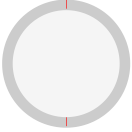



PLAGIARISM SCAN REPORT

	0% Plagiarised		100% Unique	Date	2024-11-05
				Words	942
				Characters	6721

Content Checked For Plagiarism

AIE425 INTELLIGENT RECOMMENDER SYSTEMS, FALL SEMESTER 24/25
Assignment #1: Neighbourhood CF Models (User, Item-Based CF)
Mohamed Ibrahim Fekry A20000726

Page 1 of 13

1. Introduction:

In the modern world, recommender systems are ubiquitous in almost all such online applications that assist to retrieve new content based on interests, and previous user activity. They improve the quality of the application since paraphrasing good rated content will cause increase in user's engagement and activity in the application. The present report deals with the implementation of a movie recommendation system built in Python and utilizing user based collaborative filtering technique which today is one of the prominent recommendation systems techniques.

User based collaborative filtering is based on the idea that people who rated or interacted with a set of items would be similar to other people who have a different set of items rated or interacted with. In this instance, the system finds users who have similar tastes in films and suggests films enjoyed by those users. This method draws on the power of crowdsourcing.

It is notable that the data for the system is retrieved using an API offered by TMDb, which stands for The Movie Database, the latter being a considerable source of

information regarding movies such as its respective rating, genre and other metadata also. The recommendation system aims at modeling and predicting real-life user behavior dynamics and as a result applicable suspicions, which cause the system to mimic user rating behavior. Namely, the so called user-item matrix is created where each cell represents a user's rating towards a certain film.

The focus of this project however is to look into how well user based collaborative filtering can be used to come up with effective movie recommendations. This in turn helps the system in finding films those particular individuals who are most likely to appreciate the content enhancing the viewing experience.

Page 2 of 13

Coverage of the Following Main Points

The development and significance of recommender systems over the years.
A description of collaborative filtering methods and techniques (user based versus item based).
Where TMDb API fits in this scope of work that is what does it do in the course of this work.

2. Overview of Coding Techniques:

The coding technique employs several components of which are structured. Each of these stages works toward the overall process of development and implementation of the recommendation system.

2.1 Establishing the Environment

The first step in building any software application is to prepare the environment which consists of the setting up of dependencies and frameworks.

Libraries Used:

Requests: This is an important extension to make HTTP requests to any API service such as TMDb to populate the code with retrieved information or attained answers.
Pandas: This is another powerful library for data manipulation designed explicitly for data analysis, offering data containers like DataFrames that are suitable for storing data in an organized way.

Page 3 of 13

Numpy: Numpy is a Library that is packaged with important numerical capabilities. The library provides multi dimensional array objects and a number of mathematical functions in support of data and computations.

Scikit-learn: This library is mainly designed for machine learning applications. It also provides utility functions for data mining and analysis which improves implementations of cosine similarity in collaborative filtering.

Matplotlib: This library is used for visualizing information and ideas. It supports the designing of charts and figures which helps in capturing information clearly and efficiently.

APIs Keys: It is also vital that the codes above be used with a TMDb unique identity which allows access to their film database. This code should be kept private and not

availed publicly to avoid misuse.

2.2 Acquiring and Preprocessing Data

In this segment, data from the TMDb API will be utilized to create a database where movies can be recommended.

How to Get Most Popular Movies:

functions `fetch_popular_movies` and `fetch_all_movies` are two functions that obtain movie related data. The first one gets the movies available at any one page while the second acquires data contained in several pages in order to create a complete list of movies available.

Page 4 of 13

The API will respond with JSON which will basically have a collection of movies in it. Each of the objects containing the movies has certain properties where some of the properties are the name of the movie, when it was produced out and the rates for the movie. This information is then collected and arranged in a way suitable for analysis.

Pandas Data Frame Creation:

The imported data is converted into a numpy array and stored in a Pandas DataFrame for ease of processing. A Data Frame is useful in handling data. It allows operations of that include filtering, grouping and also pivoting to be done..

2.3 Creating User Ratings

Because actual user information is generally lacking, the code emulates the User Engagement by generating random User Ratings for the Movie Database.

Generational Rating:

User ratings are randomly generated for a given number of users and movies. Each user is given a rating of 1 to 5 for the set of available movies. This aids in building a user-item matrix which resembles actual user behavior.

Why Simulation:

User rating simulation is an important aspect of designing and evaluating the recommendation algorithm. It allows one to build the required data structures and investigate user preferences and user's relations without the need of any real users, which in many cases, is not possible.

2.4 User-Based Collaborative Filtering

As soon as the user-item matrix is achieved, the following phase concerns the determination of the like users in terms of their rating patterns.

Page 5 of 13

The User-Item Matrix:

atings.

No plagiarism found

Check By:  Dupli Checker