Movie Recommendations System

Problem Description:

Recommending movies based on the personal preferences of the user.

Steps Included:

1. Loading Datasets.
2. Cleaning/Preprocessing the Datasets.

Basically Two types of recommender system:-

1. Demographic Filtering
2. Content Based Filtering

Loading Datasets:

In this we have worked on TMDB\_5000 dataset. Which consists of 2 files one is tmdb\_5000\_movies.csv and other one is tmdb\_5000\_credits.csv

Demographic Filtering:

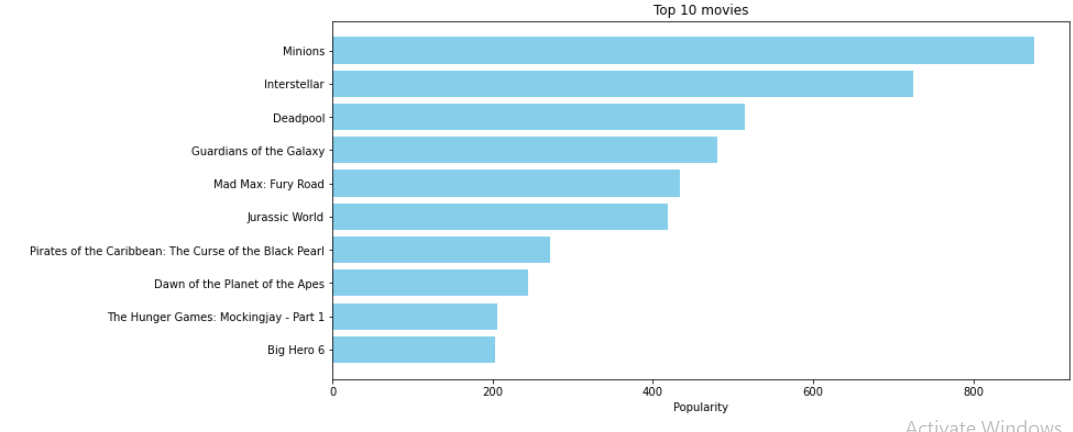
Here in this filtering we generalized recommendations to every user, based on the movie popularity and genre .The system recommends the same movies to users with similar demographic features.

1. We need a metric to rank the movies.
2. Vote\_average itself is not enough to score fairly. Since we cannot rely on vote averages based on 2-3 votes . So vote\_average a more robust metric. We can use IMDB’s metric for that. It is as follows:
3. Weighted Rating(WR)=(v/(v+m)\*R)+(m/(v+m)\*C)
4. Where,
5. V is the number of votes for the movies;
6. M is the minimum votes required to be listed in the chart.
7. R is the average rating of the movie; And
8. C is the mean vote across the whole report

After doing above things we will sort the movies based on the score and we display them.

If we want we can plot a graph.

Following is the graph for top 10 movies:



Content Based Filtering:

In this filtering we will find similar movies according to their overview, cast, crew, keyword, tagline etc.

* Here we compute Term Frequency-Inverse Document Frequency (TF-IDF) vectors for each overview.
* Here term frequency means , it is relative frequency of a word in a document.
* Inverse Document Frequency is the relative count of documents containing the term
* So using a matrix we will complete the task. Here in this matrix where each row represents a movie . By using this matrix we will reduce the words that occur frequently .
* We used bulit-in TfIdVectorizer class which is provided in scikit-learn to compute the TF\_TDF matrix
* By using that matrix we will find similarities between word matrix and cosine
* And we will Construct a reverse map of indices and movie tites.
* And we will use that map to access the movies.
* And we will display the sorted movies which are accessed or located using the map.

By using above mentioned filtering we build movie recommendation system which recommends the movies based on user preferences.

**SONG RECOMMENDATION USING MACHINE LAEARNING:**

**Probem statement:** The music recommendation system is popular ml project asias leading music streaming services provide the data said to build a better recommendation system.

There is use of **k-means clustering** algorithm

**ALGORITHM:**

K-Means Clustering is an [Unsupervised Learning algorithm](https://www.javatpoint.com/unsupervised-machine-learning), which groups the unlabeled dataset into different clusters.

**STEP-1**: Select the number K to decide the number of clusters.

**Step-2:** Select random K points or centroids. (It can be other from the input dataset).

**Step-3:** Assign each data point to their closest centroid, which will form the predefined K clusters

**Step-4:** Calculate the variance and place a new centroid of each cluster.

**Step-5:** Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

**Step-6:** If any reassignment occurs, then go to step-4 else go to FINISH.

**Steps involved in recommended system:**

1. Loading data sets
2. Analyzing and data preprocessing
3. Predicted the recommended songs

Loading datasets:

* Download the songs related dataset

(triplet\_csv) ,which consist the columns user id ,song\_id and song listening count and data set (songs\_data) consist song related information like release date and artist name and title

* Import pandas and read the csv files pd.read\_csv() function
* Then merge two data sets using merge function

Analyzing the data :

Music recommendation systems are mainly of three types. Content-based systems utilises past user choice to predict songs and recommend them. Collaborative systems predict songs based on what other similar users have previously listened to.

Here we are choosing collaborative system based recommend the song based on song listening count

Collaborative filtering is **a technique that can filter out items that a user might like on the basis of reactions by similar users**

* At first we will add a new column named “song” refereed to artist and song name.
* For easy maintenance we take only 10k rows out of all the data in the datasets.
* And we will add a new column named percentage which was calculated using the sum of listen\_count.
* We will create a popularity\_recommender object “pr” using Recommenders module.
* We will create popularity\_recommender for the dataset that we have included.

Conclusion:

* By using recommend method we will be able to display/recommend the songs where song with high score will be displayed first.
* We will also get similar songs for a given song using Recommenders module.