Project B – Movie Recommendation System

Introduction:

This project is about designing a basic recommender system for movies. This project aims to download, summarize, and analyze the movies dataset and then perform predictive analytics on the dataset. It will recommend 10 movies based on a movie and its different ratings received by different users. The dataset used in this project is the MovieLes Dataset put together by the Grouplens research group at the University of Minnesota. In this project, this data is used to build a simple item-similarity based recommender system. A simple pipeline is implemented which is to download the data, summarize the data, analyze the data, build the recommendation system, and generate the results.

Extra credits: Developing a storage model for the dataset – The data lake
Exploratory Data analysis on data

Dataset Description: This dataset (ml-latest-small) 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. Users were selected at random for inclusion. All selected users had rated at least 20 movies. The dataset contains three different files, rating.csv, movies.csv, and tags.csv. All ratings are contained in the file ratings.csv. The observation in this file represents one rating of one movie by one user and has the features userId, movie, rating, timestamp. Movie information is contained in the file movies.csv. The observation in this file represents one movie and has the features movieId, title, genres. All tags are contained in the file tags.csv. The observation in this file represents one tag applied to one movie by one user and has the features userId, movieId, tag, timestamp.

Background:

In today's era, the rise in the cut-throat competition has made it essential to adapt to survive and thrive. The business models nowadays have to be more up to date and provide the service which is beyond excellence to flourish. The companies and organizations are nowadays shifting their focus to what customers want and how they can deliver it in the best way. In this modern society there is so much variety available may it be any technology, or any kind of service delivered. People have the option of selection out of huge possibilities. This gives rise to the need for companies to provide a recommendation to the users.

A lot of companies nowadays use recommendation systems to recommend the users best suited as per their liking by monitoring their use. Companies like Netflix, Spotify, Amazon have established a whole new standard in providing recommendations to the users thus making the experience of users more satisfactory. A lot of information is being gathered regarding the users and their use and modifications are being made to improvise the recommendations to deliver the best services. This is the need of today and the companies who can catch up to this will withstand the market and truly flourish.

Methodology:

The two important types of recommendation systems are content-based and collaborative filtering recommender systems. In collaborative filtering, the behavior of a group of users is used to make recommendations to other users. The recommendation is based on the preference of other users. In content-based systems, we use metadata such as genre, actor, musician to recommend movies or music. Here in this project, a simple item similarity-based recommender system is designed.

The data from --.csv is loaded in one data frame and the data from movies.csv is loaded in another data frame. These two data frames are merged using the movie column as the pivot. Then the generated data frame is studied and statistics are observed. Then the dataset is grouped by the title column and it's mean is computed to obtain the average rating for each movie.

The number of ratings for each movie is needed so it is done by creating a number_of_ratings column. This is done to analyze the relationship between the average rating of a movie and the number of ratings the movie got. It might be possible that a 5-star movie was rated by just one person. Hence a threshold for the minimum number of ratings is needed as we build the recommender system. So, the title column is grouped by and then the count function is used to calculate the number of ratings each movie got. Now this newly augmented data frame is analyzed and visualized using the plots created by the matplotlib and seaborn packages of python. The distribution of ratings, the relationship between the rating of movies and no_of_ratings, etc. are observed through the graphs.

This data frame is converted into a matrix with the movie titles as the columns, the userId as the index, and the ratings as the values. By doing this we got a data frame with the columns as the movie titles and the rows as the user ids. Each column represents all the ratings of a movie by all users. This matrix shall be used to compute the correlation between the ratings of a single movie and the rest of the movies in the matrix. Movies that have a high correlation coefficient are the movies that are most similar to each other. In this case, the Pearson correlation coefficient is used. This number will lie between -1 and 1. 1 indicates a positive linear correlation while -1 indicates a negative correlation. 0 indicates no linear correlation.

The most rated movies are selected and two of them are chosen to work with this simple recommendation system. Consider, Shawshank Redemption (1994) and Silence of the Lambs (1991). The goal is to look for movies that are similar to the above two which we shall recommend to this user. First, a dataframe is created with the ratings of these movies from our movie_matrix. Using the corrwith functionality of pandas, the correlation between each movie's rating and the ratings of the Shawshank Redemption (1994) is calculated. Similarly, the correlation between each movie's rating and the ratings of the Silence of the Lambs (1991) is also calculated. The matrix had very many missing values since not all the movies were rated by all the users. Therefore, all those null values are dropped, and correlation results are transformed into data frames.

Now we have two data frames that show us the movies that are most similar to Shawshank Redemption (1994) and Silence of the Lambs (1991) respectively. A threshold for the number of ratings needs to be set. The histogram shows a sharp decline in the number of ratings from 100 so

it was therefore set as the threshold. Now the top 10 movies that are most similar to the movie Shawshank Redemption (1994) are selected by limiting them to movies that have at least 100 reviews and sorting them by the correlation column and view the first 10. Similarly, this is carried out for the movie Silence of the Lambs (1991).

#Refer the Jupyter notebook ProjectB.ipynb for the entire code and intermediate outputs.

Results and discussions :

A data lake is a system or repository of data stored in its raw format. A data lake is usually a single store of all enterprise data including raw copies of source system data and transformed data used for tasks such as reporting, visualization, advanced analytics. The data stored in the data lake can be structured or unstructured depending on the availability and the desired tasks to perform. For this project, a directory in HDFS is created to dump all the data. That is creating a data lake as a single repository or store for all the data.

```
[js-156-233] rgawande ~-->cd /opt/hadoop/sbin
[js-156-233] rgawande /opt/hadoop/sbin-->./start-dfs.sh
Starting namenodes on [localhost]
localhost: namenode is running as process 21873.  Stop it first.
Starting datanodes
localhost: datanode is running as process 22016.  Stop it first.
Starting secondary namenodes [js-156-233.jetstream-cloud.org]
js-156-233.jetstream-cloud.org: secondarynamenode is running as process 22258.  Stop it first.
[js-156-233] rgawande /opt/hadoop/sbin-->hadoop fs -mkdir /data-lake
[js-156-233] rgawande /opt/hadoop/sbin-->hadoop fs -ls /
Found 1 items
drwxr-xr-x   - rgawande supergroup          0 2020-04-27 15:57 /data-lake
```

```
[js-156-233] rgawande ~-->hadoop fs -ls /
Found 1 items
drwxr-xr-x   - rgawande supergroup          0 2020-04-27 15:57 /data-lake
[js-156-233] rgawande ~-->source ~/.bashrc
[js-156-233] rgawande ~-->hadoop fs -put movies.csv /data-lake
2020-04-27 16:02:06,455 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHost
se
[js-156-233] rgawande ~-->hadoop fs -put rating.csv /data-lake
put: `rating.csv': No such file or directory
[js-156-233] rgawande ~-->hadoop fs -put ratings.csv /data-lake
2020-04-27 16:02:40,080 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHost
se
[js-156-233] rgawande ~-->hadoop fs -put tags.csv /data-lake
2020-04-27 16:02:49,898 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHost
se
[js-156-233] rgawande ~-->hadoop fs -ls /
Found 1 items
drwxr-xr-x   - rgawande supergroup          0 2020-04-27 16:02 /data-lake
[js-156-233] rgawande ~-->hadoop fs -ls /data-lake
Found 3 items
-rw-r--r--   1 rgawande supergroup     494431 2020-04-27 16:02 /data-lake/movies.csv
-rw-r--r--   1 rgawande supergroup    2483723 2020-04-27 16:02 /data-lake/ratings.csv
-rw-r--r--   1 rgawande supergroup     118660 2020-04-27 16:02 /data-lake/tags.csv
```

Exploratory Data Analysis: The initial exploratory data analysis gave a brief idea of the data and statistical information about the data.

	userId	movieId	rating	timestamp
count	100836.000000	100836.000000	100836.000000	1.008360e+05
mean	326.127564	19435.295718	3.501557	1.205946e+09
std	182.618491	35530.987199	1.042529	2.162610e+08
min	1.000000	1.000000	0.500000	8.281246e+08

Table 1.1

We can tell that the max rating is 5 and the min is 5 with the average rating around 3.5. We also see that the dataset has 100003 records.

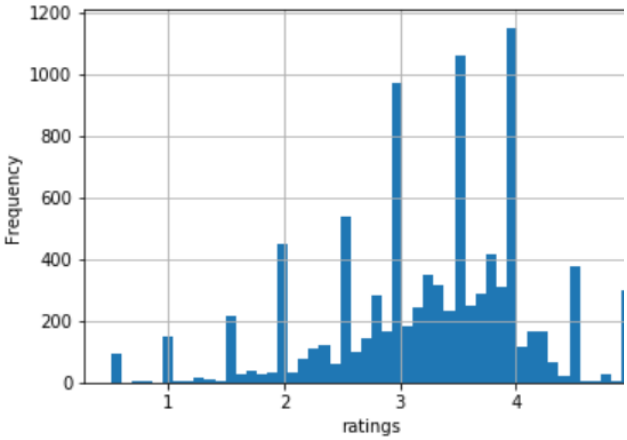


Fig 1.1

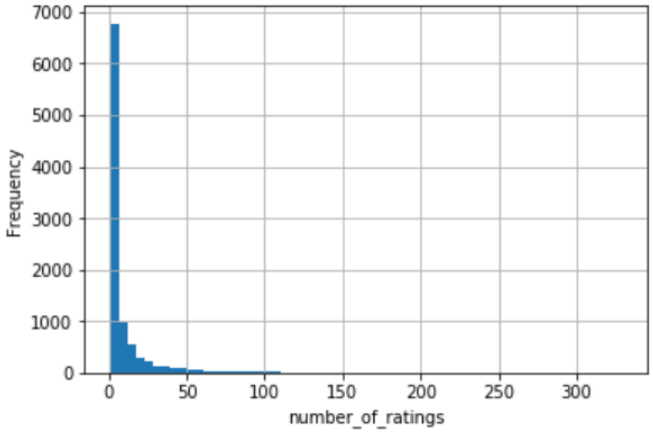
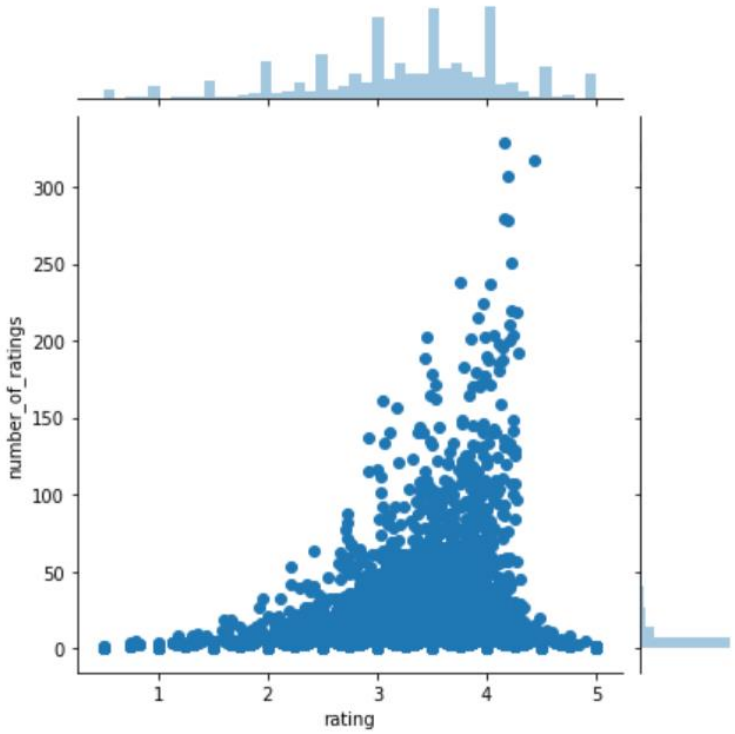


Fig 1.2

Consider fig 1.1, we can see from the distribution that most of the movies are rated between 2.5 and 4. From figure 1.2, it is clear that most movies have few ratings. Movies with most ratings are those that are most famous.



From the figure, we can see that there is a positive relationship between the average rating of a movie and the number of ratings. The graph indicates that the more the ratings a movie gets the higher the average rating it gets. This is important to note especially when choosing the threshold for the number of ratings per movie.

Recommendation System:

title	correlation	number_of_ratings
Shawshank Redemption, The (1994)	1.000000	317
Four Weddings and a Funeral (1994)	0.446212	103
Schindler's List (1993)	0.402202	220
Usual Suspects, The (1995)	0.394294	204
Ocean's Eleven (2001)	0.391546	119
Green Mile, The (1999)	0.382818	111
Inception (2010)	0.377839	143
Catch Me If You Can (2002)	0.356612	115
One Flew Over the Cuckoo's Nest (1975)	0.354215	133
Godfather: Part II, The (1974)	0.349872	129

We notice that Shawshank Redemption (1994) has a perfect correlation with itself, which is not surprising. The next most similar movie to Shawshank Redemption (1994) is Four Weddings and a Funeral with a correlation of 0.446 with 103 ratings. Clearly, by changing the threshold for the

number of reviews we get different results from the previous way of doing it. Limiting the number of ratings gives us better results and we can confidently recommend the above movies to someone who has watched Shawshank Redemption (1994).

title	Correlation	number_of_ratings
Silence of the Lambs, The (1991)	1.000000	279
Memento (2000)	0.536660	159
Aliens (1986)	0.492863	126
Shrek (2001)	0.440900	170
Amelie (Fabuleux destin d'Amélie Poulain, Le) (2001)	0.437687	120
Big Lebowski, The (1998)	0.431846	106
Groundhog Day (1993)	0.430134	143
American History X (1998)	0.420858	129
Minority Report (2002)	0.416317	120
Incredibles, The (2004)	0.408804	125

Similarly, we did it for the movie Silence of the lambs (1991). Once again we get different results. The most similar movie to Silence of the lambs (1991) is Memento (2000) with a correlation coefficient of 0.536 with 279 ratings. So, if somebody liked Silence of the lambs (1991) we can recommend the above movies to them.

Future scope*:* This system can be improved by building a Supervised Collaborative Filtering based system. We can then use techniques such as cosine similarity to compute the similarity between the movies. An alternative is to build a Model-based Collaborative Filtering system. This is based on matrix factorization. Matrix factorization is good at dealing with scalability and sparsity than the former. Deep learning is another way to build advanced recommendation engines while working with massive datasets. Nowadays autoencoders have been incorporated in the recommendation engines.

Conclusion:

A simple recommendation system was designed using the MovieLens dataset by performing merging on various data files, analyzing the data, accessing the movies, and calculating correlation amongst them based on users and their ratings. This system is responsible to recommend 10 movies to the user based on what movie the user has selected. In today's world, recommendation systems are not only a necessity, but they also have several benefits and uses in different user-centric business models. Such models help the companies to cater to the customers or users of their service perfectly. However, this recommendation system is just a basic version based on item-similarity and is not comparable to the industry standards.

References:

[1] files.grouplens.org/datasets/movielens/ml-latest-small-README.html

[2] https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

[3] https://towardsdatascience.com/how-to-build-a-simple-recommender-system-in-python-375093c3fb7d

[4] https://towardsdatascience.com/how-to-build-a-simple-song-recommender-296fcbc8c85

[5] https://pandas.pydata.org/docs/