Movie Recommendation System

Submitted for:

Statistical Machine Learning CSET211

Submitted by:

E23CSEU1900 - Kashish

Submitted to:

Shakshi Sharma

July-Dec 2024

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

INDEX

ABSTRACT

- **Objective**: The primary goal of a movie recommendation system is to filter and predict movies that a user is likely to enjoy based on their movies they search

- **Techniques Used:**

  Content-Based Filtering: This approach recommends movies based on the attributes of the movies themselves, such as genre, cast, and director, aligning with the user's past likes.

- **Implementation Steps**:

Data Collection: Gathering a comprehensive dataset that includes movie details and user ratings.
Data Preprocessing: Cleaning and preparing the data for analysis, including handling missing values and normalizing data.
Model Training: Using machine learning algorithms to train the model on the dataset, allowing it to learn patterns and relationships.
Recommendation Generation: After training, the system can provide personalized movie suggestions to users based on their input and preferences

Introduction

In the era of digital streaming, the vast selection of available movies can overwhelm users, making it challenging to find content that aligns with their interests. A movie recommendation system aims to alleviate this problem by providing personalized suggestions that enhance user engagement and satisfaction. This project focuses on developing a content-based movie recommendation system that leverages the characteristics of movies to recommend similar items based on user preferences. Content-based filtering is a technique that analyzes the attributes of items (in this case, movies) to recommend similar items to users. This method relies on the assumption that if a user liked a particular movie, they are likely to enjoy other movies with similar features, such as genre, director, actors, and keywords. By utilizing metadata associated with each movie, the content-based approach can effectively cater to individual user tastes, providing a tailored viewing experience.

Methodology

- **Data Collection**

  Gather a comprehensive dataset containing movie information, including titles, genres, descriptions, cast, directors, and other relevant attributes

- **Data Preprocessing**

  - Clean the dataset by handling missing values, removing duplicates, and standardizing formats.
  - Convert categorical variables (e.g., genres, cast) into a suitable format for analysis, such as one-hot encoding or using a bag-of-words model for textual data.

- **Feature Extraction**

- Identify and extract relevant features from the movie data. This may include:
  - **Genres**: Categorizing movies into genres (e.g., action, drama, comedy).
  - **Keywords**: Extracting significant terms from movie descriptions.
  - **Cast and Crew**: Including actors, directors, and producers as features.
- Represent the features in a structured format, such as a feature vector or a term-document matrix.

- **Similarity Calculation**
  - Use a similarity measure to evaluate the similarity between movies based on their features. Common methods include:
    - **Cosine Similarity**: Measures the cosine of the angle between two non-zero vectors in a multi-dimensional space.
    - **Euclidean Distance**: Calculates the straight-line distance between two points in feature space.
  - Create a similarity matrix that quantifies the similarity scores between all pairs of movies.

- **Recommendation Generation**

- **Deployment**
  - Implement the recommendation system in a user-friendly interface, allowing users to input their preferences and receive personalized movie suggestions.

Software Requirements

1. **Programming Language**: Python (preferred for its extensive libraries and community support).
2. **Libraries and Frameworks**:
    - **Data Manipulation**:
        - Pandas
        - NumPy
    - **Machine Learning**:
        - Scikit-learn (for similarity calculations and model evaluation)

Development Environment:

- Google colab notebook or any Python IDE (PyCharm) for code development and testing.

Future Scope

Content-Based Filtering

Real-Time Recommendations