May 3, 2023

Final Report

**Clustering in Recommendation Systems**

1. **Introduction**

Clustering is a machine learning technique that helps us find a structure or pattern in a dataset, thereby better understanding it and making use of it. The goal of clustering is to divide a dataset into groups (clusters) so that objects in the same group are more similar to each other based on some characteristics even though we have no prior knowledge of the groupings or the members' characteristics in those groups.

In this report, I will try to create a simple film recommendation system through Python using k-mean clustering. The goal of the system is to collect a data source of movies’ ratings from users as well as the movies’ genres and provide film suggestions for them based on film rating’s similarity and genres’ similarity among users. In other words, to recommend films that the user would likely give it a good rating based on a user ratings and movies’ genres dataset.

1. **Approach**

In building this film recommendation system, I simplify my approach by only using two factors for the recommendation system, which are the users’ rating of the movies and the movies’ genres. I disregard other possibly influential information like movies’ popularity, casts, and directors.

**The approach:**

1) Pick movies that people tend to like and avoid those with low ratings. I will only pick movies with an average rating score of 3 (out of 5) or higher. We will call these movies “liked movies”. They are saved into a data frame called “liked\_movies\_csv”   
2) I will create D matrix (pivot table) which rows are “liked movies” and columns are genre affiliation. Its entries [Equation] are either 0 or 1, with 0 means that the movie i does not belong to genre j; and 1 means that the movie i does belong to genre j. If we run k-means algorithm on this, we will get “liked movies” divided into clusters and those in one cluster would have similar genre

3) The info of cluster that each liked movie belonged to will be added to the original “liked movie” data frame. To recommend movies to the user, we will create another data frame “users’ favorite” that contains user id and the movies id that they like (I consider a rating of 4 or 5 as a “like” from user). Then we will merge this “user’s favorite” data frame with the “liked movie” data frame.

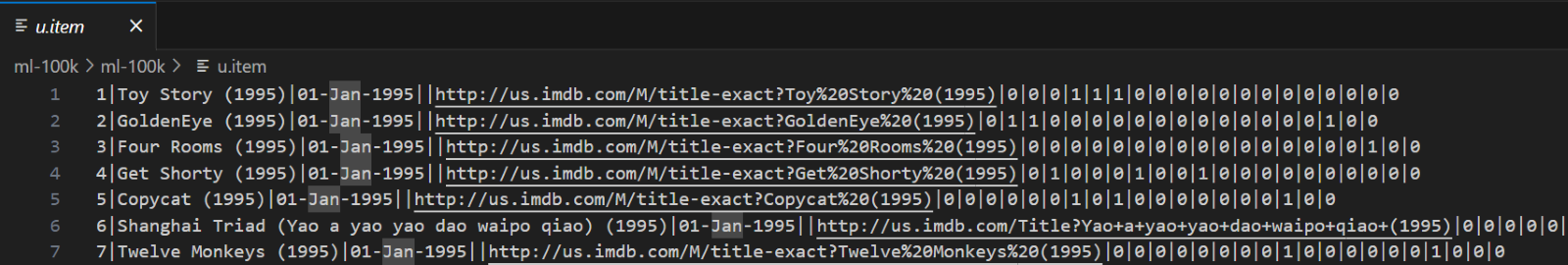
4) Based on this final “merge” data frame, I will write a function that determines the cluster that most “liked movies” of that user belong to. Then I will output 10 movies that also belong to that cluster.

1. **Application**

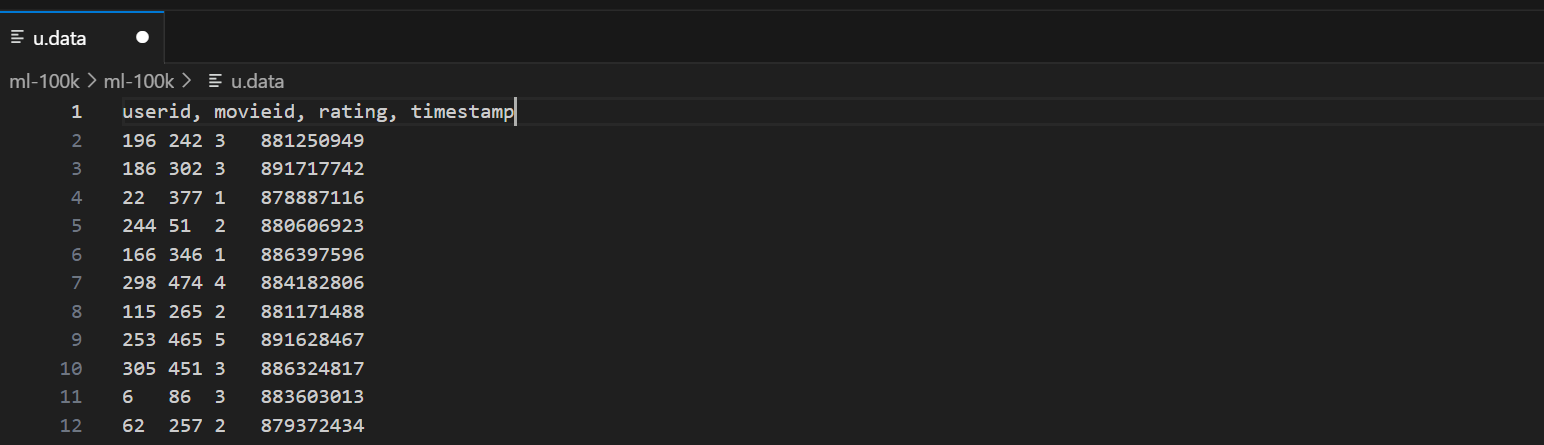
**1) Find, store, and upload the dataset**

I use Movielens100k which is a popular dataset of movie ratings. It contains 100,000 ratings from 943 users on 1,682 movies. The data set is widely used in academic research and industry applications.

**The tables I am making use of:**

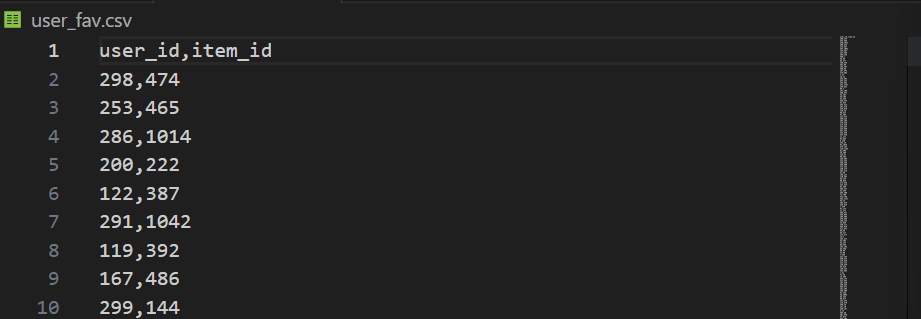
**‘u.items’** include information about the movies, including movie id, movie title, release date, IMDb URL, and genres. A snippet:   
 

**‘u.data**’ includes information of 10,000 ratings from 943 users on 1682 movies. Each row includes user id, movie id, user’s rating, and timestamp. A snippet:

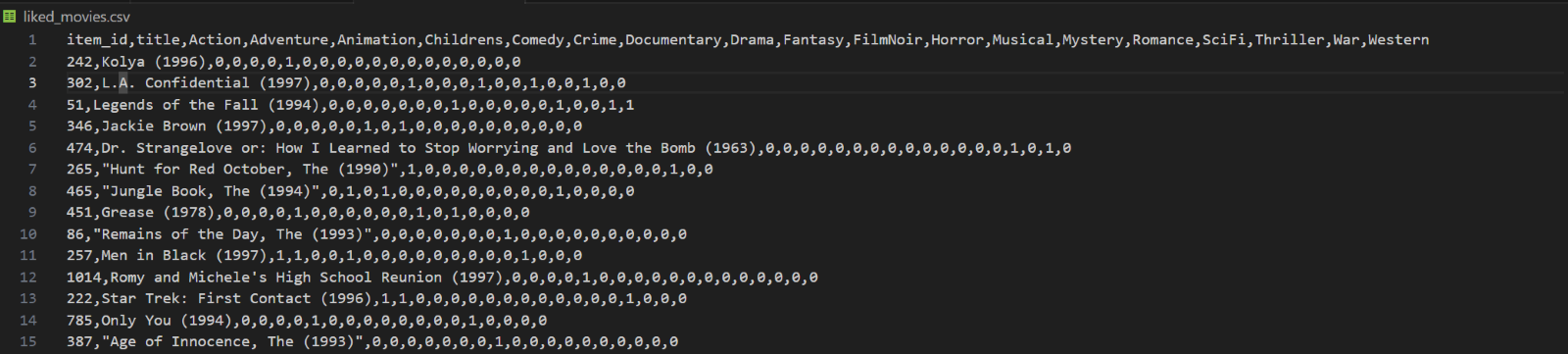


**2) Preprocess the data to produce desirable tables and matrix**

Using panda libraries and python functions, I created a file called ‘user\_fav.csv’. Each row of the file contains ‘user id’ and the ‘item id’ that they like. Here I determine that a movie is liked by a user if it receives a rating of 3 or higher. The data frame:

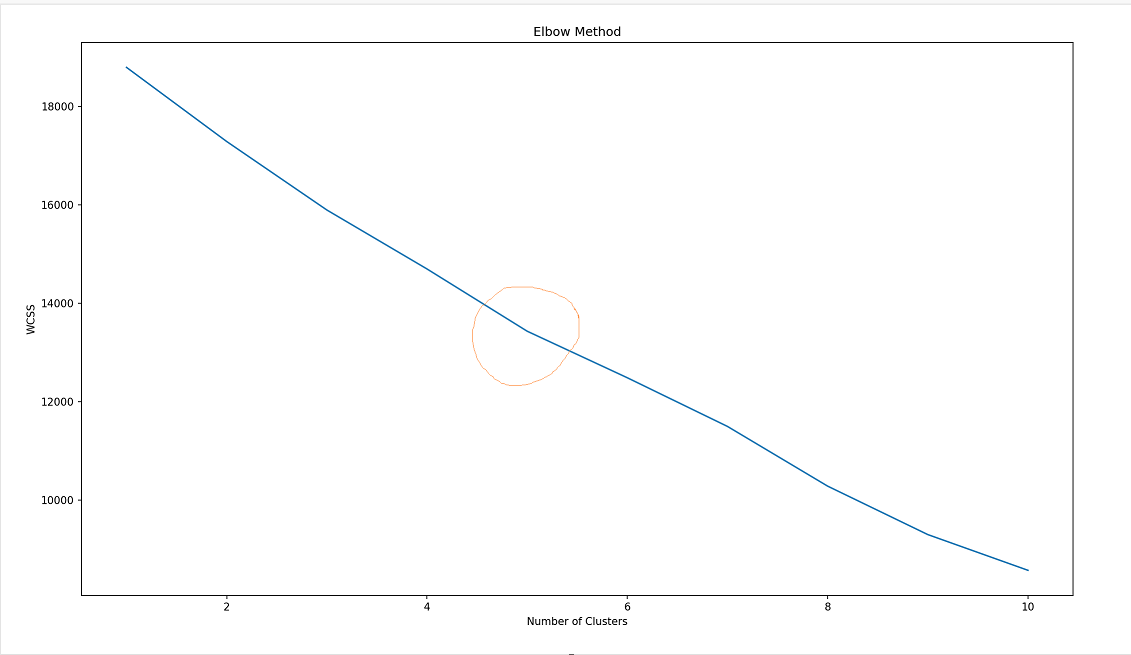


I also created a file called “liked\_movies.csv”. Movies that are included in this file receive an average rating of 3 out of 5 or higher from all users. Each row of the file contains movie information such as “item\_id”, “title”, and its genre. The data frame:



With these 2 files, I will now create a matrix D (called as a pivot table in Python) which rows are “liked movies” and columns are genre affiliation. Its entries are binary (either 0 or 1). With entry [Equation] = 0 means that the movie at row i does not belong to genre j; and [Equation] = 1 means that the movie at row i does belong to genre

**3) Use elbow method to find optimal number of clusters and run the K-means algorithm**

Before running the K-means algorithm, I need to determine an optimal number of clusters (optimal number of groups that I should divide into). Hence, I will use the Elbow Method. The first step is to plot the variance as a function of the number of clusters:   
 

Looking at the plot, we see that there is a point that indicates a change (level off) in variance. This point is called the “elbow point” and it shows that additional clusters beyond this point will not provide much more useful information in explaining the variance of the data. Hence, we choose the number of clusters at this point is 5.

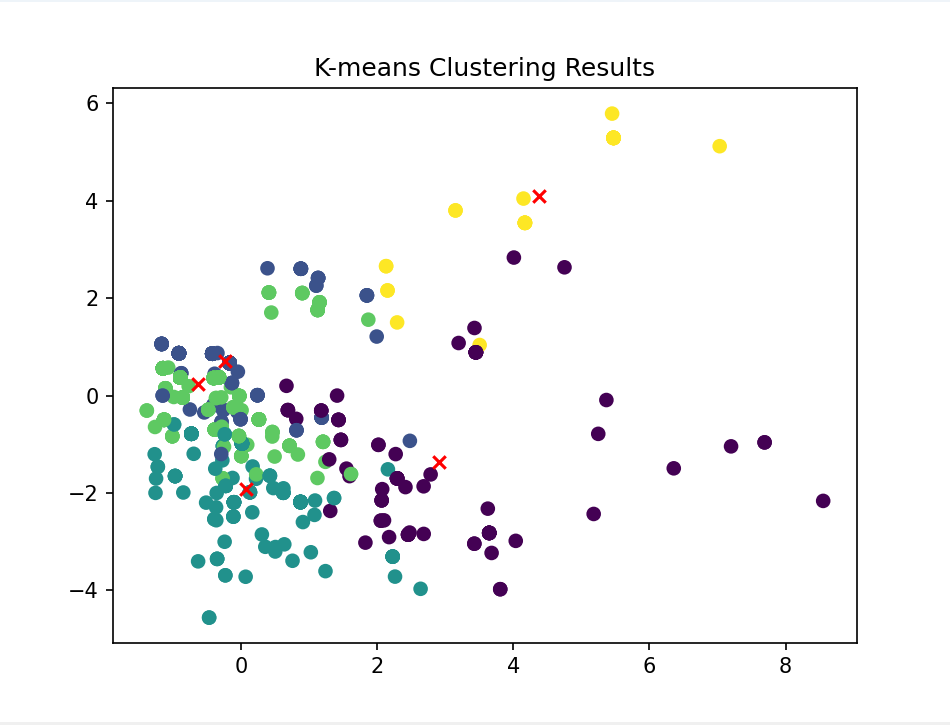
With k = 5, I will run the k-means algorithm on the matrix D (scaled) mentioned above

**4) Visualize the clustering results with PCA**

After running the K-means algorithm, we will have “liked movies” divided into clusters. Those movies in the same cluster would have similar genre

Our data frame currently has 18 movie genres, meaning that each row is an 18-vector. To represent these clustering results in a plot, we will need to use the Principal Component Analysis (PCA). This is a statistical method used to help reduce the dimensionality of a dataset by transforming data into principal components.

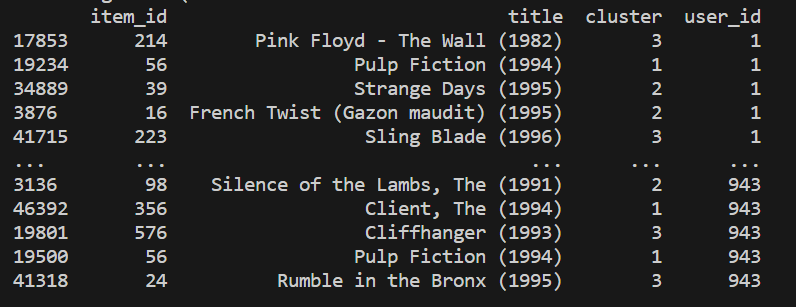
We get the following results: (the 5 red mark representing 5 centroids)



The info of cluster that each liked movie belonged to will be added to the original “liked movie” data frame.

**5) Merged the data frames for the recommendation function**

We will create another data frame “users ’favorite” that contains user id and the movies id that they like (Here, I consider a rating of 4 or 5 as a “like” from the user). Then we will merge this “user’s favorite” data frame with the “liked movie” data frame, now we have a “final merged” data frame:



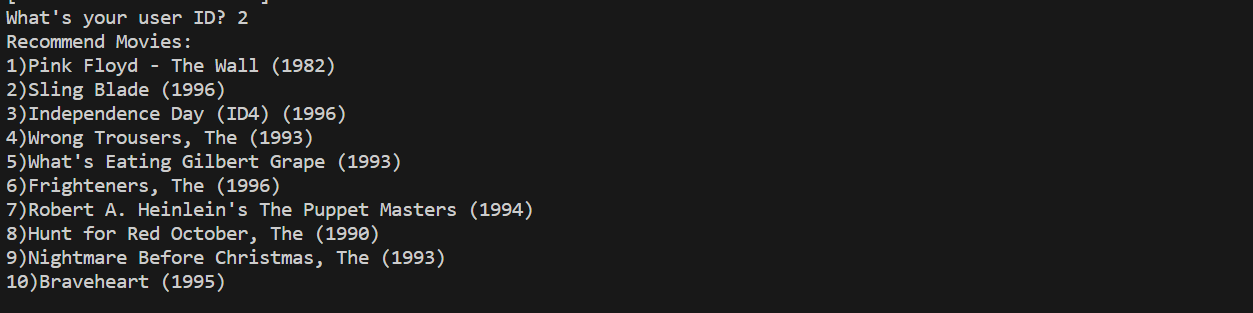
In the data frame above, we have the item\_id (movie id), title, user\_id, and the cluster the movie belongs to

.

**6) Create movie recommendation function**

To suggest movies for a user given their user-id, I will look into the record to see how many of their “liked-movies” belong to each cluster (1 to 5). I will then choose a cluster that has the highest number of that user’s “liked movies”. Then I will output 10 movies that also belong to that cluster.

Now, here is the result:



1. **Conclusion**

In this report, I attempted to create a movie recommendation system using the k-means algorithm. Two crucial indicators that I made use of are users’ ratings of movies and genre affiliations of each movie. Possible improvements in the future would be considering information like movies’ popularity, the casts, and the directors. There are definitely more complex algorithms to build the movie recommendation systems, but this report provides a simple, yet interesting way to apply k-means algorithms.

**Bibliography:**

“Content-Based Recommender with K-Means.” n.d. Rpubs.com. Accessed May 11, 2023.<https://rpubs.com/tim-dim/recommender>.

‌ “Movie Ratings and Recommendation Using KNN.” n.d. Kaggle.com. Accessed May 11, 2023.<https://www.kaggle.com/code/heeraldedhia/movie-ratings-and-recommendation-using-knn/notebook>.

‌ Taiwade, Anju, Nitish Gupta, Rakesh Tiwari, Shashi Kumar, and Upendra Singh. 2022. “Hierarchical K-Means Clustering Method for Friend Recommendation System.” IEEE Xplore. July 1, 2022.<https://doi.org/10.1109/ICICT54344.2022.9850852>.

Kleine, Daniel. 2023. “Detecting Knee- / Elbow Points in a Graph.” Medium. January 20, 2023.<https://towardsdatascience.com/detecting-knee-elbow-points-in-a-graph-d13fc517a63c>.

Saji, Basil. 2021. “K Means Clustering | K Means Clustering Algorithm in Machine Learning.” Analytics Vidhya. January 20, 2021.<https://www.analyticsvidhya.com/blog/2021/01/in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning/>?.

‌

‌

‌