Bases de Datos No Relacionales - Curso 2022

PROYECTO FINAL

Cabrera, Francisco CI: 4.904.973-6
Santellan, Agustín CI: 4.863.713-4

FACULTAD INGENIERÍA DE LA UNIVERSIDAD DE LA REPÚBLICA

4 de julio de 2022

Resumen

En este documento se detallan las decisiones tomadas y los procedimientos aplicados para la realización del proyecto final del curso de Bases de Datos No Relacionales edición 2022, cuyo producto es un sistema de recomendación de películas basado en la representación de la información en una base de datos de grafos (*Neo4j*).

I. Introducción

Los sistemas de recomendación usados por las plataformas más populares, por ejemplo *Netflix*[1] y *Amazon*[2], están basados en modelos de aprendizaje automático o *Machine Learning* (ML). La motivación de este trabajo fue realizar una prueba cualitativa del rendimiento de las bases de datos de grafos como alternativa más sencilla para un sistema de estas características, así como la aplicación de este paradigma de base de datos en un problema práctico de extremo a extremo. Por estas razones, se planteó como objetivo el desarrollo de un sistema que contara con una interfaz simple para buscar películas, marcar algunas de ellas como favoritas y obtener títulos nuevos recomendados a partir de los gustos del usuario. Se espera que los resultados de las recomendaciones sean adecuados, teniendo éstos una correlación y coherencia con los gustos del usuario. Del mismo modo se espera aprender y aplicar conocimiento en el manejo de datos dentro de este paradigma.

II. Trabajos relacionados

Dado que las buenas recomendaciones son muy importantes tanto para la experiencia de los usuarios como para la retención de los mismos en las diferentes plataformas, ya sea de compras o contenidos, se trata de una temática muy explorada pero que posee un gran desarrollo activo actualmente. Según un artículo publicado por *Neo4j* [WebberWebber] empresas grandes como *Walmart*[3] e *eBay*[4] actualmente usan esta plataforma como medio para generar recomendaciones a sus usuarios, aunque se encuentran ejemplos de sistemas de recomendaciones más complejos o de mayor volumen basados en modelos de ML como los de *Amazon* [What's new in recommender systems — AWS Media BlogWhat's new in recommender systems — AWS Media Blog] *Netflix* [How Netflix's Recommendation Engine Works? — by Springboard India — MediumHow Netflix's Recommendation Engine Works? — by Springboard India — Medium] y *Youtube* [On YouTube's recommendation systemOn YouTube's recommendation system], que se aplican a problemáticas similares al problema del proyecto.

III. Elaboración del sistema

Esta sección será dedicada a la descripción de la solución, así como el proceso de desarrollo y las decisiones tomadas en él. El código generado en el proyecto se encuentra en la plataforma *GitLab* de la Facultad de Ingeniería[5].

III-A. Arquitectura

El sistema está conformado por una interfaz de usuario elaborada en la plataforma de desarrollo *React*[6], que fue elegida por la familiaridad del equipo de trabajo con la misma, con el lenguaje *TypeScript*[7] y una base de datos de grafos *Neo4jneo4j.com* almacenada en su plataforma en la nube *AuraDB*. La interfaz de usuario se comunica con la base de datos a través de una biblioteca *use-node4j*[8] que provee funcionalidades necesarias para conectarse a la base de datos y ejecutar consultas en ella.

III-B. Generación de recomendaciones

El sistema de recomendaciones elaborado basa su funcionalidad en la noción de caminos ponderados entre nodos que representan películas. Estos caminos están formados por relaciones entre nodos de películas, personas y géneros, y representan similitudes entre los diferentes títulos. La ponderación de los caminos se realiza basándose en una heurística definida por el equipo de trabajo, que consiste en asignar un valor numérico a cada segmento (relación) del camino correspondiente al tipo y luego multiplicar estos valores para conseguir la evaluación final. Los valores numéricos asignados cambian según el tipo de relación que puede ser de, en orden de importancia, director, genero o actor. Para cada par de títulos puede existir más de un camino, lo que implica una mayor similitud, por lo que la evaluación del puntaje de la similitud consiste en la suma de las ponderaciones de todos los caminos entre ellos. Usando esto, se parte de un título para obtener todos los que están relacionados ordenados por su puntaje de similitud. En el caso de que el usuario seleccione más de un título como favorito, las recomendaciones deberían adecuarse a todos los títulos seleccionados. El sistema elaborado resuelve esta situación generando una lista que contiene las recomendaciones correspondientes a todos los títulos seleccionados y agrupa las instancias que se

[1] netflix.com
[2] amazon.com
[3] walmart.com
[4] ebay.com
[5] gitlab.fing.edu.uy/agustin.santellan/bdnr
[6] reactjs.org
[7] typescriptlang.org/
[8] npmjs.com/package/use-neo4j

repiten, acumulando su puntaje de similitud. De este modo, los títulos que resulten vinculados a muchos de los seleccionados se ven favorecidos en el puntaje y son recomendados en una posición más alta. Luego de realizar pruebas se llegó a la conclusión de que los caminos a evaluar deben ser de largo dos o cuatro, siempre par porque ningún título está relacionado directamente con otro, debido a que caminos más largos son contraproducentes a la generación de recomendaciones adecuadas. Con la misma motivación de mantener los resultados precisos, se excluyen los géneros en los caminos de largo cuatro ya que incluirlos vincularía fuertemente títulos que no tienen tanta relación. Un ejemplo de este caso serían dos títulos que tengan como géneros histórico y acción el primero y Ciencia ficción y acción el segundo, que serían vinculados por un camino de largo cuatro, pero muy probablemente no sean similares.

III-C. Diseño de la base de datos de grafos

Partiendo de un juego de datos relacional, se planteó hacer una migración teniendo en cuenta el algoritmo a utilizar para la generación de las recomendaciones, por lo que los atributos necesarios para las recomendaciones fueron modelados como nodos y relaciones de forma de utilizarlos en la generación de caminos entre títulos. Originalmente se planteó el uso del idioma de los títulos y títulos alternativos como atributos modificadores de los puntajes de las recomendaciones, que serían modelados como atributos de los nodos, pero se abandonó la idea debido a la imposibilidad de definir diccionarios como atributos en *Neo4j*. Una solución a este problema es utilizar nodos y relaciones para modelar estos atributos, aunque aumentaría la complejidad de las consultas, pero se decidió no implementar esto ya que reduciría drásticamente la capacidad de almacenamiento de nodos de títulos.

III-D. Juegos de datos

Al inicio, la información de las películas fue obtenida de los conjuntos de datos relacionales publicados oficialmente por *Internet Movie Database* (IMDb)[9] que contienen un gran volumen, aproximadamente ocho millones y medio de filas de títulos, y que incluyen no solamente películas sino que también series, episodios, videos y cortos, lo que permitiría al sistema también recomendar esos tipos de títulos. Estos datos presentan información muy amplia, como cortos de estudiantes y desde cortos previos al año 1900 hasta películas tan recientes como al año que transcurre. Por este motivo y debido al acotado espacio de almacenamiento disponible en el manejador de bases de datos de grafos usado, se presentaron problemas de calidad de la información para los propósitos de este trabajo, ya que los títulos que fueron almacenados no presentaban suficientes relaciones entre sí para generar resultados significativos. Una forma de mitigar este problema sería someter a los datos a un proceso de curación en el que se ordenen en base a su relevancia, para el que se propone utilizar la información disponible de puntajes dados por usuarios y la cantidad, de forma de obtener los títulos más populares. Cabe aclarar que de disponer el almacenamiento suficiente, este juego de datos sería adecuado. A modo de obtener mejores resultados ilustrativos del funcionamiento del sistema generado, se optó por usar otro juego de datos[10] que contiene la información de las mil películas más populares de IMDb, que puede ser almacenado completamente por el manejador de bases de datos utilizado y permite evaluar las recomendaciones de la misma manera que el conjunto anterior. Usando estos datos, se pierde la información de los otros tipos de títulos, pero extender el funcionamiento del sistema a ellos es trivial.

IV. Pruebas realizadas

Dada la naturaleza del producto, las pruebas realizadas fueron cualitativas. Todas las pruebas realizadas tuvieron un tiempo de respuesta en el entorno de tres segundos, tratándose de un valor aceptable. Las primeras pruebas realizadas tuvieron el objetivo de ayudar a definir la heurística final utilizada y consistieron en probar diferentes largos de caminos y diferentes valores utilizados para la ponderación. Con los valores finales, se seleccionaron varias películas y se evaluaron las recomendaciones obtenidas a partir de ellas teniendo en cuenta su similitud y la probabilidad de que fueran recomendadas por otros sistemas también. De los resultados observados se puede afirmar que son satisfactorios ya que estos son consistentes en cuanto al estilo, tono y público objetivo, por lo que serían buenas recomendaciones para el usuario del sistema.

V. Conclusiones y trabajo futuro

Se logró la creación de extremo a extremo de un sistema práctico de baja complejidad que permite buscar películas, guardar favoritos y generar recomendaciones a partir de estos últimos en un tiempo razonable y con una calidad adecuada. Además, mediante problemas encontrados, el equipo logró aprender sobre problemáticas con el manejo de grandes volúmenes de datos y la calidad de la información.

Como trabajo futuro se plantea el uso del conjunto completo de datos disponibles en una herramienta que no acote el espacio de almacenamiento de esta manera, así como la introducción de otros atributos que contribuyan a mejorar las recomendaciones como los títulos alternativos, lenguajes y evaluaciones de usuarios.

De forma menos inmediata, se propone la creación de relaciones entre directores que puedan implicar similitudes estilísticas y de tono para generar recomendaciones de títulos que resultan similares sin compartir elenco o dirección, que en el producto realizado no se priorizarían.

[9] www.imdb.com/interfaces/

[10] github.com/peetck/IMDB-Top1000-Movies/blob/master/IMDB-Movie-Data.csv

REFERENCIAS

[How Netflix's Recommendation Engine Works? — by Springboard India — MediumHow Netflix's Recommendation Engine Works? — by Springboard India — Medium NetflixHow Netflix's Recommendation Engine Works? — by Springboard India — Medium. How Netflix's Recommendation Engine Works? — by Springboard India — Medium. . https://medium.com/@springboard_ind/how-netflixs-recommendation-engine-works-bd1ee381bf81

[On YouTube's recommendation systemOn YouTube's recommendation system] YoutubeOn YouTube's recommendation system. On YouTube's recommendation system. . https://blog.youtube/inside-youtube/on-youtubes-recommendation-system/

[WebberWebber] neo4jPaperWebber, J. . The 1 Platform for Connected Data Powering Real-Time Recommendations with Graph Database Technology The 1 Platform for Connected Data Powering Real-Time Recommendations with Graph Database Technology.

[What's new in recommender systems — AWS Media BlogWhat's new in recommender systems — AWS Media Blog] amazonWhat's new in recommender systems — AWS Media Blog. What's new in recommender systems — AWS Media Blog. . https://aws.amazon.com/blogs/media/whats-new-in-recommender-systems/