Tecnológico de Monterrey

Avance de proyecto 3: Sistema de Recomendación

Maestría en inteligencia artificial aplicada
Asignatura: Análisis de grandes volúmenes de datos

Equipo:
Abraham Cabanzo Jimenez - A01794355
Ignacio Antonio Ruiz Guerra - A00889972
Moisés Díaz Malagón - A01208580

9 de junio de 2024

Revisión de alcance y objetivos del proyecto

El proyecto continúa siendo realizar un sistema de recomendación de películas basado en las preferencias del usuario. El sistema tiene como objetivo predecir las preferencias que un usuario tendrá de una película, y facilitar la toma de decisión del cliente al usar una plataforma de contenido de películas.

Para realizar el sistema de recomendación de películas para una plataforma de streaming, a la que llamaremos “Génesis”, se requerirán los siguientes componentes:

- Una base de datos de películas con sus características, en este caso se utiliza MovieLens 1M.
- Un algoritmo de recomendación: se presentan 4 algoritmos a utilizar, en complejidad creciente: recomendador global, sistema basado en contenido (utilizando one-hot encoding), sistema de filtro colaborativo (utilizando KNeighbors), y sistema de filtro colaborativo complejo usando deep learning.
- Evaluación del sistema utilizando alguna métrica específica: Se utilizará como base la métrica HitRatio@10 que mide la efectividad de recomendar una película que un usuario vería dentro de las 10 mejores recomendaciones del sistema recomendador.
- Sistema de despliegue para su utilización: se definirá en posteriores entregas, pero será importante que el sistema desarrollado pueda ser utilizado por otras personas, por medio de un mecanismo como una API web por ejemplo.

Después de una revisión y discusión con el equipo, se determinó que no se requieren modificaciones en el alcance y objetivos del proyecto en este momento. Continuaremos con el plan actual y evaluaremos la necesidad de ajustes en futuras revisiones.

Comparación de los algoritmos de la actividad 4.2 y 6.2

A continuación, se comparan los cinco algoritmos propuestos, de acuerdo con el rendimiento y la escalabilidad de cada uno de ellos. Se creó un nuevo Jupyter notebook con los algoritmos implementados utilizando programación orientada a objetos y se estandarizaron las entradas y salidas, de tal forma que fuera más fácil evaluar su rendimiento. El detalle de la evaluación de rendimiento puede ser encontrada en el notebook de esta entrega: https://github.com/moisesdiazm/sistema-recomendacion-bigdata-mna/blob/main/Proyecto_Avance_3%2337.ipynb

Algoritmo	Rendimiento	Escalabilidad
Sistema básico: recomendador global (Actividad 4.6)	HitRatio@10: 0.00% El recomendador global, basado en popularidad, es el que tiene menor tasa de aciertos, esto se debe a que	De los algoritmos propuestos, este es el sistema más simple de implementar. Sin embargo, el cómputo de las películas más vistas puede llegar a requerir muchos recursos, ante

	recomienda las películas más populares, no necesariamente las que le gustan a un usuario en particular, y en especial para los 100 usuarios de prueba utilizados no coincide con ninguno de sus gustos. Cabe mencionar que el porcentaje podría aumentar con otra base de usuarios de prueba.	grandes cantidades de usuarios, pues hace uso de una función de agregación. Por ello, se recomienda utilizar técnicas de caching para optimizar la consulta de recomendaciones. Su alta escalabilidad pudiera implementarse utilizando MapReduce.
Sistema de filtro colaborativo: KNeighbors (Actividad 4.6)	HitRatio@10: 57.00% El mejor rendimiento lo obtiene el filtro colaborativo que utiliza KNeighbors para encontrar similitudes basándose en la interacción entre usuarios y películas. Se puede deducir que en la práctica la suposición de que a usuarios similares les gustan películas similares puede estar correcta en muchos casos.	El algoritmo es rápido de entrenar y al ser no-supervisado no requiere de datos etiquetados. No requiere tampoco de metadatos de películas tal como descripciones o géneros. En tiempo de ejecución es el que tomó más tiempo de cómputo durante la inferencia. Este filtro es útil cuando el usuario ya ha realizado varias interacciones en la plataforma, no en un inicio, es decir, no soporta inicio en frío.. Por otro lado, requiere re-entrenarse cuando se agregan nuevas interacciones de usuarios. Aunque el entrenamiento es rápido debe estar en constante actualización lo que puede suponer retos de despliegue.
Sistema basado en contenido: one-hot encoding con similaridad coseno (Actividad 4.6)	HitRatio@10: 52.00% El segundo algoritmo con mejor rendimiento es el basado en contenido.	Este algoritmo es escalable, es no supervisado y su matriz de similaridad se puede re-calcular rápidamente cuando se agregan películas, pues no depende de la interacción de usuarios. Dado que se basa en el contenido de las películas y sus metadatos, no requiere que el usuario tenga interacciones previas, por lo que soporta arranque en frío. Hace inferencia en complejidad de tiempo constante lo que lo hace muy escalable. La matriz de similaridad se encuentra pre-calculada, y puede accederse desde diferentes servidores en paralelo, es escalable

		horizontalmente.
Sistema de filtro colaborativo complejo: capa de embedding con redes neuronales FC. (Actividad 4.6)	HitRatio@10: 42.00% El 3er algoritmo con mejor rendimiento es el más complejo de los propuestos. En este caso logra un HitRatio@10 de 42%, el cual consideramos que podría incrementar si el algoritmo se entrena con más épocas, pues solo se entrenó con 4 épocas.	Está basado en embeddings y redes neuronales. Este algoritmo genera dentro de sus pesos sinápticos una matriz de similaridad que combina usuarios y películas. Su complejidad y el poder de cómputo para su entrenamiento es considerablemente mayor que en el caso de los algoritmos anteriores. La inferencia sin embargo es rápida si se cuenta con hardware especializado como GPUs. Es más difícil de escalar comparado con el filtro colaborativo de KNeighbors y el basado en contenido de similaridad del coseno. Debe entrenarse con nuevos datos de usuarios y películas de forma constante. Implica un costo mayor de nube tanto si se utilizan instancias dedicadas con GPU como si se utiliza CPU.
Sistema basado en contenido: TF-IDF (Actividad 4.2)	No contamos con datos etiquetados de prueba que nos permitan validar el rendimiento. Se utilizó un dataset distinto. Este algoritmo toma en cuenta tanto la descripción de la película como los géneros. Debido a esto es posible que genere mejores métricas comparado con el sistema que utiliza one-hot encoding.	La inferencia sucede en tiempo constante una vez calculadas las matrices tf-idf y similaridad. Esto lo hace altamente escalable una vez entrenado. El entrenamiento no es tardado ni consume gran cantidad de recursos. Se puede comparar con el sistema previo de one-hot, sin embargo requiere de la librería sklearn para el vectorizador Tf-idf.

Diferencias y ventajas del algoritmo sencillo vs el algoritmo complejo

Los algoritmos sencillos como el que está basado en el contenido, al utilizar one-hot encoding, se tiene una ventaja significativa comparado con el algoritmo basado en deep learning, y esta es que no requiere actualizaciones cada vez que se agregan interacciones de los usuarios, además de que no requiere de mucho tiempo de

entrenamiento, ni hardware dedicado, lo que lo hace más económico y simple de escalar.

A pesar de su reducida complejidad, se logra un mejor rendimiento si el algoritmo complejo no se entrena con los suficientes datos. Por otro lado, el algoritmo simple de filtro colaborativo que utiliza KNeighbors, también supone una ventaja ante el algoritmo complejo, pues aunque el filtro colaborativo requiere actualizaciones con nuevas interacciones de los usuarios, su entrenamiento es rápido y la inferencia aunque computacionalmente más compleja no requiere de hardware especial y logra mejores rendimientos que el filtro basado en contenido.

Casos en que aplica el algoritmo sencillo vs el algoritmo complejo

De acuerdo con lo observado se utilizarían cada uno de los algoritmos mencionados siguiendo esta lógica:

- Se utilizará el sistema de recomendación global para dar las primeras recomendaciones a los nuevos usuarios, debido a que no existen interacciones previas que indique sus gustos y preferencias.
- Se utilizará el recomendador basado en contenido que utiliza one-hot encoding para las primeras recomendaciones después de que los usuarios han visto al menos una película, pero menos de un número predeterminado (e.g. 20 películas).
- Si es prioridad tener una mayor calidad de recomendaciones y se cuenta con los recursos y tiempo necesario para entrenamiento se puede utilizar el filtro colaborativo que usa deep learning.
- Si se tienen recursos limitados, pero aun así se quiere hacer recomendación personalizada basada en interacciones de usuario, se puede utilizar el filtrado colaborativo basado en KNeighbors.

Referencias:

He, X., Liao, L., Zhang, H., Nie, L., Hu, X. y Chua, T. (2017). *Neural Collaborative Filtering*. Recuperado de: https://arxiv.org/pdf/1708.05031