

Ana María Ospina Arredondo

Brahayan Stiven Gil Henao

Jhon Alexander Bedoya Carvajal

Sebastián Salamanca Méndez

INFORME SISTEMAS DE RECOMENDACIÓN.

1. Diseño de Solución

a. Problema del negocio:

- Conseguir nuevos clientes
- Fidelización y mejora en la satisfacción de los clientes
- Que los usuarios usen más la plataforma para mantener la competitividad

b. Problema analítico:

Diseñar e implementar un grupo de sistemas de recomendación efectivo que utilice los datos disponibles para ofrecer recomendaciones de películas a los usuarios en función de sus preferencias y comportamientos pasados de visualización y calificación. Estos sistemas realizarán las siguientes recomendaciones:

- Top 10 películas más vistas
- Top 10 películas mejor calificadas
- Top 10 películas más vistas por década de estreno
- Top 10 películas mejor calificadas por década de estreno
- Top 10 películas más vistas durante el mes actual
- Top 10 películas más vistas durante el año actual
- Top 10 películas más vistas por género
- Top 10 películas mejor calificadas por género
- 10 películas más similares a una vista por el usuario
- 10 películas más similares a todo el contenido visto por usuario
- 10 películas con mayor calificación predicha.

c. Implementación:

- Se tomarán datos diarios de las películas vistas por los usuarios para actualizar el sistema de recomendación basado en popularidad.
- Diariamente se toma la información de películas vistas por el usuario, se actualiza su perfil y se actualiza la tabla con las recomendaciones por usuarios, la tabla se deja en una ruta que es tomada por la aplicación para actualizar el menú de recomendaciones personalizadas de cada usuario.
- Cada semana, se toma la información de películas nuevas agregadas y se actualiza una tabla en la que se indica cuáles son para cada película las 10 películas más parecidas, esta tabla se deja en una carpeta que es subida a la aplicación, con base en esta tabla cada vez que un usuario termina una película se despliegan las 10 películas recomendadas.

- En el inicio de la aplicación se dejará una sección de "películas recomendadas que más te van a gustar", cuando el usuario inicia sesión y cada que vuelve al inicio de la aplicación, la plataforma consume un api en el que se tiene expuesto un servicio con un modelo que predice los ratings de las películas no vistas del usuario y devuelve las 10 con mayor rating. Esta API con el modelo entrenado se actualiza semanalmente por parte del área de analítica.
- Se tendrá una tabla en la base de datos que contenga los estrenos de la semana siguiente y los usuarios puedan agregarla a una lista de pendientes que le notifique una vez esté disponible la película. Esta tabla se actualizará semanalmente.

2. Exploración y Limpieza de los datos

Para abordar la problemática en cuestión, se disponía de dos tablas alojadas en una base de datos SQL denominadas "ratings" y "movies".

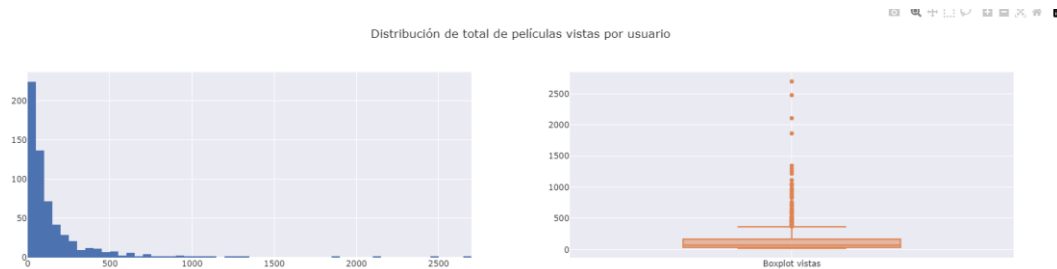
En una fase inicial, se procedió a explorar la información contenida en ambas tablas. En primera instancia se hizo una exploración de la tabla "movies", encontrando que la columna "genres" contenía todos los géneros de la película delimitados por "|", así:

movieid		title	genres
0	1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
1	2	Jumanji (1995)	Adventure Children Fantasy
2	3	Grumpier Old Men (1995)	Comedy Romance
3	4	Waiting to Exhale (1995)	Comedy Drama Romance
4	5	Father of the Bride Part II (1995)	Comedy

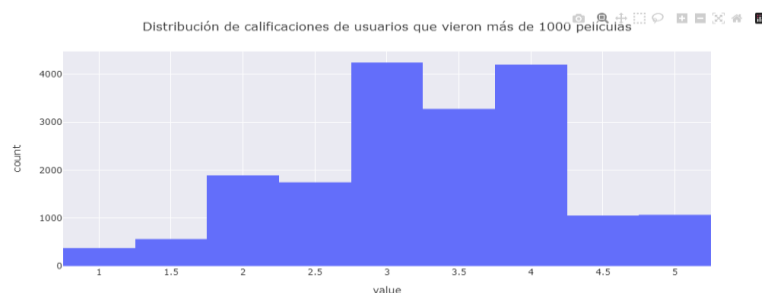
Esto se dejó para transformarla en la parte de limpieza y preprocesamiento. Luego, se hizo una exploración de la tabla "ratings", encontrando que la columna "timestamp" debe ser formateada porque es una fecha en milisegundos. Adentrándonos un poco más en la tabla "ratings" encontramos que, aunque el rating debe ser una calificación de 1 a 5, había calificaciones de 0.5:

	rating	Frecuencia
0	0.5	1370
1	1.0	2811
2	1.5	1791
3	2.0	7551
4	2.5	5550
5	3.0	20047
6	3.5	13136
7	4.0	26818
8	4.5	8551
9	5.0	13211

Estas calificaciones de 0.5 se tratarán como calificaciones implícitas, es decir, en 1370 ocasiones una película fue vista y no calificada. Por último, observamos la distribución del número de película vistas por un usuario:



Hay un sesgo claro, unos pocos usuarios han visto más de mil películas, pero decidimos no tratarlos porque no va a afectar los resultados y perderíamos mucha información, 18517 calificaciones. Además, exploramos esos usuarios y no tienen un patrón de calificaciones anormal:



3. Limpieza y Preprocesamiento:

Posteriormente, se inició un proceso de preprocesamiento de los datos en SQL con el objetivo de estructurarlos de manera más adecuada para su posterior uso en los modelos. En este contexto, se crearon dos nuevas tablas: "movies2" y "ratings2". En la tabla "movies2", se extrajeron elementos clave de la tabla original de películas, como el año de lanzamiento, el título y los géneros, los cuales fueron representados como columnas individuales para facilitar su manejo y comprensión. De manera similar, en la tabla "ratings2", se llevó a cabo una modificación en el formato de la fecha para que coincidiera con el estándar Y/M/D. Al final se unieron las 2 tablas en una sola llamada "movies_rating", la cual nos sirve para tener una visión o comprensión más completa de los datos, esto nos sirvió para modificar algunas variables a tipo objeto y llevar a cabo un análisis exploratorio de los datos completos mirando la distribución de las de las películas vistas por género.

4. Modelos:

Sistema de recomendación basado en popularidad

Para obtener sistemas útiles para el usuario por el cuál pueda interactuar y ayudarlo a decidir cuál será la siguiente película que quiere ver, se diseñaron algunos rankings basados en popularidad mencionados en el diseño de solución (*10 más vistas*, *10 mejor calificadas*, *10 más vistas por década de estreno*, *10 mejor calificadas por década de estreno*, *10 más vistas durante el mes actual*, *10 más vistas durante el año actual*, *10 más vistas por género*, *10 mejor calificadas por género*). En los sistemas que se obtiene un top basado en películas mejores calificadas, se buscó tener un mínimo de 30 calificaciones para obtener resultados con un mayor nivel de criterio y validez. Los sistemas basados en número de vistas son top 10 comunes sin alguna condición en específico.

Sistema de recomendación basado en contenido

Para este sistema, se implementó un modelo KNN para una sola película vista por el usuario y usando la métrica del coseno. Se basa en identificar recomendaciones de películas con características similares a la película seleccionada de las que hayan sido vistas por el usuario.

movies_name	Beverly Hills Cop (1984)	movies_name	Beverly Hills Cop (1984)
	Better Tomorrow II, A (Ying hung boon sik II) (1987)		Movie
	Better Tomorrow, A (Ying hung boon sik) (1986)	1	48 Hrs. (1982)
1	Betting on Zero (2016)	2	Lethal Weapon (1987)
2	Between Your Legs (Entre las piernas) (1999)	3	Fatal Beauty (1987)
3	Between the Folds (2008)	4	Miracles - Mr. Canton and Lady Rose (1989)
4	Beverly Hills Cop (1984)	5	Lethal Weapon 2 (1989)
5	Beverly Hills Cop II (1987)	6	Lethal Weapon 3 (1992)
6	Beverly Hills Cop III (1994)	7	Sonatine (Sonachine) (1993)
7	Beverly Hills Ninja (1997)	8	Best Men (1997)
8	Bewitched (2005)	9	Knockin' on Heaven's Door (1997)
9	Beyond Bedlam (1993)	10	High Heels and Low Lives (2001)
10	Beyond Borders (2003)		
	Beyond Rangoon (1995)		
	Beyond Re-Animator (2003)		
	Beyond Silence (Jenseits der Stille) (1996)		
	Beyond the Clouds (Al di là delle nuvole) (1996)		

En este modelo esperábamos que al seleccionar una película son varias partes, el modelo recomendara sus otras partes. Como sucede al seleccionar Beverly Hills Cop (1984), el modelo no recomienda sus otras partes, podría ser que el año influye mucho o simplemente las otras partes en realidad tienen poca similitud con la seleccionada en cuanto a géneros. Recomendar las otras partes de una película que tiene varias puede ser un ajuste a este sistema o por si mismo un nuevo sistema de recomendación.

KNN todas las películas vistas por el usuario

Este sistema se basa en recomendar las películas a través de todas las películas que ha visto el usuario, donde según sus calificaciones a estas, el año de estreno y sus géneros, busca las películas que sean similares, cerca al centroide de sus calificaciones y, en este caso, con la métrica del coseno. Recomienda las películas que según su perfil pueden gustarle al usuario y que no se ha visto.

user_id	4		
	title	movieid	genres
9482	Denis Leary: No Cure for Cancer (1993)	169912	Comedy
9502	Life-Size (2000)	170837	Children Comedy Fantasy
9276	The Angry Birds Movie (2016)	157340	Animation Comedy
9434	Split (2017)	166534	Drama Horror Thriller
9308	Café Society (2016)	159193	Comedy Drama Romance
9437	Passengers (2016)	166635	Adventure Drama Romance Sci-Fi
9398	Amanda Knox (2016)	164540	Documentary
9230	War and Peace (2016)	152284	Drama Romance
9089	Silence (2016)	143367	Drama Thriller
9235	Hunt for the Wilderpeople (2016)	152970	Adventure Comedy
9209	The Survivorist (2015)	151695	Drama Sci-Fi Thriller

Películas vistas	
Drama	120
Comedy	104
Romance	58
Thriller	38
Adventure	29
Crime	27
Action	25
Mystery	23
Fantasy	19
Musical	16

10 películas recomendadas

Géneros más vistos por el usuario

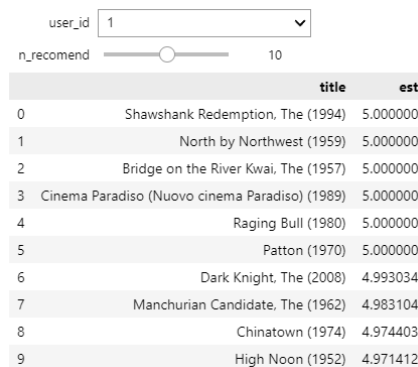
Haciendo un análisis por género, por ejemplo, para el usuario con id 4 recomienda las 10 películas de la derecha de la ilustración arriba y a la izquierda vemos los géneros más vistos por ese usuario. Drama, comedia y romance son los géneros más vistos por ese usuario y 9 de las 10 películas recomendadas por el sistema tienen alguno de estos 3 géneros.

Filtros Colaborativos

Se define un sistema mediante filtros colaborativos para darle calificaciones a las películas que no ha visto el usuario mediante las calificaciones que les han dado otros usuarios con gustos similares y se seleccionan las 10 con calificación más alta para cada usuario. Para este sistema se probaron los modelos KNN Basic, KNN With Means, KNN With Zscorey KNNBaseline usando una validación cruzada y con las métricas MAE y RMSE, obteniendo los siguientes resultados:

	MAE	RMSE	fit_time	test_time
knns.KNNBaseline	0.641356	0.829151	0.334926	1.938721
knns.KNNWithZScore	0.652017	0.848891	0.310571	1.703372
knns.KNNWithMeans	0.656976	0.849829	0.305051	1.563180
knns.KNNBasic	0.698158	0.896946	0.288089	1.386502

Elegimos el KNN Baseline ya que es el que tiene mejores resultados (menores) en MAE y RMSE. Se exporta el modelo con los mejores parámetros para luego ser usado en el despliegue.



	title	est
0	Shawshank Redemption, The (1994)	5.000000
1	North by Northwest (1959)	5.000000
2	Bridge on the River Kwai, The (1957)	5.000000
3	Cinema Paradiso (Nuovo cinema Paradiso) (1989)	5.000000
4	Raging Bull (1980)	5.000000
5	Patton (1970)	5.000000
6	Dark Knight, The (2008)	4.993034
7	Manchurian Candidate, The (1962)	4.983104
8	Chinatown (1974)	4.974403
9	High Noon (1952)	4.971412

Ejemplo de recomendación con filtros colaborativos

5. Despliegue:

Se automatiza todo el proceso descrito en los apartados anteriores, la transformación de los datos, las consultas a la base de datos y la implementación de los modelos. Se crea una función para cada modelo, las cuales retornarán un dataframe, es decir, se tendrá un dataframe por cada sistema de recomendación con las recomendaciones para cada usuario. Por último, estos dataframes se exportan como archivos .csv a una carpeta compartida entre las personas interesadas en esta información para que lo puedan consultar.

Todo este proceso se ejecutará de manera automática:

- Diariamente a las 5 AM programando dicha ejecución automática para el Script e_Despliegue_popularidad_y_KNN_todas.py para los sistemas basados en popularidad y KNN con todas las películas.
- Para los sistemas KNN y de Filtros Colaborativos, semanalmente programando la ejecución automática del Script f_Despliegue_KNN_una_y_Filtros_colab.py los viernes a las 5 AM para tener los estrenos y las nuevas recomendaciones listas para el fin de semana.