EE769 REPORT

Netflix – Movie Recommendation system

Mourya Naru
Mukesh Uppala

About

The Netflix movie recommendation system is a machine learning project that aims to provide personalized movie recommendations to Netflix users based on their Interests, search queries, and user ratings. The system uses various machine learning algorithms, such as collaborative filtering and content-based filtering, to analyse user behaviour and preferences and provide movie recommendations that are tailored to each individual user. The Netflix movie recommendation system is crucial for Netflix's success because it helps to retain customers by providing them with personalized content that matches their interests. Moreover, it allows Netflix to increase user engagement and revenue by promoting new and relevant movies to users.

Problem Statement

Netflix needs an efficient and accurate recommendation system to improve user experience and increase user engagement. The challenge is to develop a machine learning algorithm that can analyze large amounts of data and understand user preferences to provide personalized recommendations. The dataset used in this project contains over 17K movies and 500K+ customers, making it challenging to process and analyse the data effectively. The aim is to implement machine learning models like collaborative filtering and Pearson's methods to extract relevant features from the dataset and predict movie ratings based on user preferences. The ultimate goal is to improve the quality of recommendations and increase user engagement, ultimately leading to the growth of the business by retaining existing customers and attracting new ones.

Netflix provided a lot of anonymous rating data, and a prediction accuracy bar that is 10% better than what Cinematch can do on the same training data set. (Accuracy is a measurement of how closely predicted ratings of movies match subsequent actual ratings.)

Objective

- To build a movie recommendation mechanism within Netflix for better user experience
- Predict the rating that a user would give to a movie that he has not yet rated.
- Minimize the difference between predicted and actual rating (RMSE and MAPE)

Theory

In this project we used SVD (singular value decomposition) for collaborative filtering. The collaborative filtering method works by analysing the ratings given by multiple users and identifying the similarities in their ratings given for a movie to make recommendations to the individual user based on what other users with similar tastes have liked or disliked. This method is commonly implemented by using matrix factorization techniques like SVD (singular value decomposition).

In the SVD a matrix R is decomposed into three matrixes U,D,V^T ·R is the user-item rating matrix, U matrix represents user preferences and V^T matrix represents the item atrributes. D is the diagonal

matrix. Here our item is the movie_id and attribute is rating of the movie. We used scikit-surprise package for the svd model.

Data Overview

Source of Data : https://www.kaggle.com/netflix-inc/netflix-prize-data

Data files : combined_data_1.txt combined_data_2.txt combined_data_3.txt combined_data_4.txt movie_titles.csv

The first line of each file [combined_data_1.txt, combined_data_2.txt, combined_data_3.txt, combined_data_4.txt] contains the movie id followed by a colon. Each subsequent line in the file corresponds to a rating from a customer and its date in the following format:

CustomerID, Rating, Date/year

MovieIDs range from 1 to 17770 sequentially. CustomerIDs range from 1 to 2649429, with gaps. There are 480189 users. Ratings are on a five star (integral) scale from 1 to 5. Dates have the format YYYY-MM-DD

Methodology

- Data manipulation
    - Data loading
    - Data viewing
    - Data cleaning
    - Data slicing
    - Data mapping
- Recommendation models
    - Recommend with Collaborative Filtering
    - Recommend with Pearsons' R correlation
    - Recommending Top ‘n’ movies of a year based on content filtering

Data Loading

Each data file (there are 4 of them) contains below columns:

- Movie ID (as first line of each new movie record / file)
- Customer ID
- Rating (1 to 5)
- Date they gave the ratings

There is another file contains the mapping of Movie ID to the movie background like name, year of release, etc

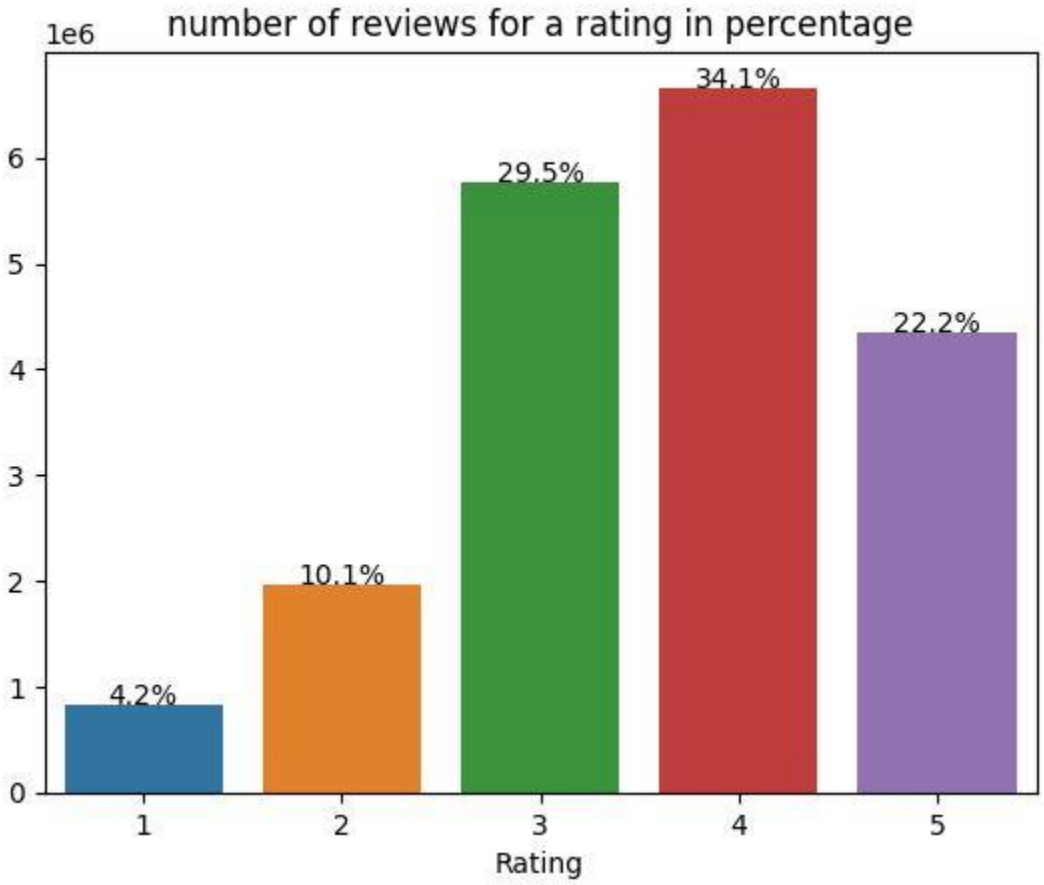
Each data file (there are 4 of them) contains below columns:

- Movie ID (as first line of each new movie record / file)
- Customer ID

- Rating (1 to 5)
- Date they gave the ratings

There is another file contains the mapping of Movie ID to the movie background like name, year of release, etc

This Plot shows the Overall rating percentage given by users



We can see that the rating tends to be relatively positive (>3). This may be due to the fact that unhappy customers tend to just leave instead of making efforts to rate. We can keep this in mind - low rating movies mean they are generally really bad

Data cleaning

Movie ID is really a mess import! Looping through dataframe to add Movie ID column WILL make the Kernel run out of memory as it is too inefficient. I achieve my task by first creating a numpy array with correct length then add the whole array as column into the main dataframe! Let's see how it is done below:

Data slicing

The data set now is super huge. I have tried many different ways but can't get the Kernel running as intended without memory error. Therefore I tried to reduce the data volumn by improving the data quality below:

- Remove movie with too less reviews (they are relatively not popular)

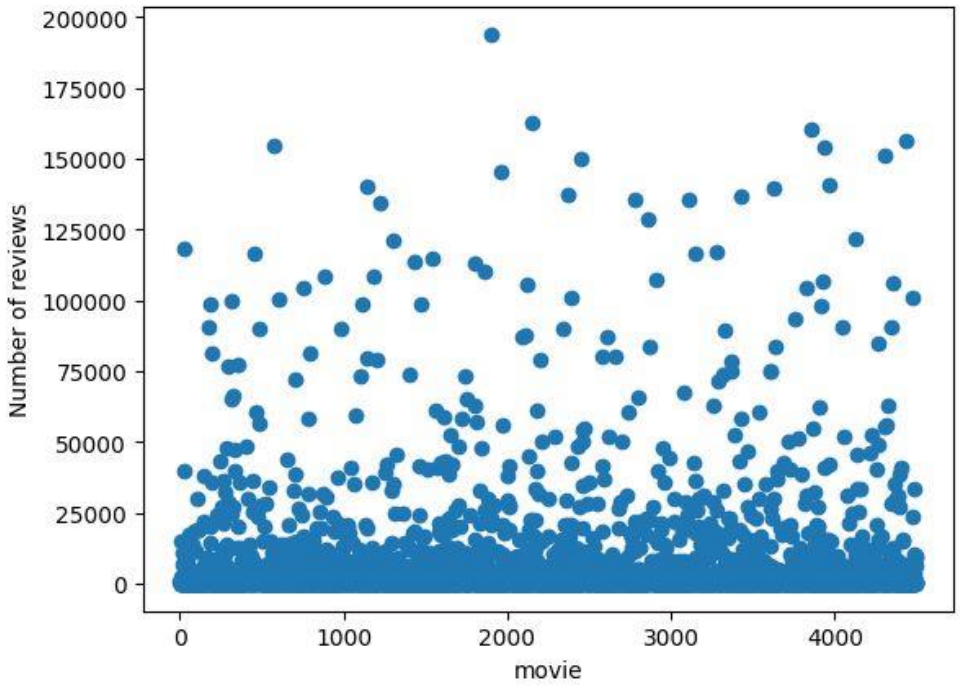
- Remove customer who give too less reviews (they are relatively less active)

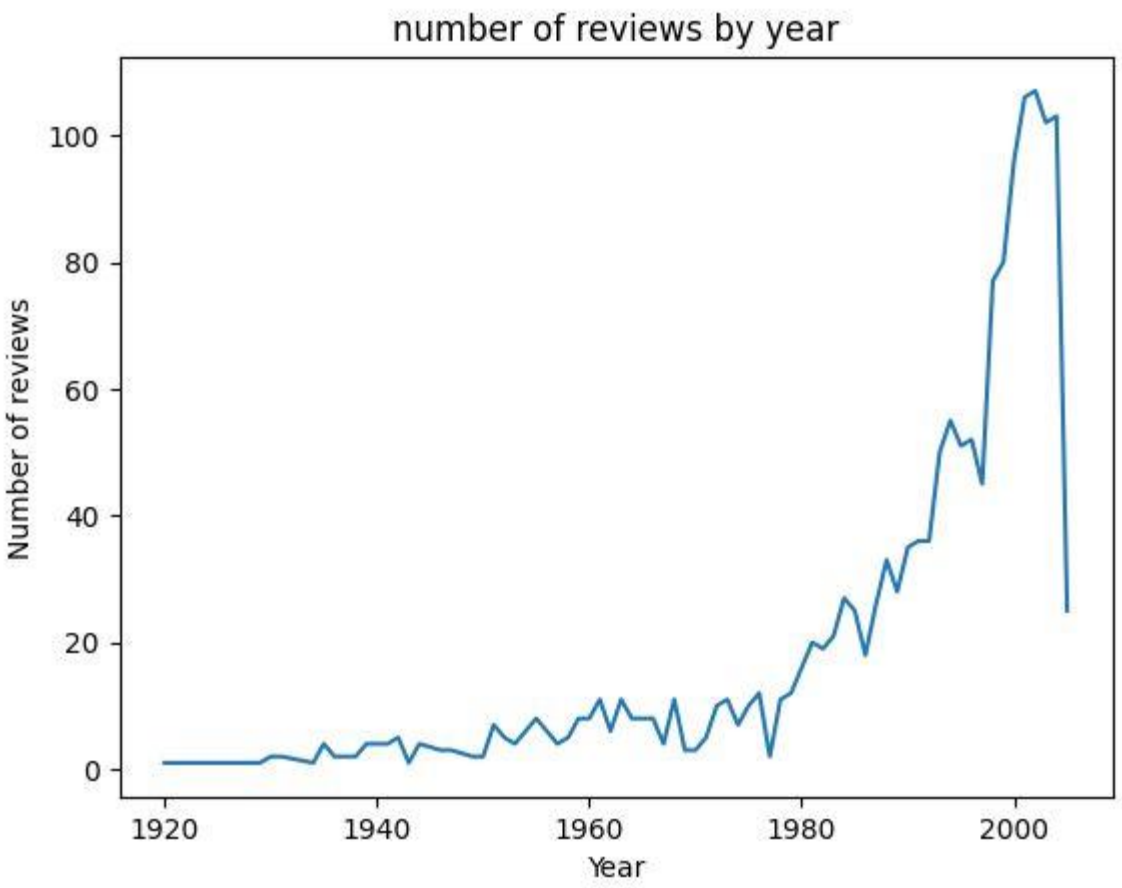
Having above benchmark will have significant improvement on efficiency, since those unpopular movies and non-active customers still occupy same volume as those popular movies and active customers in the view of matrix (NaN still occupy space). This should help improve the statistical significance too.

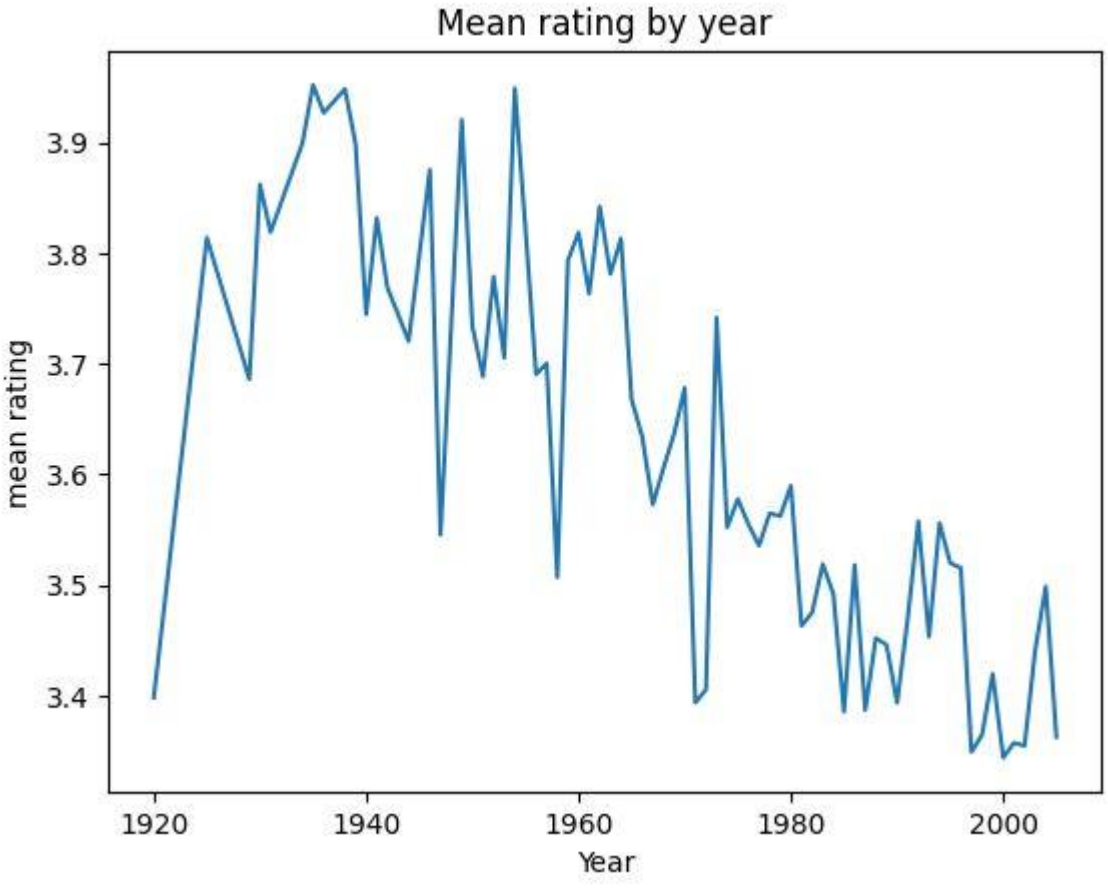
Data mapping

- load the movie mapping file

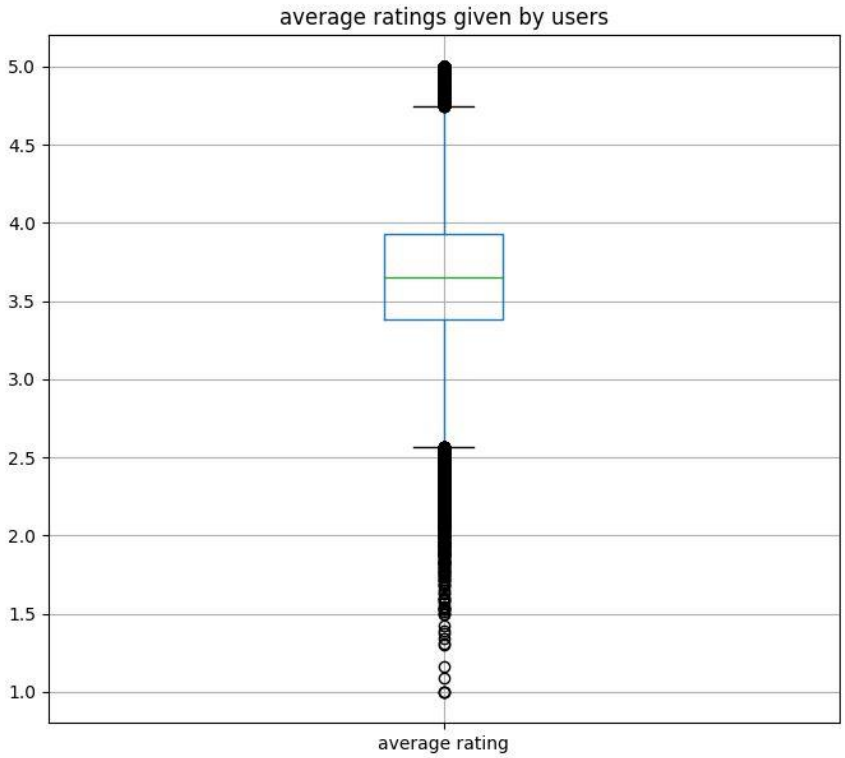
Following are plots of some data and their respective results



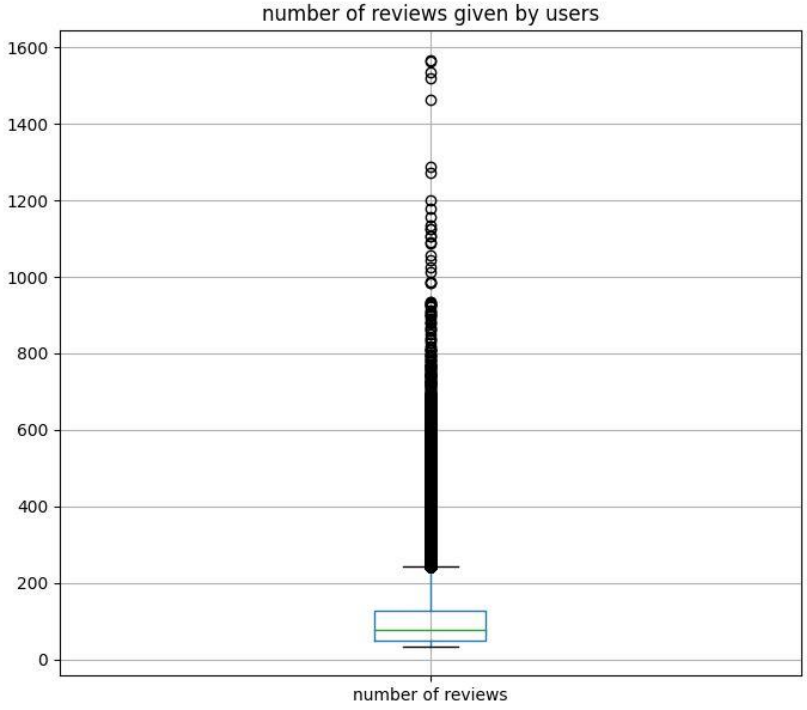

number of reviews by year
Number of reviews
0
20
40
60
80
100
1920
1940
1960
1980
2000
Year


Mean rating by year
mean rating
3.4
3.5
3.6
3.7
3.8
3.9
1920
1940
1960
1980
2000
Year

Plots 4a and 4b gives average rating given by users and no of users rated a movie



4a

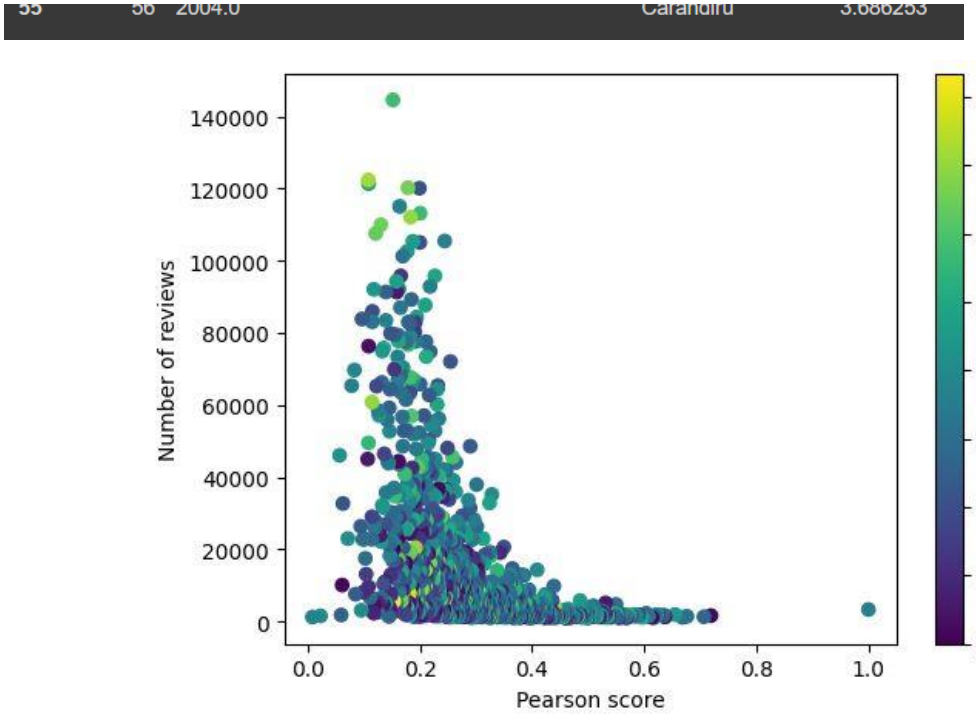


4b

Observations

Prediction using svd model:

Here we predicted top 10 movies for the user 1025579:



- When a certain movie is input, we get the Movies recommended based on pearsons model and their respective scores
- Rationalized mean square error for Movie rating given user id is 0.98

Result 1 Input (user watch) = Cloak Dagger

Movie id	Pearson score	Year	Movie name	Number of reviews	Average rating
1021	1	164	Cloak Dagger	3306	3.72
375	0.719	2003	Nine dead Gay guys	1604	3.034
2896	0.706	1960	Last metro	1293	3.55
56	0.676	2004	Carandiru	1413	3.61
4224	0.668	1996	Wish upon star	1139	3.02
895	0.655	1975	Dorsu uzala	1911	3.904

Result 2 Input (user watch) = Omen3

Movie id	Pearson score	Year	Movie name	Number of reviews	Average rating
2047	1	1981	Omen 3: final conflict	2838	3.177
3102	0.723	1978	Damien: omen 3	4821	3.37
3808	0.621	1997	Best men	1252	2.75
2708	0.609	1999	Candyman 3 : day of dead	1775	2.83
2962	0.606	1973	Coffy	1141	3.308

Complete code link and References:

https://colab.research.google.com/drive/1zMb_0OxzsCuEMqCGdIH2z-AialResI6n?usp=share_link

https://github.com/nishantml/NETFLIX-MOVIE-RECOMMENDATION-SYSTEM

https://www.kaggle.com/code/laowingkin/netflix-movie-recommendation

END