White Paper

Topic: Movie Recommendation Web Application

Business Problem

The field of entertainment has always been a significant part of human culture. Today, with the advent of digital streaming platforms, the amount of content available is overwhelming. The project aims to build a movie recommendation system, which presents a solution to the problem of information overload by providing users with suggestions that match their preferences. The goal is to increase user satisfaction and, consequently, enhance user engagement on the platform.

Background/History

In the era of digitization, recommender systems have become increasingly popular. These systems have been widely used in different online applications like e-commerce, video on demand, or music platforms. The primary purpose is to filter out the surplus of information and present users with the most relevant items.

Data Explanation (Data Prep/Data Dictionary/etc)

The datasets used for this project are the Movies and Ratings datasets. These were preprocessed to merge them on the common 'movieId' column, which resulted in a unified dataframe that includes details about the movie and user ratings. For practicality and performance reasons, we considered only the top 1000 most rated movies. The resulting data was then transformed into a movie-user matrix, which forms the basis of our recommendation system.

Movies Dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 58098 entries, 0 to 58097
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   movieId  58098 non-null  int64
 1   title    58098 non-null  object
 2   genres   58098 non-null  object
dtypes: int64(1), object(2)
memory usage: 1.3+ MB
```

Ratings Dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27753444 entries, 0 to 27753443
Data columns (total 4 columns):
 #   Column     Dtype
---  ------     -----
 0   userId     int64
 1   movieId    int64
 2   rating     float64
 3   timestamp  int64
dtypes: float64(1), int64(3)
memory usage: 847.0 MB
```

Final Dataset:

```
# Filter to include only top 1000 most rated movies
top_movies = ratings_df.movieId.value_counts().index[:1000]
df = df[df.movieId.isin(top_movies)]
df
```

	userId	movieId	rating	timestamp	title	genres
0	1	307	3.5	1256677221	Three Colors: Blue (Trois couleurs: Bleu) (1993)	Drama
1	6	307	4.0	832059248	Three Colors: Blue (Trois couleurs: Bleu) (1993)	Drama
2	56	307	4.0	1383625728	Three Colors: Blue (Trois couleurs: Bleu) (1993)	Drama
3	71	307	5.0	1257795414	Three Colors: Blue (Trois couleurs: Bleu) (1993)	Drama
4	84	307	3.0	999055519	Three Colors: Blue (Trois couleurs: Bleu) (1993)	Drama
...	...	...	...	...	...	...
24976553	283101	2366	2.5	1092337860	King Kong (1933)	Action|Adventure|Fantasy|Horror
24976554	283116	2366	4.0	1000820168	King Kong (1933)	Action|Adventure|Fantasy|Horror
24976555	283153	2366	4.0	1047143351	King Kong (1933)	Action|Adventure|Fantasy|Horror
24976556	283187	2366	3.5	1397341537	King Kong (1933)	Action|Adventure|Fantasy|Horror
24976557	283195	2366	3.5	1079282013	King Kong (1933)	Action|Adventure|Fantasy|Horror

Methods

The main technique used in this project is item-based Collaborative Filtering with K-Nearest Neighbors (KNN) algorithm. This approach focuses on finding items that are similar to those the user has rated highly in the past and recommends items that are most similar. The 'cosine' distance metric is used to compute the similarity between items.

Analysis

The analysis performed includes training the model with the movie-user matrix data and using the model to find the nearest neighbors for a given movie. The results can be interpreted as a list of movies that are most similar to the input movie based on the user ratings. This approach, while straightforward, has proven effective in making recommendations that are relevant to user preferences.

Methods

The main technique used in this project is item-based Collaborative Filtering with K-Nearest Neighbors (KNN) algorithm. This approach focuses on finding items that are similar to those the user has rated highly in the past and recommends items that are most similar. The 'cosine' distance metric is used to compute the similarity between items.

Conclusion

The movie recommendation system developed in this project has the potential to significantly improve user experience on a movie streaming platform by providing relevant recommendations. It is a step towards making the movie selection process more efficient and personalized.

Assumptions

The primary assumption in this project is that the dataset used is a good representation of the general user base and their preferences. Another assumption is that the ratings given by a user provide an accurate representation of their preferences.

Limitations

The main limitation of the system is that it only considers user ratings for making recommendations. Additional data, such as user demographics or movie genres, could help in making the recommendations more personalized. Also, it is prone to the cold-start problem, where it can't provide recommendations for new movies or users.

Challenges

The challenges encountered during this project primarily relate to data preprocessing and model tuning. Constructing the movie-user matrix was a significant step that required careful handling of the data. Also, choosing the right parameters for the KNN algorithm was crucial in achieving accurate recommendations.

Future Uses/Additional Applications

Beyond movie recommendations, the system developed in this project could be adapted for recommending other types of items such as books, music, or products in an e-commerce setting.

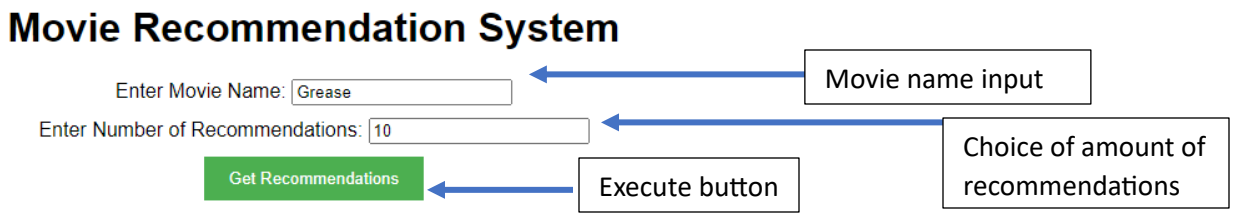
Recommendations

One possible improvement to the system could be the inclusion of additional data in the recommendation process. For example, considering the genre of the movie or the demographic information of the user might improve the quality of recommendations.

Implementation Plan

The recommendation system can be implemented in a production environment by integrating it with the existing infrastructure of a movie streaming platform. It would require setting up an appropriate data pipeline to feed the system with up-to-date user ratings data, and a user interface for displaying the recommendations.

Input page:



Output page:

Ethical Assessment

As with any data-driven application, there are some ethical considerations to take into account. First, user data should be handled responsibly, with all necessary measures taken to ensure its privacy and security. Secondly, it's important to ensure that the recommendation algorithm doesn't create a "filter bubble", where users are only shown content similar to what they've seen or liked before, potentially limiting their exposure to diverse content. Regular assessments and adjustments can help mitigate these concerns.

The system should be transparent about how recommendations are made, and users should have the ability to adjust or opt-out of personalized recommendations if they wish. It's also important to consider potential biases in the user ratings data, as these can inadvertently influence the recommendations provided by the system.