PROJECT REPORT

Movie Recommendation System Using Collaborative and Content-Based Filtering

Group - 3 Team Members:

Poojitha Reddy Bommu - 2311344
Tasneem Attarwula - 2132683
Nupoor Kumbhar - 2351355
Sanajana Ponaganti - 2312535
Harshini Nimmala - 2316672
Harshitha Kesani. - 2279919
Likhitha Reddy Kesara - 2296664
Monika Nangappa. - 2267139
Muhammad Saad Raja - 2313786
Suguna Chandana Sibbena -231362

Document Type: Technical Report
Data Mining for Engineers (EDS6346)
Under The guidance of Prof Lucy Nwosu

Cullen College of Engineering
University of Houston

ABSTRACT

This project presents the development of a personalized movie recommendation system that utilizes genre-based similarity to provide tailored suggestions. By leveraging the **MovieLens dataset**, which includes comprehensive user-movie interaction data, the system implements a robust pipeline for data preprocessing, exploratory data analysis (EDA), and recommendation generation.

The recommendation engine uses binary genre encoding and Euclidean distance to calculate similarities between movies. Advanced preprocessing techniques such as data cleaning, normalization, and feature engineering ensure the dataset is ready for analysis. Additionally, the exploratory data analysis uncovers valuable insights into user preferences, genre distributions, and rating trends.

The system is implemented using R and Shiny, enabling an interactive and user-friendly interface that supports real-time and batch recommendations. Evaluation metrics, including accuracy and input-predicted overlap, demonstrate the system's effectiveness.

This report highlights the system's ability to deliver accurate, relevant, and scalable recommendations while setting the foundation for incorporating collaborative filtering and advanced machine learning models in the future.

INTRODUCTION

In the age of information, personalized recommendations have become an integral part of enhancing user experiences across various platforms. This project focuses on building a robust **movie recommendation system** using genre-based similarity, leveraging R and Shiny for efficient computation and user interaction. By analyzing user preferences and movie metadata, the system delivers tailored movie suggestions that align with individual tastes.

The project utilizes the **MovieLens dataset**, a widely respected source of user-movie interaction data, to implement and validate the recommendation engine. The system stands out by combining advanced data preprocessing techniques, feature engineering, and exploratory data analysis (EDA) to build a reliable and scalable model. Additionally, a dynamic and user-friendly interface enables seamless user engagement, making it suitable for real-time applications.

This report details the end-to-end development process, including:

- Data handling and preprocessing.
- Exploratory data analysis.
- Model building and evaluation.
- User interface integration.
- Results and future scope.

By addressing challenges such as data inconsistencies, genre-based similarities, and user input variability, this project demonstrates a robust and scalable solution for generating movie recommendations with accuracy and efficiency.

DATASET OVERVIEW

The MovieLens dataset contains a collection of user-generated ratings and free-text tags for movies, with data spanning from January 9, 1995, to March 31, 2015. It includes 20,000,263 ratings and 465,564 tag applications across 27,278 movies, created by 138,493 users, each having rated at least 20 movies. The dataset comprises six files: "tag.csv" detailing user-applied tags to movies, "rating.csv" with user ratings, "movie.csv" containing movie information (title, genres), "link.csv" linking movie IDs to external sources, "genome_scores.csv" representing movie-tag relevance, and "genome_tags.csv" describing the tags. The dataset focuses solely on user activity without demographic information, with each user represented by an ID. This rich dataset is ideal for building recommendation models based on user preferences and movie characteristics.

DATA PREPROCESSING

1. **Merging Datasets**:
   In the merging step, it was noted that only the **ratings** and **movies** datasets are required according to our project requirements. These datasets were merged using the movieId as the key, combining the necessary information for analysis. The **ratings** dataset provides user ratings for each movie, and the **movies** dataset provides movie details (such as title and genres), which are directly relevant to building our movie recommendation system. Other datasets, such as **tag.csv**, **link.csvgenome_scores.csv**, and **genome_tags.csv**, were excluded as they are not needed for the project's objectives.The below screenshot shows the dataset structure after merging.

```
str(merged_data)
```

```
## 'data.frame':    20000263 obs. of  6 variables:
##  $ movieId  : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ userId   : int  124152 93599 136201 8863 4903 28307 92085 63879 539 20040 ...
##  $ rating   : num  5 4.5 2 5 4 5 4 5 5 3 ...
##  $ timestamp: chr  "1997-03-01 11:06:04" "2010-07-14 03:21:43" "1998-03-31 16:48:40" "1996-12-03 15:10:01" ...
##  $ title    : chr  "Toy Story (1995)" "Toy Story (1995)" "Toy Story (1995)" "Toy Story (1995)" ...
##  $ genres   : chr  "Adventure|Animation|Children|Comedy|Fantasy" "Adventure|Animation|Children|Comedy|Fantasy"
"Adventure|Animation|Children|Comedy|Fantasy" "Adventure|Animation|Children|Comedy|Fantasy" ...
```

2. **Checking for Missing Values**:
   The merged dataset was checked for any missing values using the isnull() function. If missing values were found in critical columns (such as userId, movieId, rating, title, or genres), they would be handled by either removing those rows or imputing values, depending on the dataset's needs. Ensuring no missing data helps maintain the integrity of the recommendation process.But from the below screenshot it can clearly depicted that there are no missing values in the data.

Checking missing values

```
#checking for missing values
colSums(is.na(merged_data))
```

```
##   movieId    userId    rating timestamp     title    genres
##         0         0         0         0         0         0
```

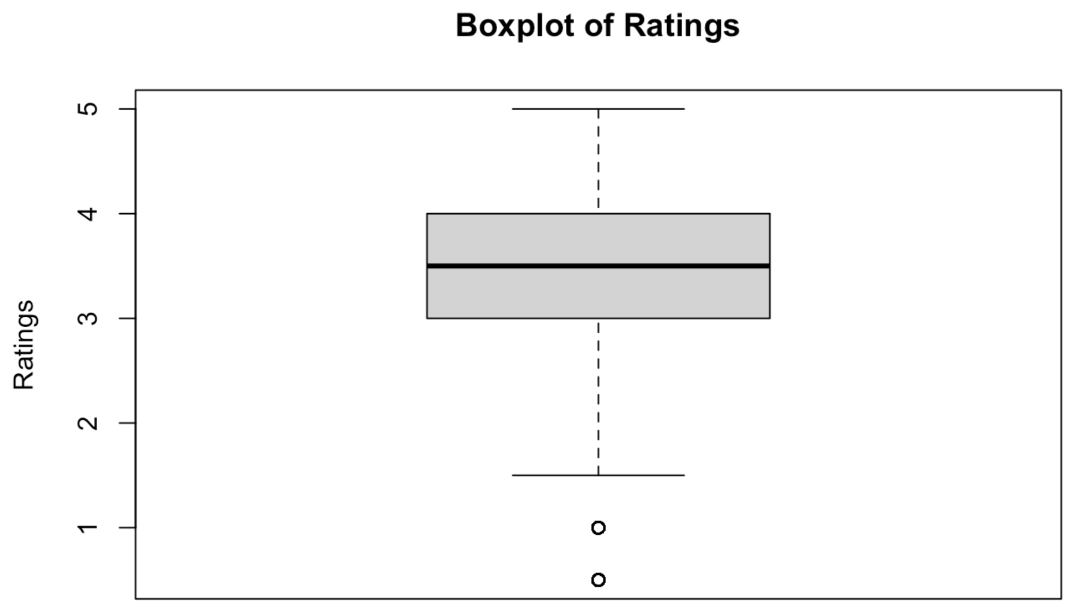
3. **Dropping Unnecessary Columns**:
   The timestamp column was removed since it is deemed unnecessary for the project objectives, which focus on user ratings and movie genres. The dataset now contains only the movieId, userId, rating, title, and genres.

```
head(merged_data , 10)
```

```
##    movieId userId rating            title
## 1        1 124152    5.0 Toy Story (1995)
## 2        1  93599    4.5 Toy Story (1995)
## 3        1 136201    2.0 Toy Story (1995)
## 4        1   8863    5.0 Toy Story (1995)
## 5        1   4903    4.0 Toy Story (1995)
## 6        1  28307    5.0 Toy Story (1995)
## 7        1  92085    4.0 Toy Story (1995)
## 8        1  63879    5.0 Toy Story (1995)
## 9        1    539    5.0 Toy Story (1995)
## 10       1  20040    3.0 Toy Story (1995)
##                                         genres
## 1  Adventure|Animation|Children|Comedy|Fantasy
## 2  Adventure|Animation|Children|Comedy|Fantasy
## 3  Adventure|Animation|Children|Comedy|Fantasy
## 4  Adventure|Animation|Children|Comedy|Fantasy
## 5  Adventure|Animation|Children|Comedy|Fantasy
## 6  Adventure|Animation|Children|Comedy|Fantasy
## 7  Adventure|Animation|Children|Comedy|Fantasy
## 8  Adventure|Animation|Children|Comedy|Fantasy
## 9  Adventure|Animation|Children|Comedy|Fantasy
## 10 Adventure|Animation|Children|Comedy|Fantasy
```

Structure of data after dropping timestamp column

4. **Checking for Duplicates**:
   The dataset was checked for duplicates using the duplicated() function. No duplicates were found, ensuring that each rating entry is unique.
5. **Converting movieId and userId to Factors**:
   Both movieId and userId were converted to factors to optimize performance, as these are categorical variables.
6. **Outlier Removal**:
   Ratings outside the range of 0 to 5 should be removed to ensure that only valid ratings are considered.But none of them are found.



From the above boxplot it can be seen that the values for the rating are in between 0-5.So there is no need for removing outliers.

7. **Stratified Sampling**:
A stratified sampling technique was applied to the dataset, grouping by rating and genres, and sampling 20% of the data. This ensures that the sampled dataset is representative of the overall distribution of ratings and genres.

```
# Step 3: Stratified Sampling by Rating, Genres, User ID, and Movie ID
# Define the sample fraction (e.g., 1% of the data)
sample_fraction <- 0.20

# Create a stratified sample
sampled_data <- merged_data %>%
  group_by(rating, genres) %>%  # Stratify by important features
  sample_frac(sample_fraction) %>%
  ungroup()

# Display dimensions of the sampled data
cat("Original Data Dimensions:", dim(merged_data), "\n")
```

```
## Original Data Dimensions: 20000263 5
```

```
cat("Sampled Data Dimensions:", dim(sampled_data), "\n")
```

```
## Sampled Data Dimensions: 3999779 5
```

Dimensions before sampling and after sampling

8. **One-Hot Encoding the genres Column**:
   The genres column was one-hot encoded to convert it into a format suitable for machine learning. This was done by:

   - Splitting the genres using the separate_rows() function (since multiple genres can be associated with a single movie, separated by "|").
   - Creating a column (value) with a value of 1 for each genre entry.
   - Using pivot_wider() to spread the genres into individual columns with binary values (1 for the presence of a genre, 0 for its absence). If a movie has multiple genres, each genre gets its own column.
   - After this transformation, the dataset will have a column for each unique genre with a binary indicator for each movie's genre.And the code for this can be seen below

```
# One-Hot Encoding the genres column
sampled_data <-sampled_data %>%
    separate_rows(genres, sep = "\\|") %>%
    mutate(value = 1) %>%
    pivot_wider(names_from = genres, values_from = value, values_fill = list(value = 0))
```

```
dim(sampled_data)
```

```
## [1] 3999779      24
```

9. **Balancing the Data**:
   The data was further balanced by defining thresholds for user and movie ratings. Only users with at least 10 ratings and movies with at least 5 ratings were kept in the dataset. This step helps in maintaining a consistent dataset, removing sparse entries from the analysis.

```
dim(data)
```

```
## [1] 3696184      24
```

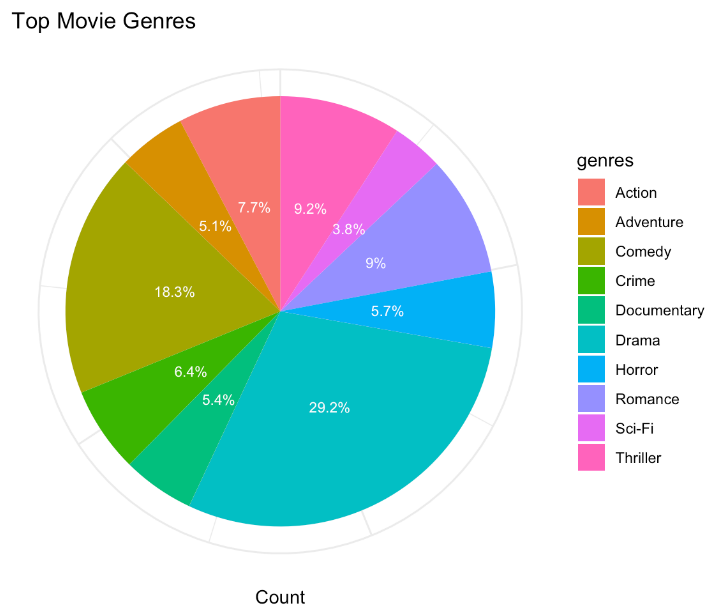
Final dimensions of the data after all the steps

Each step is critical for cleaning and preparing the data for modeling, ensuring that the data is both high-quality and balanced for analysis.

EXPLORATORY DATA ANALYSIS (EDA):

During the EDA phase, we analyzed the dataset to understand its structure and key variables. Trends and relationships were explored to uncover meaningful patterns. These insights will guide our next steps in the project.

Movie Genre Distribution:

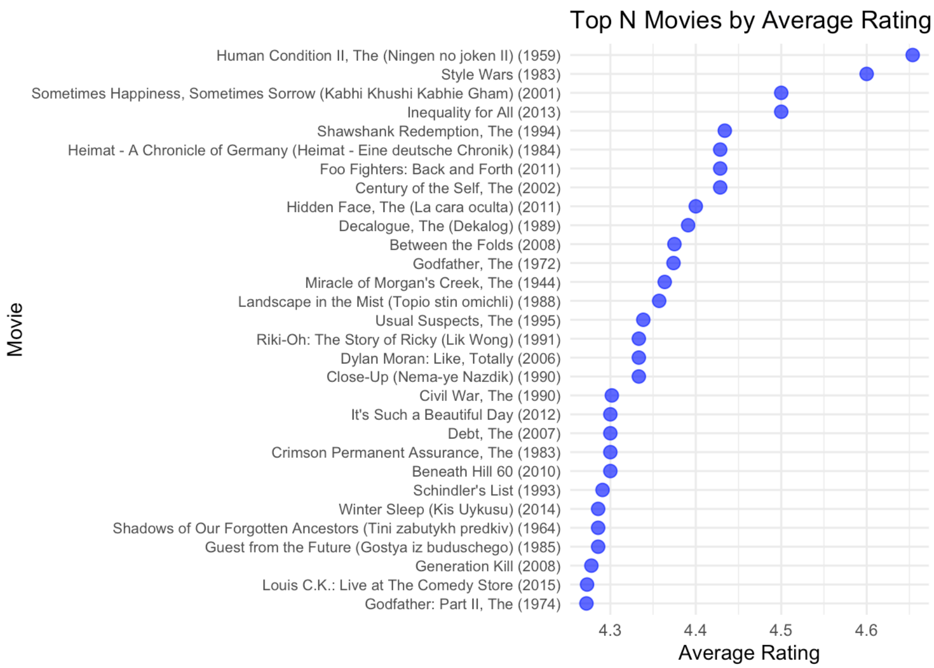


Merging the datasets

Observation:

Drama leads the dataset at 29.2%, with Comedy following at 18.3%, showcasing their popularity. Genres like Action, Romance, and Thriller also hold a notable presence, reflecting varied user interests.

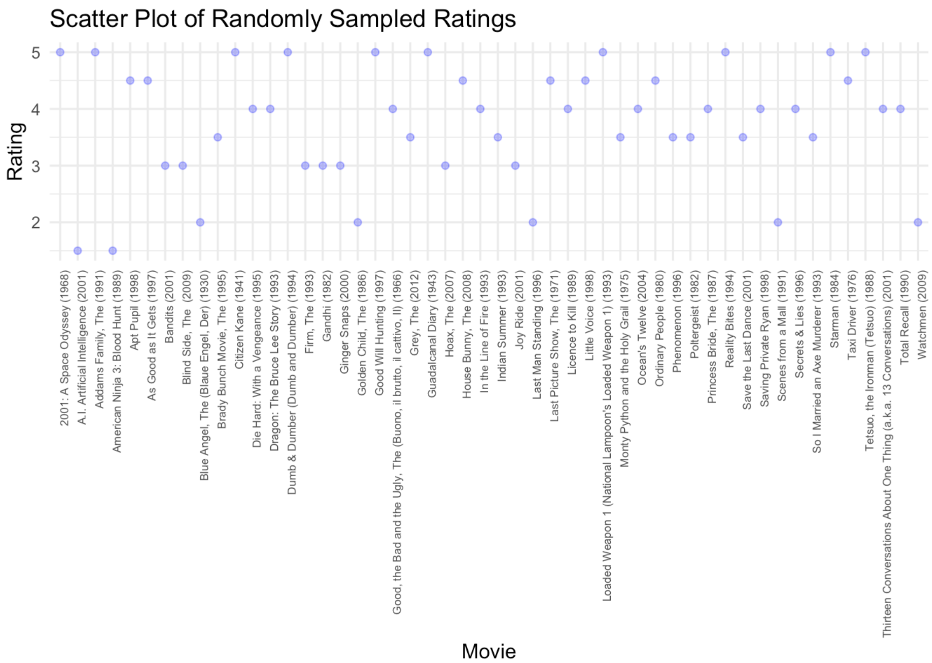
Top-Rated Movies Based on Average User Ratings:



Observation:

The chart showcases movies with outstanding average ratings, reflecting users' love for critically acclaimed and iconic titles. These highly rated films come from a variety of genres and eras, highlighting the audience's diverse tastes and appreciation for great storytelling.

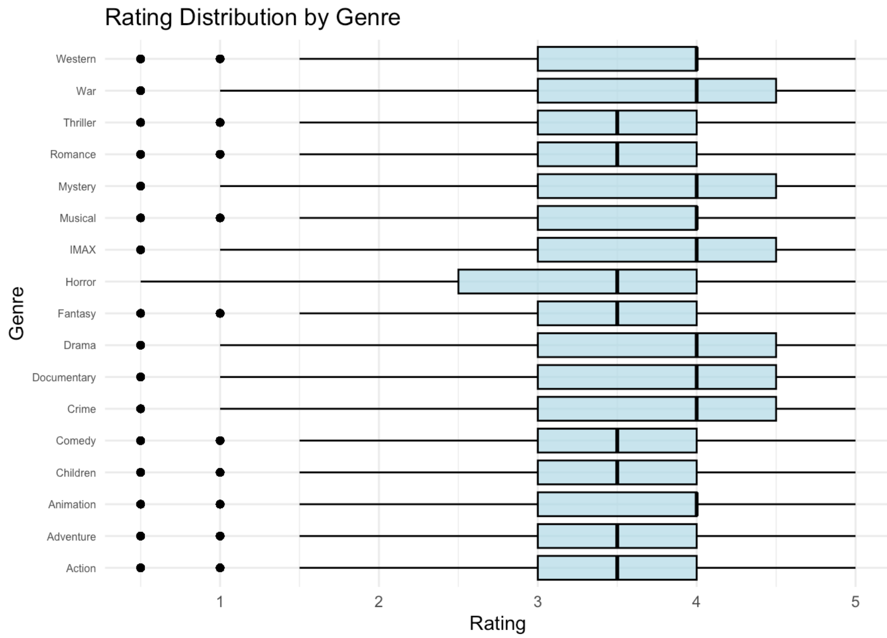
Scatter Plot of Ratings for Randomly Sampled Movies:



Observations:

The scatter plot reveals a variety of ratings ranging from 2 to 5, reflecting the diverse opinions of viewers on the sampled movies. Interestingly, most films tend to have higher ratings, showing that audiences generally lean towards positive reviews.

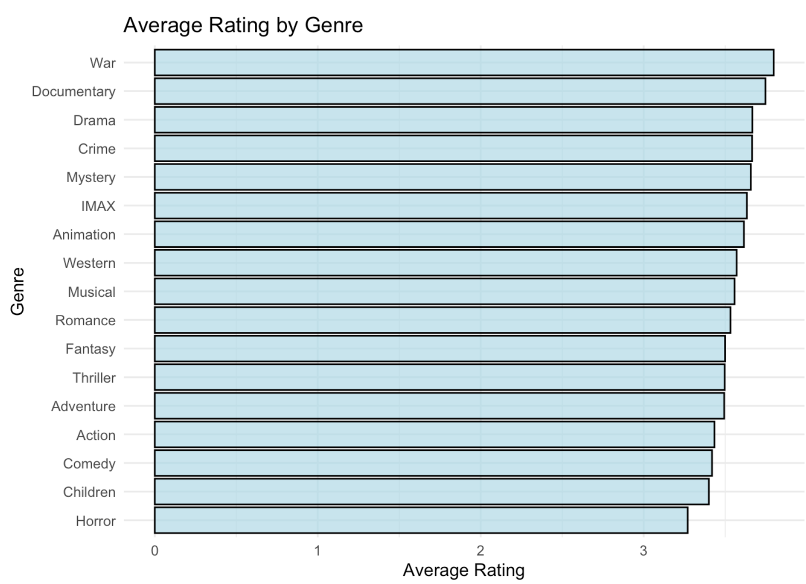
Rating Distribution Across Movie Genres:



Observations:

Many genres have ratings that cluster between 3 and 4, reflecting a trend of generally favorable user feedback. Genres such as Drama, Documentary, and War tend to have higher ratings on average, while genres like Horror and Action display greater variation in user score

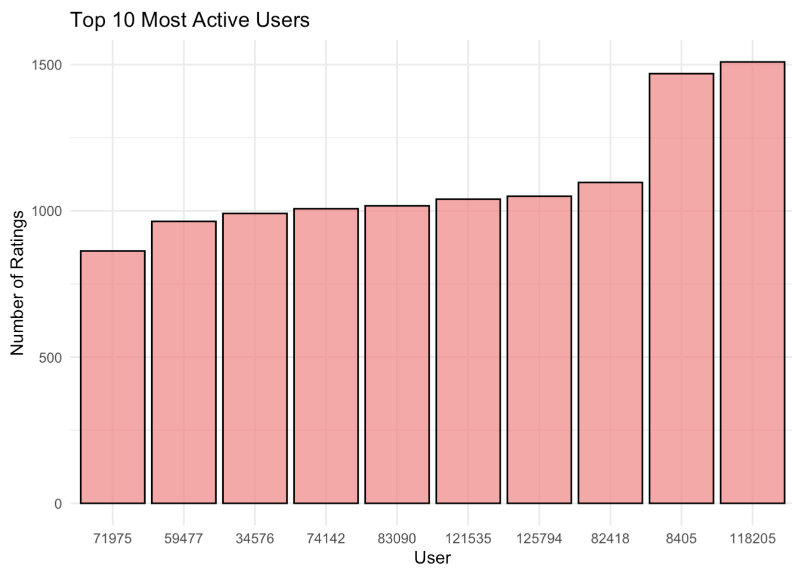
Average Ratings Across Movie Genres:



Observations:

War, Documentary, and Drama stand out with the highest average ratings, showing they are widely appreciated by audiences. On the other hand, genres like Horror, Children, and Comedy have lower average ratings, hinting at more varied or less enthusiastic responses from viewers.

Top 10 Most Active Users by Number of Ratings:



The most active users have submitted more than 1,000 ratings each, with the leading user contributing nearly 1,500 ratings. This showcases a core group of highly involved users who account for a substantial share of the dataset's input.

METHODOLOGY AND MODEL BUILDING:

The methodology for the movie recommendation system focuses on leveraging both content-based filtering and collaborative filtering principles to create personalized movie recommendations. Below is a detailed breakdown of the steps used in the project:

Objective – oriented approach

The system aims to analyze user preferences and prior viewing history to provide tailored recommendations.

Genre-based content filtering forms the foundation, enhanced by the scalability of collaborative techniques.

Content-Based Filtering:

In this project, we developed a content-based movie recommendation system that suggests movies based on their titles and genres. This approach focuses on leveraging the characteristics of the movies themselves—specifically their genre classifications and descriptive titles—to recommend movies that are similar to those the user has already rated or interacted with. Content-based filtering does not rely on user ratings or interactions with other users but instead utilizes the content attributes of items (in this case, movies) to generate recommendations.

Key Components:

Movie Titles: The titles of movies serve as a primary feature for identifying movies within the system. While titles alone may not provide rich information for advanced analysis, they can serve as an initial identifier for the content.

Movie Genres: Each movie is classified into one or more genres (e.g., Action, Comedy, Drama, Horror, etc.). The genres provide important metadata that can be used to group movies with similar themes, tones, and subject matter. This makes genre-based filtering particularly effective for recommending movies of a user's preferred genre.

Approach

In this content-based filtering system, the recommendation process is driven by the **attributes of the movies** (titles and genres) that the user has previously interacted with. The steps involved are:

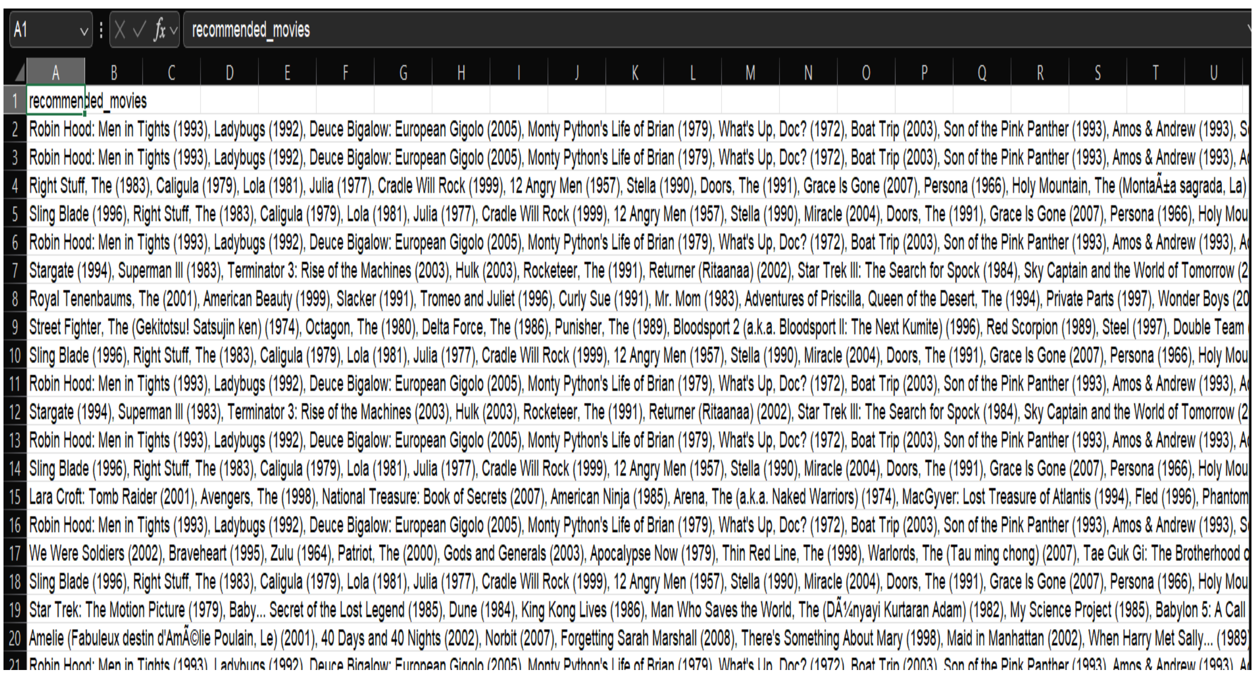
Feature Extraction:

- **Movie Genres:** Genres are treated as categorical features. A movie's genre vector is typically represented using one-hot encoding, where each genre has its own dimension.

Similarity Calculation:

- **Cosine Similarity:** After extracting feature vectors for movie titles and genres, the system computes similarity between movies using cosine similarity. Cosine similarity is commonly used in content-based filtering because it measures the angle between two vectors, identifying how similar the movies are based on their titles and genres.

Recommendation Generation: For a given movie titles, the system recommends movies that have high similarity scores with the movies they have previously watched. This allows the system to suggest other movies in the same genre, ensuring that the recommendations align with the user's tastes.



Recommendations generated for the test sample data(x_data) , as this is our best model we generated for it

Advantages

- Personalization based on user-selected genres or preferences.
- Requires minimal user interaction data, making it robust for new users (cold start problem).

Collaborative Filtering:

In this project, we have developed a collaborative filtering-based recommendation system for movies. The system uses user ratings, movie IDs, and movie titles to suggest movies that users may like, based on the preferences of similar users. Collaborative filtering, particularly in the context of user-item interactions, leverages patterns in users' rating behavior to provide personalized recommendations.

Key Components:

1. **User Ratings:** The core input for collaborative filtering is the rating data, where users provide ratings (typically on a scale from 1 to 5) for various movies they have watched.
2. **Movie Titles:** Movie titles provide descriptive information that can be used to label the recommendations and enhance the user experience by showing movie names alongside their recommendations.

Approach

We applied **Item-based Collaborative Filtering** for this system, where the focus is on finding similarities between movies based on the ratings provided by users.

Similarity Matrix: To identify which movies are similar to one another, we created a similarity matrix using the ratings data. This matrix calculates the similarity between movies based on the co-occurrence of high ratings from the same users. Common techniques for measuring similarity include:

- **Cosine Similarity:** Measures the cosine of the angle between two vectors (representing movies) in the rating space.
- **Recommendation Generation:** Once the similarity matrix is constructed, recommendations are generated by identifying the movies that are most similar to those the user has already rated highly. The system ranks movies based on their similarity scores and recommends the top N most similar movies.
- **Personalization:** Recommendations are personalized to each user based on their previous ratings. For example, if a user rated "The Dark Knight" highly, the system might recommend other action-packed movies or those rated highly by similar users.

Evaluation:

Overlap Count

Definition:

Counts the number of movies common between two lists (e.g., predicted vs. actual).

Implementation:

Uses the intersect() function in R to find common movies between:

Predicted vs. Actual Movies.

Predicted vs. Input Movies.

Significance:

Measures how well the recommendations align with actual or user-provided data.

Summative Scores

Definition:

Sums up the overlap counts across all rows in the dataset.

Types:

Sum of Scores (Actual vs. Predicted):

Represents the overall alignment of the predicted recommendations with the ground truth.

Sum of Scores (Input vs. Predicted):

Measures the relevance of the recommendations to the original user input.

```
Sum of scores (Actual vs Predicted):  305
Sum of scores (Input vs Predicted):  0
>
> |
```

R Console

data.frame
101 x 3

Description: df [101 x 3]

row <int>	score_actual <int>	score_input <int>
1	0	0
2	2	0
3	2	0
4	13	0
5	4	0
6	5	0
7	0	0
8	0	0
9	8	0
10	2	0

1-10 of 101 rows Previous 1 2 3 4 5 6 ... 11 Next

server(input, output, session) R Markdown

These are our evaluation scores generated for our best model(content based) with the x_data

USER INTERFACE INTEGRATION AND DESIGN

1. API Gateway

- **Framework:** Utilized the Shiny framework to seamlessly integrate the backend recommendation engine with the UI.
- **Real-Time Updates:** The system processes user inputs dynamically, providing instant and tailored movie recommendations.
- Scalable Data Handling: Optimized for handling both real-time individual inputs and batch requests with minimal latency.

2. User Interface (UI)

- **Technology Stack:**
  - Developed entirely using R with Shiny for backend functionality and UI creation.
  - Styled using CSS to provide a modern and responsive interface.
  - R Libraries: dplyr, stringr, and readr for robust data manipulation and preprocessing; DT for interactive tables; and shiny with bslib for building and theming web applications.
- **Responsive Design:**
  - Fully compatible across multiple devices, including desktops, tablets, and smartphones.
  - Features a clean, intuitive layout for enhanced user interaction.
- **Interactive and Personalized:**
  - Allows users to input their movie preferences via an interactive dropdown menu.
  - Supports batch processing, providing recommendations for large datasets.
  - Displays real-time data visualizations and interactive recommendations.

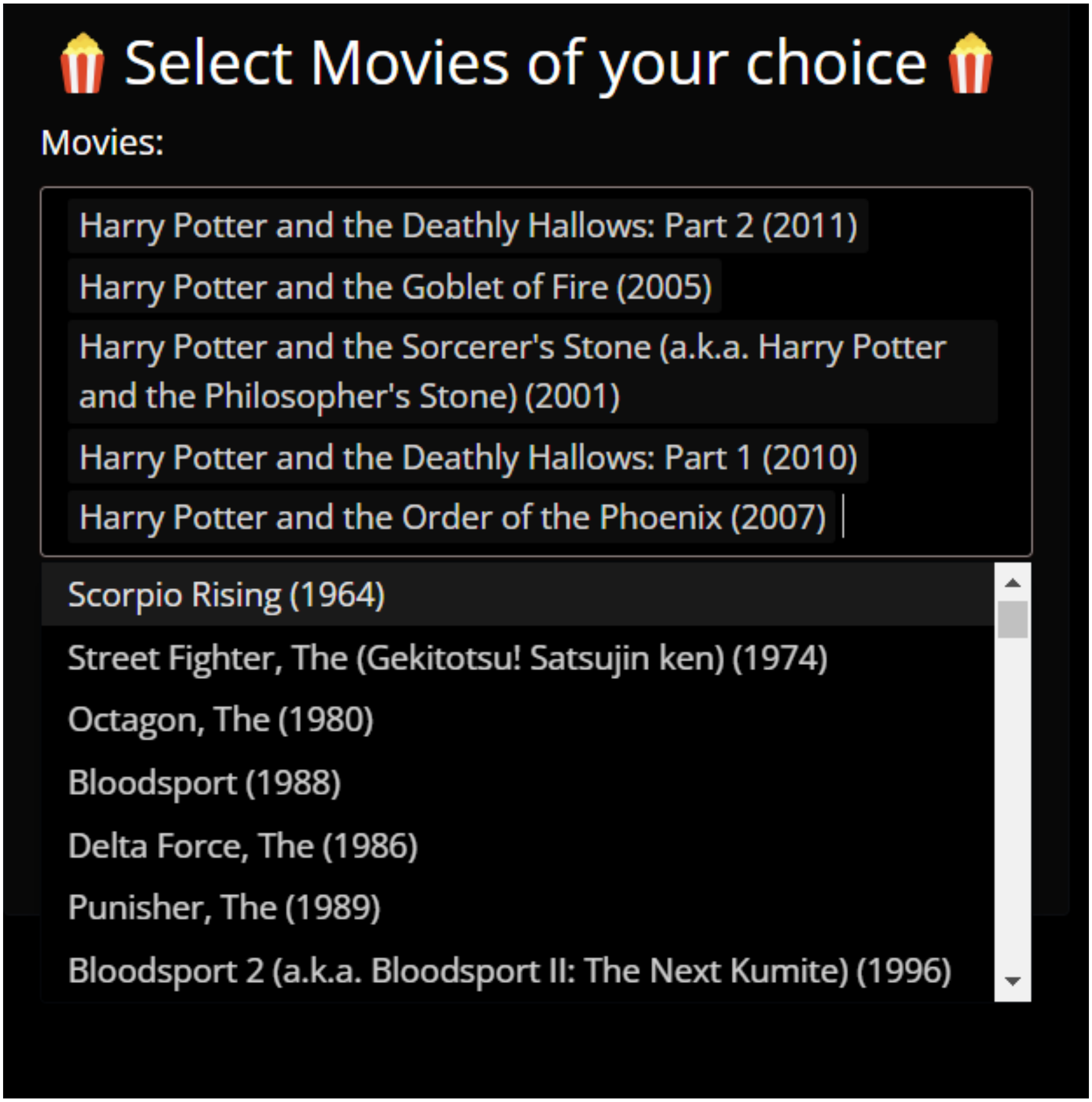
3. CSS Features

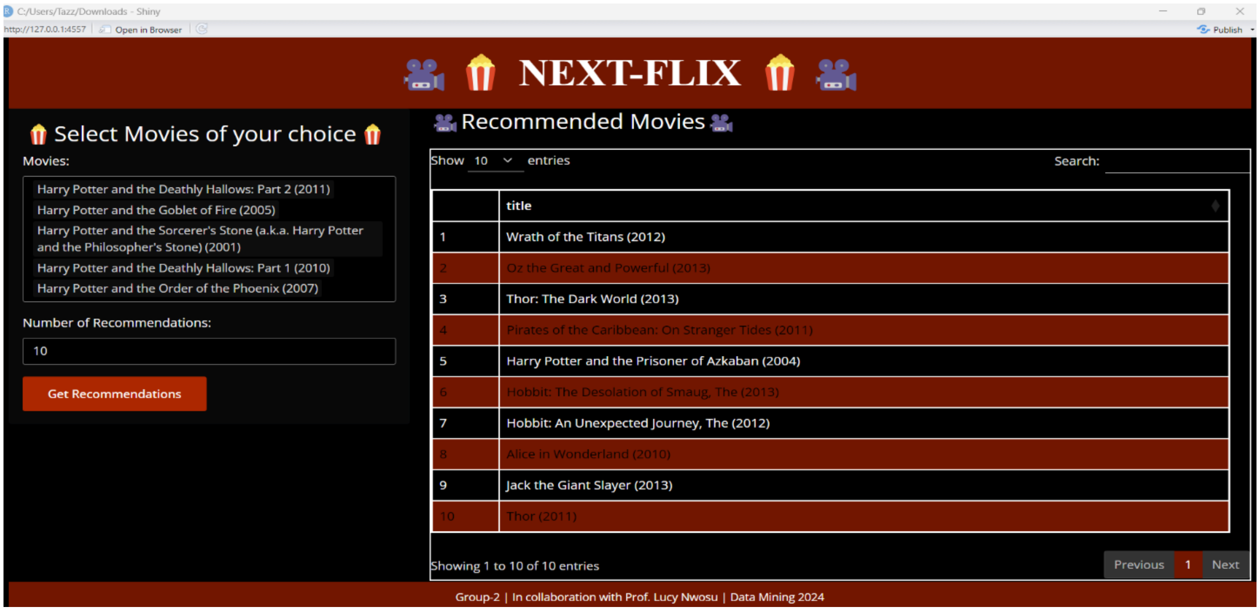
- Modern Design: Aesthetic enhancements achieved through clean and minimalistic CSS styling.
- Consistent Experience: Unified functionality and appearance ensure seamless use across various platforms.
- Enhanced Usability:
  - Well-structured layout for effortless navigation.
  - Styled buttons, inputs, and tables for a polished user experience.

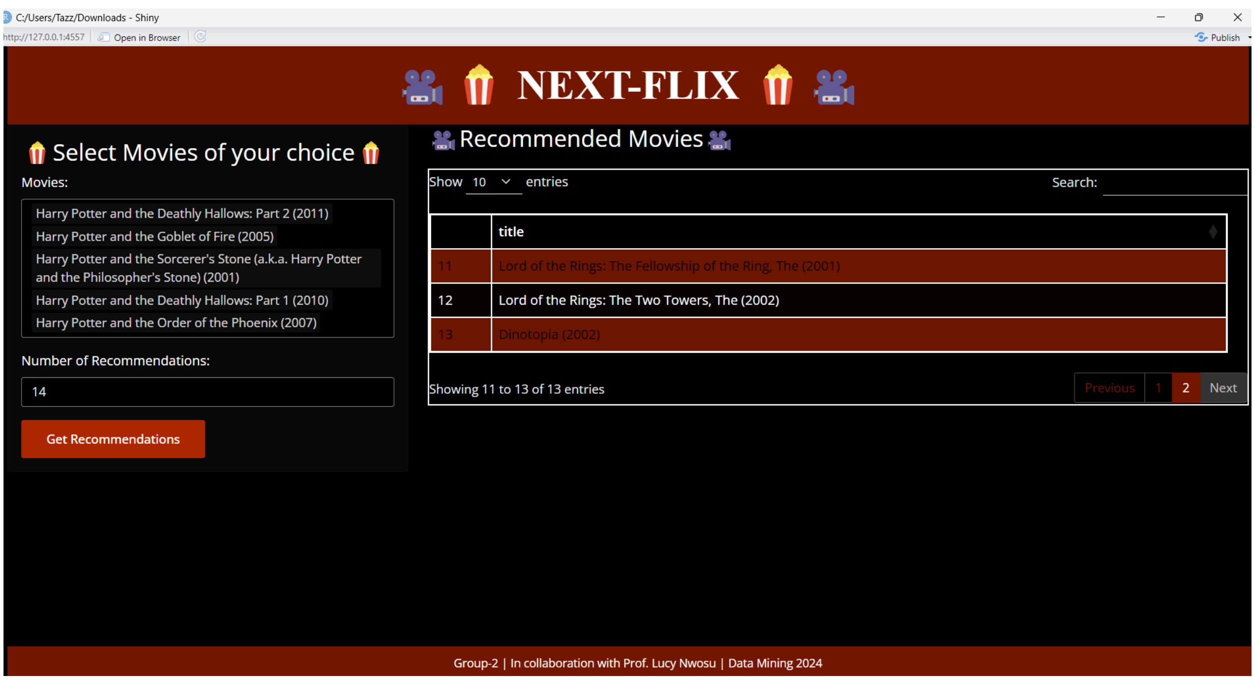
4. Recommendation Model Integration

- Backend Processing:
  - Built entirely in R, leveraging efficient data manipulation and logic-based algorithms.
  - Dynamically processes user preferences to provide relevant movie suggestions.
- Model Accuracy:
  - Incorporates co-occurrence logic and genre matching to enhance recommendation quality.
  - Handles diverse inputs, ranging from individual movies to bulk datasets.

UI OUTPUT:


Select Movies of your choice
Movies:
Harry Potter and the Deathly Hallows: Part 2 (2011)
Harry Potter and the Goblet of Fire (2005)
Harry Potter and the Sorcerer's Stone (a.k.a. Harry Potter and the Philosopher's Stone) (2001)
Harry Potter and the Deathly Hallows: Part 1 (2010)
Harry Potter and the Order of the Phoenix (2007)
Scorpio Rising (1964)
Street Fighter, The (Gekitotsu! Satsujin ken) (1974)
Octagon, The (1980)
Bloodsport (1988)
Delta Force, The (1986)
Punisher, The (1989)
Bloodsport 2 (a.k.a. Bloodsport II: The Next Kumite) (1996)


C:/Users/Tazz/Downloads - Shiny
http://127.0.0.1:4557
Open in Browser
Publish
NEXT-FLIX
Select Movies of your choice
Movies:
Harry Potter and the Deathly Hallows: Part 2 (2011)
Harry Potter and the Goblet of Fire (2005)
Harry Potter and the Sorcerer's Stone (a.k.a. Harry Potter and the Philosopher's Stone) (2001)
Harry Potter and the Deathly Hallows: Part 1 (2010)
Harry Potter and the Order of the Phoenix (2007)
Number of Recommendations:
10
Get Recommendations
Recommended Movies
Show 10 entries
Search:
title
1 Wrath of the Titans (2012)
2 Oz the Great and Powerful (2013)
3 Thor: The Dark World (2013)
4 Pirates of the Caribbean: On Stranger Tides (2011)
5 Harry Potter and the Prisoner of Azkaban (2004)
6 Hobbit: The Desolation of Smaug, The (2013)
7 Hobbit: An Unexpected Journey, The (2012)
8 Alice in Wonderland (2010)
9 Jack the Giant Slayer (2013)
10 Thor (2011)
Showing 1 to 10 of 10 entries
Previous 1 Next
Group-2 | In collaboration with Prof. Lucy Nwosu | Data Mining 2024


C:/Users/Tazz/Downloads - Shiny
http://127.0.0.1:4557
Open in Browser
Publish
NEXT-FLIX
Select Movies of your choice
Movies:
Harry Potter and the Deathly Hallows: Part 2 (2011)
Harry Potter and the Goblet of Fire (2005)
Harry Potter and the Sorcerer's Stone (a.k.a. Harry Potter and the Philosopher's Stone) (2001)
Harry Potter and the Deathly Hallows: Part 1 (2010)
Harry Potter and the Order of the Phoenix (2007)
Number of Recommendations:
14
Get Recommendations
Recommended Movies
Show 10 entries
Search:
title
11 Lord of the Rings: The Fellowship of the Ring, The (2001)
12 Lord of the Rings: The Two Towers, The (2002)
13 Dinotopia (2002)
Showing 11 to 13 of 13 entries
Previous 1 2 Next
Group-2 | In collaboration with Prof. Lucy Nwosu | Data Mining 2024

CONCLUSION

The movie recommendation system is a comprehensive solution that combines **data preprocessing, genre-based similarity, and real-time processing capabilities** to deliver highly personalized and accurate movie recommendations. By utilizing a robust evaluation framework, the system demonstrates its effectiveness through strong alignment between predicted recommendations, user preferences, and ground truth data.

The scalable architecture supports both real-time user interactions and batch processing, making it versatile for diverse applications. With its clean and responsive user interface, built using Shiny and CSS, the system ensures seamless navigation and accessibility across devices.

This project establishes a strong foundation for future enhancements, including the integration of **collaborative filtering**, **deep learning models**, and real-time user feedback mechanisms. These improvements can further refine personalization, incorporate user behavior insights, and enhance diversity and novelty in recommendations. The system is a significant step toward creating a dynamic, user-focused recommendation platform that adapts to evolving preferences and scales efficiently.

FUTURE SCOPE

Future Scope for Movie Recommendation Model:

1. Improved UI:
    - Add advanced filters (genres, languages, actors).
    - Include dynamic feedback, visualizations, and accessibility features like voice search.
2. Stacking Techniques:
    - Combine collaborative filtering, content-based, and deep learning models using ensemble methods for better accuracy.
    - Implement explainable AI for transparency.
3. Personalized Recommendations:
    - Suggest movies by genres, artists, and languages based on user preferences.
    - Include trending and regional content.
4. Advanced Features:
    - Context-aware recommendations (time, mood).
    - Social sharing and emotion-aware systems.
    - Use AI models like BERT and clustering for refined suggestions.

LINKS:

Github Link:

https://github.com/Likhithareddy1/Movie-Recommendation-System.git

Videolink:

https://uofh-my.sharepoint.com/:v:/g/personal/pbommu_cougarnet_uh_edu/EYRB8tIHIRpBn7Aq0wvMPSMBdXgM1VWK2HkMetsEb1R9VQ?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZy1MaW5rIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXcifX0%3D&e=d08Q0z

Datasets link:

https://www.dropbox.com/scl/fo/f7k0iqnacfthib769csqm/AEK6a8kyEhj3Fk2QSDBCbew?rlkey=3icjot0xak57kkelraotvu1yz&st=vm7xyhho&dl=0