

SCHOOL OF COMPUTER SCIENCES UNIVERSITI SAINS MALAYSIA

CDS503: Machine Learning Semester 2, 2020/2021

**FINAL REPORT**

*Group 03*

*Movie Recommender*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Matric No. | USM Email Addresses | Experiment Set |
| Chin Jun Hong | P-COM0007/21 | [tituschin@student.usm.my](mailto:tituschin@student.usm.my) | 1: Partitional clustering |
| Nur Aquilah Zulkifli | P-COM0088/21 | [aquilahzulkifli@student.usm.my](mailto:aquilahzulkifli@student.usm.my) | 2: Hierarchical clustering |
| Zhao Dan | P-COM0255/20 | [zhaodan@student.usm.my](mailto:zhaodan@student.usm.my) | 3: Generate association rules from the full data |
| Lee Kah Win | P-COM0230/20 | [leekahwin84000@student.usm.my](mailto:leekahwin84000@student.usm.my) | 4: Generate association rules from subsets of data |

Date of Submission

2 July 2021

**Task 2: Data Set Selection**

We selected [The Movies Dataset](https://www.kaggle.com/rounakbanik/the-movies-dataset) from Kaggle which consists of 7 CSV files, containing metadata for all 45,000 movies released on or before July 2017 that are listed in the Full MovieLens Dataset, and 26 million ratings from 270,000 users for all 45,000 movies. According to the author, the movies metadata are collected from [TMDB Open API](https://www.themoviedb.org/documentation/api), and the users’ ratings are obtained from the [Official GroupLens](https://grouplens.org/datasets/movielens/latest/) website. Since the attributes that we are interested in are stored in different CSV files, we decided to merge them together. We also briefly described the steps of data merging in data preparation. In the end, we will be using two datasets namely movies\_metadata\_cleaned.csv for clustering analysis, and ratings\_all.csv for association rules mining. The reasons why we need these two datasets are justified in problem definition and business application. The attributes of each dataset are shown as table below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Datasets** | **Attributes** | | | |
| movies\_metadata\_cleaned.csv  (6,919 instances,  12 attributes) | 1. belongs\_to\_collection  2. genres  3. id  4. popularity  5. production\_companies  6. release\_date  7. runtime | (object)  (object)  (int64)  (float64)  (object)  (datetime64)  (float64) | 8. title  9. vote\_average  10. vote\_count  11. directors  12. characters | (object)  (float64)  (float64)  (object)  (object) |
| ratings\_all.csv  (265,662 instances,  2 attributes) | 1. userId  2. movieId | (int64)  (object) | | |

**Task 3 Problem Framing**

Nowadays, the movie streaming service providers are competing intensely to improve their users watching experience, and meet users’ diverse preferences to retain users in their service environment. Therefore, it’s important for service providers to provide a great user experience, and recommend relevant contents to the users, or even predict what movies that the users are going to watch next. Any service providers that fail to do so might suffer from high customer defections, poor service rating or decrease of customer lifetime value (CLV).

How to provide a great user experience? Is it like only showing a search bar on the homepage? Is it like displaying all movies at once without proper labels? Definitely not! To provide a great user experience, the service providers should have a great understanding of their contents and group similar movies together before displaying them to the users.

In addition, some users would like to watch several movies in a row. How can service providers predict what their users are going to watch next and recommend relevant contents to them? Should they just randomly show other movies that belong to the same group with the previous movie? It makes sense but it’s not an ideal solution. To predict the movies that the users are going to watch next, the service providers should analyze the users watching histories, and infer the likely association patterns in the data. For example, service providers might find out that one who watched Avengers: Age of Ultron will likely to watch Avengers: Infinity War, so they can recommend the right content to the users, and improve overall user satisfaction.

Therefore, there are two machine learning problems that we are going to address in this project. Firstly, given the attributes of the movies, group the similar movies together by using clustering analysis. The movies\_metadata\_cleaned.csv will be used for clustering analysis because it contains a lot of interesting movie attributes such as genres, collections, main characters, etc. When users come in, the system will recommend a few movies from each cluster in the homepage. In addition, the system will recommend the closest movies that are within the same cluster to the users after they finished watching a movie. Secondly, given the users rating histories, identify the association rules among the movies to predict what movies the users are going to watch next. The ratings\_all.csv will be used for association rules mining because it contains the list of movies rated by each user in the sequence of time.

By applying the two algorithms above, we will be able to create a reliable movie recommendation system that can provide a better user experience. Hence, movie service providers will gain popularity by recommending the right movies, thus the user retention rate and the service rating will increase.

**Task 4: Data Preparation**

1. Movies Metadata (for Experiment Set 1 and Set 2)

We want to ensure that the movies that we recommended are of good quality, so we extracted the movies that satisfied certain conditions from the movies metadata. The movies must be an English movie with released status, should have an average vote score of 6 (out of 10) or higher, at least 20 vote counts, runtime between 60 and 200 minutes. Then, we dropped irrelevant attributes. The dropped attributes are: ('adult', 'budget', 'homepage', 'imdb\_id', 'original\_title', 'overview', 'poster\_path', 'revenue', tagline', 'video', 'status', 'production\_countries', 'original\_language', 'spoken\_languages'). We also converted attributes into correct data types, and dropped all null values and duplicates. Furthermore, we also extracted the information of collections, genres and production companies from the stringified JSON object. For the credits dataset, we are only interested in the directors and characters of each movie, and this information is stored in the format of stringified JSON objects. So, we first matched the ID of the credits dataset with the movies metadata, then we sorted all the instances by the ID in ascending order. After that, we extracted the directors and 3 main characters from the stringified JSON object, and merged them into the movies metadata, and exported it to movies\_metadata\_cleaned.csv. Lastly, we do one-hot encoding on the categorical attributes namely (‘*collection’, ‘genres’, ‘production companies’, ‘characters’, ‘directors’)*, and export it to movies\_metadata\_ohe.csv.

1. All Ratings Data (for Experiment Set 3)

We filtered out the ratings data based on the ‘movieId’ remaining in the preprocessed movies metadata, and the ratings should be at least 3.5 (out of 5). Then, the ratings data is sorted by ‘userId’ and ‘timestamp’ in ascending order to identify which movies the users will watch next. After sorting, the ‘userId’ and ‘movieId’ attributes are dropped. Next, we grouped ‘movieId’ by ‘userId’. Therefore, each user has a list of movies sorted according to the ‘timestamp’, and the data is exported to ratings\_all.csv. Before conducting the experiment, the ratings\_all.csv had been one-hot-encoded to convert the list of movies into a binary format.

1. Ratings Data Subsets (for Experiment Set 4)

We prepared 20 ratings data subsets and each subset is from a different genre (we have 20 genres in total). All the ratings data subsets are exported to their respective csv file. Before conducting the experiment, all the ratings data subsets had been one-hot-encoded to convert the list of movies into a binary format.

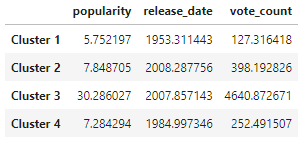
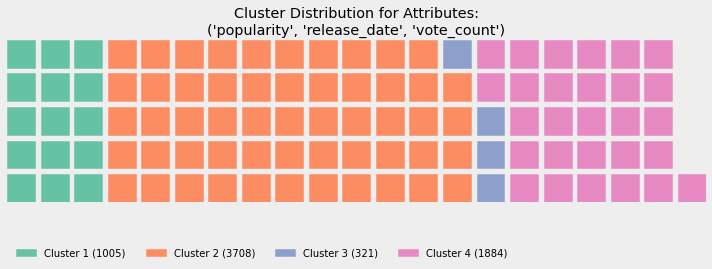
**Task 5: Experiment Setup**

**Experiment Set 1: Partitional clustering**

In this section, partitional clustering is performed on movie metadata to group similar movies together. Since movie metadata consists of mixed numerical and categorical attributes, and the distance measures for numerical and categorical attributes are different, the analysis is performed separately. The table below summarizes the experiment set up for partitional clustering:

|  |  |  |
| --- | --- | --- |
|  | Numerical Attributes | Categorical Attributes |
| Algorithm | K-means clustering | K-modes clustering |
| Selected combination | * popularity, release\_date, runtime, vote\_average, vote\_count * popularity, release\_date, vote\_average, vote\_count * popularity, release\_date, vote\_count | * collection, genre, company, director, character * collection, genre, director, character |
| Analysis for each combination | * Normalization using MinMaxScaler. * Plot line graph of Sum of Squared Error (SSE) against the number of clusters (k). * Select the best value of k using the elbow method. * Fit k-means clustering algorithm with the best value of k. * Waffleplot, scatterplot and centroid table are generated to analyze and interpret the clusters. | * Plot line graph of costs against the number of clusters (k). * Select the best value of k using the elbow method. * Fit k-modes clustering algorithm with the best value of k. * Waffleplot, wordclouds and top items table are generated to analyze and interpret the clusters. |

For numerical attributes, after running k-means clustering algorithms on three different numerical combinations, some results of the best k-means model are shown in the figure below.

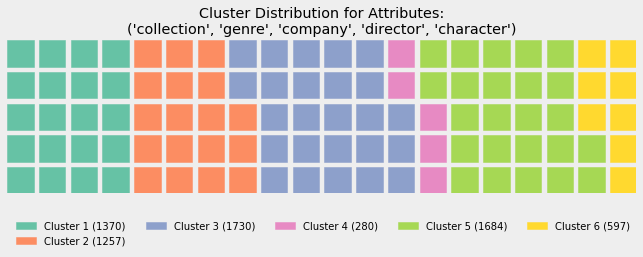


The best value of k is 4 according to the elbow method. Waffleplot shows that the cluster distribution is uneven, the smallest cluster has 321 movies whereas the largest cluster has 3708 movies. Based on the centroid table, the clusters can be described as follow (note that all movies in movie metadata are of high rating which is at least 6 out of 10):

* Cluster 1 - Extremely old but high rating movies
* Cluster 2 - Popular movies
* Cluster 3 - Extremely popular movies
* Cluster 4 - Old but popular movies

Based on the cluster description, although Cluster 3 has only 321 movies, all movies are extremely popular. This model is selected as the best k-means clustering model for numerical attributes because the result is more interpretable and more interesting. Therefore, the selected numerical attributes are popularity, release\_date and vote\_count. The number of k for the final k-means model is 4.

For categorical attributes, after running k-modes clustering algorithms on two different categorical combinations, some results of the best k-modes model are shown in the figure below.



The best value of k is 6 according to the elbow method. Wordclouds show that it is quite a clean split between each cluster because each cluster is highlighting different items. Waffleplot shows that the cluster distribution is quite even, ranging from 280 to 1730. The clusters can be described as follow:

* Cluster 1 - Thriller & Crime
* Cluster 2 - Action & Adventure
* Cluster 3 - Comedy & Drama
* Cluster 4 - Family & Animation
* Cluster 5 - Drama & Romance
* Cluster 6 - Horror & Mystery

This model is selected as the best k-modes clustering model for categorical attributes because the result is more interpretable, and it has more clusters for users to choose. Therefore, the selected categorical attributes are collection, genre, company, director and character. The number of k for the final k-modes model is 6.

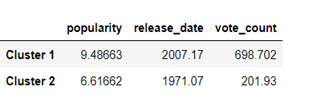
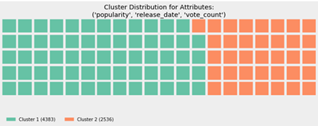
**Experiment Set 2: Hierarchical clustering**

For hierarchical clustering, dataset movies\_metadata\_ohe.csv was used. The file contains 6919 rows and 992 columns after being processed and containing both numerical and categorical attributes. The data was also divided into a few combinations.

For numerical, there are three selected combination which are:

* popularity, release\_date, runtime, vote\_average, vote\_count
* popularity, release\_date, vote\_average, vote\_count
* popularity, release\_date, vote\_count

MinMaxScaler was used to normalize the numerical attributes. Then, the selected numerical combination was tested to find the best agglomerative clustering model. We selected the best number of clusters by referring to dendrograms. To visualize, analyze and interpret the clusters, waffle plot, scatterplot and centroid table are generated. After running agglomerative clustering algorithm on three different combinations of numerical attributes, we can see the best result as per below:



The third clustering model was selected as our best model because the results are more interpretable and have sufficient movies for each cluster. Based on the waffle plot, it shows that the clusters are evenly distributed which range from 2536 to 4383. Cluster 1 represents the movies that were released recently, having high popularity, vote count and ratings. Cluster 2 represents the old movies with high ratings, but not gaining much attention by the users. The selected numerical attributes are popularity, release\_date and vote\_count. For the final parameters for the agglomerative clustering, the number of clusters is 2, linkage is complete and affinity is Manhattan.

For categorical, there are two selected combinations, which are:

* collection, genre, company, director, character
* collection, genre, director, character

Since the data is too big, we cannot generate dendrograms for categorical attributes. So, a trial-and-error method is applied to find the best number of clusters. In addition, complete linkage and hamming distance was applied to perform clustering analysis to the selected combination. We need to calculate the hamming distance of the selected categorical combination first before fitting the agglomerative algorithm with the hamming distance matrix. Next, we interpreted the result by using waffle plot, scatterplot and top item table.

After running agglomerative clustering algorithm on two different combinations of categorical attributes the best result is as per below:



After trial and error, the best number of clusters for this combination is 2. Based on the waffle plot, the distribution of these two clusters is even and both are having enough movies. In addition, the split is quite clean based on the word cloud and the results are very interpretable even though the clusters produced from this model are not much.

As a conclusion, the second clustering model is selected as our best model because the results are more interpretable. The selected categorical attributes are collection, genre, director, and character. For the final parameters for the agglomerative clustering the number of clusters is 2, linkage is complete and for affinity is hamming distance. Therefore, the final cluster description chosen as below:

* Cluster 1 - Drama & Comedy & Thriller
* Cluster 2 - Drama & Adventure & Action

**Compare partitional clustering with hierarchical clustering**

Since the best number of clusters produced by agglomerative clustering is only 2 for both numerical and categorical attributes, the interpretation of clusters is limited, and there will be too few clusters for users to choose. Hence, it is less interesting compared to the clusters produced by k-means and k-modes clustering. Therefore, the best models for numerical and categorical attributes are k-means model with k = 4 and k-modes model with k = 6 respectively. By grouping movies into these clusters, it could give new insights to the users. For example, some users might be curious about how good those old and classic movies are. Out of curiosity, they might start watching the movies in the cluster of "extremely old but high rating movies". However, some people might prefer certain genres. In that case, they can click on the cluster that suits them the best such as "Horror & Mystery", "Family & Animation", etc. In addition, two functions are defined to recommend 5 closest movies within the same cluster to the users after they finish watching a movie. For numerical attributes, Euclidean distance is used to recommend the closest movies whereas hamming distance is used for the categorical attributes.

**Experiment Set 3: Generate association rules from the full data**

Ratings\_all.csv, which includes all the rating information, has been selected as the dataset. There are 265,662 instances with 6,893 columns after one-hot encoding, and 842 frequent itemset were found. The total number of association rules is 2816, and 562 were finally selected by using support>=0.07, lift>=2 and confidence>=0.7. And then, we explored 5 aspects. They are movie titles, directors, production companies, popularity, and characters. Though some interesting insights have been seen, only a few of them will be given below.

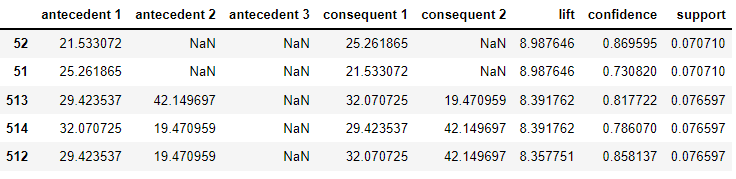
After replacing the Movie ID with Movie title, I discovered that if customers watch a movie from a series, then they also tend to watch another movie from the same series, no matter how many movies there are in the series. Though in the general case, we predict that people will watch all the movies in a series, they actually only tend to watch two of them. The system will recommend another movie from the same series, if a client has already watched a movie from that series.

|  |  |  |
| --- | --- | --- |
| Antecedent | Consequence | Lift |
| Kill Bill Vol.2 | Kill Bill Vol.1 | 8.987 |
| Aliens (1986) | Alien | 6.722 |

Next, after replacing the Movie ID with Movie title, I found that if customers watch movies from a director, then they tend to watch other movies directed by the same director, though customers also watch movies from other directors. The system will recommend the movies directed by the same director, if a client has already watched a movie from that director.

|  |  |  |
| --- | --- | --- |
| Antecedent | Consequence | Lift |
| Quentin Tarantino | Quentin Tarantino | 8.988 |
| Peter Jackson  George Lucas | Peter Jackson  Irvin Kershner' | 8.392 |

Furthermore, after replacing the Movie ID with the popularity values. I discovered that if people watch a movie with popularity values from 20 to 40, then they tend to watch other movies with popularity values from 20 to 40. The system will be prioritized to recommend the movies with popularity values between 20 and 40, if a client has already watched a movie from 20 to 40.



**Experiment Set 4: Generate association rules from selected attributes**

The selected attribute to generate association rules is “Genre”, which consist of 20 categories. Genre is chosen because it can show different thematic categories for each movie. Based on the constant parameters of support >= 0.07; confidence >= 0.7 and lift >= 2, there are 10 genre subsets that can generate association rules, but another 10 genre subsets cannot generate association rules. Table below summarised Genre Subsets with and without Association Rules.

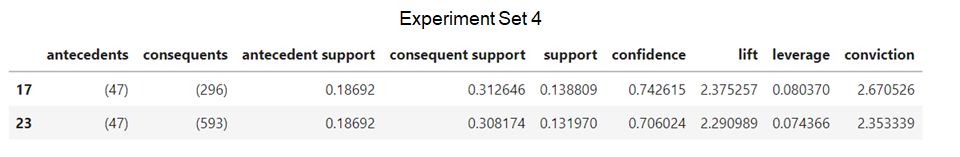
|  |  |  |  |
| --- | --- | --- | --- |
| **Genre Subsets that Generate Association Rules** | | **Genre Subsets that do not Generate Association Rules** | |
| 1. Action 536 Rules  2. Science fiction 326 Rules  3. Adventure 255 Rules  4. Crime 111 Rules  5. Drama 55 Rules | 6, Thriller 44 Rules  7. Animation 14 Rules  8. Fantasy 9. Rules  9. Horror 5 Rules  10. Family 1 Rule | 1. Comedy  2. Documentary  3. Foreign  4. History  5. Music | 6. Mystery  7. Romance  8. TV Movie  9. War  10. Western |

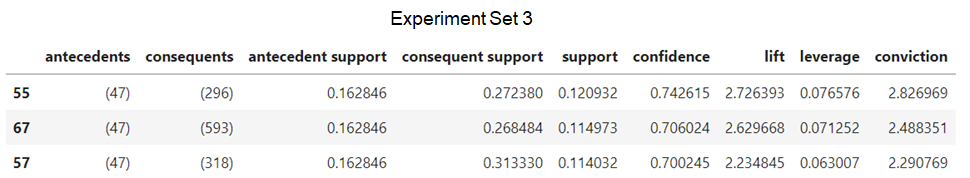
It is expected that the horror movies generate only a few rules because it will scare some audiences away from continuing watching horror movies. But it is quite unexpected to see drama movies, which have the largest collection of movies in the dataset, but only generate 55 rules. It is nearly 10 times less than action movies. On the other hand, it is also quite surprising to notice comedy movies do not generate association rules (comparatively horror movies at least generate 5 rules). Apart from that, I also notice an interesting finding in this experiment when one antecedent to one consequent for movie series, audiences who watched a movie that belongs to a series will more likely to watch an earlier version of movies (i.e. generates higher lift), rather than later version. The observations are listed below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Genre** | **antecedent** | **consequent** | **lift** | **Interesting Findings** |
| Drama  Fantasy  Horror | The Godfather: Part II  The Lord of the Rings (2002)  Aliens (1986) | The Godfather  The Lord of the Rings (2001)  Alien (1979) | 4.5  3.5  3.1 | Higher lift than consequent of Part 3  Higher lift than consequent of the last series in (2003)  Higher lift than consequent of Alien 3 in (1992) |

In business applications, we normally assume when audiences finish a movie that belongs to a series will continue to watch a subsequent movie. But the data-driven insights tell us that, in some scenarios, it can be the earlier version of movie series audiences will be more likely to continue watching. The 10 genre subsets with association rules are summarized in this [link](https://docs.google.com/spreadsheets/d/1VxiYVrrwq3y2SM7EhnPdAvIKI_1X_t2mEPM_dYqxfVY/edit?usp=sharing); and this [workbook](https://docs.google.com/spreadsheets/d/1rzt4CO9F0YhW_Y_UtxPanfMBwEj3JS-gg0U4NXQuDqg/edit?usp=sharing) summarized all the rules with replaced titles and sorted lift in descending order.

In comparison, let's say an audience has watched “Se7en” (Movie Id =47), which belongs to a thriller subset. Based on this experiment set 4, it can generate 2 rules in a setting of one antecedent to one consequent, which are consequents of “Pulp Fiction” (296) and “The Silence of the Lambs” (593). Both the consequents are belonging to thriller subsets. In comparison with experiment set 3, it can generate similar consequents as set 4, but provide one additional rule, which is “The Shawshank Redemption” (318). The additional consequent does not belong to the thriller subset, but to the drama subset.





**Conclusion**

K-Means and K-Modes clustering are chosen to cluster similar movies together. By having these clusters, it helps service providers to manage their contents more efficiently, and it also enables users to choose their favorite movies easily. Furthermore, by using association rules, we discovered a lot of rules with high lift that can be used to recommend the right movies to the users. So, the overall user experience will improve. The figure below shows a movie system interface that applies clustering and association rules to an audience who has watched "The Sixth Sense".

