Capstone Project - Project **NETFLIX**

Table of Contents:

1. Problem Statement

2. Project Objective

3. Data Description

4. Data Pre-processing Steps and Inspiration

5. Choosing the Algorithm for the Project

6. Motivation and Reasons for Choosing the Algorithm

7. Assumptions

8. Model Evaluation and Techniques

9. Inferences from the Same

10. Future Possibilities of the Project

11. Conclusion

12. References

Problem Statement:

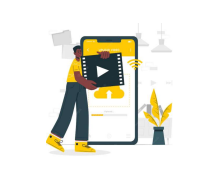
Customer Behaviour and its prediction lie at the core of every Business Model. From Stock Exchange, e-Commerce and Automobile to even Presidential Elections, predictions serve a great purpose. Most of these predictions are based on the data available about a person’s activity either online or in-person.



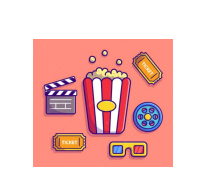
Recommendation Engines are the much needed manifestations of the desired Predictability of User Activity. Recommendation Engines move one step further and not only give information but put forth strategies to further increase user’s interaction with the platform.



In today’s world OTT platform and Streaming Services have taken up a big chunk in the Retail and Entertainment industry. Organizations like Netflix, Amazon etc. analyse User Activity Pattern’s and suggest products that better suit the user needs and choices.



# For the purpose of this Project we will be creating one such Recommendation Engine from the ground-up, where every single user, based on their area of interest and ratings, would be recommended a list of movies that are best suited for them.



Project Objective:

The objective of this project is to draw insights that can give them a clear perspective on the following:

1. Find out the list of most popular and liked genre.

2. Create Model that finds the best suited Movie for one user in every genre.

3. Find what Genre Movies have received the best and worst ratings based on User Rating.



Data Description:

The dataset available is provided with the weekly sales data for their various outlets.

What we can infer from the given data is that, it is provided with various features and they are as follows:

1. **ID** – Contains the separate keys for customer and movies.

2. **Rating** – A section contains the user ratings for all the movies.

3. **Genre** – Highlights the category of the movie.

4. **Movie Name** – Name of the movie with respect to the movie id.



# Recommendation System for Netflix Prize Dataset using SVD

Data Pre-processing: Steps and Inspiration

1. Importing the required Libraries



# To load the 'combined\_data\_1' dataset

2. The second step is to mount the Google drive account to the Colab notebook.



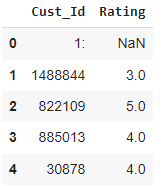
3. Reading dataset file



3.1 Reading the first 5 columns of the dataset



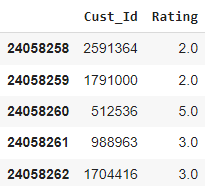
Output:



3.2 Reading the last 5 columns of the dataset



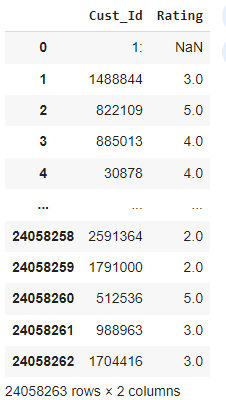
Output:



3.3 Reading the full dataset

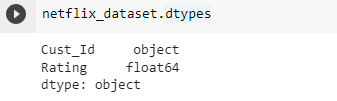


Output:

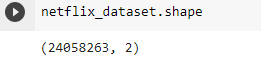


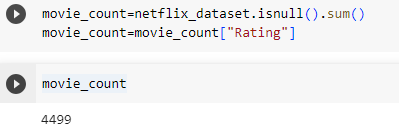
Here 1: 🡪 indicates the 1st movie and 2: indicates 2nd movie and likewise others indicates

4. To print the datatype of columns

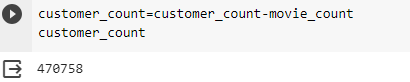
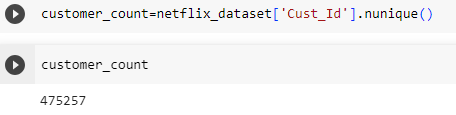


5. To inspect the shape of the dataset

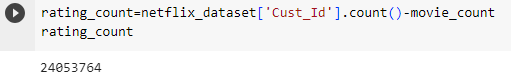


6. To get movie count by counting Nan values

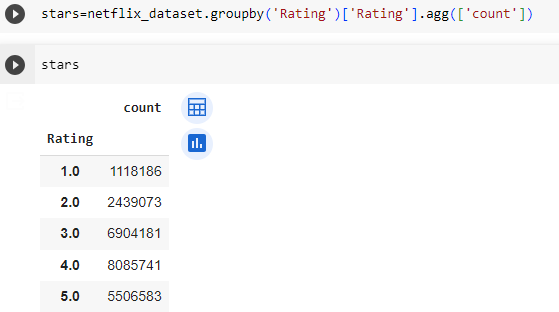
7. To get customer count



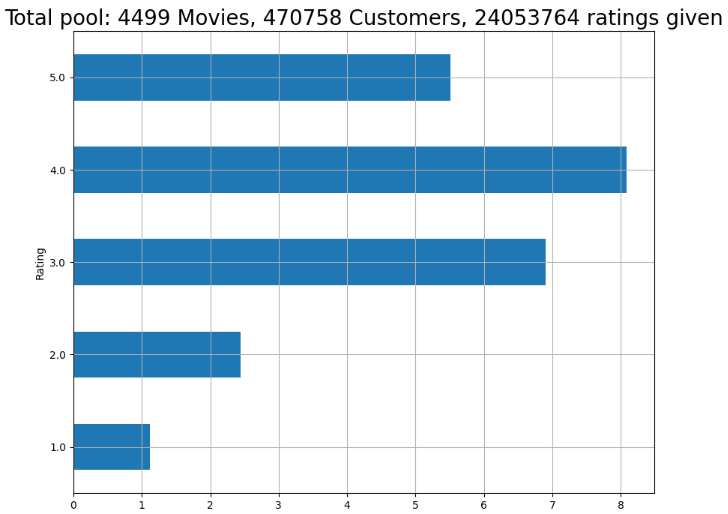
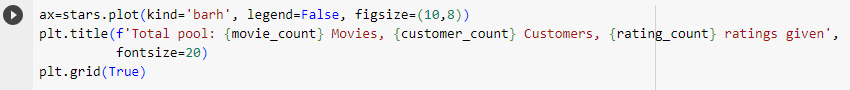
8. To get rating count



9. To find the distribution of different ratings in the dataset



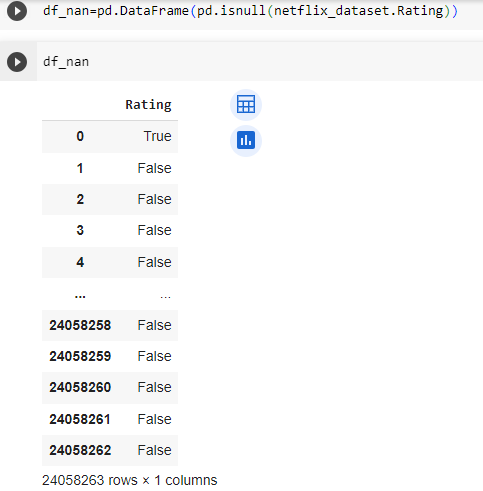
10. To plot the distribution of the ratings in as a bar plot



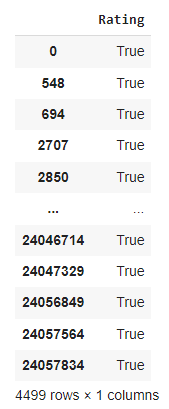
# To create a numpy array containing movie ids corresponding to the rows in the 'ratings' column in Netflix dataset

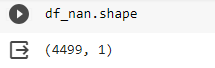
11.

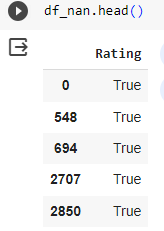
11.1 To count all the ‘nan’ values in the Rating column

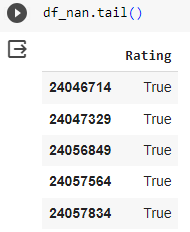


11.2 To store the index of all the rows containing 'nan' values

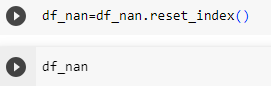


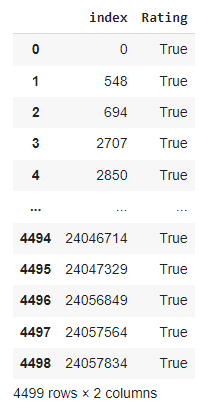




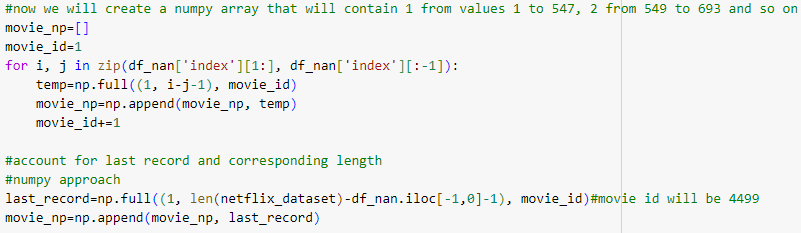


11.3 To reset the index of the dataframe

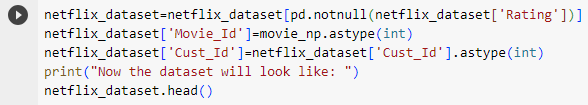


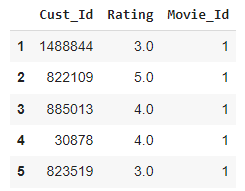


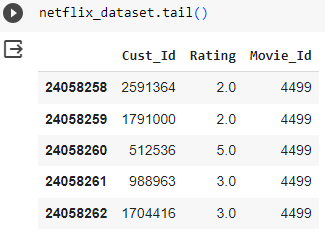
11.4 To create a numpy array containing movie ids according the 'ratings' column in the Netflix dataset



12. To append the above created array to the Netflix dataset after removing the 'nan' rows

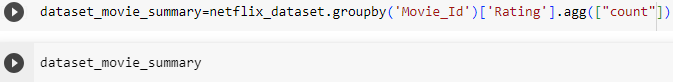


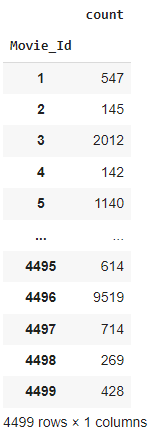




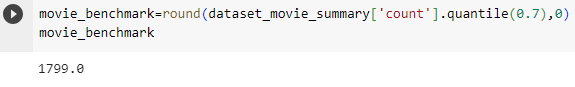
# Data Cleaning

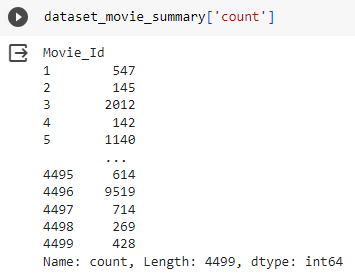
13. To create a list of all the movies rated less often (only include top 30% rated movies)



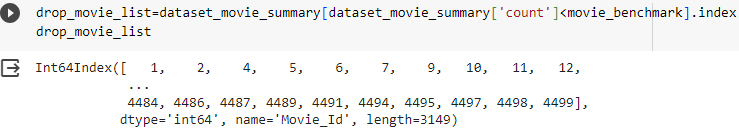


13.1 Now I will create a movie benchmark



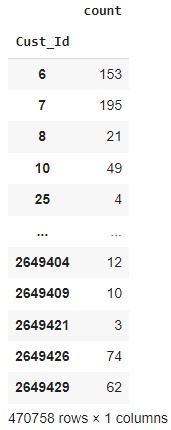


13.2 Creating a separate list containing movies index that have been rated less often than the benchmark

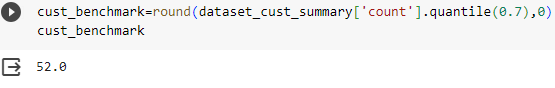


14. To create a list of all the inactive users (users who rate less often)

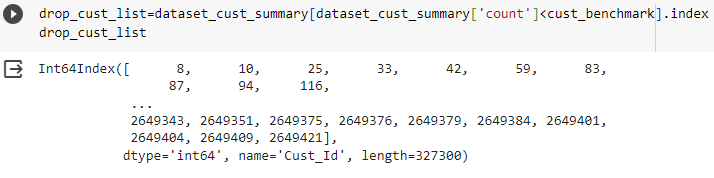




14.1 Now I will create a customer benchmark



14.2 Creating a separate list containing user index that have rated less often than the benchmark

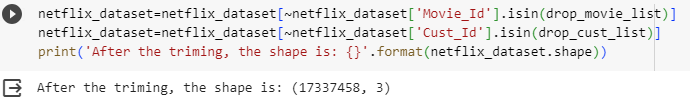


The original dataframe shape is:

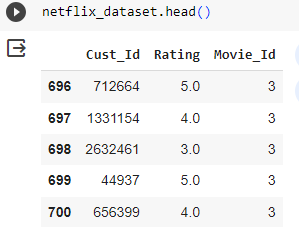


15. I will remove all the customers and movies that are below the benchmark

15.1 And the shape after the trim is:

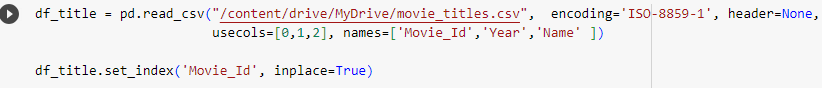


15.2 Now our Netflix dataset looks like:



### To load the movie\_titles dataset

16.





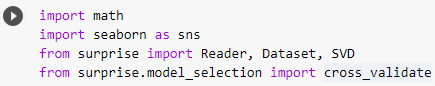
# To install the scikit-surprise library for implementing SVD

17.





18. Importing the required libraries



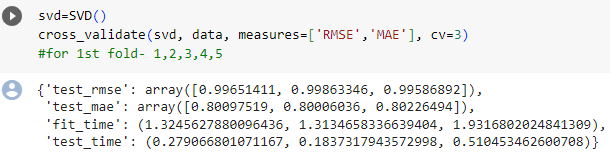
19. Load Reader library

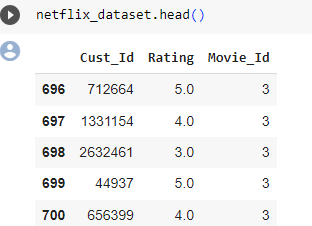


20. Getting just top 100K rows for faster run time and converting it in the matrix format so that it would be readable by SVD

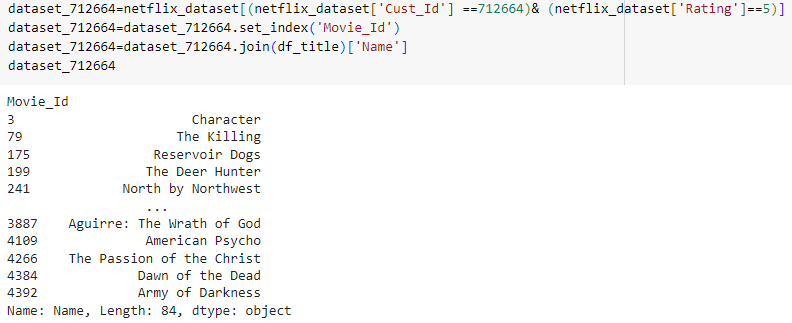


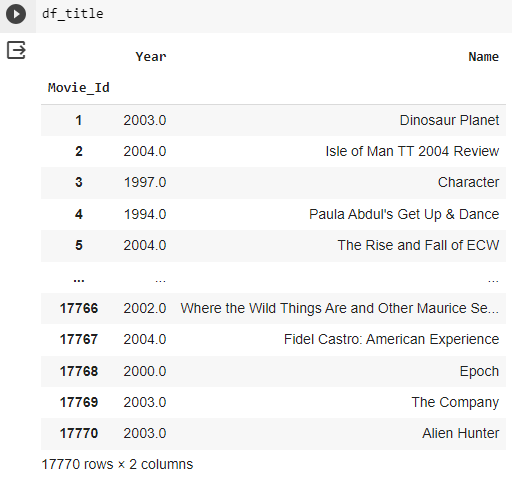
21. Using the SVD algorithm





## 22. To find all the movies rated as 5 stars by user with userId = 712664





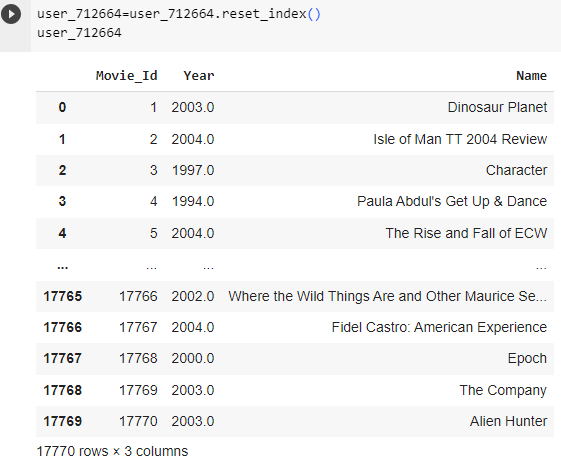
# 23. Train an SVD to predict ratings for user with userId = 1

23.1 Create a shallow copy for the movies dataset

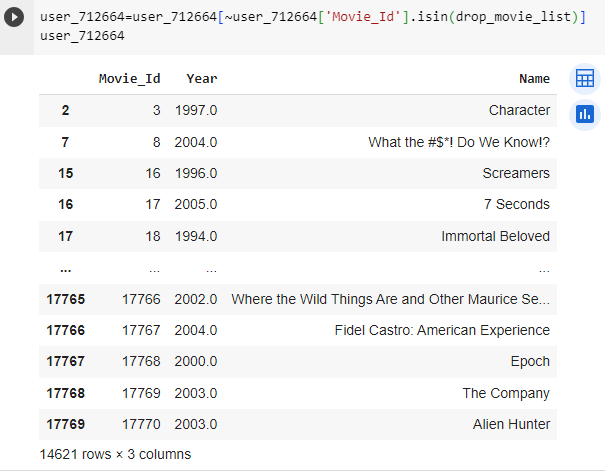




23.2 Reset the index



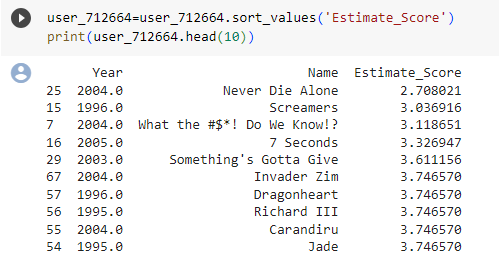
23.3 To remove all the movies rated less often



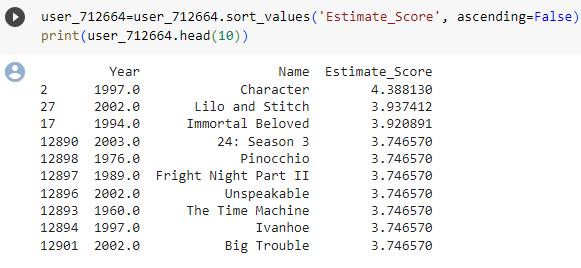
24. Predict the ratings for user\_712664



24.1 Sort predicted ratings for user\_712664



24.2 Sorting predicted ratings for user\_712664 in descending order and Printing top 10 recommendations



Choosing the Algorithm for the Project

SVD: Single Value Decomposition

The SVD is often denoted  ���T**UΣV**T

A **recommender system**, or a **recommendation system** (sometimes replacing 'system' with a synonym such as platform or engine), is a subclass of **information filtering system** that provide suggestions for items that are most pertinent to a particular user. Typically, the suggestions refer to various decision-making processes, such as what product to purchase, what music to listen to, or what online news to read. Recommender systems are particularly useful when an individual needs to choose an item from a potentially overwhelming number of items that a service may offer.

Recommender systems usually make use of either or both collaborative filtering and content-based filtering (also known as the personality-based approach),as well as other systems such as knowledge-based systems. In this recommendation system we used one of the approach among the above i.e., **collaborative filtering.**

**Collaborative filtering approaches** build a model from a user's past behaviour (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users. This model is then used to predict items (or ratings for items) that the user may have an interest in.

Collaborative filtering methods are classified as memory-based and model-based. A well-known example of memory-based approaches is the user-based algorithm,while that of model-based approaches is Matrix Factorization (recommender systems).

**Singular value decomposition** is used in recommender system to predict people's item ratings. Distributed algorithms have been developed for the purpose of calculating the SVD on clusters of commodity machines.

Motivation and Reasons for Choosing the Algorithm

A key advantage of the collaborative filtering approach is that it does not rely on machine analysable content and therefore it is capable of accurately recommending complex items such as movies without requiring an "understanding" of the item itself.

Singular Value Decomposition (SVD), a method from linear algebra is getting popular in the field of data science and machine learning. This popularity is because of its application in developing recommender systems.

Finding and recommending many suitable items that would be liked and selected by users is always a challenge. There are many techniques used for this task and SVD is one of those techniques.

The Singular Value Decomposition (SVD), a method from linear algebra that has been generally used as a dimensionality reduction technique in machine learning. SVD is a matrix factorisation technique, which reduces the number of features of a dataset by reducing the space dimension from N-dimension to K-dimension (where K<N). In the context of the recommender system, the SVD is used as a collaborative filtering technique. It uses a matrix structure where each row represents a user, and each column represents an item. The elements of this matrix are the ratings that are given to items by users.

Assumptions

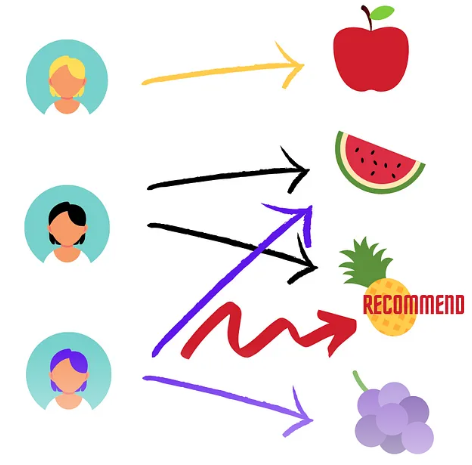
# Collaborative Filtering:

The assumption of this approach is that people who have liked an item in the past will also like the same in future. This approach builds a model based on the past behaviour of users. The user behaviour may include previously watched movies, liked movies, given ratings on list of movies. In this way, the model finds an association between the users and the items. The model is then used to predict the item or a rating for the item in which the user may be interested. Singular value decomposition is used as a collaborative filtering approach in recommender systems.

Model Evaluation and Techniques

User-based collaborative filtering algorithm usually has the following steps:

1. Find similar users based on interactions with common items.
2. Identify the items rated high by similar users but have not been exposed to the active user of interest.
3. Make use of the SVD to decompose the matrix and perform the collaborative filtering.
4. List items based on the score and pick the top n items to recommend.



Inferences

We can see that there is relationship between the past likings of the customers and the ratings provided by them. So we can definitely know that there is an indirect link between one user and the other, as in here the prediction is made from the past likings of the user whose information we are going to predict and the other users who have rated different movies. If the possibilities are high that the respective predictive user will surely like the movie according to his and his peer’s ratings then we will predict those movies for him and the same will be applied on other users as well.

Future Possibilities of the Project

Recommender systems provide users with personalized suggestions for products or services. These systems often rely on Collaborating Filtering (CF), where past transactions are analysed in order to establish connections between users and products. The two more successful approaches to CF are latent factor models, which directly profile both users and products, and neighbourhood models, which analyse similarities between products or users. In this work we introduce some innovations to both approaches. The factor and neighbourhood models can now be smoothly merged, thereby building a more accurate combined model. Further accuracy improvements are achieved by extending the models to exploit both explicit and implicit feedback by the users. The methods can be performed on the Netflix data. Results can be better than the previous ones. In addition, i suggest a new evaluation metric, which highlights the differences among methods, based on their performance at a top-K recommendation task.

Conclusion

Most of the existing approaches to collaborative filtering cannot handle very large data sets. In few cases it shows how a class of two-layer undirected graphical models, called Restricted Boltzmann Machines (RBM's), can be used to model tabular data, such as user's ratings of movies. It presents efficient learning and inference procedures for this class of models and demonstrate that RBM's can be successfully applied to the Netflix data set, containing over 100 million user/movie ratings. It is also shown that RBM's slightly outperform carefully-tuned SVD models. When the predictions of multiple RBM models and multiple SVD models are linearly combined, we achieve an error rate that is well over 6% better than the score of Netflix's own system.

Reference

-Intellipaat

-Wikipedia (SVD, Recommendation system, Collaborative filtering)

-[ICML '07: Proceedings of the 24th international conference on Machine learning](https://dl.acm.org/doi/proceedings/10.1145/1273496) June 2007

-R. Bell and Y. Koren, "Lessons from the Netflix Prize Challenge", SIGKDD Explorations 9 (2007)

-J. Bennet and S. Lanning, "The Netflix Prize", KDD Cup and Workshop, 2007

-https://dl.acm.org/doi/abs