Report on the Project: Movie Recommendation Using Pearson Correlation

Alessio Marinucci, 546778 → repo GitHub: https://github.com/alemari7/AdvancedTopics

Introduction

This report describes the development process of a movie recommendation system based on Pearson correlation between users. The system was implemented by using Python programming language and key libraries such as Pandas and Math for data manipulation and statistical calculations.

Methodology

1. Data Preparation

Movie ratings were loaded from a CSV file using the Pandas library. After loading, unnecessary columns such as the timestamp of ratings were removed to simplify data analysis.

2. Calculation of Pearson Correlation

Pearson correlation between the user of interest (USER_1) and all other users in the dataset was calculated by Using Panda's data aggregation and manipulation capabilities. This was achieved by iterating over all users and using scipy's `pearsonr()` method to compute the correlation between movie ratings. The reason `pearsonr` was used is because it's a common way to calculate the correlation between two variables, in this case the movie ratings of two users. Pearson correlation is a measure of how two variables are linearly related to each other. In this context, we're interested in identifying users who have similar preferences for movies, and Pearson correlation provides a measure of this similarity. By using Pearson correlation, we can find users who have similar movie preferences and therefore recommend movies based on what similar users have liked

3. Prediction of Movie Ratings

A prediction function was implemented to calculate the predicted ratings of movies for the user of interest. This function utilizes the Pearson correlation calculated in the previous phase and the ratings of other users to estimate the rating of a movie for the user of interest. This approach is chosen because it leverages the similarity between users to make predictions. It assumes that users who are similar to each other in terms of their movie preferences are likely to rate movies similarly. By using a weighted sum of the ratings given by similar users, the function captures the influence of these similar users on the prediction, with more similar users having a greater impact. If there are no similar users, the function resorts to the mean rating of the user, providing a baseline prediction. Overall, this method provides personalized predictions based on the user's own ratings and the ratings of similar users.

4. Movie Recommendation

Finally, a function was developed to recommend movies to the user of interest. Using the predicted ratings calculated in the previous phase, the function recommends movies that have not yet been rated by the user of interest, sorting them based on the predicted rating. This approach is chosen because it provides personalized recommendations for the user based on their own preferences and the preferences of similar users. By predicting ratings for unrated movies and selecting the top recommendations, the function suggests movies that the user is likely to enjoy, leveraging the collective wisdom of similar users. Overall, this method aims to improve user satisfaction by offering relevant and tailored movie suggestions.

5. Calculation of Euclidean Similarity

This function effectively finds users who have rated similar movies similarly to the target user. The Euclidean distance metric is used to quantify the dissimilarity between ratings, and the inverse of this distance serves as a measure of similarity between users. This approach is chosen because Euclidean similarity provides an intuitive measure of how close two sets of ratings are in a multi-dimensional space. By calculating the Euclidean distance between the ratings of `USER_1` and each other user, the function identifies users with similar preferences, thereby enabling personalized

recommendations. The inverse of the Euclidean distance is used to ensure that higher scores indicate greater similarity, making it consistent with other similarity metrics.

Euclidean similarity serves as an alternative to Pearson correlation for evaluating similarity between two data vectors. While Pearson correlation assesses linear correlation, Euclidean similarity focuses on geometric distance in Euclidean space. It's advantageous for several reasons: It offers conceptual simplicity and ease of implementation, grounded in the intuitive Euclidean distance metric. It remains robust even in the presence of missing values, considering only available data entries. Normalization isn't necessary, unlike with Pearson correlation, allowing direct use on raw data. It's sensitive to scale differences in data, potentially treating differently scaled vectors more appropriately.

Results

The recommendation system was able to successfully identify users similar to the user of interest and recommend movies to them based on the preferences of similar users. The recommendations were sorted based on the predicted rating, allowing users to view the most relevant movies at the top of the list.

```
Top 10 similar users for user 4 using Pearson Correlation are: {299: 1.0, 360: 1.0, 396: 1.0, 518: 1.0, 544: 1.0, 378: 0.9864400504156212, 44: 0.9827076298239907, 502: 0.9271726499455306, 539: 0.8910
421112136304, 521: 0.8874119674649424}

Predicted rating for user 4 and movie 23 : 3.5555555555555554

Top 10 recommended films for user  4 are:
Movie ID: 293 | Predicted Rating: 5.412698412698413
Movie ID: 1210 | Predicted Rating: 5.201388888888889
Movie ID: 1429 | Predicted Rating: 5.201388888888889
Movie ID: 1476 | Predicted Rating: 5.201388888888889
Movie ID: 737 | Predicted Rating: 5.201388888888889
Movie ID: 1620 | Predicted Rating: 5.195555555555556
Movie ID: 1608 | Predicted Rating: 5.195555555555556
Movie ID: 1616 | Predicted Rating: 5.195555555555556
Movie ID: 1687 | Predicted Rating: 5.195555555555556
Movie ID: 1754 | Predicted Rating: 5.195555555555556

Top 10 similar users for user 4 using Euclidean similarity are: {4.0: 1.0, 158.0: 0.6666666666666666, 396.0: 0.6666666666666666, 502.0: 0.5358983848622454, 44.0: 0.5, 87.0: 0.5, 151.0: 0.5, 214.0: 0.
5, 291.0: 0.5, 306.0: 0.5}
```

Conclusions

In conclusion, the recommendation system implemented using Pearson correlation proved effective in providing personalized movie recommendations to users. The Pearson correlation function calculates the correlation coefficient between the user of interest and all other users in the dataset. This metric measures the linear relationship between two variables, in this case, the movie ratings of different users. Scipy's built-in `pearsonr()` method facilitates the computation of Pearson correlation efficiently. By iterating over all users and computing their correlation with the user of interest, we obtain a similarity score that indicates how closely their movie preferences align. The recommendation function suggests movies to the user of interest based on their predicted ratings. It iterates over all movies in the dataset and identifies those that the user has not yet rated. For each unrated movie, the function calculates its predicted rating using the prediction function described above. The recommended movies are then sorted in descending order based on their predicted ratings, ensuring that the most relevant and potentially enjoyable movies appear at the top of the list. At the end, we calculate the top ten similar users using the Euclidean Similarity. However, further improvements could be made by exploring more advanced techniques and recommendation algorithms.