MOVIE RECOMMENDATION SYSTEM: A DATA-DRIVEN APPROACH

Developed using the MovieLens dataset

Presented by: Ehab Herein

INTRODUCTION

- Objective: Build a content-based movie recommendation system
- Dataset source: MovieLens, including four key files (movies, ratings, tags, links)
- Approach: Content-based filtering using movie features like genre, release year, and tags

DATA COLLECTION

- Datasets used:
  a. movies.csv: Titles, genres, release years
  b. ratings.csv: Over 100,000 user ratings
  c. tags.csv: User-generated tags describing movie content
  d. links.csv: External identifiers for IMDb and TMDb
- Purpose: These datasets form the foundation for our feature extraction and model training.

DATA PROCESSING

- Key Steps:
  a. Extracted release years from titles (e.g., Toy Story (1995)→1995)
  b. Reformatted movie titles ("Title, The"→"The Title")
  c. Aggregated tags and cleaned missing data
  d. Calculated average ratings for each movie
- Goal: Ensure data consistency and prepare for feature engineering

MERGING AND FEATURE ENGINEERING

- Merging Process:
  Step 1: Merge movies.csv and links.csv
  Step 2: Merge the result with ratings.csv and tags.csv
- Feature Engineering:
  a. Created a release_decade feature (e.g., 1990s, 2000s)
  b. Constructed a combined_features column (genres + tags + ratings)

MODEL DEVELOPMENT

- Methodology:
  a. TF-IDF vectorization: Converts text data into numerical features
  b. Nearest Neighbors with cosine similarity: Finds similar movies by comparing feature vectors
- Goal:
  - Identify the top 20 most similar movies for any given input

EVALUATiON OF RESULTS

- Quality Improvements:
  - Filtered out duplicate and irrelevant titles
  - Sorted recommendations by average rating to prioritize highly-rated movies
- Outcome:
  - The system provides relevant, high-quality recommendations

RECOMMENDATION FUNCTION

- Steps:
  a. Check if the input movie title exists in the dataset
  b. Identify 20 similar movies using the model
  c. Filter and rank results by average rating, returning the top 10
- Output:
  - Recommendations include movie titles, ratings, genres, and URLs for further exploration

EXAMPLE RECOMMENDATiONS

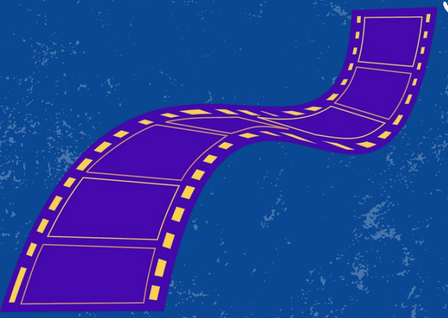
- Input Movie: "The Godfather"
- Top Recommendations:
  - Sympathy for the Underdog (Rating: 4.50)
  - The Godfather: Part II (Rating: 4.26)
  - Goodfellas (Rating: 4.25)
- Links:
  - Direct URLs to MovieLens, IMDb, and TMDb

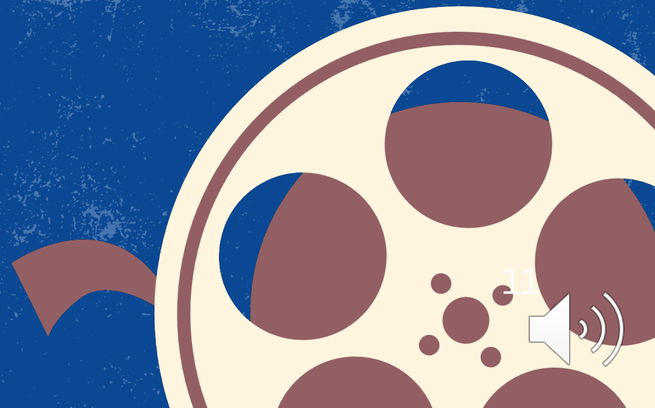
CHALLENGES AND IMPROVEMENTS

- Challenges:
  - Missing data for some movies
  - Balancing recommendation diversity and similarity
- Potential Improvements:
  - Implement collaborative filtering for hybrid recommendations
  - Experiment with different similarity metrics

RECOMMENDATiONS FOR FUTURE WORK

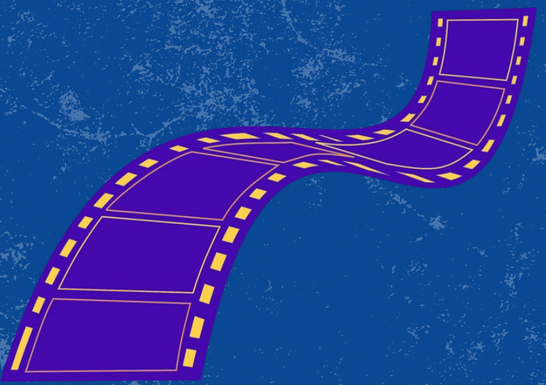
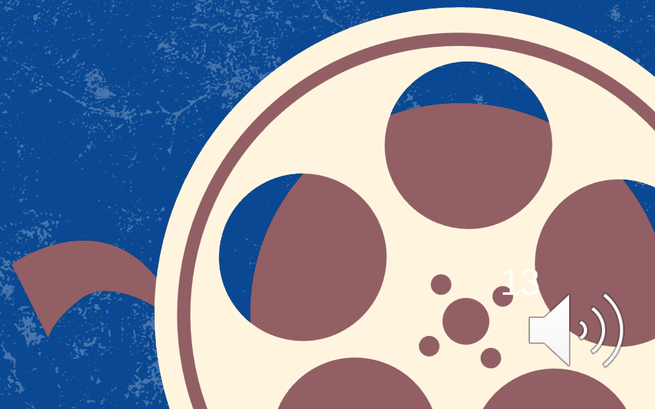
- Future Steps:
  - Expand the dataset with more genres and languages
  - Incorporate user feedback for continuous learning
  - Develop a scalable API for wider integration
- Visual Suggestion:
  - Roadmap graphic showing short-term and long-term goals.





CONCLUSION

- Summary:
  - Developed a robust movie recommendation system
  - Applied machine learning to improve user engagement
- Impact:
  - Enhanced content discovery and personalization

QUESTIONS AND DISCUSSION

"Do you have any questions or suggestions for improvement?"