Team Members:

هادي أحمد عبد السلام عبد الحميد1.

هادي عاطف سيد محمد2.

هادي إيهاب رجاء أحمد3.

وسيم عبده فتحي عبده4.

هيثم محمود السيد محمود5.

هاني محمد سيد أحمد 6.

Movie recommender

[Team ID\_8]

Movie Recommender

Introduction: /\* Discuss in brief the project main idea and the objectives\*/

The main idea of the project is to select the top 10 similar movies for each movie in the dataset based on the user’s ratings.

Methodology:

/\*Discuss the methodologies used in order to fulfil your objectives (i.e. The feature sets and the models implemented) \*/

1.Data Preprocessing:

1. Firstly, we performed an outer merge between two csv files; “movies” and “ratings” on “movieId” column into a new data frame called “RatedMovies”. We did this step in order to be able to use pivot\_table() function in the next step which works on one dataframe.
2. Out of the whole data, the most important features we will need are “userId”, “title” & “movieId” features. So, we continued preprocessing by using pivot\_table() function which is used to reshape a given dataframe organized by given index and column values. Pivot\_table() function is applied on “RatedMovies” dataframe setting the index to a list containing two columns; “movieId” and “title” while setting columns to “userId”. Then, creating a new dataset called “final\_dataset” which is set to that new dataframe.
3. Filling undefined cells: it is done by just filling these cells by 0 instead of “NaN” using fillna() function. We also set “inplace” variable to true to apply the changes on the data frame itself.

2. Features extraction:

We have applied this step to remove unnecessary data from both users and movies datasets based on two conditions:

1. Removing movies that have been rated by 5 users or less.
2. Removing users that have rated 15 movies or less.

3. Model training and testing

Clustering: We have used K-Nearest Neighbor algorithm (KNN) to get the 10 nearest neighbors of each movie based on the users’ ratings for this movie.

At the end we accessed the final dataset using surprise package to store the final dataset in a dataset object called “data” which we split afterwards into trainset (80%) and testset (20%) using train\_test\_split() and setting test\_size=0.20.

Training: we use KNN Algorithm to apply training on trainset using KNNBasic() prediction algorithm (in surprise package) in which we fit our trainset.

Testing: we use test () function which takes trainset as a parameter to apply testing. Afterwards, we compute the mean absolute error and the model’s accuracy and print them.

4. Function getting similar movies

We implemented “get\_movie\_recommendation” function which takes the entered movie name by user and it checks if it is a substring of any movie title in “title” column in our dataset then it appends all matching results into “movie\_list”. If the list is empty, it will display to the user that the movie is not found. In case there is one matching result only, the 10 most similar movies will be displayed with their distances to the required movie. Otherwise, it displays the matching results with their indices to the user. The user picks the required movie by entering the movie’s index then the function displays the 10 most similar movies with their distances.

Data Set Summary:

/\*Answer the following Questions:

1-What is the data set used?

🡪This dataset describes 5-star rating and free-text tagging activity from [MovieLens], a movie recommendation service. It contains 100836 ratings and 3683 tag applications across 9742 movies. These data were created by 610 users between March 29, 1996 and September 24, 2018. This dataset was generated on September 26, 2018. The data are contained in the files `movies.csv` and `ratings.csv`.

2- What is the summary of the dataset columns?

Movies Data File structure(movies.csv):

Movie information is contained in ‘movies.csv’ which has three columns; “movieId” containing a unique ID for each different movie, “title” carrying movie titles and “genres” column which contains the movies’ genres.

Ratings Data File Structure (ratings.csv):

All ratings are contained in the file `ratings.csv`. Each line of this file after the header row represents one rating of one movie by one user, and has the following format:

userId, movieId, rating, timestamp

The lines within this file are ordered first by userId, then, within user, by movieId.

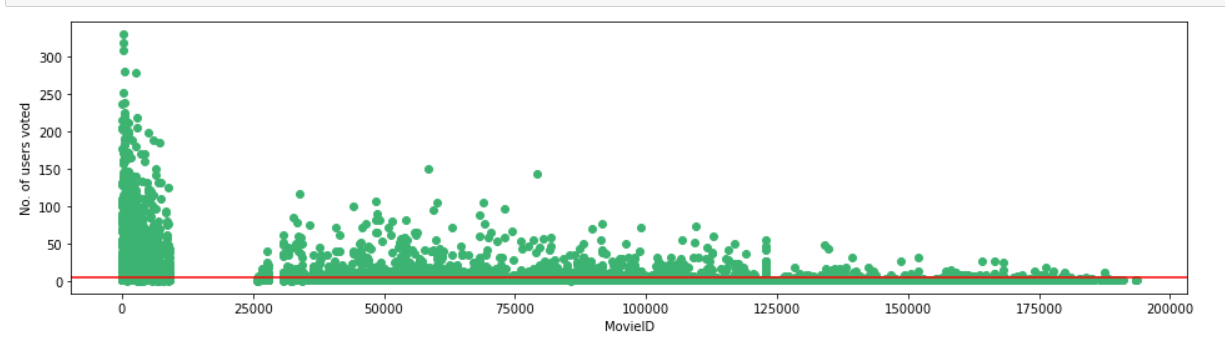
Ratings are made on a 5-star scale, with half-star increments (0.5 stars - 5.0 stars).

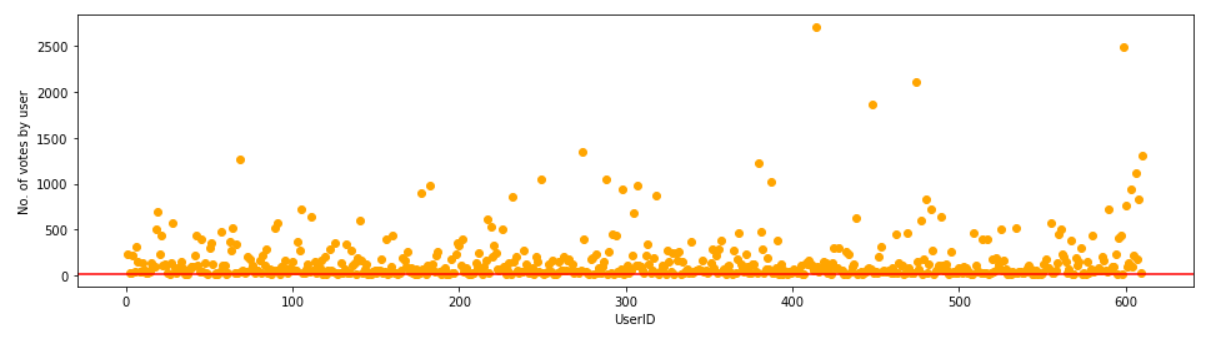
Timestamps represent seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970.

3- Visualize the dataset statistics

The following two graphs show important relations that we needed to make the decision of performing features extraction on our dataset.

1. Graph showing the relation between movies and the number of users rating these movies

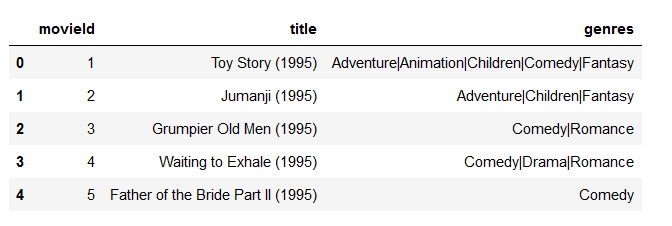


1. Graph showing the relation between users and the number of movies rated by them

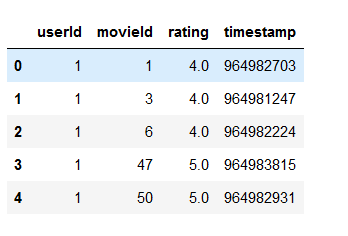
Results:

/\*Use suitable graphs to visualize your models results\*/

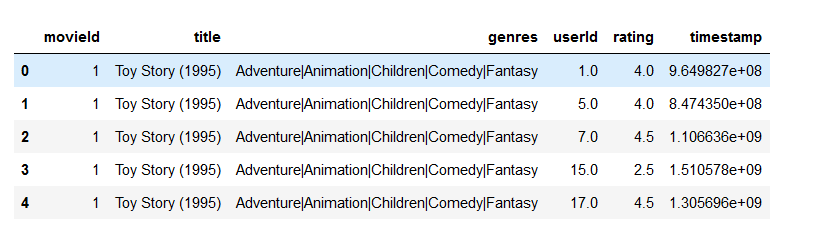
Sample of movies dataset



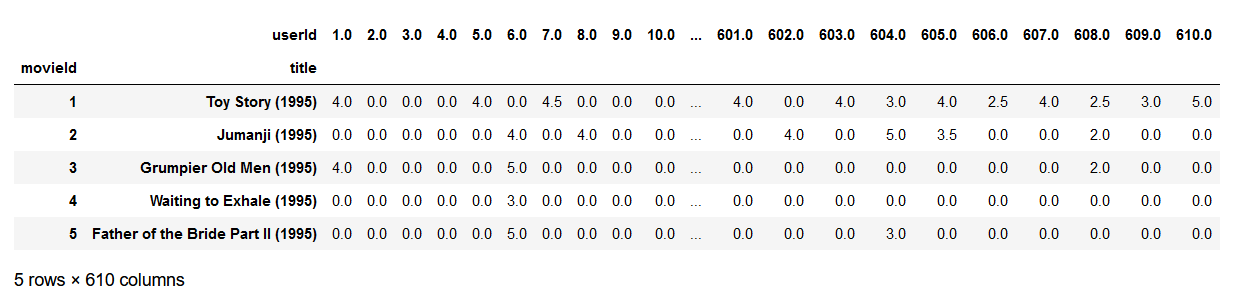
Sample of ratings dataset



Sample of the result after merging both movies and ratings into a new dataset



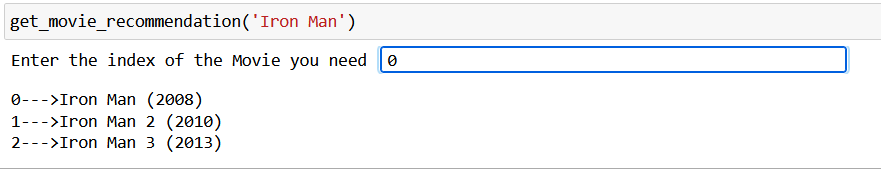
Sample of the final dataset after being preprocessed:

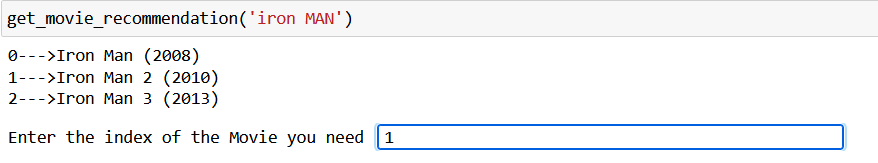


Getting the 10 most similar movies to “Iron Man” and “Iron Man 2” Movies.

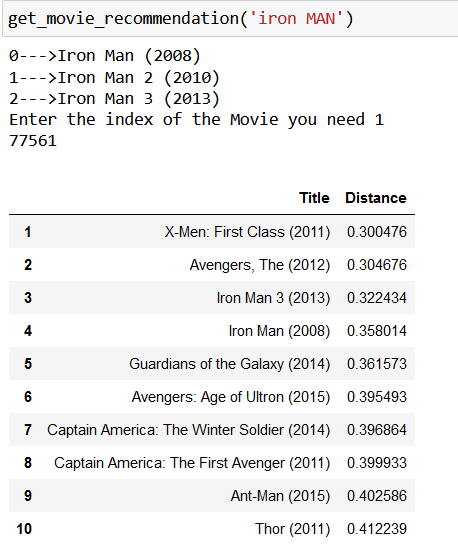
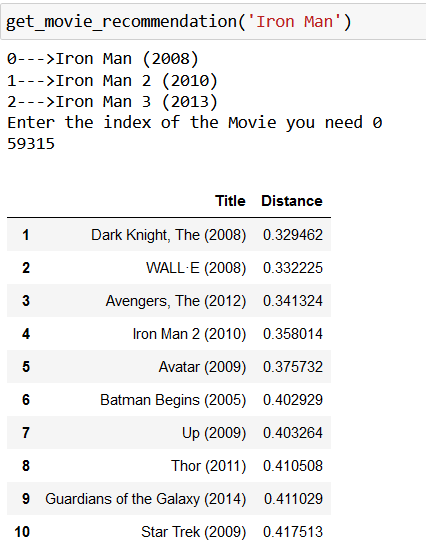
**Notice: the function is working without considering case sensitivity**

STEP1: Displaying the matching results to let the user pick the required movie





STEP2: Displaying 10 most similar movies for “Iron Man” and “Iron Man 2”



Finally, displaying Mean Absolute Error and model’s accuracy