearl',
...     'Charlie and the chocolate factory'])
                                    title  similarity
0                     The Golden Hawk    0.239585
1       The Legend of the Candy Cane    0.212481
2                     The Black Swan    0.210171
3                         The Pirate    0.204521
4     Arabella, the Pirate's Daughter    0.200829
5   Willy Wonka & the Chocolate Factory    0.199479
6               The Boy and the Pirates    0.198282
7  Pirates of the Caribbean: On Stranger Tides    0.197687
8                 Romantics Anonymous    0.182286
9                         Bad Girls    0.179633
>>>
```

```
>>> recsys.set_cgk_similarity_metric()
Creating new similarity matrix...
>>> recsys.get_content_recommendations([
...     'Pirates of the Caribbean: The Curse of the Black Pearl',
...     'Charlie and the chocolate factory'])
                                    title  similarity
0          Pirates of the Caribbean: At World's End    0.700000
1       Pirates of the Caribbean: Dead Man's Chest    0.600000
2  Pirates of the Caribbean: Dead Men Tell No Tales    0.500000
3                         Behind Enemy Lines    0.447214
4                              Radiopiratene    0.447214
5                                    Uncle P    0.447214
6                               Mystery Men    0.400000
7                                 Ned Kelly    0.400000
8                          Alice in Wonderland    0.400000
9       Pirates of the Caribbean: On Stranger Tides    0.400000
>>>
```

```
>>> recsys.get_content_recommendations_for_user(2, positiveThresh=3.5)
                                    title  similarity
0  The Lost World: Jurassic Park    0.737865
1                  Patriot Games    0.700000
2                The Terminator    0.572078
3                Batman Returns    0.555556
4      Cirque du Soleil: Varekai    0.547723
5                      Last Exit    0.547723
6        The Ghost and the Whale    0.547723
7         Puff, the Magic Dragon    0.547723
8                           Rege    0.547723
9                Yedyanchi Jatra    0.547723
>>>
```

Figura 12: Recomendaciones con el sistema basado en contenido. A la izquierda, usando la similitud basada en la descripción de la película. En el centro, usando la similitud basada en créditos, géneros y palabras clave. Podemos ver que en el primer caso prácticamente todas las recomendaciones son sobre películas que tratan de piratas o en las que aparecen fábricas de chocolate. En el segundo caso aparecen películas que tratan otros temas, pero que comparten géneros o actores (en este caso Johnny Depp) con las películas indicadas. A la derecha se muestran las recomendaciones para un usuario concreto, de acuerdo con las películas que ha valorado positivamente.

didas de evaluación de los sistemas. Para la evaluación no supervisada, se disponen los métodos `evaluate_popularity_recommendations(top)`, `evaluate_content_recommendations(top, positiveThresh)`, `evaluate_collaborative_recommendations(top)`, `evaluate_hybrid_cascade_recommendations(top, content_top, positiveThresh)` y `evaluate_hybrid_weighted_recommendations(top, positiveThresh)`. Estos métodos calculan las recomendaciones para todos los usuarios según los parámetros especificados y calculan las métricas no supervisadas para dichas listas. Para la evaluación supervisada, se disponen los métodos `validate_popularity_recommendations(n_folds, movie_train, top, positiveThresh, random_state)`, `validate_content_recommendations(n_folds, movie_train, top, positiveThresh, random_state)`, `validate_collaborative_recommendations(n_folds, movie_train, top, positiveThresh, random_state)`, `validate_hybrid_cascade_recommendations(n_folds, movie_train, top, content_top, positiveThresh, random_state)` y `validate_hybrid_weighted_recommendations(n_folds, movie_train, top, positiveThresh, random_state)`. Los parámetros `top` y `content_top` vuelven a especificar el tamaño de las recomendaciones tal y como se explicó previamente; `n_folds` indica el número de particiones para la validación cruzada, `movie_train` indica la fracción de películas que se va a utilizar para calcular las recomendaciones sobre los usuarios de test, `positiveThresh` es el umbral que indica cuándo una película se considera relevante para el usuario (tanto a la hora de recomendar como a la hora de validar), y `random_state` es la semilla aleatoria utilizada para generar las particiones. Los métodos, tanto supervisados como no supervisados, devuelven un diccionario, donde las claves son el nombre de las distintas métricas, y los valores son el valor numérico de dicha métrica, para el caso de los no supervisados, o una lista de valores, desde 1 hasta `top`, para el caso de la precisión, recall y precisión restringida. En la Figura 15 se muestra el uso de los métodos de validación.

```
>>> recsys.get_collaborative_recommendations_for_user(2, top=5)
                  title  estimation
1930       Seven Samurai    4.558221
900   The Maltese Falcon    4.512047
1182        12 Angry Men    4.466737
5947         City of God    4.445690
3369        Modern Times    4.445577
>>> recsys.get_collaborative_recommendations_for_user(3, top=5)
                      title  estimation
841            The Godfather    4.399836
1856       On the Waterfront    4.378583
734             A Close Shave    4.363280
885   The Philadelphia Story    4.311532
49         The Usual Suspects    4.274739
>>> recsys.get_collaborative_recommendations_for_user(4, top=5)
                      title  estimation
3369         Modern Times         5.0
23075        Interstellar         5.0
913          All About Eve         5.0
49     The Usual Suspects         5.0
600                  Fargo         5.0
```

Figura 13: Recomendaciones del sistema colaborativo (con SVD) para distintos usuarios.

```
>>> recsys.get_hybrid_cascade_recommendations_for_user(2, top=5, content_top=25)
                        title  estimation
2553               Frankenstein    3.884741
12231  The Jane Austen Book Club    3.821778
3165              Patriot Games    3.732831
17202      Voices from the List    3.692205
15010               Persuasion    3.675946
>>> recsys.get_hybrid_cascade_recommendations_for_user(3, top=5, content_top=25)
                            title  estimation
11220            Stranger Than Fiction    3.675638
17818  A Very Harold & Kumar Christmas    3.476360
5062                 We Were Soldiers    3.428025
4737                  The Big Red One    3.412723
20157                 The Awful Truth    3.395446
>>> recsys.get_hybrid_weighted_recommendations_for_user(2, top=5)
                        title  hybrid_score
1313       Bride of Frankenstein      0.642377
12231  The Jane Austen Book Club      0.599542
1216              The Terminator      0.587296
1254          Young Frankenstein      0.570474
3165              Patriot Games      0.568356
>>> recsys.get_hybrid_weighted_recommendations_for_user(3, top=5)
                            title  hybrid_score
4737                  The Big Red One      0.454909
11220            Stranger Than Fiction      0.443315
17818  A Very Harold & Kumar Christmas      0.437154
2259                    The Big Chill      0.424724
10523               Angels in America      0.423016
```

Figura 14: Recomendaciones de los sistemas híbridos (con similaridad por descripciones y SVD) para distintos usuarios.

```
>>> recsys.evaluate_content_recommendations(top=10, positiveThresh=3.5)
Obtaining recommendations for every user (this may take a while)...
Preprocessing...
Calculating coverage...
Calculating personalization...
Calculating intra-list overview similarity...
Calculating intra-list CGK similarity...
Calculating novelty...
{'coverage': 0.03140013726835964, 'personalization': 0.9521676268434281, 'intralist_overview_similarity': 0.050898866284827586, 'intral
ist_cgk_similarity': 0.07197151916799592, 'novelty': 1.0974287944022798}
>>>
>>> recsys.validate_content_recommendations(n_folds=5, movie_train=0.5, top=20, positiveThresh=3.5,
...     random_state=28)
Partitioning...
Validating Fold 1...
Obtaining recommendations for every user (this may take a while)...
Obtaining test values (1/3)...
Obtaining test values (2/3)...
Obtaining test values (3/3)...
Evaluating...
Validating Fold 2...
Obtaining recommendations for every user (this may take a while)...
Obtaining test values (1/3)...
Obtaining test values (2/3)...
Obtaining test values (3/3)...
Evaluating...
Validating Fold 3...
Obtaining recommendations for every user (this may take a while)...
Obtaining test values (1/3)...
Obtaining test values (2/3)...
Obtaining test values (3/3)...
Evaluating...
Validating Fold 4...
Obtaining recommendations for every user (this may take a while)...
Obtaining test values (1/3)...
Obtaining test values (2/3)...
Obtaining test values (3/3)...
Evaluating...
Validating Fold 5...
Obtaining recommendations for every user (this may take a while)...
Obtaining test values (1/3)...
Obtaining test values (2/3)...
Obtaining test values (3/3)...
Evaluating...
{'precision': 1     0.002342
2     0.005617
3     0.007945
4     0.010302
5     0.011500
6     0.012504
7     0.014270
8     0.015656
9     0.016477
10    0.017328
11    0.017933
12    0.018739
13    0.019454
14    0.020059
15    0.021077
16    0.021703
17    0.022684
18    0.023282
19    0.023838
20    0.024277
dtype: float64, 'recall': 1     0.002188
2     0.003982
```

Figura 15: Ejemplo de evaluación de un sistema de recomendación y resultados.

Referencias

[1] R. Banik. *The Movies Dataset. Metadata on over 45,000 movies. 26 million ratings from over 270,000 users.* https://www.kaggle.com/rounakbanik/the-movies-dataset

[2] N. Hug. *Surprise. A Python scikit building and analyzing recommender systems.* http://surpriselib.com/

[3] C. Longo. *Recmetrics. A Python library with diagnostic and evaluation metrics useful for evaluating recommender systems.* https://github.com/statisticianinstilettos/recmetrics

[4] S. Vargas, *Novelty and Diversity Enhancement and Evaluation in Recommender Systems (Trabajo de Fin de Máster).* Universidad autónoma de Madrid, 2012.