THE MOVIE BUFF-REPORT

ABSTRACT:

Movie Buff is a movie recommendation system made on python. Here movies are recommended based on:

1. Similar movie based on a particular movie.
2. Based on movie popularity and/or genre.

DATASET USED:

The Movies Dataset: *Metadata on over 45,000 movies. 26 million ratings from over 270,000 users.*

Used

- **credits.csv**
- **movies_metadata.csv**
- **keywords.csv**

ALGORITHM USED:

1. **TF-IDF**

TF = (Number of times term t appears in a document)/(Number of terms in the document).

IDF = 1+log(N/n), where, N is the number of documents and n is the number of documents a term t has appeared in.

2. **Cosine Similarity**

Cosine Similarity (d1, d2) = Dot product(d1, d2) / ||d1|| * ||d2||

STEPS:

1. Importing necessary libraries.

2. Reading the dataset(CSV) using pandas.

3. Merging the data frames into 1 based on ‘id’.

4. As the code runs user is given two options

4.1 Similar movies based on a particular movie.

4.2 Based on movie popularity and/or genre.

5. Calculate the similarity scores for the overview of each movie entry in the dataset.
6. Compute Term Frequency-Inverse Document Frequency (TF-IDF) vectors for each overview.
7. The data set was trimmed in order to find the cosine-similarity score.
8. Create a *get_recommendation* function which accepts a movie title from the user and recommends similar movies to the user based on the title by taking the cosine similarity scores of the most similar movies. This function will recommend the 10 most similar movies.
9. Wrote some functions to clean and extract important features like directors, genres, and keywords from the data.
10. Created a metadata string function which will help to concat all the important features.
11. Used CountVectorizer instead of tf-idf because tf-idf only gives unique values.
12. If the user chooses 4.2

12.1 We need a metric to score or rate movie.

12.2 Calculate the score for every movie using equation:

Weighted Rating(WR)=(((v/(v+m))*R)+((m/(v+m))*C)))

- v is the number of votes for the movie;
- m is the minimum votes required to be listed in the chart;
- R is the average rating of the movie; And
- C is the mean vote across the whole report.

12.3 Sort the scores and recommend the best-rated movie to the users.

DOMAINS, TOOLS, AND LIBRARIES USED:

Domain: Machine Learning, NLP,
Tools: Kaggle Kernel(Earlier colab was used but since kaggle gave more resource shifted to it).
Libraries: Pandas, Numpy, Scikit learn.

HOW TO RUN:

In case of using the kaggle kernel:
recommender1.ipynb, recommender2.ipynb
Here data set can be loaded as default. Both types of recommendations is done as a separate interactive python notebook.

In case of running in CPU(Enough computational power needed)(Recommended):
recommender.py
Here data set is inside /dataset folder. Also cannot find the dataset here because of the high size, please download and extract inside the folder.

DIFFICULTIES FACED:

Struggled with colab and jupyter-notebook(anaconda) until got the idea about kaggle kernel :-)

REFERENCE:

I referred to several notebooks and scripts associated with the recommendation system. Also, StackOverflow visiting became a routine.

CONCLUSION:

This recommendation system works fine. And also can enhance it's features in the course of a long time. For the time being it works really cool.