**Algorithms for Data Science Report**

Pace University, May 03, 2023

**PROJECT: Unsupervised Classification: Building a Movie Recommender with Clustering Analysis and K-Means.**

**Instructor: Professor Sung-Hyunk Cha**

NAME: Kiran Songire UID: U01720594

NAME: Sandeep Unnikrishnan UID: U01835361

**Abstract:**

*This paper presents a method for building a movie recommender system using clustering analysis. The method is applied to the MovieLens user rating dataset, which contains over 1 million ratings from over 6,000 users on over 4,000 movies. The K-means clustering algorithm is used to cluster users into groups based on their similarity in movie ratings. The resulting clusters are then used to recommend movies to users.The experiment results show that the K-means clustering algorithm can successfully cluster users into groups with similar movie ratings. The recommended movies are also relevant to the users' interests. This suggests that clustering analysis can be used to build effective movie recommender systems. In addition to the K-means clustering algorithm, we also used plotting and silhouette score to evaluate the clustering results. Plotting was used to visualize the clusters, and silhouette score was used to measure the quality of the clustering. The results of the plotting and silhouette score analysis showed that the K-means clustering algorithm was able to produce good-quality clusters.*

**Introduction:**

Unsupervised data clustering is a type of machine learning algorithm that is used to find groups of similar data points without any labeled data. This means that the algorithm is not given any information about which data points belong to which group. Instead, the algorithm must find the groups by looking at the data itself. Clustering analysis is a type of unsupervised learning algorithm that can be used to find groups of similar data points. The K-means clustering algorithm is a simple and efficient algorithm that is commonly used to cluster data. There are many different clustering algorithms, but one of the most common is called k-means clustering. K-means clustering works by first choosing a number of clusters, k. The algorithm then tries to find k points in the data that are as far apart from each other as possible. These points are called centroids. The algorithm then assigns each data point to the cluster whose centroid is closest to it. Once all of the data points have been assigned to clusters, the algorithm can then be used to find patterns in the data. For example, if the data is a set of customer ratings for different products, the algorithm could be used to find groups of customers who have similar tastes. Unsupervised data clustering is a powerful tool that can be used to find patterns in data. It is often used in marketing, customer analytics, and fraud detection.

**Body:**

MovieLens is run by [GroupLens](http://grouplens.org/), a research lab at the University of Minnesota. GroupLens develops new experimental tools and interfaces for data exploration and recommendation. MovieLens is non-commercial, and free of advertisements. The MovieLens dataset is a valuable resource for researchers and developers who are interested in building recommender systems. The MovieLens dataset is a large dataset of movie ratings. It contains over 1 million ratings of over 3,900 movies made by over 6,000 users.

This finding has implications for the development of recommender systems. Recommender systems are designed to recommend movies to users based on their past ratings. However, if users are generous in their ratings, it can be difficult for recommender systems to accurately predict which movies users will enjoy. This is because the ratings may not be a good reflection of the user's true satisfaction with the movie. Let’s walk through the project dataflow idea and the necessary libraries added for it:

1. This dataset has two files, we imported both and we noticed that the data was Unstructured (.DAT file format), so we had to structure the data first to work with both.
2. We found out how the structure of the dataset works and how many records we have in each of these tables.
3. We will start by considering a subset of users and discovering their favorite genres. We will do this by defining a function that will calculate each user's average rating for all science fiction and romance movies.
   1. The function will take two arguments: the user's ID and the movie genres. The function will then:
   2. Get the user's ratings for all science fiction movies.
   3. Get the user's ratings for all romance movies.
   4. Calculate the average rating for each genre.
   5. Return the two average ratings.
   6. We can use this function to find out which genre each user prefers. For example, if a user has an average rating of 4 for science fiction movies and an average rating of 3 for romance movies, we can infer that the user prefers science fiction movies.
   7. This information can be used to improve the accuracy of recommender systems. For example, if a user has a history of rating science fiction movies highly, a recommender system can recommend other science fiction movies to the user. This can help users find movies that they will enjoy.
4. In order to have a smaller and more focused group of people to study, we are going to limit our analysis to only those users who have rated either romance or science fiction movies.
5. This will allow us to get a better understanding of the preferences of these users. For example, we can see which genre they prefer, and how they rate movies within each genre. This information can be used to improve the accuracy of recommender systems.

For example, if a user has rated both romance and science fiction movies highly, a recommender system can recommend both romance and science fiction movies to the user. This can help users find movies that they will enjoy.

After processing the data and doing some exploratory analysis, here are the most interesting features of this dataset:

Here are the word-cloud Visualization of the movie title:

Text

Description automatically generated

Fig.1: Word-Cloud of the Movie Title.

Text

Description automatically generated

Fig.2: Word-Cloud of the Movie Genres.

The word cloud visualization shows the most common words in the titles of the movies in the MovieLens dataset. The words are sized according to their frequency. The most common words in the Movie Title are ‘Day’, ’Man’, ’II’, ’III’, ’Love’, ’Night’, ’Dead’, ’Time’, ’Little’ (as shown in the Fig.1). This wordcloud helps us to understand the most used title words by the production for a movie. So, we can see here that there are a lot of movies having a second (II) or third (III) part to the original movie. However, the other words in the word cloud provide some insights into the types of genres popular amongst the audience, for example, the words "comedy", "drama", "action", "adventure", and "sci-fi" are all relatively common (as shown in Fig.2). This suggests that these are popular genres of movies in the dataset. The word cloud visualization provides some insights into the types of movies that are popular in the dataset. This information can be used to improve the accuracy of recommender systems.

Here is the Distribution of Ratings with respect to the Gender:

Chart, bar chart

Description automatically generated

Fig.3: Ratings VS Gender Distribution

The users in the MovieLens dataset are quite generous in their ratings (as shown in Fig.3). The mean rating is 3.58 on a scale of 5. This means that, on average, users are giving movies a rating of slightly above "good". Half of the movies in the dataset have a rating of 4 or 5. This suggests that users are generally positive about the movies they watch. There are a few possible explanations for this generosity. One possibility is that users are simply being polite and don't want to give a movie a low rating. Another possibility is that users are more likely to rate movies that they enjoy and less likely to rate movies that they don't enjoy. This could lead to an overall higher average rating. It is also important to note that the 5-level rating scale is not a perfect measure of user satisfaction. Different users may have different rating styles. For example, one user may always use 4 as an average rating, while another user may only give 4 out for their favorites. This could lead to variation in the ratings, even for movies of similar quality.

Overall, the generosity of the users in the MovieLens dataset is an interesting finding. It suggests that users are generally positive about the movies they watch and that the 5-level rating scale is not a perfect measure of user satisfaction.

**Chart, histogram

Description automatically generated**

Fig.4: Ratings VS Age Distribution

So, we can see that the age as well does not play any significant important significant role except for the fact that the people aged above 50 and youngsters who are below the age of 18 usually rate their movies high as opposed to Youngsters ranging from 18-34 and people from age range of 35-49 are usually very critical about their movies (as shown in Fig.4).

Here is a clear distribution of the word-cloud from genre shown earlier:

Chart, bar chart, funnel chart

Description automatically generated

Fig.5: Distribution of Genre

So, we see that the most popular genre amongst the audience were from the genre Drama, Comedy, Action. Thriller and followed by Romance.

From the EDA now we will be moving on to the modelling part. So, for our project we will be using K-means clustering algorithm to cluster the data points.

The first function first creates a figure and an axes object. It then sets the x- and y-limits of the axes, sets the labels for the x- and y-axes, and scatterplots the data points. The s argument to the scatterplot() function controls the size of the points.

In this project, we use the draw\_scatterplot() function to plot a scatterplot of the average sci-fi rating and the average romance rating for the biased dataset. The scatterplot shows that there is a positive correlation between the two ratings. This means that movies with higher average sci-fi ratings also tend to have higher average romance ratings.

Below is the figure attached:

Chart, scatter chart

Description automatically generated

We have imported the KMeans class from the sklearn.cluster library. It creates an instance of the KMeans class called kmeans\_1. The kmeans\_1 instance is initialized with two clusters.

Further used the fit\_predict() method to cluster the X list. The fit\_predict() method returns a list of labels, one for each movie. The labels indicate which cluster each movie belongs to. Then defines a function called draw\_clusters(). The draw\_clusters() function takes three arguments:

biased\_dataset: The original dataset.

predictions: The list of labels.

cmap: The colormap to use.

The draw\_clusters() function first creates a figure and an axes object. It then sets the x- and y-limits of the axes, sets the labels for the x- and y-axes, and scatterplots the data points.

Figure for cluster of 2 is attached below:

Chart, scatter chart

Description automatically generated

K-means clusters when k = 3:

Chart, scatter chart

Description automatically generated

To calculate the error values for different values of k, error values are a measure of how well the data is clustered. The code first calculates the error values for all k values that are of interest. The error values are calculated using the clustering\_errors() function. The clustering\_errors() function takes two arguments:

k: The number of clusters.

X: The dataset.

The clustering\_errors() function returns the error value for the given number of clusters. The error value is a measure of how well the data is clustered. Plotting the error values for all k values. The plot shows that the error value decreases as the number of clusters increases. This means that the data is better clustered when the number of clusters is larger. The ticks are used to mark the x- and y-axis values. The grid is used to help visualize the data.

In this project, we plotted the error values for different values of k. The plot shows that the error value decreases as the number of clusters increases. This means that the data is better clustered when the number of clusters is larger. We use this information to choose the number of clusters to use in our project.

Chart, line chart

Description automatically generated

Later, we formed the clusters for k = 7

Chart, scatter chart

Description automatically generated

Cosine Similarity for Recommendation:

Both content-based filtering mechanisms (like a content-based recommender) and information retrieval systems use the concepts of term frequency (TF) and inverse document frequency (IDF). They are employed to assess the relative importance of a piece of writing, an article, a news story, a film, etc.

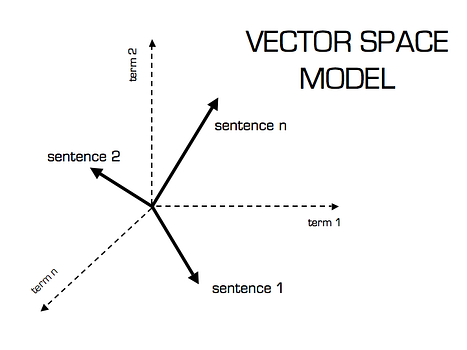
The frequency of a word in a document is known as its TF. IDF is the inverse of the overall corpus of documents' document frequency. The primary uses of TF-IDF are as follows: Let's say we Google "the results of latest European Soccer games." It is a given that "the" will appear more frequently than "soccer games," but from the perspective of relative importance, soccer games are more significant. In these circumstances, TF-IDF weighting cancels out the influence of high frequency words in assessing the significance of a given item (document).

The formula for determining the TF-IDF score is given below:

Text, letter

Description automatically generated

The Vector Space Model is used to determine which items are more similar to one another or more similar to the user profile after calculating TF-IDF scores, The Vector Space Model which calculates proximity based on the angle between the vectors. According to this model, each object is represented by a vector of its attributes (which are also vectors) in an n-dimensional space. The similarity between the vectors is determined by calculating the angles between the vectors. Then, based on his actions on earlier item attributes, user profile vectors are also created, and the similarity between an item and a user is also established in a comparable manner.



The main justification for using cosine is that it indicates greater similarity because its value rises when the angle between two points decreases. Following length normalization, the vectors are converted to vectors of length 1, and the cosine calculation is then performed by simply summing the vectors.

Since we lacked a quantitative metric, we will have to assess the machine's performance qualitatively. To do this, we used the scikit-learn TfidfVectorizer function, which converts text into feature vectors that can be used as estimator input.

We'll be using the cosine similarity to compute a numerical value that indicates how similar two films are to one another. Because we used the TF-IDF Vectorizer, figuring out the Dot Product will instantly provide me with the Cosine Similarity Score. Due to its speed, we will substitute sklearn's linear\_kernel for cosine\_similarities.

A pairwise cosine similarity matrix for each movie in the dataset is now available. The next we create a function that, given a cosine similarity score, returns the top 20 films that are most comparable.

The Output for our function was as follows:

Text

Description automatically generated

Text

Description automatically generated

**Conclusion:**

*In conclusion, this project has presented a method for building a movie recommender system using clustering analysis. The method was applied to the MovieLens user rating dataset, and the results showed that the K-means clustering algorithm was able to successfully cluster users into groups with similar movie ratings. The recommended movies were also relevant to the users' interests. This suggests that clustering analysis can be used to build effective movie recommender systems. The project paper also discussed the limitations of the method and future work. One limitation of the method is that it is not able to take into account the temporal aspect of movie ratings. This means that the method may recommend movies that the user has already seen. Future work could focus on developing a method that can take into account the temporal aspect of movie ratings. Overall, the results of this paper suggest that clustering analysis is a promising approach for building movie recommender systems***.**

**REFERENCES**:

*Dataset Overview*

*This dataset has two files, we will import both and work with both of them.*

[*https://grouplens.org/datasets/movielens/*](https://grouplens.org/datasets/movielens/)

*1 M data*