Tecnológico de Monterrey

Avance de proyecto 2: 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

26 de mayo del 2024

Introducción

En esta segunda entrega del sistema de recomendación se presentan un algoritmo complejo nuevo y dos sencillos basados en el contenido de la materia. Debido a que inicialmente la base de datos utilizada IMDB solo contiene datos de las películas pero no de las interacciones de usuario, se decidió para esta entrega utilizar la base de datos MovieLens que contiene una tabla de calificaciones otorgadas por los usuarios. Esto último fue necesario para la implementación de filtros colaborativos no supervisados y supervisados que se proponen en esta entrega.

La implementación de este trabajo se detalla en el documento de Jupyter notebook contenido en el github del equipo:

https://github.com/moisesdiazm/sistema-recomendacion-bigdata-mna/blob/main/Proyecto_Avance_2%2337.ipynb

1. Descripción del algoritmo de recomendación avanzado elegido

El algoritmo de recomendación avanzado que se incluye en esta práctica está basado en el artículo de He et al. (2017) titulado *Neural Collaborative Filtering* que consiste en la utilización de redes neuronales para resolver un problema de clasificación binaria, si el usuario tendrá o no interacción con una película dada.

Como se puede observar en el Jupyter lab entregado, inicialmente se propone un filtro colaborativo utilizando una técnica no supervisada para hacer recomendaciones, y se basa en encontrar similitudes entre usuarios que permiten recomendar películas que perfiles similares han visto.

A diferencia de esta primera aproximación cuya implementación se puede observar en el Jupyter notebook, el approach complejo busca derivar las relaciones entre usuarios y películas que pueden ayudar a determinar si la película será vista o no por el usuario.

La técnica consiste en reemplazar el producto punto o cálculo de similaridad con una arquitectura de redes neuronales que puede aprender una función arbitraria, y que incluso puede generalizar una factorización de matrices por medio de las no-linealidades que permiten las múltiples capas de perceptrones (He et al., 2017). El objetivo de esta red neuronal será aprender de la interacción entre usuarios y películas.

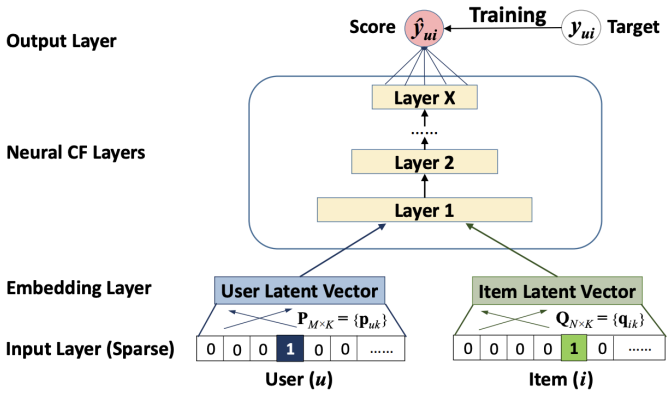

Figura 1. Sistema de recomendación de filtro colaborativo propuesto por He et al. (2017)

La propuesta de He et al. (2017) es vectorizar el usuario y el id de la película como un vector one-hot representando su número o id de forma vectorial. Posteriormente ambos vectores deberán ser convertidos en embeddings de menor dimensionalidad, en nuestro caso se eligieron vectores de 10 dimensiones. Sin embargo es un parámetro que se puede ajustar de acuerdo con las necesidades.

Utilizando el framework PyTorch se definirán dos capas de tipo Embedding cuyo número de entradas será la cantidad de dígitos 1 y 0 para representar los ids enteros en forma one-hot, es decir el número más grande a representar. Y el tamaño de la salida de esta capa será las dimensiones de embedding que hemos definido, en este caso un vector de 10 elementos. Esta capa es entrenable, lo que significa que sus parámetros serán ajustados durante el entrenamiento o ajuste.

```
   | Name            | Type      | Params
-----------------------------------------------
 0 | user_embedding  | Embedding | 36.2 K
 1 | movie_embedding | Embedding | 23.7 K
```

Figura 2. Capas de Embedding, con la cantidad de parámetros que se entrenarán.

Posteriormente, los vectores de salida de los embedding del usuario y película deberán ser concatenados para formar una sola entrada a las capas de la red neuronal. Considerando que los vectores tienen dimensión de 6 elementos cada uno, el vector de entrada a la red neuronal después de la concatenación tendrá dimensión de 12 elementos.

Se utilizarán 3 capas de 64, 32 y 1 neuronas respectivamente. El tipo de capa será Fully Connected, lo que significa que cada una de las neuronas de una capa están conectadas a cada una de las neuronas de la siguiente capa. De esta forma para la primera capa tenemos una dimensión de entrada de 12 elementos (embeddings concatenados) y una salida de 64 elementos (cantidad de neuronas en capa 1).

Para la segunda capa tenemos 64 elementos de entrada y 32 de salida, para la tercera capa tenemos 32 entradas y una única salida (una neurona) que será utilizada para predecir un valor entre 0 y 1 que representa la probabilidad de que la película sea vista o no.

A fin de que las neuronas en las capas puedan representar internamente funciones no lineales será necesario utilizar una función de activación no lineal. Para capas intermedias utilizaremos la función de activación ReLU. En el caso de la salida, debido a que estamos tratando con un problema de clasificación binaria utilizaremos la función Sigmoide que entrega valores entre 0 y 1 de forma suavizada.

```
    | Name            | Type       | Params
-----------------------------------------------
  2 | fc1             | Linear     | 832
  3 | fc2             | Linear     | 2.1 K
  4 | fc3             | Linear     | 33
```

Figura 2. Capas Fully Connected, con la cantidad de parámetros que se entrenarán.

Finalmente la predicción final será comparada con el valor real de que la película fue vista por el usuario. Se calculará la función de pérdida (error) utilizando Binary Cross Entropy, que es útil en problemas de clasificación binaria. Para la optimización se utilizará Adam de PyTorch que realiza la optimización de gradiente descendente con técnicas que favorecen una convergencia rápida y estable, por ejemplo haciendo uso de momento.

Previo al entrenamiento del algoritmo supervisado será necesario hacer algunos ajustes al set de datos:

- Dado que se tiene un problema de clasificación binaria, las interacciones a considerar serán de si el usuario vio o no una película en lugar de la calificación asignada. Para ello se asume que si el usuario calificó la película entonces la vio, y si no la calificó no la ha visto. Esto no necesariamente significa que no la quiere ver, puede ser que no la conozca. Este tipo de problemas se pueden resolver combinando con resultados de filtro basado en contenido, como abordaremos en posteriores entregas.
- Se debe hacer una partición de datos en conjunto de entrenamiento y validación. Para ello se considerará la última interacción del usuario como parte del conjunto de validación y el resto para el entrenamiento.

2. Métricas de evaluación de desempeño de sistemas de recomendación

Para evaluar los sistemas de recomendación, es necesario utilizar métricas que nos den una visión precisa y clara de cómo se satisface la necesidad de los usuarios y los objetivos planteados. A continuación algunas de ellas.

Recall@k or HitRatio@

Esta es una métrica esencial en la evaluación de sistemas de recomendación, proporcionando una medida clara de cuántos ítems relevantes son capturados entre las principales recomendaciones. Su facilidad de cálculo y alineación con el comportamiento del usuario la hacen una herramienta valiosa para mejorar la precisión y efectividad de los modelos de recomendación (Z. Deutschman, 2023). Se calcula de la siguiente manera:

$$HR = \frac{Número\ de\ recomendaciones\ principales\ relevantes}{Número\ de\ todos\ los\ elementos\ relevantes}$$

Esta métrica se aplicó en el modelo de la siguiente forma:

1. Para cada usuario seleccionamos de manera aleatoria 99 títulos de película que el usuario no ha visto (en este caso que no ha evaluado). A estos 99 le agregamos una que el usuario haya visto. De esta forma tenemos 100 películas.
2. Ejecutamos la inferencia sobre estas 100 películas, ordenamos el resultado de las probabilidades de manera descendente y seleccionamos el top 10.
3. Si la película que ha visto está presente en el top 10, se considera un acierto.
4. Se repite el proceso para todos los usuarios y se hace el promedio de aciertos.

Precision@k

Es una medida del cociente de los ítems recomendados entre los top k que son relevantes para el usuario.

$$P = \frac{Número\ de\ recomendaciones\ principales\ relevantes}{Número\ de\ todos\ los\ elementos\ relevantes}$$

La justificación reside en que es una métrica esencial para evaluar la exactitud de las recomendaciones. En proyectos donde es crucial que las recomendaciones sean altamente relevantes (por ejemplo, recomendaciones de productos en un comercio electrónico), una alta precisión indica que el sistema está sugiriendo elementos que realmente interesan al usuario, reduciendo la probabilidad de que los usuarios ignoren las recomendaciones.

Aquí un ejemplo:

Supongamos que un sistema de recomendación proporciona una lista de 5 ítems recomendados (top 5) a un usuario, y sabemos que los ítems relevantes para este usuario son los siguientes:

- Ítems recomendados: A, B, C, D, E
- Ítems relevantes (los que al usuario realmente le interesan): B, D, F, G

Para calcular precision@k, identificamos cuántos de los ítems recomendados son relevantes. En este caso, los ítems relevantes dentro de los recomendados son B y D.

Entonces, tenemos:

Número de recomendaciones relevantes entre las top 5=2 Número de recomendaciones relevantes entre las top 5=2 Número de ítems recomendados=5
Número de ítems recomendados=5

Aplicamos la fórmula:

P@5=2/5=0.4

Así, la precision@5 es 0.4, lo que significa que el 40% de los ítems recomendados en las top 5 son relevantes para el usuario.

F1@k

Es la media armónica de precision@k y recall@k, que ayuda a simplificarlos en una única métrica. Estas métricas pueden calcularse con base en la matriz de confusión (Z. Deutschman, 2023). Las fórmulas exactas se presentan a continuación:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

TP = True positive

TN = True negative

FP = False positive

FN = False negative

Fórmulas Precision, recall, F1. De Z. Deutschman, 2023, Recommender Systems: Machine Learning Metrics and Business Metrics, Recuperado de https://neptune.ai/blog/recommender-systems-metrics

Como podemos ver, el coeficiente F1 no considera los valores de verdaderos negativos. Estos son los casos en los que el sistema de recomendación no recomendó un ítem irrelevante para el usuario. Esto significa que podemos asignar cualquier valor a los verdaderos negativos y no afectará el puntaje F1. Una alternativa interesante y perfectamente simétrica es el coeficiente de correlación de Matthews (MCC).

Cobertura

La cobertura de un sistema de recomendación es una medida del rango de elementos sobre los cuales el sistema puede hacer predicciones o recomendaciones. Es especialmente importante para la tarea de "Encontrar Todos los Buenos Elementos", ya que los sistemas que no pueden evaluar muchos de los elementos en el dominio no pueden encontrar todos los buenos elementos. También es crucial para la tarea de "Anotar en Contexto", ya que no es posible realizar anotaciones para elementos donde no hay predicciones disponibles. Se puede definir directamente en las predicciones preguntando: "¿Para qué porcentaje de elementos puede este recomendador formar predicciones?" Este tipo de cobertura

se llama comúnmente cobertura de predicción. Otra métrica de cobertura puede formarse para recomendaciones, en términos de "¿Qué porcentaje de los elementos disponibles recomienda este recomendador a los usuarios?" Para un sitio de comercio electrónico, este último tipo de cobertura mide cuánto del catálogo de artículos del comerciante se recomienda; por esta razón, se denomina cobertura del catálogo (Herlocker, J. L., et al, 2004).

La **justificación** para usar esta métrica, tiene que ver con la importancia para evaluar la diversidad del sistema de recomendación. En proyectos donde es importante que el sistema sea capaz de recomendar una amplia gama de productos o contenidos (por ejemplo, en una plataforma de streaming de video), una alta cobertura asegura que el sistema puede sugerir una variedad significativa de opciones, lo cual es crucial para mantener el interés del usuario a largo plazo.

Curva ROC

Existen dos definiciones populares para el acrónimo ROC: “relative operating characteristic” introducido por Swets (1963, 1969) y “receiver operating characteristic” utilizado en la teoría de detección de señales (Hanley y McNeil, 1982). En ambos casos, ROC se refiere a la misma métrica subyacente.

El modelo ROC intenta medir en qué medida un sistema de filtrado de información puede distinguir exitosamente entre señal (relevancia) y ruido. Asume que el sistema asignará un nivel de relevancia predicho a cada posible elemento. Esto da lugar a dos distribuciones: una para la probabilidad de que el sistema predice un nivel de relevancia para un elemento que no es relevante, y otra para elementos relevantes. Cuanto más separadas estén estas dos distribuciones, mejor será el sistema diferenciando elementos relevantes de los no relevantes (Herlocker, J. L., et al, 2004).

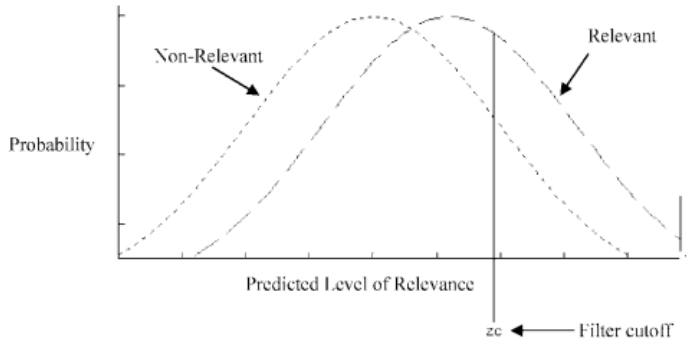

Fig. 1. A possible representation of the density functions for relevant and irrelevant items.

La **justificación** para usar AUC - ROC reside en su utilidad para evaluar el desempeño del sistema de recomendación en términos de su capacidad para distinguir entre elementos relevantes e irrelevantes. Esta métrica es especialmente valiosa cuando se necesita un balance entre precisión y cobertura, como en sistemas donde tanto los aciertos como los errores tienen un impacto significativo (por ejemplo, en sistemas de recomendación médica).

Estas métricas permiten obtener una evaluación integral del desempeño del sistema de recomendación, considerando tanto la exactitud de las recomendaciones como la capacidad del sistema para abarcar una amplia gama de elementos y su habilidad para balancear correctamente entre verdaderos y falsos positivos.

3. Experimentación con al menos 1 algoritmo básico

Modelo KNN

Se genera una matriz dispersa que contiene usuario-película, lo que significa que la mayoría de sus entradas son ceros. Esto es típico en sistemas de recomendación donde no todos los usuarios han visto todas las películas y se genera con una matriz dispersa en formato CSR (Compressed Sparse Row), que es eficiente en términos de memoria y operaciones de acceso para matrices dispersas.

Se preparan los datos extrayendo los identificadores únicos de usuarios y películas y se crean diccionarios para mapear estos identificadores a índices de la matriz y se convierten las listas de identificadores de usuarios y películas en listas de índices correspondientes para que con ellos se construya una matriz dispersa donde las filas representan usuarios, las columnas representan películas y los valores son las calificaciones dadas por los usuarios a las películas para que con ello se calculae la proporción de entradas no nulas en la matriz y se verifica que supere un umbral mínimo para asegurar que la matriz sea adecuada para recomendaciones.

Filtrado Colaborativo con el Modelo KNN:

Se utiliza el algoritmo de K Vecinos Más Cercanos (KNN) para encontrar películas similares basándose en las calificaciones de los usuarios y se emplea la métrica de distancia coseno, comúnmente utilizada en filtrado colaborativo, ya que mide la similitud entre vectores en función del ángulo entre ellos, ignorando la magnitud, se entrena el modelo con la matriz transpuesta (película-usuario) para encontrar las películas más similares a una película dada para que el modelo KNN se ajuste a la matriz transpuesta para buscar la película objetivo en la matriz y se calculan las distancias a todas las demás películas y se obtiene una lista de los índices de las películas más similares.

Se traducen los índices de las películas de vuelta a sus identificadores originales y se muestran los títulos de las películas recomendadas junto con sus distancias (si se solicita).

1. **Sistemas de Recomendación**:

Son herramientas que sugieren productos (películas en este caso) a los usuarios basándose en varios tipos de información, como el historial de calificaciones, ya que con el filtrado colaborativo hace recomendaciones utilizando la información de usuarios o ítems similares, asumiendo que si dos usuarios han calificado ítems de manera similar en el pasado, probablemente coincidan en sus preferencias futuras.

2. **Matrices Dispersas**:

Utilizan estructuras de datos eficientes para manejar grandes cantidades de ceros, permitiendo operaciones matemáticas y almacenamiento más eficientes dado que la eficiencia es crucial cuando se manejan grandes volúmenes de datos típicos en sistemas de recomendación.

3. **Filtrado Colaborativo Basado en Vecindario (KNN)**:

Utiliza métodos como KNN para identificar elementos similares basándose en las interacciones pasadas y con dichas similitudes se puede usar la métrica de coseno ya que se centra en la dirección de las preferencias, no en su magnitud, pero existen métricas como la Ecluidiana, La distancia de Manhattan, Minkowski, Hamming que cada una nos aporta distintos acercamientos o métricas de los datos.

Modelo One-Hot

Se preparan los datos de género de la película para convertirlos en una lista separada por el carácter “|” y se eliminan las películas que no tengan un género al igual que los títulos de las películas, donde se elimina el año del lanzamiento entre paréntesis en el título y se crea una columna nueva para tener el título limpio y así mismo para tener el año y la década en dos columnas separadas.

Se Utiliza el modelo One-Hot

Se utiliza el modelo One-Hot para transformar los géneros en vectores binarios agregando 1 si la película pertenece a un género o 0 si no permitiendo que cada película tenga un vector de características basada en los géneros, de manera similar se codifican las décadas de las películas e incluir esta información en los vectores. Se crea la matriz de similitud con la métrica del seno que nos mide la similitud entre dos vectores basándose en el coseno de los ángulos entre ellos y nos regresa una matriz cuadrada (i, j) donde existe una similitud entre película (i) y la película (j). Se realiza una búsqueda por título y se generan recomendaciones a

partir de una película en específico ordenando las películas similares en función a su similitud y se traducen los índices en los títulos de las películas.

1. **Sistemas de Recomendación Basados en Contenido**:

Utilizan las características de los ítems (en este caso, películas) para calcular la similitud entre ellos y hacer recomendaciones. A diferencia del filtrado colaborativo, que se basa en las interacciones de los usuarios, el filtrado basado en contenido se enfoca en las propiedades intrínsecas de los ítems.

2. **Codificación One-Hot**:

Es una técnica para representar variables categóricas como vectores binarios permitiendo que los algoritmos de aprendizaje automático y análisis de similitud manejen datos categóricos de manera efectiva.

3. **Similitud Coseno**:

Es una métrica que evalúa la similitud entre dos vectores considerando la orientación, no la magnitud y es adecuada para datos dispersos y de alta dimensionalidad, como las características de géneros y décadas de películas, pero también existen métricas como la Euclidiana, Hammming, Jaccard.

4. **Procesamiento de Cadenas y Expresiones Regulares**:

Se utilizan para limpiar y extraer información relevante de los títulos de las películas, mejorando la calidad de los datos para el análisis posterior.

Conclusiones:

En esta segunda entrega del proyecto de sistema de recomendación, presentamos un algoritmo basado en redes neuronales y dos con enfoques más simples, basados en filtros colaborativos no supervisados y supervisados.

La base usada es MovieLens, que contiene información de usuarios con películas, a diferencia que la usada previamente, IMDB, que solo contenía datos de películas. Esto fue necesario para poder implementar los filtros colaborativos.

El algoritmo avanzado, se basa en el artículo *Neural Collaborative Filtering* de He et al (2017), que utiliza redes neuronales para estimar si un usuario verá una película o no. Utiliza técnicas de embeddings y capas completamente conectadas, con lo que se logra modelar de manera más compleja las interacciones entre usuarios y películas, superando las limitaciones de los métodos tradicionales que se basan en similitudes.

Para evaluar el desempeño del sistema de recomendación, se utilizaron varias métricas clave:

HitRatio@10 o Recall@10: Esta métrica mide la fracción de las 10 recomendaciones principales que son relevantes para el usuario. En este caso, se obtuvo un valor de 0.4619, indicando que el 46.19% de las veces, la película relevante estaba entre las 10 principales recomendaciones.

Precision@10**:** Mide la fracción de las 10 recomendaciones principales que son relevantes. Se obtuvo un valor de 0.4611, lo que significa que el 46.11% de las recomendaciones en el top 10 eran relevantes para los usuarios.

F1@10: Es la media armónica de la precisión y el recall, proporcionando una métrica única que balancea ambos aspectos. Se obtuvo un valor de 0.4540.

Los resultados obtenidos indican un buen desempeño del sistema en términos de precisión y recall, lo que demuestra la efectividad del enfoque avanzado basado en redes neuronales.

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

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1), 5-53.

Z. Deutschman, 2023, Recommender Systems: Machine Learning Metrics and Business Metrics, Recuperado de https://neptune.ai/blog/recommender-systems-metrics