**HYBRID MOVIE RECOMMENDATION SYSTEMS**

**-**

**–**

**Project Overview:**

**Movie recommendation is done using two methods:**

1. Recommend movies based on the user profile i.e movies similar to which user has liked/highly rated.
   1. This is done using k-Nearest Neighbors and kmeans clustering.
2. Movies can also be recommended by predicting the rating the users would give to certain movie and recommend based on that.
   1. This is done using two techniques:
      1. Multi Layer perceptron
      2. Neural Matrix Factorization(is better compared using Root mean square error)

**Project Implementation Details:**

**Type 1 Recommendation:**

**Input:** Around 26 user profiles has been created with details userid, movieid, ratings. userId 1-25 are normal users who has watched and rated few movies. userId = 100 is new user.

**Identify movies which user likes from input:** We have written logic to find top few movies which the user has liked and rated the most. We will use those identified movies as input to our model KNN and Kmeans. If it is new user then highly rated movies from popular genre will be recommended.

**K-Nearest Neighbor:**

This model uses **Euclidean distance as measure,** and **algorithm uses** **brute.** Because the above performs better when compared with measures and algorithms like cosine, kd-trees.

Fuzzy matching is used to find the exact movie name and id even if it is found or typed wrongly. And it also used in kMeans to identify the movie name.

Model is fitted with the training data and nearest neighbors movies are found with closest distance to the test data. And this identified 10 closest movies are recommended.

**K-Means Clustering:**

Kmeans++ is initialized to cluster the movies perfectly by initializing it with the model.

Elbow method is used to identify the right number of clusters required to cluster movies perfectly. The code is commented below in MovieRecommender.py. k=6 or 7 as screen shot shown below.

A picture containing screenshot

Description automatically generated

Model is trained and clusters are formed for the training data. Movies are recommended by our logic i.e the movie cluster is identified first and then the closest 10 movies to the user input movie is recommended.

**Normal User** output:

C:\Users\rikky\AppData\Local\Programs\Python\Python37\python.exe C:/Users/rikky/PycharmProjects/PracticeHR/dataanalysis/KNNitembasedCF.py

User input movie: Jaws 2 (1978)

1257

Finding the most similar movies through KNN

......

**Recommendations similar to Jaws 2 (1978):**

1: Jaws 3-D (1983), with distance of 165.71058654785156

2: Jaws: The Revenge (1987), with distance of 177.74911499023438

3: King Kong Lives (1986), with distance of 182.54930114746094

4: Amityville II: The Possession (1982), with distance of 182.93031311035156

5: Exorcist II: The Heretic (1977), with distance of 183.03688049316406

6: Poltergeist III (1988), with distance of 183.20684814453125

7: Amityville 3-D (1983), with distance of 183.36166381835938

8: Prom Night II (1987), with distance of 183.37939453125

9: Graveyard Shift (Stephen King's Graveyard Shift) (1990), with distance of 183.43731689453125

10: Howling II: Your Sister Is a Werewolf (1985), with distance of 183.4570770263672

Recommendations of similar movies through K-means:

**Recommendations similar to Kicked in the Head (1997):**

['Fistful of Dollars, A (Per un pugno di dollari) (1964)', 'RoboCop 2 (1990)', 'The Glass Shield (1994)', 'Fast Food, Fast Women (2000)', "Mummy's Ghost, The (1944)", 'Firefox (1982)', "Mo' Better Blues (1990)", 'Heartbreak Kid, The (1972)', "Big Momma's House (2000)", 'Juice (1992)', 'Dead Again (1991)']

Recommendations of similar movies through K-means:

**Recommendations similar to Simple Plan, A (1998):**

["2 ou 3 choses que je sais d'elle (2 or 3 Things I Know About Her) (1967)", 'Get Over It (1996)', 'Open Season (1996)', 'Looking for Mr. Goodbar (1977)']

--- 219.54859924316406 seconds ---

**New User** Output:

C:\Users\rikky\AppData\Local\Programs\Python\Python37\python.exe C:/Users/rikky/PycharmProjects/PracticeHR/dataanalysis/KNNitembasedCF.py

**Highly recommended movies for Action genre:**

Inception

The Dark Knight

Fight Club

The Lord of the Rings: The Return of the King

Star Wars

The Empire Strikes Back

Leon: The Professional

Scarface

The Good, the Bad and the Ugly

**Highly recommended movies for Comedy genre:**

Pulp Fiction

Forrest Gump

Back to the Future

The Intouchables

The Grand Budapest Hotel

Life Is Beautiful

Dilwale Dulhania Le Jayenge

Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb

Modern Times

**Highly recommended movies for Thriller genre:**

Inception

Fight Club

Pulp Fiction

Se7en

The Imitation Game

The Silence of the Lambs

The Prestige

Leon: The Professional

Memento

--- 217.49652194976807 seconds ---

**Type 2 Recommendation:**

**Train/Test data: 80% Train data, 20% Test data**

**Input:** Users profiles i.e userId,MovieId,ratings will be given as the input data.

Predict: How user will **rate** a particular movie?

**Models:**

**Common:** Data dimensionality is reduced at the beginning.

1. Multi Layer Perceptron:
   1. Model is created using keras tensorflow frameworks. We have used distinguished layers, we train the model based on training data EPOCH =15.
   2. We will predict the test data rating by loading the trained MLP model, and we have used Root square mean error measure to evaluate the model.
2. Neural Matrix Model:
   1. The model is trained with the training data for EPOCH configured as 15, also callback early stoping is used.
   2. Then we will predict the test data rating by loading the trained Neural Matrix Model, and we have used Root square mean error measure to evaluate the model.

**Evaluating models:**

Root square mean error outputs:

MLP - The out-of-sample RMSE of rating predictions using MLP is 1.2184253

Neural Matrix Factorization- The out-of-sample RMSE of rating predictions is 1.1920696

**Lower the value of RMSE the model is better.**

**Based on our testing NEURAL MATRIX FACTORIZATION MODEL works better.**

**MLP Training ScreenShot:**

**A screenshot of a social media post

Description automatically generated**

**MLP Output ScreenShot:**

**A screenshot of a social media post

Description automatically generated**

**Neural Matrix Factorization Training:**

**A screenshot of a social media post

Description automatically generated**

**Neural Matrix Factorization Output:**

**A screenshot of a social media post

Description automatically generated**

References:

<https://machinelearningmastery.com/start-here/#deeplearning>

https://machinelearningmastery.com/start-here/#python

https://towardsdatascience.com/unsupervised-classification-project-building-a-movie-recommender-with-clustering-analysis-and-4bab0738efe6

<https://towardsdatascience.com/how-did-we-build-book-recommender-systems-in-an-hour-part-2-k-nearest-neighbors-and-matrix-c04b3c2ef55c>

<https://towardsdatascience.com/creating-a-hybrid-content-collaborative-movie-recommender-using-deep-learning-cc8b431618af>