Department of Computer Science

Missouri State University

CSC735 - Data Analytics

Movie recommendation system with clustering algorithm using Movie lens dataset

Project Report

Submitted by

Section: 1 Group number: 4
Ayesha Siddiqua ID: M03526026
Shusmoy Chowdhury ID: M03386286

Submitted on November 27, 2023

Abstract

The movie recommendation system aims to selectively predict and suggest films that a specific user is most likely to be interested in watching. Our project utilizes clustering algorithms for movie recommendations, a system that forecasts user preferences based on their past movie-watching history or activity. The recommendation process begins by extracting features from the Movielens Dataset and subsequently employing clustering algorithms, including K-Means clustering, Gaussian Mixture Model, and Latent Dirichlet Allocation Model, to form movie clusters. Subsequently, we identify the user's last highest-rated movie, determine its cluster, and present the top ten highest and most-rated movies from that cluster as recommendations. The evaluation metric for the clustering algorithms is the silhouette coefficient, with results indicating that K-Means clustering yields superior clusters compared to other algorithms.

Keywords: Movie Recommendation System, Clustering Algorithm, Kmeans, GMM, LDA, Silhouette Co-efficient.

1 Introduction

The concept of recommendation systems is an active platform for researchers. The system could be built to recommend a book, movies, songs, venues, sites, and many more depending on the user's needs. There are multiple techniques to build a recommendation system, they are Content-based filtering, Memory-based filtering, and Model-based filtering. Most recommendation systems rely on user input, ratings, preferences, and similarities. The collected information is analyzed for these to produce recommendations. There have been many proposed recommendation systems that used KNN, Neural networks, and other deep learning algorithms with either content-based, memory-based, or model-based filtering techniques.

Movie recommendation system is one of the most popular applications of big data analysis using machine learning. The recommendation is done by studying the users' past ratings and observed behaviors.

In this project, we will be using the Movielens dataset[1] for the movie recommendation system. GroupLens Research has collected Rating data of movies from users over periods of time and made them available in the MovieLens website. The movielens latest dataset were created by 330975 users between January 09, 1995 and July 20, 2023. This dataset was generated on July 20, 2023, and it contains 33832162 ratings and 2328315 tag applications across 86537 movies. The dataset contains the following files: genome-scores.csv, genome-tags.csv, links.csv, movies.csv, ratings.csv, tags.csv. This project mostly utilises data from ratings.csv for extracting the feature's popularity and average rating of a specific movie, and data from movies.csv for extracting the feature genre and movie-specific metadata.

The movie recommendation system filters and predicts only those movies that a corresponding user is most likely to want to watch. In our project, we have used a content-based filtering approach for movie recommendation. It is a type of recommendation system that tries to guess what a user would like based on the user's activity or prior liking. The recommendation system will be created using clustering algorithms such as (LDA, K means, GMM) unsupervised machine learning algorithms that can be used to solve recommendation problems.

The rest of the report is structured as follows Section 2 provides the summary of previous

work done in the movie recommendation system. Section 3 explains the methodology of our movie recommendation system. Section 4 shows the comparison of our clustering algorithms based on different matrices and the final results of our movie recommendation system. Finally, Section 5 concludes this report with a summary of our work and future extension of this project.

2 Related Work

In recent years, there have been numerous research works dealing with the problem of movie recommendation systems using Artificial Intelligence (AI) and Machine Learning (ML) Techniques. Goyeni et al. [2] have discussed the limitations and challenges of the movie recommendation system. R. Singh et al [3] describes an approach that offers generalized recommendations to every user, based on movie popularity and/or genre. They have illustrated the modeling of a movie recommendation system by making the use of content-based filtering in the movie recommendation system. The KNN algorithm is implemented in this model along with the principle of cosine similarity.
B.-B. Cui et al. [4] designed and implemented a movie recommendation system prototype combined with the actual needs of movie recommendation through researching of KNN algorithm and collaborative filtering algorithm. Ahuja et al. [5] have proposed a movie recommendation system based on K-means clustering and the KNN algorithm. In our project, we have implemented three distinct clustering algorithms for movie recommendation and presented a comparative analysis.

3 Methodology

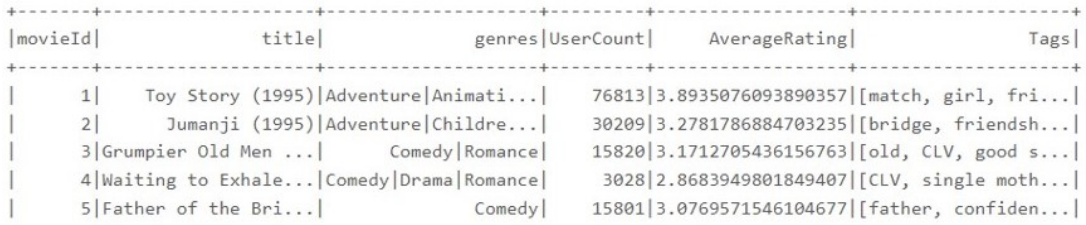
3.1 Data Prepossessing

Data preprocessing is the process of transforming unusable raw data into data that can potentially be used by machine learning techniques. Data cleaning, data integration, data reduction, and data transformation are the stages of this data pre-processing. Data cleaning refers to techniques that clean data by removing outliers, replacing missing values, smoothing noisy data, and correcting inconsistent data. The practice of merging data from various source systems to produce a single set is known as data integration. Without compromising the integrity of the original dataset, data reduction aids in the removal of unnecessary data. The transformation of data into a format suitable for data modeling is the last stage of data pre-processing. We have utilized Movielen's latest dataset where the data is divided into multiple CSV files. For our project, we have used three files movies.csv, ratings.csv, and tags.csv. The Movies.csv file contains movieId, title, and genres. Tags.csv file contains userId, movieId, tag, and timestamp. Ratings.csv file contains userId, movieId, rating, and timestamp. The data preprocessing steps are described in the following subsections.

3.1.1 Join Dataset

The first step of our preprocessing is joining the datasets into a single dataframe. We have used Spark Scala for this project. First, we calculated average rating and user count for each

movie from the rating dataframe. Next, we modified the tag dataframe to group the tags of each movie into a single row. Finally, we have joined three dataframe based on movieId and the joined dataframe is shown in Figure 1.



movieId	title	genres	UserCount	AverageRating	Tags
1	Toy Story (1995)	Adventure|Animati...	76813	3.8935076093890357	[match, girl, fri...
2	Jumanji (1995)	Adventure|Childre...	30209	3.2781786884703235	[bridge, friendsh...
3	Grumpier Old Men ...	Comedy|Romance	15820	3.1712705436156763	[old, CLV, good s...
4	Waiting to Exhale...	Comedy|Drama|Romance	3028	2.8683949801849407	[CLV, single moth...
5	Father of the Bri...	Comedy	15801	3.0769571546104677	[father, confiden...

Figure 1: Joined dataset

3.1.2 Data Tranformation

Data transformation refers to the process of converting raw data into a more suitable format for analysis, interpretation, or presentation. This process is a crucial step in data preprocessing, which aims to enhance the quality of data and make it more valuable for various analytical tasks. For our project, we first transform the genre column into a set of words using RegexTokenizer. The RegexTokenizer is a tokenizer relying on regular expressions employed to extract tokens from text, achieved either by employing a specified regex pattern for text segmentation or by iteratively matching the regex. Additional parameters offer the option to filter tokens based on a minimum length. The output is an array of strings, which may be empty. The transformation is shown in Figure 2a.

Finally, we converted the list of words to meaningful features using hashingTF. HashingTF is a Transformer that takes sets of terms and converts those sets into fixed-length feature vectors. A raw feature is mapped into an index by applying a hash function. The transformation is shown in Figure 2b.

3.1.3 Feature Extraction

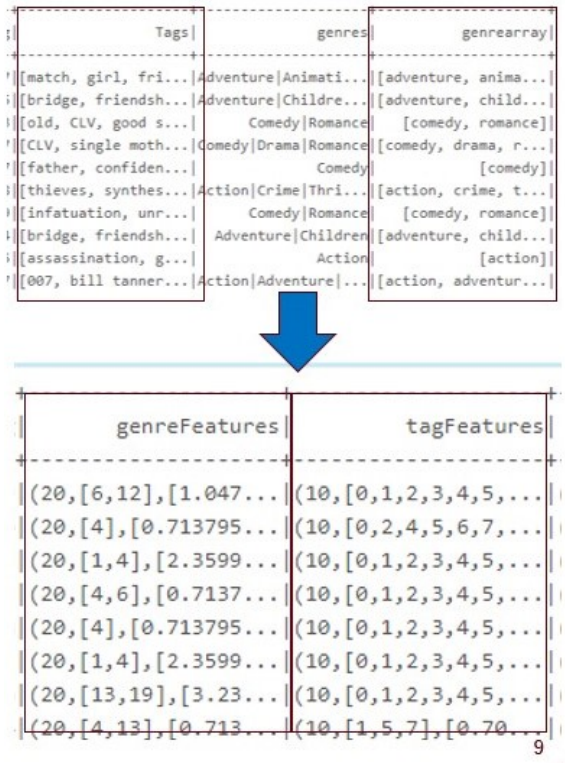
Feature selection is a method that aims to decrease the number of features in machine learning. It involves the automatic or manual selection of features that have the most significant impact on the target prediction variable. This process is crucial in machine learning because irrelevant or partially relevant features can significantly affect the accuracy of the model. For our project, we have selected four distinct features to use in our clustering algorithms. The features are genre, tag, user count and average rating. We have merged these four columns in a single feature column using vectorAssembler. The vectorAssembler is a feature transformer that merges multiple columns into a vector column. The feature column is shown in Figute 3.

3.2 Clustering Movies

After processing the data, we tried to cluster the movies so that similar kinds of movies could be grouped into the same groups. Then, we will filter the top 10 movies based on the highest

genres	movieId	words
Adventure|Animation|Children|Comedy|Fantasy	1	[adventure, animation, children, comedy, fantasy]
Adventure|Children|Fantasy	2	[adventure, children, fantasy]
Comedy|Romance	3	[comedy, romance]
Comedy|Drama|Romance	4	[comedy, drama, romance]
Comedy	5	[comedy]
Action|Crime|Thriller	6	[action, crime, thriller]
Comedy|Romance	7	[comedy, romance]
Adventure|Children	8	[adventure, children]
Action	9	[action]
Action|Adventure|Thriller	10	[action, adventure, thriller]

(a) Convert genre data into a set of words



(b) Convert tag and genre into numeric values

Figure 2: Data transformation

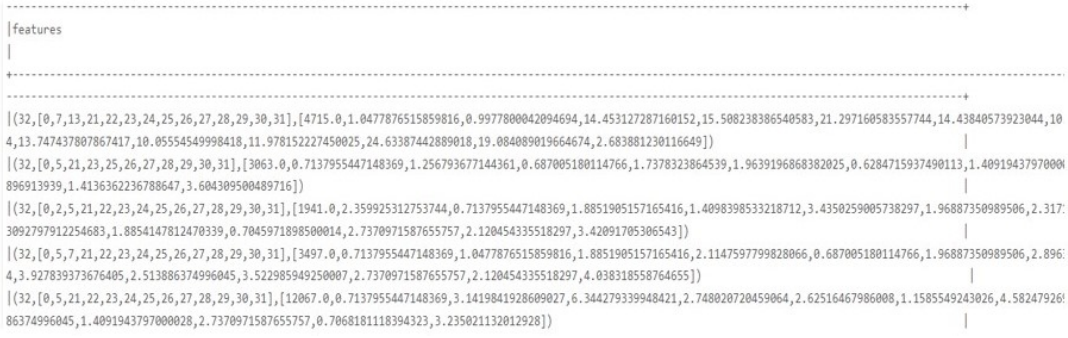


Figure 3: Feature Extraction

rating and the highest number of user-rated movies. For clustering, we have used three clustering methods from the spark's ml.clustering package in this project. They are described below.

3.2.1 K Means Clustering.

K-means is one of the most widely used clustering algorithms. In this method, a user-specified number of clusters (K) is initially assigned randomly to distinct points in the dataset. Subsequently, the unassigned points are allocated to a cluster based on their distance from the previously assigned point, measured using Euclidean distance. Following this assignment, the centroid (center) of the cluster is calculated, and the entire process is iteratively repeated. Each point is assigned to a specific centroid, and a new centroid is recalculated. This iteration continues for a defined number of steps or until convergence, denoting a point when the centroid locations stabilize and cease to change significantly. Figure 4 shows the K means clustering in this project. K means clustering can detect circular clusters and is very sensitive to the initial centroids of the algorithm. K means may not be able to detect the overlapping clusters. However, the movies can be into overlapping clusters based on similar genres and tags.

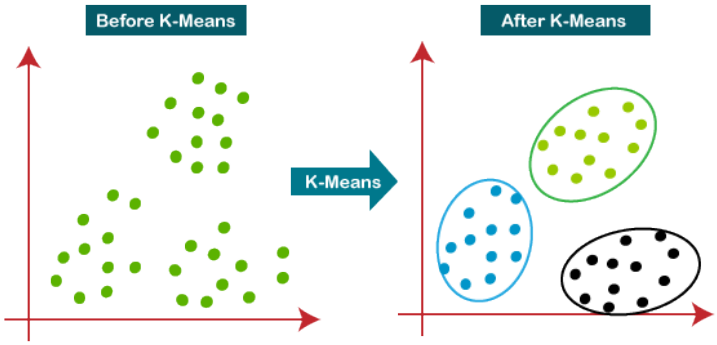


Figure 4: K Means Clustering algorithms

3.2.2 Gaussian Mixture Model

Gaussian Mixture Model is a probabilistic model used in machine learning for clustering and density estimation. It assumes that the data points are generated from a mixture of several Gaussian distributions with unknown parameters. In the context of clustering, GMM aims to identify the underlying clusters in a dataset by estimating the parameters of these Gaussian distributions. A Gaussian distribution represents each cluster, and the model calculates the probability that a data point belongs to each cluster. The data points are then assigned to the cluster with the highest probability. Figure 5 shows the clustering with GMM algorithm. K-means forms strict clusters with each point exclusively belonging to a single cluster. In contrast, Gaussian Mixture Models enable more flexible clusters by associating probabilities with data points, offering a nuanced representation without rigid boundaries.

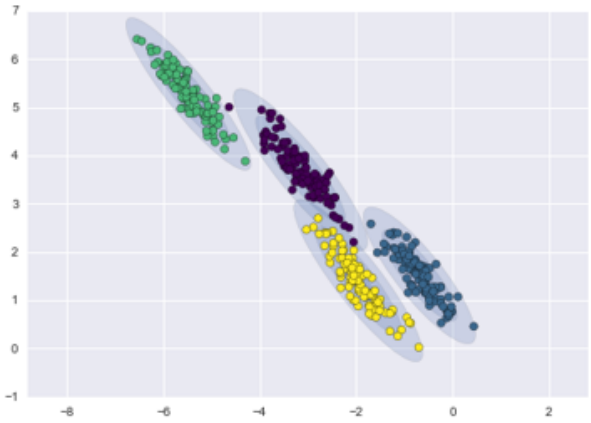


Figure 5: Gaussian Mixture Model Clustering algorithms

3.2.3 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a clustering model often used for topic modeling in text documents. It aims to identify the main topics and associated keywords in a set of documents. LDA interprets each document as a mix of various topics, allowing for a variable number of contributions from different input topics. Figure 6 shows the clustering methods of LDA. As our movielens dataset has words in genres and tags, we want to use LDA to find the main genres and tags for the movie clusters.

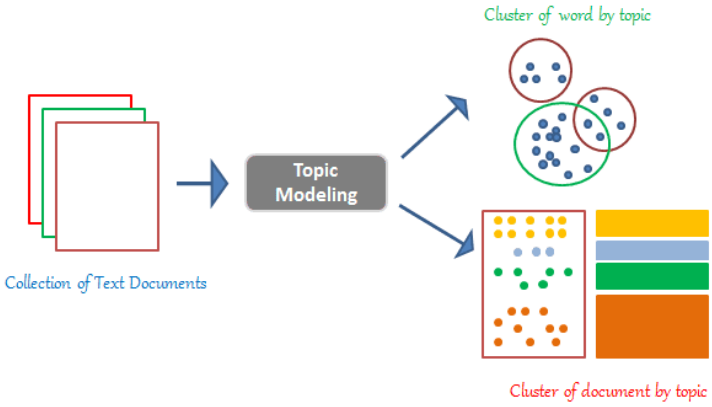


Figure 6: Latent Dirichlet Allocation Clustering algorithms

3.3 Movie Recommendation for the User

After clustering the movies with clustering algorithms, we will suggest ten movies based on the following criteria to the users.

1. Maximum Average rating
2. Maximum number of user rated movies

We have filtered the latest movies that have been rated by the users recently. We have done that by filtering the data from the tag's dataset based on the timestamp. Then we have got the highest-rated movie from the latest movies. Then, we found the cluster for that highest-rated

movie. Then we have suggested the user's top ten movies from that cluster based on the two criteria we have previously mentioned.

4 Results

4.1 Finding the value of K

Selecting the appropriate K value is crucial for successfully applying the K-means algorithm. Since there is no fixed rule for determining the number of clusters, experimentation with different values is necessary. The Elbow method is commonly used to identify the optimal K value. This method involves calculating the Within Cluster Sum of Squared Errors (WCSS), the sum of squared distances between each point and the centroid in a cluster. Table 1 displays WCSS values for various cluster numbers, and in Figure 7, we've plotted WCSS against the number of clusters. The plot resembles an elbow, and as the number of clusters rises, the WCSS value decreases. The WCSS is greatest when K equals 5. Upon analyzing the graph, we can visualize a noticeable change occurring at K=10, forming an elbow shape. Beyond this point, the graph follows a nearly parallel path to the X-axis. According to the elbow method, the K value corresponding to the elbow is considered the optimal number of clusters, making K=10 the optimal choice in our case.

Table 1: WCSS for different number of Clusters

Number of Clusters	WCSS
5	44687659115.550160
10	13269938837.774704
15	4562471358.5815170
20	2859394958.236343
25	1842899211.0363803
30	1435770940.071001

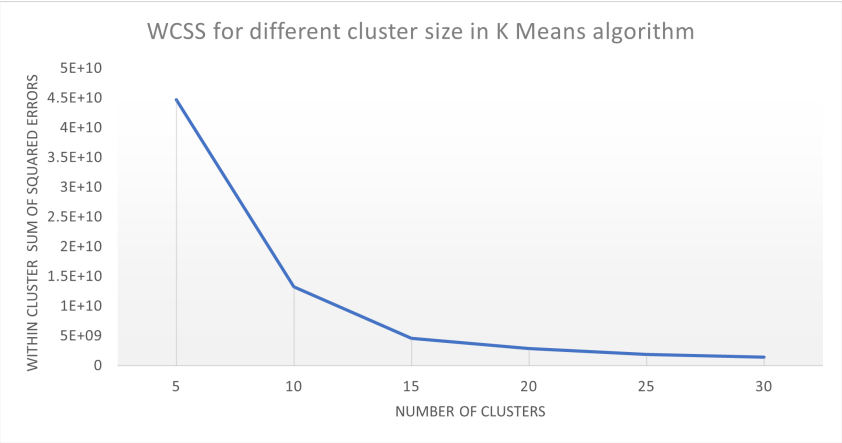


Figure 7: WCSS for different cluster size in K Means algorithm

4.2 Cluster Number and Cluster Size from Different Clustering Algorithms

The elbow method identified the optimal K value as 10, which we applied across all clustering algorithms. However, distinct algorithms yielded varying numbers of clusters with different sizes. Table 2 displays the number of clusters determined by each algorithm. K-means consistently forms 10 clusters, adhering to the specified K value. In contrast, GMM results in only two clusters, while LDA produces eight clusters for the movies. Figure 8 visually compares the cluster numbers generated by different clustering algorithms.

Table 3 shows different cluster sizes found by different clustering algorithms. K means clusters

Table 2: Cluster number for different of Clustering algorithm

Algorithm	Number of Clusters
K Means	10
GMM	2
LDA	8

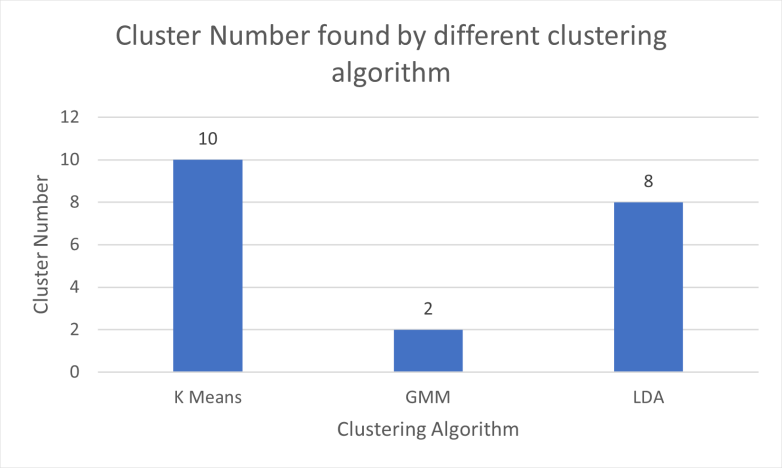


Figure 8: Cluster Number found by different clustering algorithm

the movies into 10 different sizes. GMM divides the movies into only two clusters with large number of movies. GMM struggle in high-dimensional spaces as the number of parameters to be estimated grows with the dimensionality. This can lead to over-fitting and difficulties in accurately capturing the underlying structure of the data. The clustering from LDA also may not be appropriate, as we can see one of the clusters only have one movie.

4.3 Comparison of Different Clustering Algorithms

We evaluated our clustering algorithm by analyzing both training time and silhouette coefficient. Table 4 presents a comparative overview of the clustering algorithm. Figure 9 illustrates that LDA requires less time for model training compared to other algorithms, with GMM taking the longest. However, it's crucial to note that training time alone does not necessarily reflect the effectiveness of the clustering algorithm.

For this reason, we have used the silhouette coefficient to measure how well-defined clusters are in a given clustering configuration. It quantifies the separation between clusters and the

Table 3: Cluster size for different of Clustering algorithm

Cluster	K Means	GMM	LDA
0	45016	19510	1
1	116	0	1937
2	545	0	10038
3	214	0	26784
4	20	0	0
5	39	0	5574
6	74	0	564
7	1052	0	0
8	362	0	2731
9	2716	30644	2525

cohesion within clusters. The silhouette coefficient ranges from -1 to 1, where:

1. A high silhouette coefficient indicates that the object is well-matched to its own cluster and poorly matched to neighboring clusters.

2. A low silhouette coefficient indicates that the object is poorly matched to its own cluster and may be well matched to a neighboring cluster.

From table 4 we can see that K Means clustering algorithm has the highest silhouette coefficient among all other algorithms. So, we can say that K means creates better clusters compared to GMM and LDA.

Table 4: Training Time and Silhouette coefficient different of Clustering algorithm

Algorithm	Training Time(minutes)	silhouette coefficient
K Means	5.58	0.941107604
GMM	5.78	0.251992028
LDA	2.97	-0.52210153

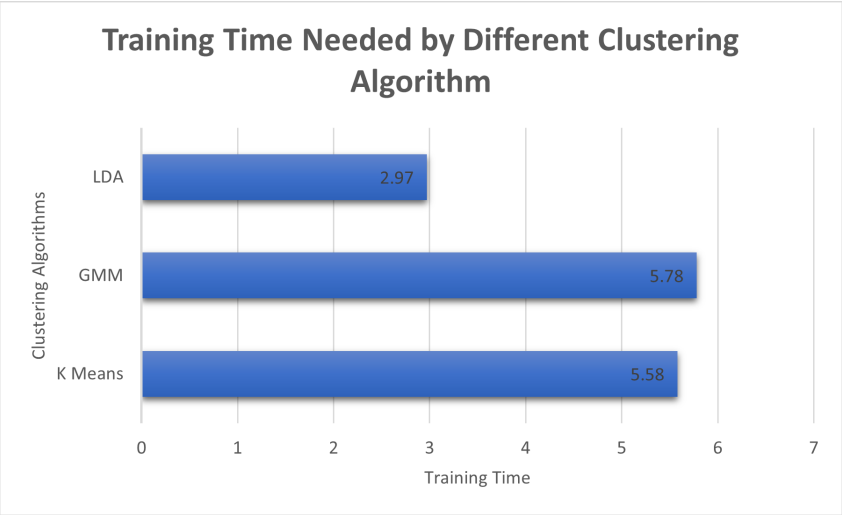


Figure 9: Training time comparison of different clustering algorithm

4.4 Recommendations Results

We have filtered the user's top ten latest-rated movies from the rating dataset. Then, we chose the highest-rated movie from those ten movies. The top-rated movie for user Id 1 is shown in Figure 10. Then we searched for the cluster in which the movie resides. Finally, we have shown

```
val recent = ratings.where(col("userId").equalTo(1)).sort(desc("timestamp")).limit(10)
val top = recent.sort(desc("rating")).limit(1)
top.show()
```

```
+------+-------+------+----------+----------------+
|userId|movieId|rating| timestamp|           title|
+------+-------+------+----------+----------------+
|     1|   8132|   5.0|1225865124|Gladiator (1992)|
+------+-------+------+----------+----------------+
```

Figure 10: Top rated recently watched movie for user Id 1

the top ten movies based on the highest average rating, and the user number rated that movie for movie recommendation for that user. Figure 11 shows the recommended movies for the user.

K Means Clustering

title	AverageRating	UserCount
Planet Earth (2006)	4.448092868988391	3015
Band of Brothers (2001)	4.423985890652557	2835
Whiplash (2013)	4.231990831696136 5	3054
Paths of Glory (1957)	4.180317630264097	5604
Your Name. (2016)	4.16751269035533	3940
Yojimbo (1961)	4.167050691244239	5208
Thin Man, The (1934)	4.139336016096579	3976
When We Were Kings (1996)	4.123484668409793	4207
It Happened One Night (1934)	4.115296585924089 5	5243
Raise the Red Lantern (Da hong deng long gao gao gua) (1991)	4.112258233141662	3826

GMM Clustering

title	AverageRating	UserCount
Final Recourse (2013)	5.0	3
Ciao Ni! (1979)	5.0	3
Who Killed Chea Vichea? (2010)	5.0	2
Look At Me: XXXTENTACION (2022)	5.0	2
Fatal Memories (1992)	5.0	2
Patterns of Evidence: Journey to Mount Sinai II (2023)	5.0	2
One Track Heart: The Story of Krishna Das (2013)	5.0	2
The Light in the Forest (1958)	5.0	2
One Winter Proposal (2019)	5.0	2
American Gospel: Christ Alone (2018)	5.0	2

LDA Clustering

title	AverageRating	UserCount
Planet Earth II (2016)	4.451739343459089	2041
Planet Earth (2006)	4.448092868988391	3015
Band of Brothers (2001)	4.423985890652557	2835
James Acaster: Cold Lasagne Hate Myself 1999 (2020)	4.421052631578948	19
Shawshank Redemption, The (1994)	4.416792045528881	122296
The Work of Director Chris Cunningham (2003)	4.395833333333333	24
Cosmos	4.3432	625
Come From Away (2021)	4.342105263157895	19
Parasite (2019)	4.329945963384143 5	12399
Godfather, The (1972)	4.32660258119567	75004

Figure 11: Recommended 10 movies for user 1 based on different clustering algorithms

5 Conclusion

In this study, we have implemented a movie recommendation system with clustering algorithms. We have used the MovieLens Dataset to train our model and provide recommendations based on the model. We filtered the user's top ten latest-rated movies from the rating dataset. Subsequently, we selected the highest-rated movie among these ten. Next, we identified the cluster to which the chosen movie belongs. The resulting cluster information guided the presentation of the top ten movies, determined by the highest average rating and the user count that rated the recommended movie for personalized movie recommendations. We have used three different clustering algorithms in our project. They are K means clustering, Gausian Mixture Model, Latent Dirichlet Allocation model. We have seen that based on three different movies, we get various movie recommendations with a few similar movies or even no similar ones. From the evaluation matrix with training time and silhouette co-efficient, we can say that although Latent Dirichlet Allocation takes less time compared to K means clustering and Gausian Mixture Model, yet K means the clustering algorithm creates better clusters than both Gaussian mixture and latent Dirichlet allocation models. In our future work, we can use the intersection result of these different kinds of clustering algorithms results and show them as recommended movies.

We can also use different kinds of feature selection models for feature extraction and train our model with other clustering and prediction algorithms for a better movie recommendation system.

References

[1] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.

[2] M. Goyani and N. Chaurasiya, "A review of movie recommendation system: Limitations, survey and challenges," *ELCVIA: electronic letters on computer vision and image analysis*, vol. 19, no. 3, pp. 0018–37, 2020.

[3] R. Singh, S. Maurya, T. Tripathi, T. Narula, and G. Srivastav, "Movie recommendation system using cosine similarity and knn," *International Journal of Engineering and Advanced Technology*, vol. 9, pp. 2249–8958, 06 2020.

[4] B.-B. Cui, "Design and implementation of movie recommendation system based on knn collaborative filtering algorithm," in *ITM web of conferences*, vol. 12. EDP Sciences, 2017, p. 04008.

[5] R. Ahuja, A. Solanki, and A. Nayyar, "Movie recommender system using k-means clustering and k-nearest neighbor," in *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. IEEE, 2019, pp. 263–268.