*Subject:*

**COMP4434 (Big Data Analytics)**

THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學

*Topic:*

# PolyTube Recommender Systems

*Name & Student ID:*

**Andrew Thenedi**

*Presentation Video Link:*

**https://drive.google.com/file/d/1eTB_RBybX-DSMJplYGBXspAbdMl1ZUMi/view?usp=sharing**

# Introduction

This project aims to predict teleplay ratings for the Teleplay dataset, in the hope to improve PolyTube systems with the knowledge of COMP4434 (Big Data Analytics). The data preprocessing and analytics will be conducted in advance, to find meaningful insights and remove or replace any irrelevant information.

# Data Preprocessing

Import Libraries

- ➢ NumPy
- ➢ Pandas
- ➢ Matplotlib
- ➢ Seaborn
- ➢ scikit-learn
- ➢ Tensorflow

Data Cleaning

- ➢ Get and Preprocess the Training Dataset ('Teleplay.csv' and 'Rating.csv') as 'df_teleplay' and 'df_rating' data frames.
    - ○ Remove 'rating' or 'members' column values less or equal to 0, in the 'df_teleplay' data frame.
    - ○ Remove 'rating' column values less or equal to 0, in the 'df_rating' data frame.
    - ○ Find the mean ratings for each 'teleplay_id' column values in the 'df_rating' data frame.
        - ■ Merge such values in the 'df_teleplay' data frame on the 'rating_2' column.
    - ○ Check empty values existence in 'df_teleplay' data frame, per column.
        - ■ In this case, there exists an empty value in the 'genre' and 'rating_2' column values.
    - ○ Drop the empty values of the 'genre' column in the 'df_teleplay' data frame.
        - ■ Check empty values existence in the 'df_teleplay' data frame, per column.
            - ● There is an empty value in the 'rating_2' column values in this case.
- ➢ Get and Preprocess the Testing Dataset 'New_Teleplay.csv' as 'df_new_teleplay' data frame.
    - ○ Merge the mean ratings for each 'teleplay_id' column values in 'df_rating' dataframe values in 'df_new_teleplay' dataframe on 'rating_2' column.
    - ○ Check empty values existence in 'df_new_teleplay' dataframe, per column:
        - ■ In this case, there exists an empty value in 'genre' , 'type', 'rating' , and 'rating_2' column values.

    - Replace empty values in 'df_new_teleplay' dataframe with "nan" string datatype, except 'rating' and 'rating_2' column values.
        - Check empty values existence in 'df_new_teleplay' dataframe, per column.
            - In this case, there exists an empty value in the 'rating' and 'rating_2' column values.

Predict the empty values of 'rating_2' column from 'df_teleplay' and 'df_new_teleplay' dataframes

- Split 'df_teleplay' dataframe into 'df_teleplay_train_predict_rating_2' dataframe as training dataset and 'df_teleplay_test_predict_rating_2' dataframe as test dataset.
    - Reset index of 'df_teleplay_train_predict_rating_2' and 'df_teleplay_test_predict_rating_2' dataframes.
- Split 'df_new_teleplay' dataframe into 'df_new_teleplay_train_predict_rating_2' dataframe as training dataset and 'df_new_teleplay_test_predict_rating_2' dataframe as test dataset:
    - Reset index of 'df_new_teleplay_train_predict_rating_2' and 'df_new_teleplay_test_predict_rating_2' dataframes.
- Encode and Normalize the values of 'type', 'genre' , and 'members' columns:
    - Concatenate 'df_teleplay_train_predict_rating_2' , 'df_teleplay_test_predict_rating_2' , 'df_new_teleplay_train_predict_rating_2' and 'df_new_teleplay_test_predict_rating_2' dataframes as 'df_merge_encode_normalize' dataframe, for 'rating' , 'rating_2' , 'members' , 'type', and 'genre' column values.
        - Reset index of 'df_merge_encode_normalize' dataframe.
        - Normalize 'rating' , 'rating_2' , and 'members' columns values in 'df_merge_encode_normalize' dataframe, which automatically skip the empty values.
        - Encode 'type' and 'genre' columns values in 'df_merge_encode_normalize' dataframe.
    - Merge normalized 'members' , 'rating' & 'rating_2' and encoded 'type' and 'genre' columns values of 'df_merge_encode_normalize' into **'arr_merge_encode_normalize'** nparray.
        - Split **'arr_merge_encode_normalize'** nparray into *'x_noNA_teleplay'*, *'x_NA_teleplay'*, *'y_noNA_teleplay'*, *'y_NA_teleplay'*, *'x_noNA_new_teleplay'*, *'x_NA_new_teleplay'*, *'y_noNA_new_teleplay'*, and *'y_NA_new_teleplay'* nparrays.
- Predict the empty values of the 'rating_2' column from the 'df_teleplay' data frame.
    - Evaluate the model performance using the non-empty values of the 'rating_2' column from the 'df_teleplay' data frame:

        - Split *'x_noNA_teleplay'* and *'y_noNA_teleplay'* nparrays into *'x_train'*, *'x_test'*, *'y_train'*, and *'y_test'* np arrays.
        - Perform Artificial Neural Network (ANN) Model in the training dataset.
        - Calculate RMSE of using ANN model in the training dataset.
            - In this case, the value of RMSE is 0.7051800875594707.
    - Perform ANN model, with *'x_NA_teleplay'* nparray as the given input values, to predict the empty values, in *'y_NA_teleplay',* of 'rating_2' column from 'df_teleplay' data frame:
        - Replace the empty values with the predicted values of *'y_NA_teleplay'* nparray to 'df_teleplay_test_predict_rating_2' dataframe of 'rating_2' column.
        - Concatenate 'df_teleplay_train_predict_rating_2' and 'df_teleplay_test_predict_rating_2' dataframes as 'df_teleplay' dataframe.
- Predict the empty values of 'rating_2' column from 'df_new_teleplay' dataframe:
    - Evaluate the model performance using the non-empty values of the 'rating_2' column from the 'df_new_teleplay' data frame.
        - Combine *'x_noNA_new_teleplay'* with *'x_noNA_teleplay'* (excluding 'rating' column values).
        - Split *'x_noNA_new_teleplay'* and *'y_noNA_new_teleplay'* nparrays into *'x_train'*, *'x_test'*, *'y_train'*, and *'y_test'* np arrays.
        - Perform Artificial Neural Network (ANN) Model in the training dataset.
        - Calculate RMSE of using ANN model in the training dataset.
            - In this case, the value of RMSE is 1.0958634072744535.
    - Perform ANN model, with *'x_NA_new_teleplay'* nparray as the given input values, to predict the empty values, in *'y_NA_new_teleplay'*, of 'rating_2' column from 'df_new_teleplay' dataframe:
        - Replace the empty values with the predicted values of *'y_NA_new_teleplay'* nparray to 'df_new_teleplay_test_predict_rating_2' dataframe of 'rating_2' column.
        - Concatenate 'df_new_teleplay_train_predict_rating_2' and 'df_new_teleplay_test_predict_rating_2' dataframes as 'df_new_teleplay' dataframe.

Check empty values existence in 'df_teleplay' data frame, per column.

- In this case, there does not exist an empty value.

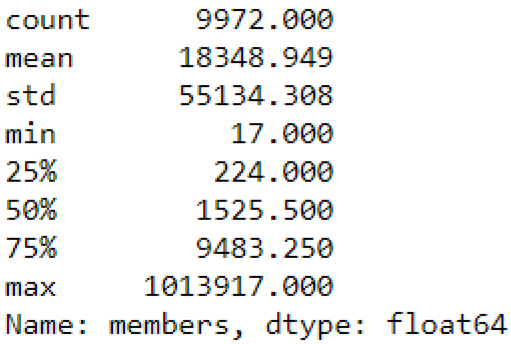
Check empty values existence in 'df_new_teleplay' data frame, per column.

- There is an empty value in this case in the 'rating' column values.

## Exploratory Data Analysis (EDA)

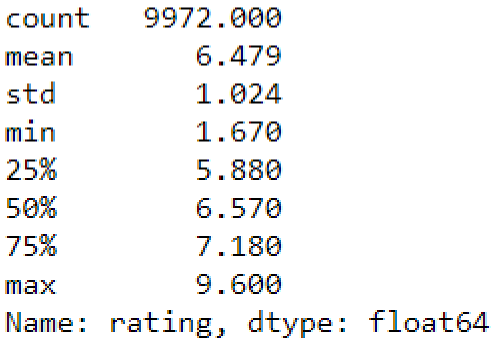
Describe the values of the 'df_teleplay' data frame on 'members' and 'rating' columns.

- Check the distribution of the 'df_teleplay' data frame 'members' values. It is shown that the minimum values are the teleplays that have at least 17 'members' value.



```
count       9972.000
mean       18348.949
std        55134.308
min           17.000
25%          224.000
50%         1525.500
75%         9483.250
max      1013917.000
Name: members, dtype: float64
```

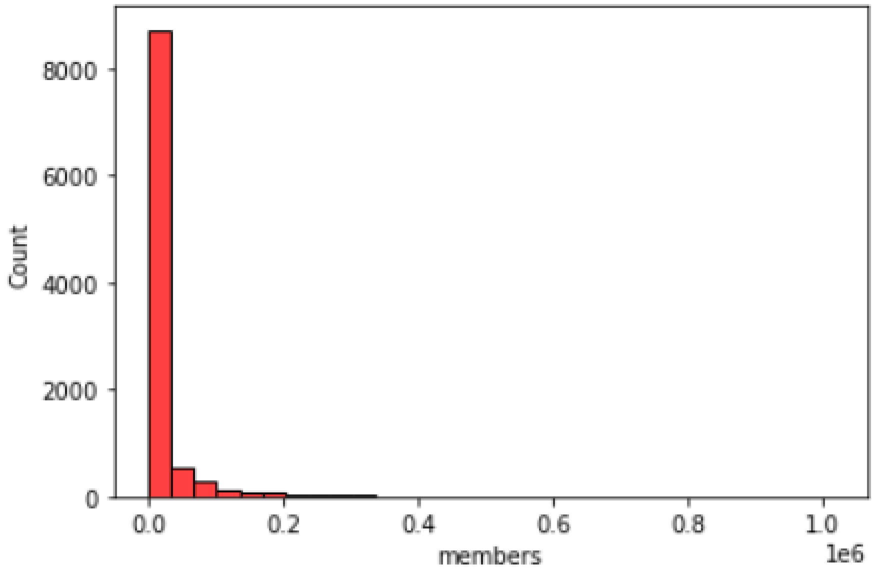
- Check the distribution of the 'df_teleplay' data frame 'rating' values. It is shown that the minimum values are the teleplays that have at least 1.670 'rating' value.



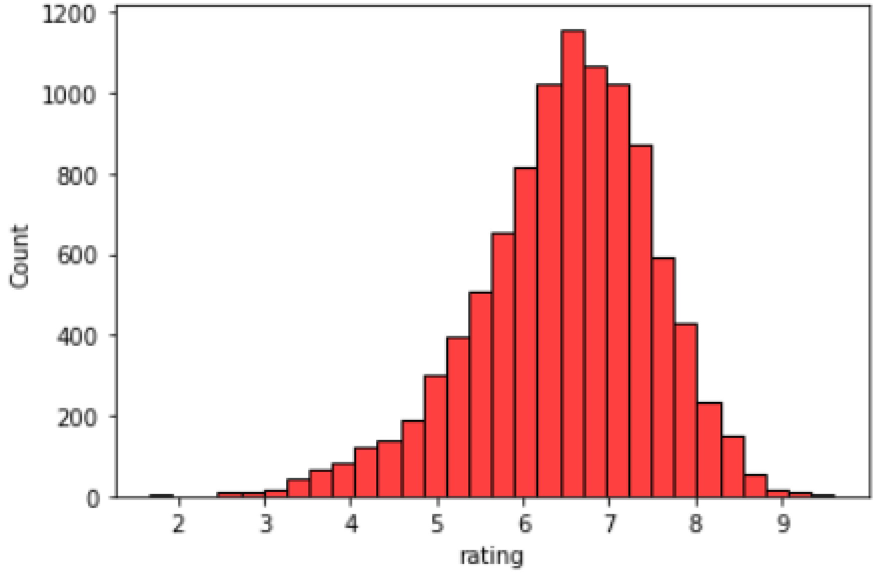
```
count    9972.000
mean        6.479
std         1.024
min         1.670
25%         5.880
50%         6.570
75%         7.180
max         9.600
Name: rating, dtype: float64
```

Visualize the trend and correlations among data points of the 'df_teleplay' data frame.

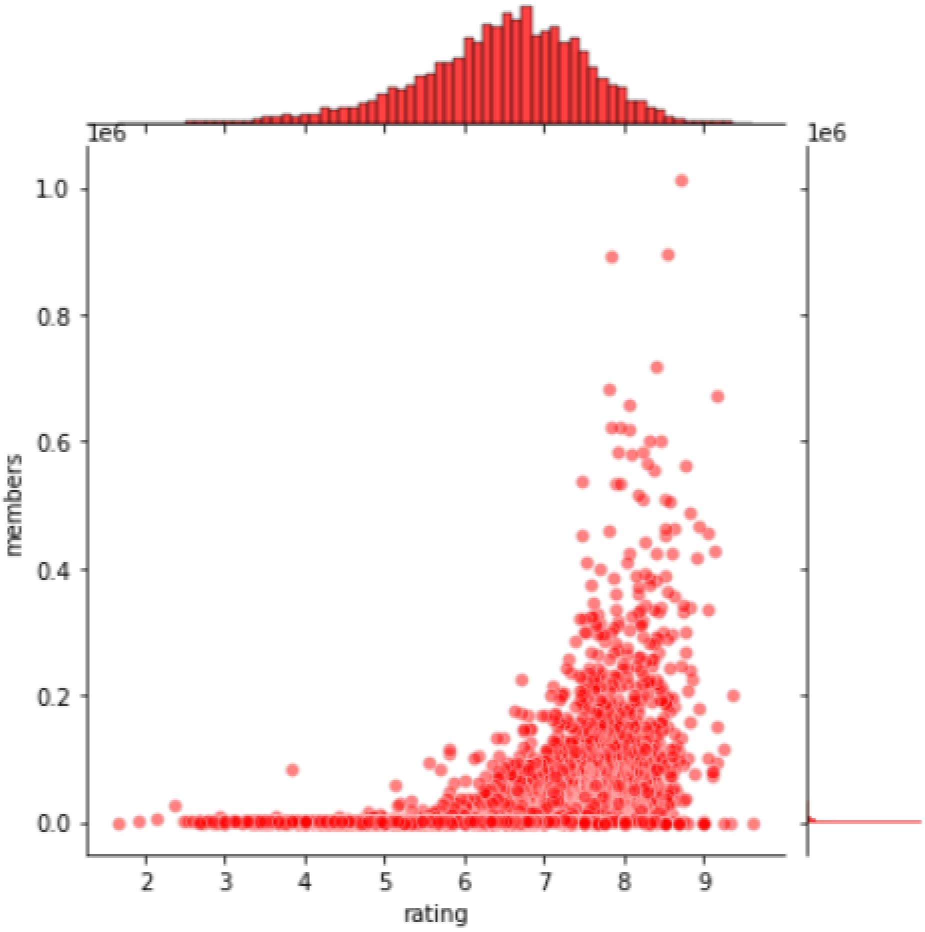
- 'members' column:
    - It is shown that the number of 'members' is dominated by a small group of teleplays that are blockbusters. The distribution plummets for other teleplays subsequently.



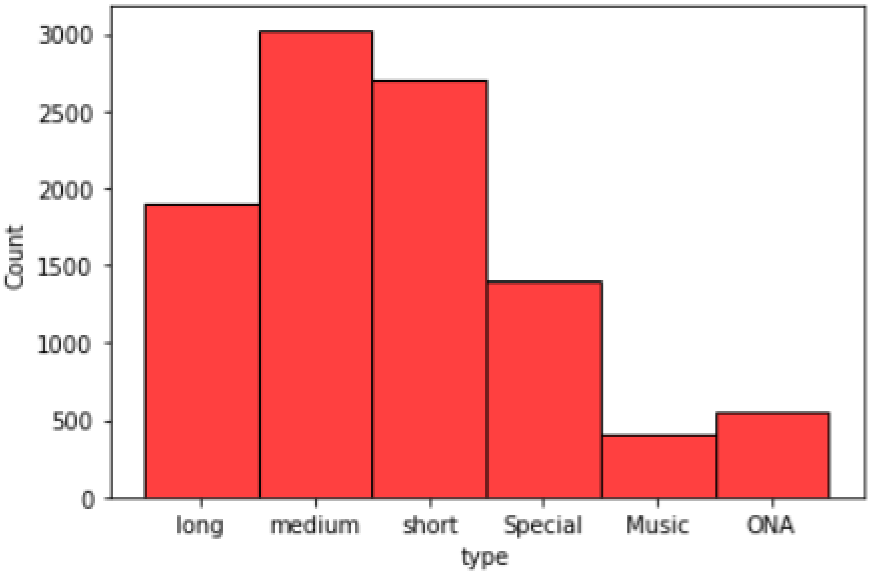
- ➢ 'rating' column:
  - ○ It is shown that the average 'rating' value is around 6.5, as the bell-curve shape permits.



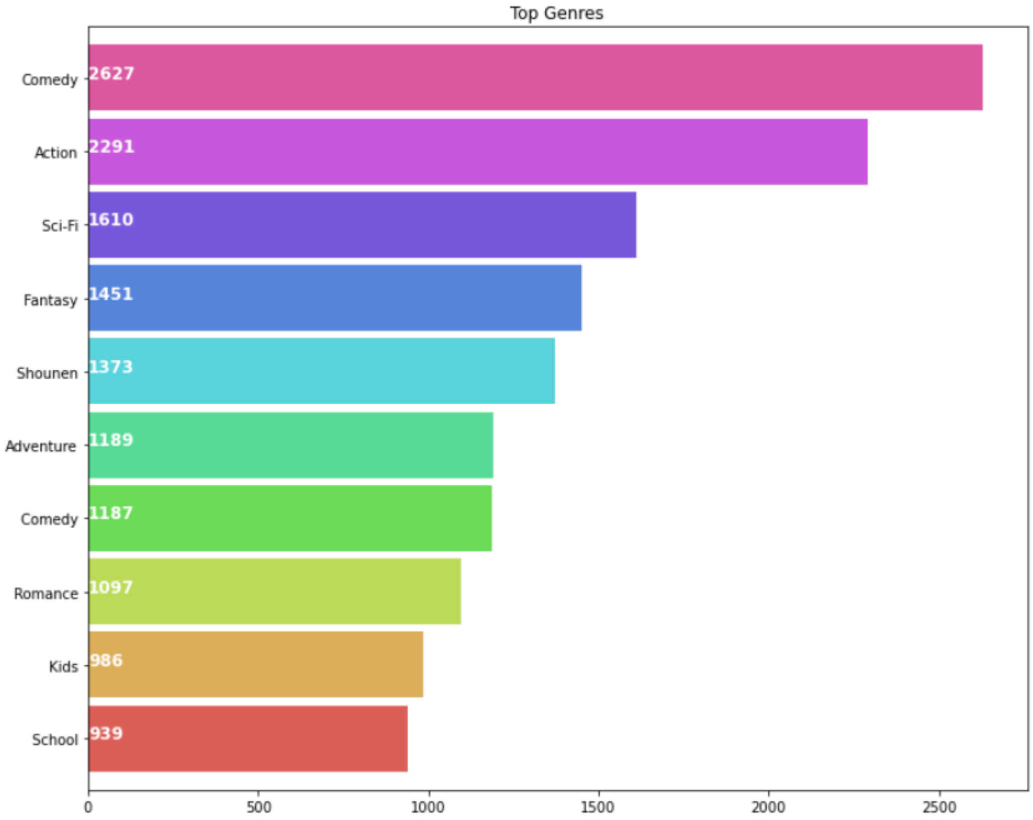
- ➢ 'members' and 'rating' columns:
  - ○ It is shown that the 'rating' values are increasing with the number of 'members'.



- 'type' column:
  - It is shown that medium-'type' teleplays are the most sought-after, while music-type teleplays have the least demand.



- 'genre' column:
  - It is shown that the comedy-'genre' has the most demand, followed closely by the action-'genre'.



- After considering all possible relevant relationships among columns in the 'df_teleplay' data frame, 'members', 'type', and 'genre' column values are being chosen as the input values for task 1, in addition to the 'rating_2' column values.

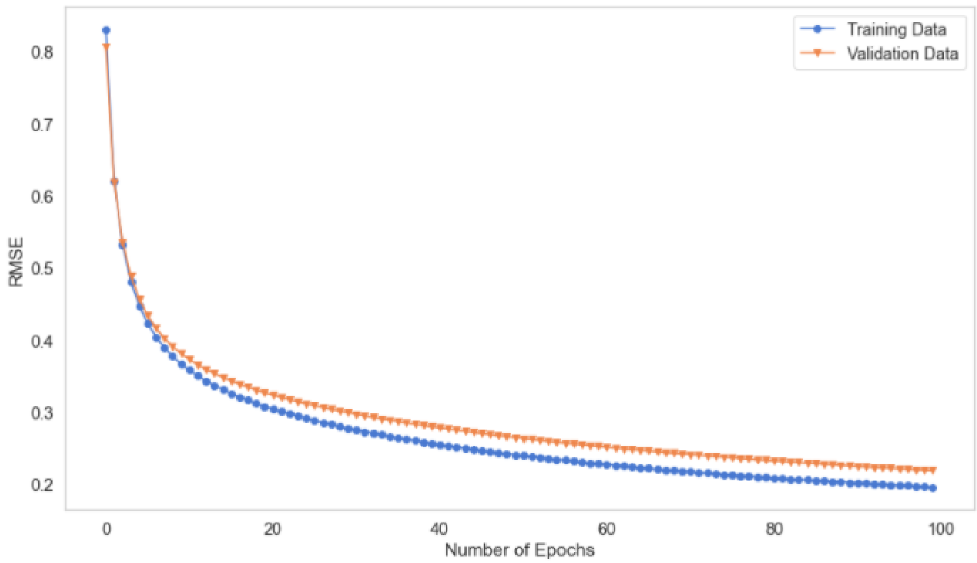
## Task 1: Predict the empty values of the 'rating' column from the 'df_new_teleplay' data frame

- Concatenate 'df_teleplay' and 'df_new_teleplay' dataframes as 'df_merge_encode_normalize' dataframe, for 'rating_2' , 'members' , 'type', and 'genre' column values.
  - Reset index of 'df_merge_encode_normalize' dataframe.
  - Save indexes information details about 'df_merge_encode_normalize'.
  - Normalize 'rating_2' , and 'members' columns values.
  - Encode 'type' and 'genre' columns values.
  - Merge normalized 'rating_2', and 'members' and encoded 'type' and 'genre' columns values into **'arr_merge_encode_normalize'** nparray.

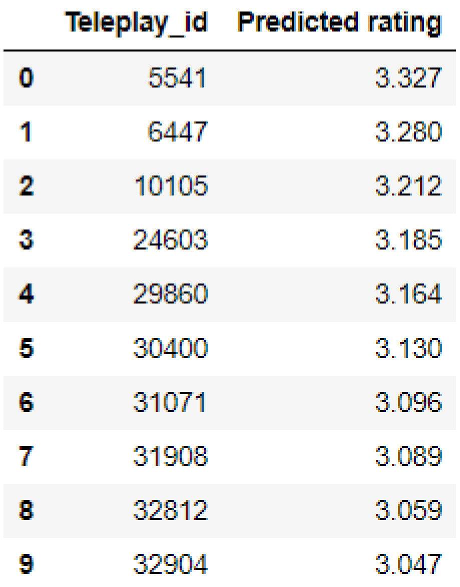
        - Split **'arr_merge_encode_normalize'** nparray into *'x_teleplay'*, *'y_teleplay'*, *'x_new_teleplay'*, and *'y_new_teleplay'* nparrays.
- Evaluate the model performance from the 'df_teleplay' data frame:
    - Split *'x_teleplay'* and *'y_teleplay'* nparrays into *'x_train'*, *'x_test'*, *'y_train'*, and *'y_test'* np arrays.
    - Perform Artificial Neural Network (ANN) Model in the training dataset.
    - Calculate RMSE of using ANN model in the training dataset.
        - In this case, the value of RMSE is 0.49096476235066167.
- Perform ANN model, with *'x_new_teleplay'* nparray as the given input values, to predict the values of 'rating' column from 'df_new_teleplay' data frame.
    - Replace the values with the predicted values of *'y_new_teleplay'* nparray to the 'df_new_teleplay' data frame of the 'rating' column.
    - Submission.
        - The resulted values are stored in the ‘18079969D_task1.csv’ file.

## Task 2: Predict user 53698’s personalized rating of all teleplays

- Get the average rating of each teleplay, based on the rating of each user, to each teleplay in 'df_rating'.
- Normalize the column values of 'rating_per_user_id' and perform aggregation to get the input values for collaborative filtering.
    - Store the resulted aggregation values as ‘ratings’ nparray.
    - Split ‘ratings’ nparray to *'x_train'*, *'x_test'*, *'y_train'*, and *'y_test'* np arrays.
- Perform Collaborative Filtering Model in the training dataset.
- Plot the RMSE value of using the Collaborative Filtering model in the training dataset with the number of epochs. As it shows, the regularization penalties for this model are optimal, in which the training data is below validation data in a reasonable distance.



- ➢ Perform a prediction for user 53698’s personalized rating of all teleplays.
    - ○ Output:



| | Teleplay_id | Predicted rating |
|---|---|---|
| **0** | 5541 | 3.327 |
| **1** | 6447 | 3.280 |
| **2** | 10105 | 3.212 |
| **3** | 24603 | 3.185 |
| **4** | 29860 | 3.164 |
| **5** | 30400 | 3.130 |
| **6** | 31071 | 3.096 |
| **7** | 31908 | 3.089 |
| **8** | 32812 | 3.059 |
| **9** | 32904 | 3.047 |

- ○ Submission:
    - ■ The resulted values are stored in the ‘18079969D_task2.csv’ file.

## Summary and future work

In conclusion, the report has described thoroughly the processes of getting a prediction result for task 1 using Artificial Neural Network (ANN) and task 2 using Collaborative Filtering. Moreover, Exploratory Data Analysis (EDA) process has been introduced to understand the correlation of each feature and analyzed its importance for the model implementation.

In the future, model optimization methods will be applied, such as k-cross-validation and grid search, to better understand how to tune the model in the best way possible. If any, other relevant approaches will also be considered, in the hope that the model can perform the lowest RMSE possible for task 1 and perform a more accurate recommendation of teleplays for a given user.

# References

Abadi, M. et al., (2016). TensorFlow: A system for large-scale machine learning. CoRR, abs/1605.08695.

Buitinck, L. et al., (2013). API design for machine learning software: experiences from the scikit-learn project. *ECML PKDD Workshop: Languages for Data Mining and Machine Learning* (pp. 108–122).

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science &amp; Engineering*, *9*(3), 90–95.

McKinney, W. et al, (2010). Data structures for statistical computing in python. *Proceedings of the 9th Python in Science Conference* (Vol. 445, pp. 51–56).

Oliphant, T. E. (2006). *A guide to NumPy* (Vol. 1). Trelgol Publishing USA.

Waskom, M. et al., (2017). *mwaskom/seaborn: v0.8.1 (September 2017)*, Zenodo.