RECOMMENDATION SYSTEM USING COLLABORATIVE FILTERING AND K-MEANS

(Movie Recommendation System)

A PROJECT REPORT

Submitted by

UDHAYAKUMAR R

18BCB0158

CSE4020 – MACHINE LEARNING

SLOT-F1

Under The Guidance Of

Prof. BALMURUGAN R

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

MAY 2021

TABLE OF CONTENETS

LIST OF FIGURES:

1. ABSTRACT:

Nowadays the Recommendation System was very useful to predict the good products in online shopping and suggest the best or top rated movies to users in online OTT platforms and recommend the good stories or articles in Google and book recommendations in the kindle. So in many places the recommendation system plays a major role and many of the top companies use this recommendation and recommend good and suitable products for our customers.

In this paper we have created a Movie Recommendation system using Collaborative filtering and K-means clustering algorithm. Basically the recommendation system was two types one is content-based-filtering another one is collaborative-based-filtering. The recommendation was information filtering and information retrieval systems.

Content-based-filtering methods have user profiles because the system understands what type of movies user's watch and what type of movies do not like the particular users so gather that information after the system has recommended the suitable movies. Collaborative filtering method does not take a single user's profile.

The collaborative filtering takes a lot of users' ratings and profiles and analyse the data and makes a group of similar users and finally recommends the movies from every user not only single user and also the top rated movies to new users. so the collaborative filtering had a large size of data otherwise this method was not suitable.

Our project we have implemented the Collaborative filtering method and separated the data into groups using the K-means clustering algorithm. We have taken the movie id from and also match with ratings table and that movie id match with ratings movies so that movies was selected along with movie’s genres and ratings. Also we have taken only for best ratings movies only our project we have taken of above 2.5 and 3 ratings only.

We apply the k-means clustering algorithm and find that optimal k and display the cluster and use the collaborative filtering approach for recommendation part we apply that and recommend the movies for new users and also recommend the particular users.

KEYWORDS:- Recommendation Systems, Collaborative Filtering, K-means clustering.

2. INTRODUCTION

The Movie Recommendation System using collaborative filtering and KMeans clustering method. In this project first we have to use a movie-lens dataset. The movie-lens dataset contains 5 csv files, we choose only Movies.csv and Rating.csv files the movies files contains 3 columns movie-id, title and genres. The rating file contains 4 columns user-id, movie-id, rating and timestamp. So first we have created a new data-frame using movies and rating files. Select a genres from the movies dataset and movie-id that particular movie-id compare with ratings dataset movie-id both movie-id is same select that particular movie user-id and genre based ratings of movies and create a one data-frame. Our project we have selected three genres like Romance, science-fiction and action related movies and we have selected the best rating movies because that only we recommend the good movies from the customers.

After we have used k means clustering method to make the group of users based on our genre based ratings or similarity. Each cluster has the same type of taste so we have easily recommended the movies. Before visualizing the cluster first we choose the optimal number of k in k means clustering algorithm. In the k-means clustering many ways to find the k value our project we have use two methods one is Elbow method another one silhouette coefficient method. After finding the optimal k we have visualized the cluster along with the centroids. We have evaluated the clusters using various methods.

Our project we have using the DB-Index method two evaluate the clusters. We have created a data-frame that contains a merging of ratings and movies dataset and we have selected features like movie names along with movies id and ratings and user id. The new dataset contains a lot of null values. Every user must not be rated for everyone. It was optional one and one more reason we have not dropped the data because the collaborative filtering method needs a lot of data so we haven't dropped the rows.

So we have found which users have rated the most ratings and also find the most rated movies in the dataset. We have retrieved the most rated movies from the user ratings data-frame and after getting the most rated movies dataset we have converted that dataset into a sparse matrix. Basically the sparse matrix contains a lot of null values so we have used some preprocessing techniques to reduce the half of missing values after applying the sparse matrix to k-means algorithm to finally recommend the movies to particular users.

3. LITERATURE REVIEW:

NO	TOPIC	AUTHOR	METHODOLOGY	PROS	CONS
1	Movie Recommendation system using KMeans clustering and K-Nearest Neighbour	Rishabh Aujha, Anand Nayyar, Arun solanki	In this paper the movie recommendation system was built using KMeans and KNN. First the dataset was split into two one is train another one is test data.convert the train data was utility matrix and using wcss find optimum K and find the clustered matrix and using KNN to find similarity users and finally recommend the movies.	This proposed system contains both supervised and unsupervised techniques. KMeans use for clustering and KNN use for recommending movies. The utility matrix represented which user rated which movie showed it was useful for clustering.Finally the evaluation was made by RMSE method. This method was overcome by the Existing Cuckoo search method.	The Main Drawbacks is only 1 lakhs data was used for this project and prediction or recommendation part the KNN was very slow or took more time for recommendation the movies.

2	Movie Recommendation Using Collaborative and Content Based Filtering	Saurabh bhalla, Baibhav Kumar, Rajat Tiwari, Dr.P.A Jadhav	In this Paper Build a Movie Recommendation using Collaborative filtering and Content Based filtering. K-means was used for Collaborative Filtering and KNN is used for Content Based Filtering. in the collaborative method two types are used, one item based and another one is user based and many similarity measures were used to recommend the movies to users. The data contains movies id, ratings and users details.	In this recommendation system one webapp using django and bootstrap was very close or interactive to the users.	This paper doesn't detaily tell the dataset information and how to find the optimal k values. These types of details are not there in this paper.
3	Movie Recommendation System using Machine Learning Algorithms	Akansh Surender, Aditya Kumar Yadav, Aditya kumar	In this paper the movie recommendation system was building using collaborative and content based filtering algorithms was used and some similarity measures was used for prediction part of the this project	The main advantage of this project is the web interface is created it was easily interact with the users and users choose the best movies in short time	In this paper the dataset and algorithm definition is not clear and collaborative filtering based recommendation system need a large dataset

4	A User Rating Based Collaborative Filtering Approach to Predict Movie Preferences	Md. Asif Shahjalal, Zubair Ahmad, Mohammad Shamsul Arefin, Mohammad Rubaiyat Tanvir Hossain	In this Paper Build a Recommendation system using Collaborative Filtering Method. We have built a collaborative method to solve the 3 problems: cold start and optimization and reduce the cost function. This efficiently solves 3 problems and maximizes the implementation in matlab because this proposed system uses more mathematical methods.	The project implementation of doing in the matlab was the best and fastest tool for matrix related works so it was efficient work.	The dataset contains 1 lakhs rating so we can easily reduce the cold start using normalization methods but the data is large and it was not recommended for good movies to users.
5	Movie Recommendation System using Collaborative filtering and K Means	Phongsavanh phorasim, Lashing Yu	movie recommendation system using collaborative filtering and k-means algorithms. The collaborative filtering method have large amount of data and no outliers of data so using k means separate users in to the group after recommending the movies of this paper.	In this paper separating the users into clusters using WEKA software and also implementing the web interface of this recommendation part.	In this method only small amount users and movies are selected and finding the ratings and recommend to the users

4. PROBLEM FORMULATION

The main problem of the recommendation system we have need a large amount of data but in case the real time the large size of data have more null values or missing values contains or unnecessary values are there because the humans don't give the correct data for every time and every user not given the ratings or review of movies or products because it was optional we don't force for it.

In our method we have been using collaborative filtering and k-means clustering algorithm was used to build a movie recommendation system. The k-means algorithms don't work with large sizes of data and the data has no null values or any missing values the dataset contains any outliers. The k-means algorithm did not work properly. One of the big issues of the k-means clustering algorithm is that we have initially assigned the k-values.

Collaborative filtering methods have some limitations such as cold start is one of the issues in collaborative filtering. We recommend movies for particular users. We need some data for that user. It was called cold start. The number of users grows the algorithm suffers scalability issues.

5. OBJECTIVE

We have implemented the movie recommendation system using collaborative filtering and k means clustering algorithm. Basically the recommendation system has a large amount of data otherwise we don't need to recommend the good movies or not suitable movie for particular users.

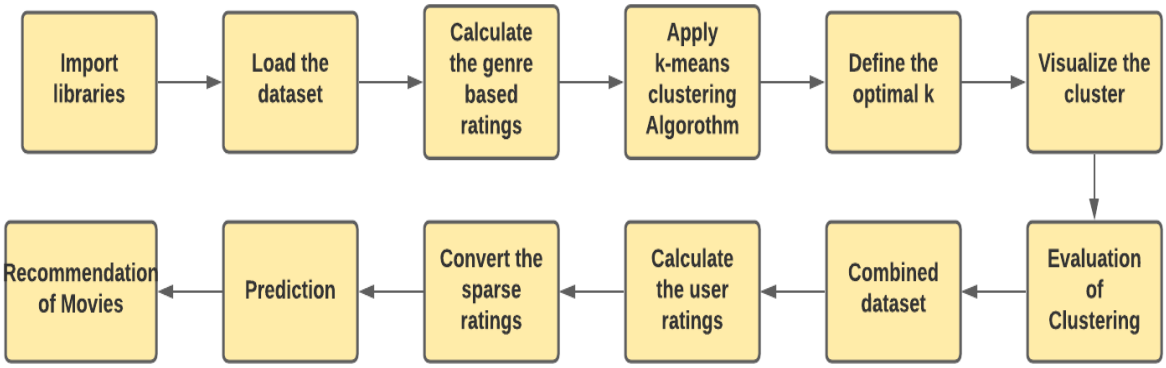
In the existing recommendation system many of doing the content-based system or doing the collaborative system use the small size of data or using K-Nearest neighbor algorithm was used. In our project we have implemented a recommendation system using collaborative filtering and k-means clustering.

Our project we have directly taken the ratings to use the recommendations part rather than calculating the genre based ratings after we have applied the k-means algorithm to make clusters. In this project we have taken only 3 genres like Romance, science fiction and action related movie ratings. So we have recommended the best movies and suitable genres based movies for users. We have shown the most rated movies and recommend the movies from particular users.

6. METHODOLGY

6.1. PROCESS DIAGRAM

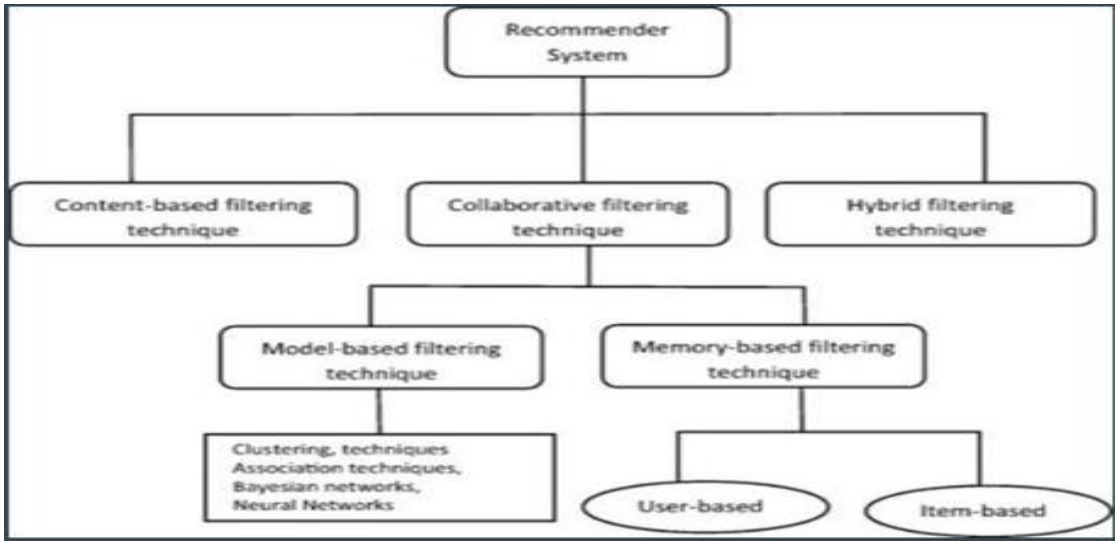
Fig-[1] : Project Process Diagram



6.2. Recommendation System

The Recommendation system is basically information retrieval system. Now a days it was used in many online platforms. In our project we have do the Movie Recommendation system using Collaborative filtering using K-means. The recommendation system contains three types one is content-based, collaborative filtering based and last one is hybrid filtering based recommendations systems. Our project we have do the collaborative based filtering system. Collaborative filtering based system also contains two types one is memory based and another one is model based system. The memory based system also separate into two parts one is user-user and another one is item-item filtering methods. Our project we have do model based filtering method using clustering approach algorithms for using grouping the data.

Fig-[2]: Overview of Recommendation system



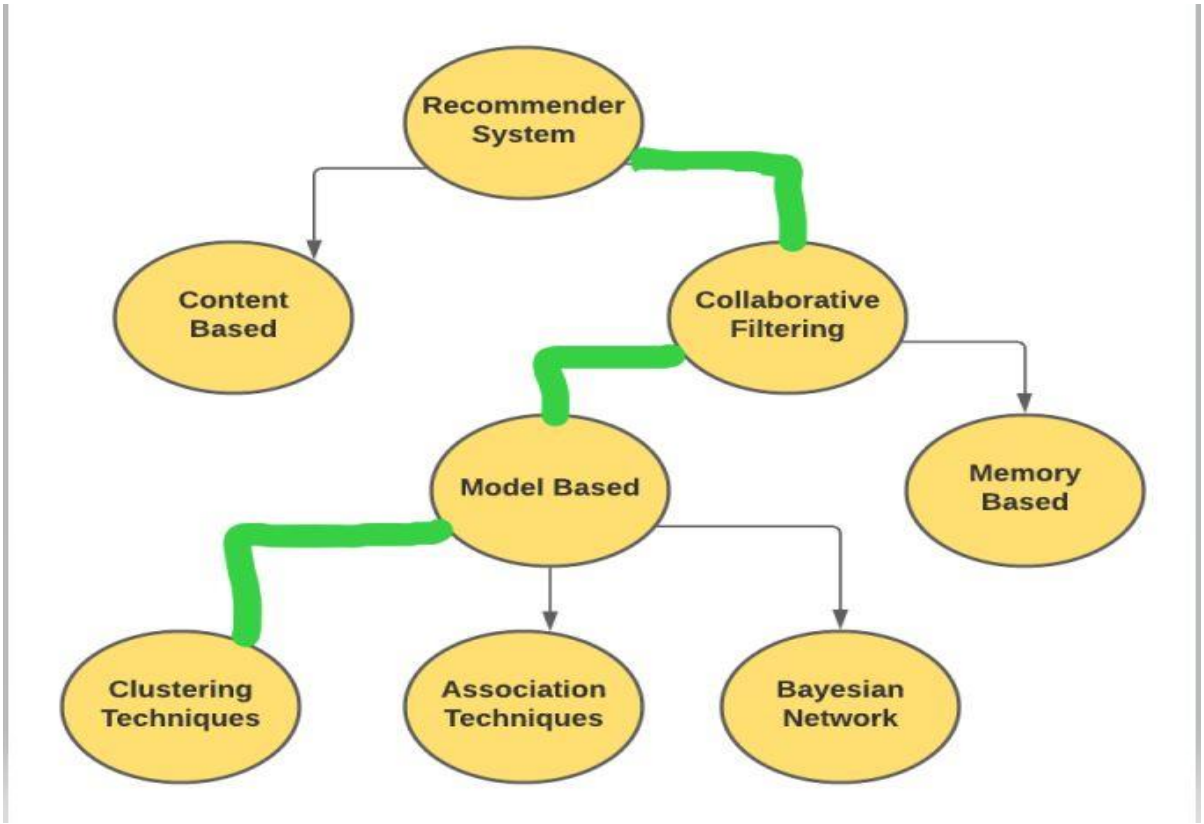
6.3. Collaborative Filtering

Recommendation system used two types of filtering methods one is collaborative filtering and another one is content based filtering. Our project we have using collaborative filtering methods. Collaborative filtering was used to predict the user preferences by using similar user preferences and find the unrated movies ratings.

Collaborative filtering contains two methods, one is memory based or nearest neighbour system, second one is model based system in the model based system using some Bayesian network and clustering algorithms was used. Our project we have using model based systems and clustering algorithms.

In the collaborative filtering methods, a recommendation system has a large size of dataset, otherwise the recommendation system was not good at predicting the user preferences or ratings. Our project we have collaborative filtering methods and use the model based systems and in the model-based we have using the clustering algorithm. The clustering algorithms we use the k-means clustering algorithm used.

Fig-[3]: Flow of Our Project

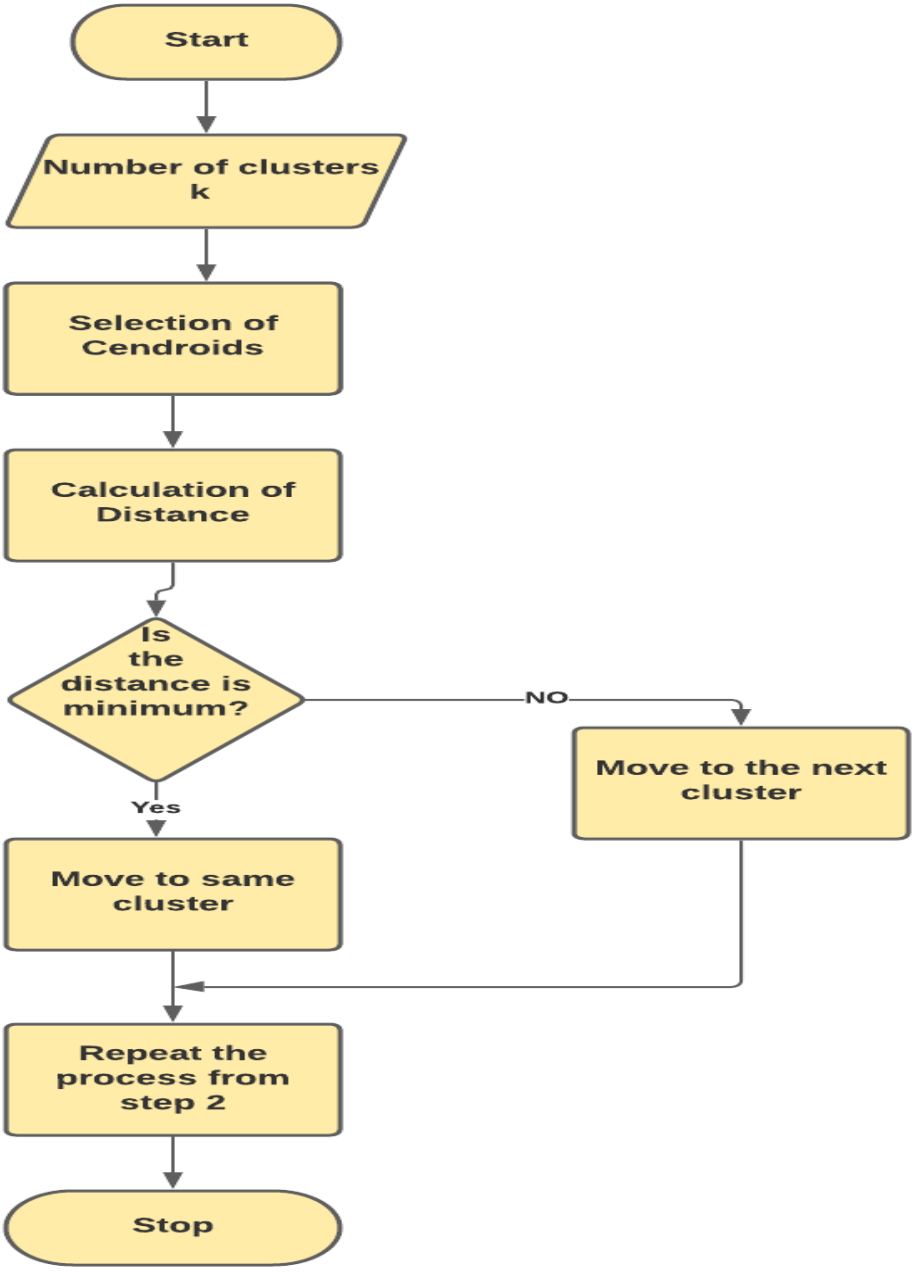


6.4. K-MEANS ALGORITHM

K-means clustering is one of simplest and most popular unsupervised learning. It was useful for grouping or clustering the data. K-means clustering algorithm was used for many recommendation systems and many real world application the algorithm works iteratively to assign each data point to one of k groups based on features of the data.

Process of K-means

Fig-[4] K-means Algorithm Process



6.5. Calculate the Genre Based Ratings

We have use two datasets in our project one is movie dataset and another one ratings dataset the movie dataset contains Movies Id, user Id and genres. Ratings dataset contains ratings, Movies Id, user id and timestamp. So we have select the genres along with Movie Id from movies dataset and select the ratings and user Id for matching movies id from movies dataset. Our project we have select only for 3 genres like that Romance, action and science-fiction movies.

We have built a best recommendation system so we have taken only ratings values for greater than 3.First we have Calculate the Average ratings for Romance and Science Fiction Movies and add the action movies.

Fig-[5]: Genre Based Ratings

	userId	Avg_romance_rating	Avg_scifi_rating
0	4	3.50	3.00
1	8	3.95	3.18
2	9	4.00	2.67
3	15	3.33	3.00
4	18	3.88	2.45

Fig-[6]: Ratings for all 3 Genres

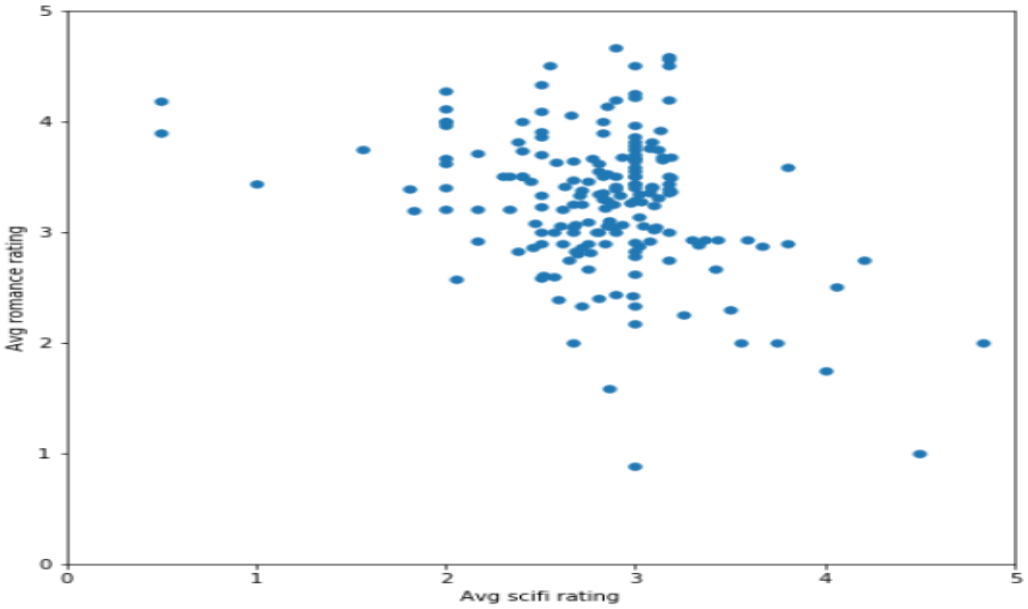
	userId	Avg_romance_rating	Avg_scifi_rating	Avg_action_rating
0	4	3.50	3.00	3.54
1	8	3.95	3.18	3.63
2	9	4.00	2.67	3.20
3	15	3.33	3.00	2.73
4	18	3.88	2.45	2.54

6.6. Visualize the Cluster

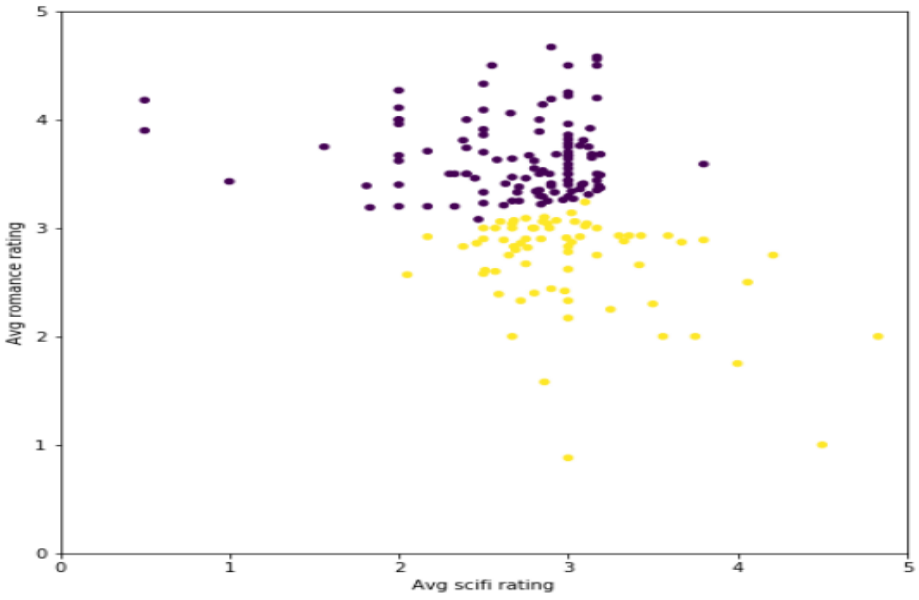
Visualize the Cluster for Before Find the Optimal K value

Average Ratings for Romance and Science Fiction Movies

K=1 Fig-[7]: Initial Cluster



K=2 Fig-[8]: Cluster Visualization of K=2



Separate the groups by ratings. Greater than 3 is one group less than 3 is one group

6.7. ELBOW METHOD

The Fundamental steps of every unsupervised learning especially clustering or grouping related algorithms we have find the optimal k values. Many methods are there to find the optimal k values like that elbow method and silhouette methods.

Elbow Method

The Elbow method is one of the most popular methods to find the optimal number of k values. Compute clustering algorithm (e.g. K-means clustering) for different values of k. for instance by varying k from 1 to 10 clusters.

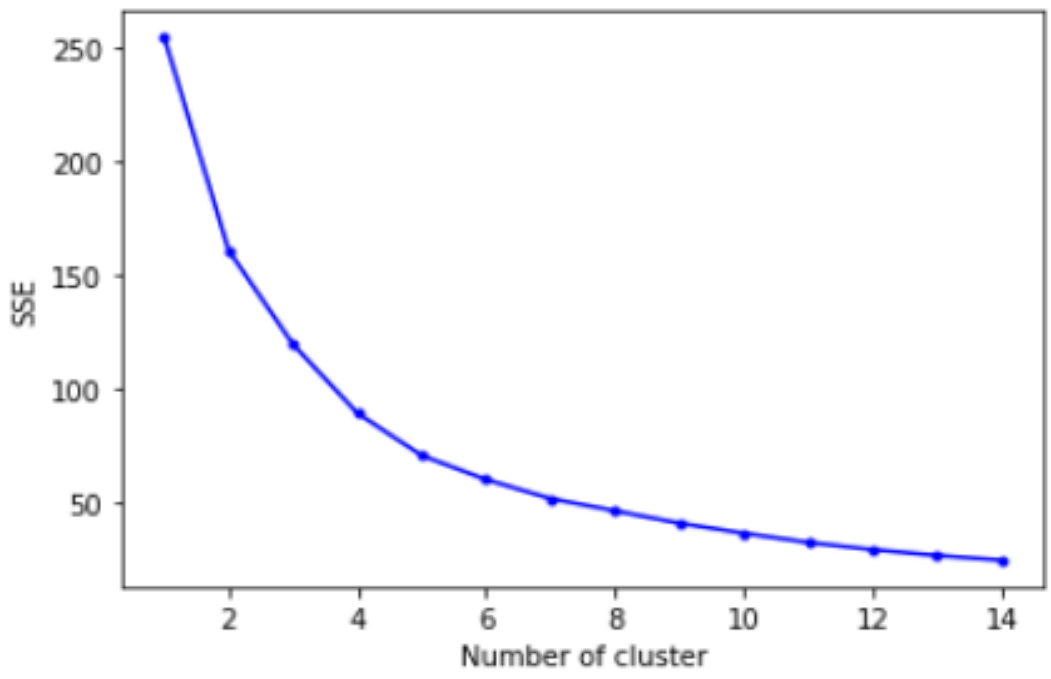
For each k calculate the total within-cluster sum of square (wcss) values.

Plot the curve of wcss according to the number of clusters k.

The location of bend in the plot is generally considered as an indicator of the appropriate number of clusters.

Our Project Optimal K Value is 5 (K=5)

Fig-[9]: Elbow Method



6.8. Silhouette Coefficient Method

The class labels are not found in the dataset the evaluation process must do by the model itself. The silhouette coefficient method was used for this type of evaluation process. The higher silhouette coefficient score is related to models with better defined clusters. This method was each sample and composed two values.

A - The mean distance between samples and all other points in the same cluster class.

B - The mean distance between samples and all other points in the next nearest cluster class.

S- Silhouette coefficient for single samples

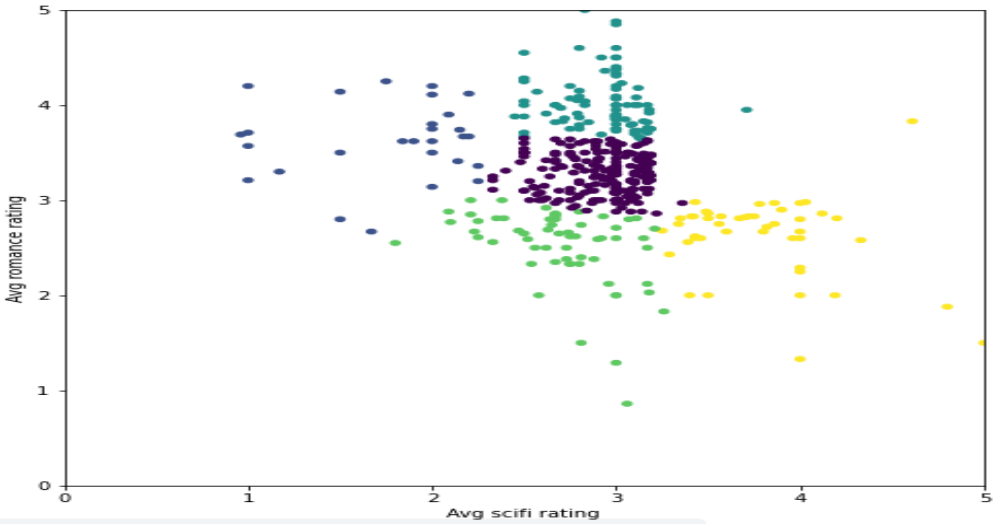
S = B-A/ MAX (A, B)

Fig-[10]: Silhouette Method

```
For n_clusters=2, The Silhouette Coefficient is 0.34447713391038837
For n_clusters=3, The Silhouette Coefficient is 0.3472401702342571
For n_clusters=4, The Silhouette Coefficient is 0.3762261237073798
For n_clusters=5, The Silhouette Coefficient is 0.398840703139085
For n_clusters=6, The Silhouette Coefficient is 0.37695572665280713
For n_clusters=7, The Silhouette Coefficient is 0.34568731821625476
For n_clusters=8, The Silhouette Coefficient is 0.3563601807284739
For n_clusters=9, The Silhouette Coefficient is 0.3563187356226249
For n_clusters=10, The Silhouette Coefficient is 0.3558750029904473
For n_clusters=11, The Silhouette Coefficient is 0.3576052149153519
For n_clusters=12, The Silhouette Coefficient is 0.36197235605979355
For n_clusters=13, The Silhouette Coefficient is 0.3765652128040539
For n_clusters=14, The Silhouette Coefficient is 0.3827259457055984
```

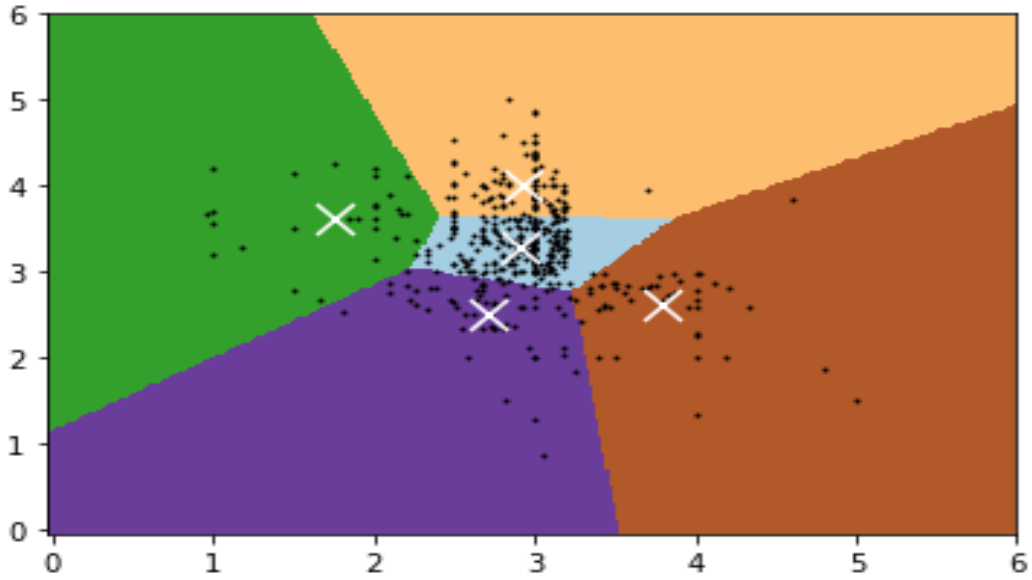
Visualize the Cluster after Find the Optimal K Value

K=5 Fig-[11]: Visualize the Optimal Cluster



6.9. Visualize the Clusters with centroids

Fig-[12]: Clusters with Centroids



7. CLUSTER EVALUATION METHODS

There are three types of clustering validation methods. One is internal cluster validation, external cluster validation and another one is relative validation method. The internal cluster validation method uses the cluster data itself and the external cluster validation method uses some external labels and the relative cluster validation method uses the different parameters in the same algorithms.

Our project we have evaluated the cluster using internal evaluation methods, especially the Dunn index and DB index methods.

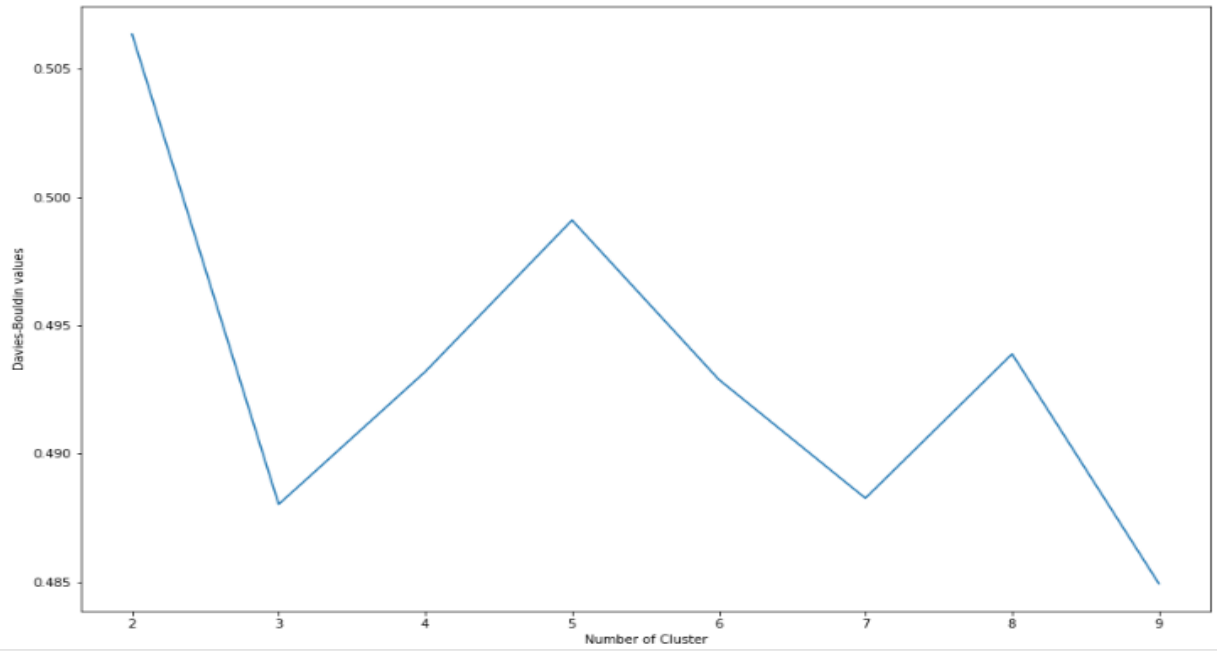
Dunn Index Method

Dunn introduced by J.C. Dunn in 1974. It uses the cluster data itself. The aim of dunn index is to identify the set of clusters compactly and with variance of cluster members and separate each cluster. The higher the number of Dunn index it was better clustering and the Dunn index is increasing the k value is optimal k value.

DB Index

DB index was introduced by David. L Davies and Donald .W Bouldin in 1979. It was an internal validation method. The clustering was done by using quantiles and features of the dataset. DB index value is low then the cluster is better.

Fig-[13]: DB Index Cluster



8. User Ratings

We have use two datasets in our project one is Movies another one is Ratings dataset the movie dataset contains Movies ID, names and genres and rating dataset contains user Id, ratings, movies Id and timestamp. We have merge the two datasets and take only the user Id, Movies names and ratings of the dataset this type of datasets contains more null values.

9.Sparse Ratings Dataset

The basically the recommendation System need more data for recommend the movies for users. The more data contains more null values and because most of the users not rate the all movies so the data contains more null values this type of matrix called sparse matrix and the k-means clustering not suitable for outliers data.so we have find the most rated movies and also find the most rated users it was useful to remove the sparsity of the dataset.

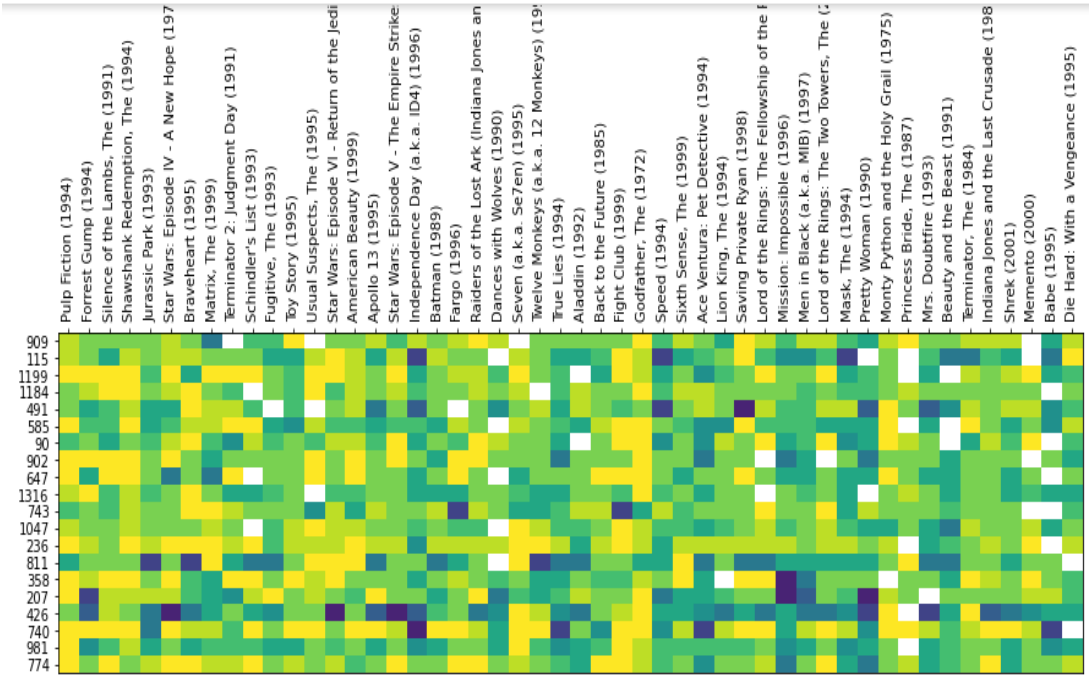
Fig-[14]: User ratings

title	'Hellboy': The Seeds of Creation (2004)	'Neath the Arizona Skies (1934)	'Round Midnight (1986)	'Salem's Lot (2004)	'Til There Was You (1997)	'burbs, The (1989)	'night Mother (1986)	(500) Days of Summer (2009)	(Untitled) (2009)	*batteries not included (1987)	...All the Marbles (California Dolls, The) (1981)	...And Justice for All (1979)	.45 (2006)	1-900 (06) (1994)	10 (1979)	10 Items or Less (2006)
userId																
1367	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1368	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1369	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1370	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1371	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Fig-[15]: Most rated movies & users

title	Pulp Fiction (1994)	Forrest Gump (1994)	Silence of the Lambs, The (1991)	Shawshank Redemption, The (1994)	Jurassic Park (1993)	Star Wars: Episode IV - A New Hope (1977)	Braveheart (1995)	Matrix, The (1999)	Terminator 2: Judgment Day (1991)	Schindler's List (1993)	Fugitive, The (1993)	Toy Story (1995)	Usual Suspects, The (1995)	Star Wars: Episode VI - Return of the Jedi (1983)	American Beauty (1999)
774	5.0	4.0	5.0	4.0	4.5	5.0	5.0	4.5	4.5	5.0	4.0	4.5	4.0	4.5	3.5
981	5.0	2.5	3.5	4.5	3.0	4.0	3.0	3.5	3.5	3.5	3.0	3.0	4.0	3.5	4.0
740	5.0	5.0	5.0	5.0	2.0	5.0	4.5	4.5	3.5	5.0	4.0	5.0	5.0	4.0	5.0
426	4.0	1.5	4.5	4.0	2.0	0.5	2.0	3.0	4.0	3.0	2.5	4.0	4.0	0.5	4.0
207	5.0	1.0	4.5	4.5	4.5	4.5	3.5	3.0	2.5	4.5	5.0	4.0	4.5	4.0	3.5

Fig-[16]: Display the Heat-map



Each column is a different movie.

Each row is a different user.

The cell's colour is the rating that each user has given to each film. The values for each colour can be checked in the scale of the right.

The white values correspond to users that haven't rated the movie.

The more vertical lines of the same colour in the cluster, the more similar the ratings will be in that cluster. Some clusters are more sparse than others, that show that the algorithm tends to group also people that watch and rate less movies.

Clusters tend to have a dominant colour : Yellowish if they liked their rated movies an blueish if don't. Horizontal lines with the same colour correspond to users with low variety in their ratings, they tend to like or dislike most of the movies.

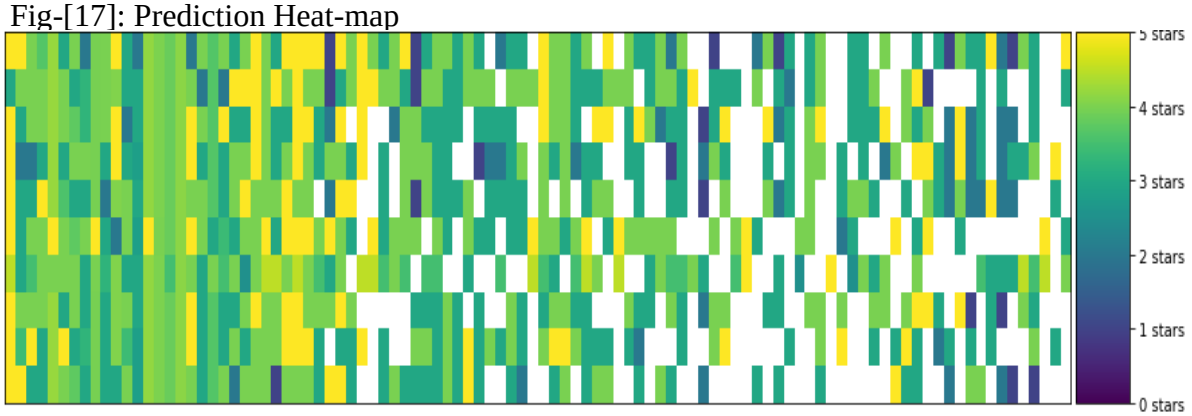
10. PREDICTION

Pick the cluster Id

Select number of users and number of movies from the dataset

Sort and print the cluster using ratings of the cluster

Fig-[17]: Prediction Heat-map



11. RECOMMENDATIONS

Average Ratings for 20 Movies

Fig-[18]: Recommendation for new users

```
Pulp Fiction (1994)                 4.302721
Rain Man (1988)                     3.928586
True Lies (1994)                    3.556270
Aladdin (1992)                      3.781649
Fight Club (1999)                   4.261532
Speed (1994)                        3.670834
Apocalypse Now (1979)               4.082981
Lion King, The (1994)               3.805448
Mask, The (1994)                    3.271062
X-Men (2000)                        3.666508
Fargo (1996)                        4.048624
Good Will Hunting (1997)            4.059010
Forrest Gump (1994)                 4.132653
Goodfellas (1990)                   4.176245
Waterworld (1995)                   2.924329
Alien (1979)                        4.033990
GoldenEye (1995)                    3.525598
Outbreak (1995)                     3.438776
Dances with Wolves (1990)           3.762252
Clueless (1995)                     3.353717
dtype: float64
```

The above recommendations was useful to new user of this system because the new user don't have any movie preferences so the average ratings of 20 movies it was useful to new users.

Movie Recommendation of Particular User

Fig-[19]: Recommendation for Single user

```
Maltese Falcon, The (1941)                                    4.625000
Inception (2010)                                              4.413043
City of God (Cidade de Deus) (2002)                           4.368421
Hoop Dreams (1994)                                            4.318182
One Flew Over the Cuckoo's Nest (1975)                        4.304878
Vertigo (1958)                                                4.295455
Amelie (Fabuleux destin d'Amélie Poulain, Le) (2001)          4.283333
Lord of the Rings: The Return of the King, The (2003)         4.256098
Lord of the Rings: The Two Towers, The (2002)                 4.250000
Memento (2000)                                                4.250000
Rear Window (1954)                                            4.250000
Sound of Music, The (1965)                                    4.250000
Casablanca (1942)                                             4.231707
Lord of the Rings: The Fellowship of the Ring, The (2001)     4.215686
Departed, The (2006)                                          4.205882
Cool Hand Luke (1967)                                         4.203704
Seven Samurai (Shichinin no samurai) (1954)                   4.200000
WALL·E (2008)                                                 4.194444
Bridge on the River Kwai, The (1957)                          4.173913
Iron Man (2008)                                               4.166667
Name: 0, dtype: float64
```

12.RESULT AND DISCUSSION

The Result part of our project we have is to evaluate the cluster using DB index score and Dunn index method is not do it in project. In the DB index part we have got the small index value we better clustering otherwise we don't have better clustering.

Our project DB Index value is = **0.4991045608232513.**

The above is a little lower but it was applicable for our project.

Dataset Details

Our project we have using Movie-lens dataset. It contains 6 csv files but our project we have used only two datasets one is rating.csv and another one is movies.csv.

Rating.csv

It contains 4 columns and 20000263 rows

User Id, Movies Id, ratings and timestamp

Movies.csv

It contains 3 columns and 27278 rows.

Movies Id, Movie names and genres

Totally we have used 27278 movies, 20000263 ratings and 138493 users. This data was collected from 09 January 1995 to 31 march 2015. But we have used only 2,00,000 ratings only for our project.

Software Details:

IDE- Google colab

Runtime: 3 types of runtime in google colab GPU, TPU and Normal one our project we execute normal runtime.

Google colab options mounted to our Google Drive. This IDE was given to the 64GB space.

13. Conclusion

In our project we have built a movie recommendation system using collaborative filtering and k-means algorithm we have calculate the genre based rating in our dataset and we have using this genre based rating used in recommendation the movies for users but we have taken only three genres like that romance, science-fiction and action movies in future we have taken more genres to predict the movie recommendation system.

We have till now built only for machine learning. We have also attached the front-end or web app for this project. Web-app using Django or flask because the user interface is one of the best to interact with the users and users easily understand the recommendation results.

We have been using only collaborative filtering methods for our project. In the future we have implemented the content-based and collaborative based system in a single one.

14.REFERENCES

[1]. P. Phorasim and L. Yu, “Movies recommendation system using collaborative filtering and k-means,” International Journal of Advanced Computer Research, vol. 7, no. 29, p.52, 2017.

[2]. Xingyuan Li.2011 “Collaborative Filtering Recommendation Algorithm Based on Cluster”, International Conference on Computer Science and network Technology(ICCSNT), IEEE, 4: 2682-2685

[3].Md.Asif Shahjalal, Zubaer Ahmad, Mohammad Shamsul Arefin,Mohammad Rubaiyat Tanvir Hossain,"A User Rating Based Collaborative Filtering Approach to Predict Movie Preferences",3rd international Conference on Electrical Information and Communication Technology (EICT),7-9 December 2017.

[4].Akansh Surendran, Aditya Kumar Yadav, Aditya Kumar,"Movie Recommendation system using Machine Learning Algorithm",International Research Journal of Engineering and Technology (IRJET),Volume:07 Issue:04|Apr 2020.

[5].Saurabh Bhalla, Baibhav Kumar, Rajat Tiwari, Dr.P.A.Jadhav,"Movie Recommendation System Using Collaborative and Content-Based Filtering", BV(DU)COE,Pune,September 2020 IJSDR|Volume 5 Issue 9.

[6].Rishabh Ahuja, Arun Solanki, Anand Nayyar,"Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor",9th International Conference on Cloud Computing,Data Science & Engineering,978-1-5386-5933-5/19/$31.00 2019 IEEE.

[7].Vikash Yadav, Rati Shukla, Aprna Tripathi,Anamika Maurya,"A New Approach for Movie Recommendation System using K-Means Clustering and PCA",Journal of Scientific & Industrial Research, Vol.80,February 2021,pp. 159-165, Received 07 August 2020.

[8].Rahul Katarya, Om Prakash Verma,"An effective collaborative movie recommender system with cuckoo search", Department of computer Science& Engineering Delhi Technological University, Delhi,india, http://dx.doi.org/1ij.2016.10.002.0.1016.

15. APPENDIX:

Import the Libraries & Packages

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from scipy.sparse import csr_matrix
from sklearn.metrics import silhouette_samples,silhouette_score
import itertools
from mpl_toolkits.axes_grid1 import make_axes_locatable

%matplotlib inline
```

Load the movies dataset

```
movies=pd.read_csv("drive/My Drive/Ai/movie.csv")
movies.head()
```

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

Load the ratings dataset

```
rating=pd.read_csv("drive/My Drive/Ai/rating.csv")
rating.head()
```

	userId	movieId	rating	timestamp
0	1	2	3.5	2005-04-02 23:53:47
1	1	29	3.5	2005-04-02 23:31:16
2	1	32	3.5	2005-04-02 23:33:39
3	1	47	3.5	2005-04-02 23:32:07
4	1	50	3.5	2005-04-02 23:29:40

Movies dataset Info

```
movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27278 entries, 0 to 27277
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   movieId  27278 non-null  int64 
 1   title    27278 non-null  object
 2   genres   27278 non-null  object
dtypes: int64(1), object(2)
memory usage: 639.5+ KB
```

Rating dataset Info

```
ratings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 4 columns):
 #   Column     Non-Null Count   Dtype  
---  ------     --------------   -----  
 0   userId     200000 non-null  int64  
 1   movieId    200000 non-null  int64  
 2   rating     200000 non-null  float64
 3   timestamp  200000 non-null  object 
dtypes: float64(1), int64(2), object(1)
memory usage: 6.1+ MB
```

Calculate the Genre Based Ratings

```
def get_genre_ratings(ratings, movies, genres, column_names):
    genre_ratings = pd.DataFrame()
    for genre in genres:
        genre_movies = movies[movies['genres'].str.contains(genre) ]
        avg_genre_votes_per_user = ratings[ratings['movieId'].isin(genr
e_movies['movieId'])].loc[:, ['userId', 'rating']].groupby(['userId'])[
'rating'].mean().round(2)

        genre_ratings = pd.concat([genre_ratings, avg_genre_votes_per_u
ser], axis=1)

    genre_ratings.columns = column_names
    return genre_ratings

genre_ratings = get_genre_ratings(ratings, movies, ['Romance', 'Sci-
Fi'], ['Avg_romance_rating', 'Avg_scifi_rating'])
genre_ratings.head()
```

	Avg_romance_rating	Avg_scifi_rating
1	3.95	3.71
2	3.83	4.61
3	4.06	4.00
4	3.50	3.00
5	3.94	4.60

Take only best rating score

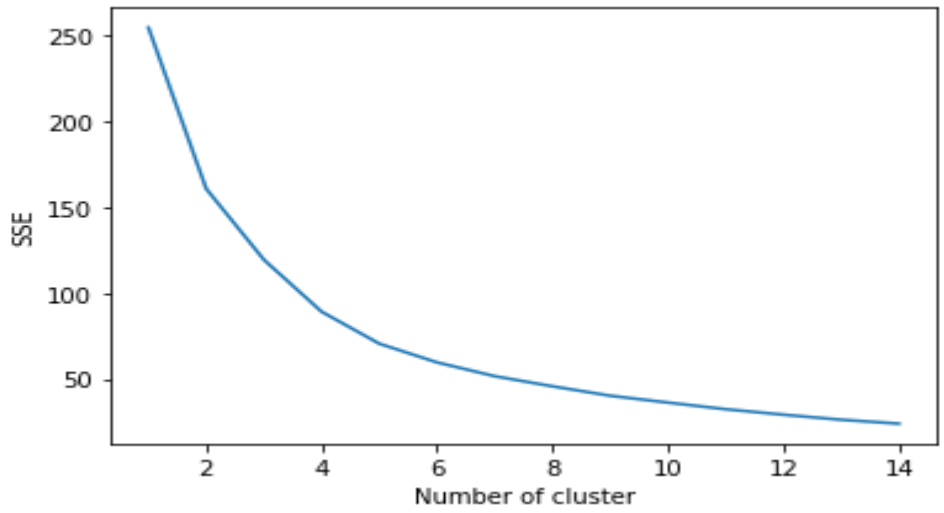
```
def best_genre_rating_dataset(genre_ratings, score_limit_1, score_limit
_2):
    best_dataset = genre_ratings[((genre_ratings['Avg_romance_rating']
< score_limit_1 - 0.2)&(genre_ratings['Avg_scifi_rating'] > score_limit
_2)) | ((genre_ratings['Avg_scifi_rating'] < score_limit_1) & (genre_ra
tings['Avg_romance_rating'] > score_limit_2))]
   best_dataset = pd.concat([best_dataset[:702], genre_ratings[:2]])
   best_dataset = pd.DataFrame(best_dataset.to_records())
   return best_dataset

   best_dataset = best_genre_rating_dataset(genre_ratings, 3.2, 2.5)
```

	userId	Avg_romance_rating	Avg_scifi_rating
0	4	3.50	3.00
1	8	3.95	3.18
2	9	4.00	2.67
3	15	3.33	3.00
4	18	3.88	2.45

Elbow Method

```
sse = {}
for k in range(1, 15):
    kmeans = KMeans(n_clusters=k, max_iter=1000).fit(X)
    best_dataset["clusters"] = kmeans.labels_
    sse[k] = kmeans.inertia_ # Inertia: Sum of distances of samples to
their closest cluster center
plt.figure()
plt.plot(list(sse.keys()), list(sse.values()))
plt.xlabel("Number of cluster")
```

```
plt.ylabel("SSE")
plt.show()
```



Silhouette method

```
for n_cluster in range(2, 15):
    kmeans = KMeans(n_clusters=n_cluster).fit(X)
    label = kmeans.labels_
    sil_coeff = silhouette_score(X, label, metric='euclidean')
    print("For n_clusters={}, The Silhouette Coefficient is {}".format(
n_cluster, sil_coeff))
```

```
For n_clusters=2, The Silhouette Coefficient is 0.34319264631605945
For n_clusters=3, The Silhouette Coefficient is 0.3472401702342571
For n_clusters=4, The Silhouette Coefficient is 0.3762261237073798
For n_clusters=5, The Silhouette Coefficient is 0.3979720401051306
For n_clusters=6, The Silhouette Coefficient is 0.39757917151714356
For n_clusters=7, The Silhouette Coefficient is 0.34647417582503093
For n_clusters=8, The Silhouette Coefficient is 0.3517175512770122
For n_clusters=9, The Silhouette Coefficient is 0.3643569347527321
For n_clusters=10, The Silhouette Coefficient is 0.36904807957882313
For n_clusters=11, The Silhouette Coefficient is 0.37195478447807057
For n_clusters=12, The Silhouette Coefficient is 0.3667028287560942
For n_clusters=13, The Silhouette Coefficient is 0.3726684247850386
For n_clusters=14, The Silhouette Coefficient is 0.3817598821922288
```

Cluster Evaluation – DB Index

```
from sklearn.cluster import KMeans
from sklearn.metrics import davies_bouldin_score
kmeans = KMeans(n_clusters=5,max_iter=1000,random_state=10).fit(best_da
taset_3_genres)
labels = kmeans.labels_
```

```
print(davies_bouldin_score(best_dataset_3_genres, labels))
```

DB Index value is = **0.4991045608232513.**

Calculate the User Based Ratings

```
# Merge the two tables then pivot so we have Users X Movies dataframe
ratings_title = pd.merge(ratings, movies[['movieId', 'title']], on='mov
ieId' )
user_movie_ratings = pd.pivot_table(ratings_title, index='userId', colu
mns= 'title', values='rating')
# Print he number of dimensions and a subset of the dataset
print('dataset dimensions: ',
      user_movie_ratings.shape, '\n\nSubset example:' )
user_movie_ratings.iloc[:6, :10]
```

title	'Hellboy': The Seeds of Creation (2004)	'Neath the Arizona Skies (1934)	'Round Midnight (1986)	'Salem's Lot (2004)	'Til There Was You (1997)	'burbs, The (1989)	'night Mother (1986)	(500) Days of Summer (2009)	(Untitled) (2009)	*batteries not included (1987)
userId										
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Calculate the most rated users and most rate movies

```
n_movies = 50
n_users = 20
most_rated_movies_users_selection = sort_by_rating_density(user_movie_r
atings, n_movies, n_users)
print(most_rated_movies_users_selection.shape)
most_rated_movies_users_selection.head()
```

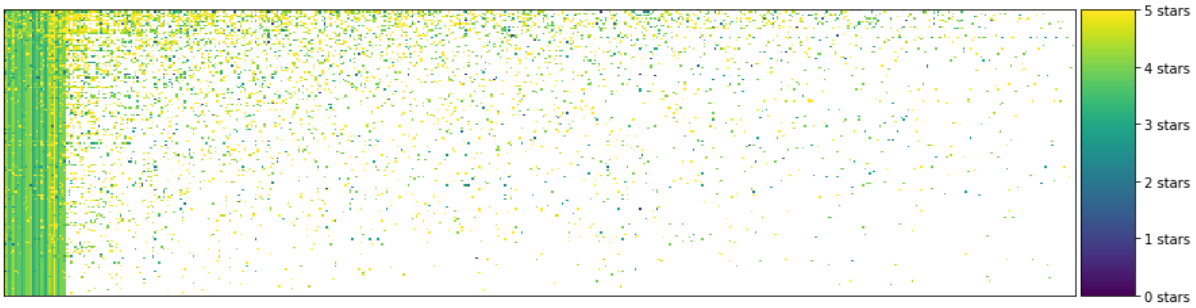
title	Pulp Fiction (1994)	Forrest Gump (1994)	Silence of the Lambs, The (1991)	Shawshank Redemption, The (1994)	Jurassic Park (1993)	Star Wars: Episode IV - A New Hope (1977)	Braveheart (1995)	Matrix, The (1999)	Terminator 2: Judgment Day (1991)	Schindler's List (1993)	Fugitive, The (1993)	Toy Story (1995)	Usual Suspects, The (1995)	Star Wars: Episode VI - Return of the Jedi (1983)	American Beauty (1999)
774	5.0	4.0	5.0	4.0	4.5	5.0	5.0	4.5	4.5	5.0	4.0	4.5	4.0	4.5	3.5
981	5.0	2.5	3.5	4.5	3.0	4.0	3.0	3.5	3.5	3.5	3.0	3.0	4.0	3.5	4.0
740	5.0	5.0	5.0	5.0	2.0	5.0	4.5	4.5	3.5	5.0	4.0	5.0	5.0	4.0	5.0
426	4.0	1.5	4.5	4.0	2.0	0.5	2.0	3.0	4.0	3.0	2.5	4.0	4.0	0.5	4.0
207	5.0	1.0	4.5	4.5	4.5	4.5	3.5	3.0	2.5	4.5	5.0	4.0	4.5	4.0	3.5

Covert the most rated movies data frame to sparse matrix

```
sp_arr = csr_matrix(most_rated_movies_500)
sdf = pd.DataFrame.sparse.from_spmatrix(sp_arr)
sparse_ratings = sdf1.sparse.to_coo()
```

Visualize the cluster Density

```
cluster_number = 1
n_users = 300
n_movies = 1000
cluster = clustered[clustered.group == cluster_number].drop(['index', '
group'], axis=1)
cluster = sort_by_rating_density(cluster, n_movies, n_users)
draw_movies_heatmap(cluster, axis_labels=False)
```



Particular Movie rating

```
movie_name = "Speed (1994)"
cluster[movie_name].mean()
```

```
3.6719806403574164
```

Average movies ratings

```
cluster.mean().head(20)
```

```
Pulp Fiction (1994)              4.423529
Apocalypse Now (1979)            4.177510
True Lies (1994)                 3.613955
Aladdin (1992)                   3.730985
Fight Club (1999)                4.260129
Speed (1994)                     3.671981
Rain Man (1988)                  3.911100
Lion King, The (1994)            3.843869
Mask, The (1994)                 3.322624
X-Men (2000)                     3.737285
Fargo (1996)                     4.106967
Good Will Hunting (1997)         4.030711
Goodfellas (1990)                4.160773
Waterworld (1995)                2.860859
Outbreak (1995)                  3.457983
Alien (1979)                     4.065666
GoldenEye (1995)                 3.466161
Forrest Gump (1994)              4.164706
Dances with Wolves (1990)        3.826351
Clueless (1995)                  3.315444
dtype: float64
```

Recommend the movies for particular user id

```
# Pick a user ID from the dataset
userId = 7

# Get all this user's ratings
user_2_ratings  = cluster.loc[userId , :]

# Which movies did they not rate?
user_2_unrated_movies =  user_2_ratings[user_2_ratings.isnull()]

avg_ratings = pd.concat([user_2_unrated_movies, cluster.mean()], axis=1
, join='inner').loc[:,0]

# highest rated movies are presented first
avg_ratings.sort_values(ascending=False)[:20]
```

```
Remains of the Day, The (1993)                                    5.000000
Fantasia (1940)                                                   5.000000
Last of the Mohicans, The (1992)                                  5.000000
Tombstone (1993)                                                  5.000000
Nell (1994)                                                       5.000000
Big Fish (2003)                                                   5.000000
Chasing Amy (1997)                                                5.000000
Postman, The (Postino, Il) (1994)                                 4.954545
Christmas Story, A (1983)                                         4.923077
Some Like It Hot (1959)                                           4.916667
Mulholland Drive (2001)                                           4.900000
Lawrence of Arabia (1962)                                         4.884615
Notting Hill (1999)                                               4.875000
Good, the Bad and the Ugly, The (Buono, il brutto, il cattivo, Il) (1966)    4.875000
Fried Green Tomatoes (1991)                                       4.875000
Dogma (1999)                                                      4.857143
African Queen, The (1951)                                         4.833333
Dracula (Bram Stoker's Dracula) (1992)                            4.833333
Amadeus (1984)                                                    4.833333
Léon: The Professional (a.k.a. The Professional) (Léon) (1994)    4.818182
Name: 0, dtype: float64
```

Another output for different user id

```
# Pick a user ID from the dataset
userId = 20

# Get all this user's ratings
user_2_ratings  = cluster.loc[userId , :]

# Which movies did they not rate?
user_2_unrated_movies =  user_2_ratings[user_2_ratings.isnull()]
```

```
avg_ratings = pd.concat([user_2_unrated_movies, cluster.mean()], axis=1
, join='inner').loc[:,0]

# highest rated movies are presented first
avg_ratings.sort_values(ascending=False)[:20]
```

```
Last of the Mohicans, The (1992)                                   5.000000
Big Fish (2003)                                                    5.000000
Chasing Amy (1997)                                                 5.000000
Remains of the Day, The (1993)                                     5.000000
Fantasia (1940)                                                    5.000000
Postman, The (Postino, Il) (1994)                                  4.954545
Mulholland Drive (2001)                                            4.900000
Lawrence of Arabia (1962)                                          4.884615
Good, the Bad and the Ugly, The (Buono, il brutto, il cattivo, Il) (1966)  4.875000
Dogma (1999)                                                       4.875000
Dracula (Bram Stoker's Dracula) (1992)                             4.833333
African Queen, The (1951)                                          4.833333
Tombstone (1993)                                                   4.833333
Spirited Away (Sen to Chihiro no kamikakushi) (2001)               4.812500
Deep Impact (1998)                                                 4.800000
Father of the Bride Part II (1995)                                 4.800000
Like Water for Chocolate (Como agua para chocolate) (1992)         4.800000
Seven Samurai (Shichinin no samurai) (1954)                        4.791667
Gone with the Wind (1939)                                          4.785714
Some Like It Hot (1959)                                            4.785714
Name: 0, dtype: float64
```

Links:-

Full Code Link:-

https://colab.research.google.com/drive/1PLFHKpo44KlE0iszVgN_Jz32itHrgMNC?usp=sharing

Dataset Link:-

Movies.csv

https://drive.google.com/file/d/1Do9g9Lqy8v_d4vfgx03gTNGY-gvaUReh/view?usp=sharing

Ratings.csv

https://drive.google.com/file/d/1oBaha1XzXXDhqIVsYbzOb47baB7AglIw/view?usp=sharing