Graph-Based Recommendation System for Movies

Sijjil Khan and Humayun Hasan
Graph Data Science Course Project

INTRODUCTION

The "Graph-Based Movie Recommendation System" addresses the challenge of delivering accurate, personalized recommendations by using a graph-based approach. By representing users, movies, genres, and ratings as interconnected nodes in a Neo4j graph database, this system leverages advanced graph analysis and machine learning algorithms to reveal patterns and preferenceS. This comprehensive model enables a more accurate and relevant recommendation experience compared to traditional methods.

OBJECTIVE

The objective of this project is to design and implement a graph-based recommendation system that suggests movies to users by leveraging a graph database. By analyzing interactions between users and movies—such as ratings and genres—we aim to personalize recommendations, ensuring they are relevant and engaging to each user.

Methodology

Data Acquisition

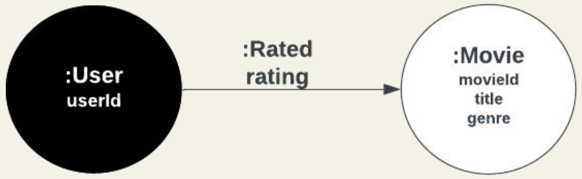
We sourced our dataset from Kaggle, consisting of two CSV files: one for movies (movies.csv) with columns for movie ID, title, and genre; and another for ratings (ratings.csv) with user ID, movie ID, and ratings. These datasets form the foundation for our graph-based model.

Data preparation

We used Python and Pandas for data preprocessing:
- Movies Data: Trimmed whitespace, removed problematic characters from titles, and handled missing genres.
- Ratings Data: Converted data types, and removed rows with missing values.

Data Modelling

We designed a graph model with:
Nodes: Movies (with properties: movie ID, title, genre) and Users (with property: user ID).
Relationships: Rated (connecting Users to Movies, with property: rating).



Graph Statistics

Total Nodes and Relationships

Count of movies, users, genres, ratings, and genre assignments

Isolated Nodes

Identified movies with no ratings to capture data integrity issues.

Graph Analytics

User Centrality Analysis

Identifies the most connected users (those who rate the most) using Neo4j's centrality algorithm to find key influencers.

Path Analysis

Finds paths between users to identify those with similar tastes using Neo4j's shortest path algorithm.

Community Detection

Groups users into clusters based on movie genre preferences using Neo4j's community detection algorithm.

Machine Learning

1.Graph Model Construction
Built `userMovieGraph` linking users and movies via `RATED` relationships for efficient graph analytics.
2.Node Similarity Analysis
Predicted potential movie recommendations by comparing user ratings and creating new relationships based on similar tastes.
3.Node Embeddings
Used FastRP for node embeddings to uncover hidden patterns across the dataset.
4.User-Movie Similarity Calculation
Calculated cosine similarities between user and movie embeddings to identify personalized recommendations with high similarity scores.

Results

- **Evaluation Dataset Creation:** By randomly selecting 10% of user ratings as test data, we ensured accurate testing of model predictions. The evaluation dataset comprised 36,400 relationships labeled as "test".
- **Link Prediction Accuracy:** The model achieved high accuracy in predicting user preferences by recommending movies based on similarities between rated movies and the SIMILAR_TO relationships. Multiple users had accuracy scores above 80%, indicating the model effectively identifies potential recommendations.
- **Precision and Recall:** The model demonstrated high precision, achieving perfect scores (1.0) for many users. Recall scores, though slightly lower, still proved strong, highlighting the ability of the model to retrieve relevant recommendations.

CONCLUSION

The machine learning model provided a reliable and efficient means to predict user preferences and recommend movies that align with existing tastes.Future work could involve fine-tuning the model's similarity metrics, improving the recall rates, and exploring additional algorithms for more personalized recommendations.