MOVIE RECOMMENDATION SYSTEM

PRML PROJECT

TOPICS COVERED

1 : INTRODUCTION

2 : OVERVIEW OF DATASET COMPONENTS

3 : DATA PREPROCESSING

4 : APPROACHES

5 : LIMITATIONS AND CHALLENGES

6 : COUNTVECTORIZER AND COSINE SIMILARITY

7 : WEBSITE DEVELOPMENT

8 : INTERFACE

INTRODUCTION

- Title: Exploring Movie Recommendation System Dataset

- Overview:
  - Description of the dataset and its components.
  - Brief statistics: 100,836 ratings, 3,683 tag applications, and 9,742 movies.
  - Consistency in movie IDs across files.

DATASET COMPONENTS

1. Movies.csv

- Information about 9,742 movies.
- Unique movie ID, title with release year, and genres.
- Example format: movieId, title, genres.

2. Tags.csv

- User-generated tags for movies.
- Includes user ID, movie ID, tag, and timestamp.
- Example format: userId, movieId, tag, timestamp.

3. Ratings.csv

- Ratings on a 5-star scale.
- Includes user ID, movie ID, rating, and timestamp.
- Example format: userId, movieId, rating, timestamp.

4. Links.csv

- External identifiers for movies.
- Includes MovieLens, IMDb, and TMDb IDs.
- Example format: movieId, imdbId, tmdbId.

DATA PREPROCESSING

- **Data Loading and Exploration:**
  - Utilized pandas library for CSV file loading.
  - Conducted exploratory data analysis to understand dataset.
  - Checked for missing values:
    - None in movies, tags, ratings.
    - 8 missing in links, neglected.

- **Data Cleaning Steps:**
- Selected relevant columns from tags DataFrame.
- Removed timestamp column from tags and ratings DataFrames.
- Split genres in movies DataFrame.
- Removed extraneous whitespace from tags DataFrame.
- Combined tags for each movie in tags DataFrame.

DATA PREPROCESSING

Data Merging and Transformation:

- Merged movie information with combined tags.
- Created a new column "description" and dropped redundant columns.
- Lowercased text in "description" column for consistency.
- Applied remove_repetitive_words function to remove repeated words.
- Applied stemming to standardize text using NLTK Porter Stemmer algorithm.

APPROACHES

DBSCAN:

- DBSCAN, or Density-Based Spatial Clustering of Applications with Noise, is a clustering algorithm particularly suited for datasets with irregular shapes and varying densities.
- In our context, DBSCAN is employed to group movies based on their feature vectors derived from movie descriptions.
- By analyzing the density of data points, DBSCAN identifies core points (movies with many similar features) and expands clusters around them.
- This approach enables the system to recommend movies that are closely related in terms of their thematic content, even if they don't belong to predefined genres.

K-means Clustering

- K-means clustering is a widely used unsupervised learning algorithm for partitioning a dataset into K distinct, non-overlapping clusters.
- In our application, K-means clustering is utilized to group movies based on their genre information.
- We transform categorical genre data into a binary matrix representation, with each genre represented as a binary feature.
- The number of clusters for K-means is set to the total number of unique genres, ensuring each cluster represents movies sharing similar genre characteristics.
- This approach allows for the recommendation of movies with similar thematic elements within the same cluster, facilitating personalized recommendations based on genre preferences.

APPROACHES

Agglomerative Clustering

- By grouping together movies that have similar descriptions or characteristics. This grouping helps identify clusters of movies that share common themes, genres, or content.
- Agglomerative clustering builds a hierarchy of clusters by iteratively merging pairs of clusters that are closest to each other until all data points belong to a single cluster.
- Once the clustering is done, each movie belongs to a specific cluster. Recommendations are then made by suggesting other movies within the same cluster as the input movie

COUNTVECTORIZER AND COSINE SIMILARITY

- Text to Numerical Vectors: Text data in the description column is converted to numerical vectors using the countVectorizer class.
- Transformation to Matrix: CountVectorizer t ransforms the text data into a matrix of token counts represented as a NumPy array. Each row of the vectors array corresponds to a movie in the dataset.
- Cosine Similarity Calculation: The cosine similarity between all pairs of movie descriptions represented as vectors is calculated.
- Recommendation Process: For a given movie its cosine similarity score is sorted in descending order and top 5 movies are selected for recommendation.

WEBSITE DEVELOPMENT

- Enhanced User Experience: Description of the developed website for better user interaction.
- Intuitive Interface: Mention of user-friendly features designed to improve the user experience.
- Movie Selection and Recommendations: Overview of browsing movies and receiving recommendations based on user selections.

INTERFACE

1.

Movie recommender System

Select a movie

Toy Story

Recommend movies

2.

Movie recommender System

Select a movie

Click to get movie recommendations

Recommend movies

3.

Movie recommender System

Select a movie

Toy Story

Recommend movies

Antz | Bug's Life, A | Adventures of Rocky and Bullw | Emperor's New Groove, The | Monsters, Inc.