Anime Recommendation System

Riddhi Narayan,[1] Saachi Chandrashekhar[2]
Northeastern University [1,2]
narayan.ri@northeastern.edu,[1] chandrashekhar.s@northeastern.edu[2]

Abstract

The objective of this project was to develop an anime recommendation system that suggests anime to different users by making use of unsupervised machine learning techniques and data mining tasks like clustering and frequent patterns that can recommend anime to people based on their interests.

Introduction

Today's market is flooded with data, and as a result, we are well into a data-driven era where consumers have an abundance of options. The issue of information overload, which has become a barrier for many users, must be solved by filtering, prioritizing, and efficiently delivering vital information. This issue is resolved by recommender systems, which sift through vast amounts of data to offer customers specialized products and services. Recommender systems increase revenue in an e-commerce environment since they are an efficient way to increase sales.

Anime is a computer-generated animation of hand-drawn cartoons that originated in Japan. Today, the popularity of Anime has grown all over the world and is still only ever growing. Due to the multitude of options available today and their different genres, users find it difficult to make a well-informed choice. Because of this, it would be beneficial to build a recommendation model to meet this need. This allows for Anime recommendations to be more personalized, thus increasing customer engagement and in turn revenue.

In our project, we have aimed to build a recommendation system that focuses solely on recommending Anime. This will allow users to make a sophisticated choice that better suits their preference. In order to do so, we have explored 2 systems, them being- the Popularity-based system and the Collaborative Filtering system where we implemented both the User and Item based filtering techniques. In addition, we also performed clustering in order to learn more about the cluster characteristics and the dominant genres present in each cluster. In order to perform clustering, we have employed the K-Means algorithm and performed k-value selection using the elbow method and silhouette scores.

Background Information

A. Dimensionality Reduction

The practice of lowering the number of dimensions, or characteristics, in a dataset, is known as "dimensionality reduction." This can be helpful for a variety of reasons, including making the data more easily observable, using less memory or computing power to evaluate the data, and making it simpler to create models using the data. Principal component analysis, singular value decomposition, and t-distributed stochastic neighbor embedding are just a few of the several methods available for dimensionality reduction (t-SNE).

The method of principal component analysis (PCA) is often used to reduce the number of dimensions. It uses a linear combination of the original dimensions to discover a new set of dimensions since it is a linear procedure. Principal components are the names of these new dimensions, and they are arranged in such a way that the first principal component explains the greatest variation in the data, the second principal component explains the second most variance, and so on. You select the number of dimensions you wish to reduce the data to before projecting the data onto those dimensions to apply PCA for dimensionality reduction. This may make it easier for you to view the data or create models with fewer dimensions.

B. Clustering

K-means clustering is a popular unsupervised machine learning algorithm. The objective of this algorithm is to group similar data points together and discover underlying patterns. A group of data points that share these similar characteristics are said to belong to one such cluster. The question of choosing the parameter k which signifies the number of clusters, now arises. For this, we make use of a technique called the elbow method. We also investigate the silhouette score. The elbow method is a method for calculating the ideal number of clusters in a dataset used in data analysis. It is known as the "elbow approach" because the goal is to identify the location on a graph where the inaccuracy or distortion within a cluster starts to rise quickly. The analyst first shows the number of clusters on the x-axis and the sum of squared errors (SSE) for each cluster on the y-axis before using the elbow approach. Then, at the "elbow" on the graph, when the rate of change in the SSE starts to level out, they select the number of clusters. This is considered the optimal number of clusters because adding more clusters after this point does not significantly reduce the error within the clusters.

A statistic for assessing a dataset's clustering quality is the silhouette score. Its values vary from -1 to 1, with a greater number indicating that a cluster's data points are more similar to one another and more dissimilar from those in other clusters. By averaging the silhouette coefficient of each data point, the silhouette score is determined. The silhouette coefficient is derived as follows:

$$silhouette\_coefficient = (b - a) / max(a, b)$$

In this case, a represents the average distance between a data point and every other data point in the same cluster, whereas b represents the average distance between a data point and every other data point in the cluster that is closest to it. The average of these values is used to get the final silhouette score, which is determined by first calculating the average silhouette coefficient for each data point. A score of 1 denotes complete separation between the data points in each cluster, whereas a score of -1 denotes substantial overlap between the clusters.

C. Word clouds

Visualizations of text data are called word clouds or tag clouds. Each word's magnitude in a word cloud reflects how frequently that word appears in the text. This can be a helpful method for rapidly seeing the most frequent terms in a document and gaining an understanding of its general subject matter. Word clouds are frequently used to summarize and simplify enormous volumes of text data. Additionally, you may use them to draw attention to certain phrases or ideas in the text. Word clouds are frequently used for things like studying research papers, consumer reviews, and social media postings.

D. Collaborative Filtering

Making recommendations using other users' ratings or preferences is known as collaborative filtering. It is predicated on the notion that individuals with comparable ratings or preferences would also have comparable interests, increasing the likelihood that they will enjoy similar items. This concept is the basis for collaborative filtering algorithms, which utilize it to propose products that other users who have similar interests enjoy but haven't yet reviewed or rated for the target user. The usage of collaborative filtering is widespread, with applications in a variety of fields like music, literature, and film. In the realm of recommendation systems, it is a commonly employed strategy.

- **Item-Item Collaborative Filtering**

A kind of collaborative filtering called item-item collaborative filtering concentrates on the connections between various objects rather than the connections between users and items. This strategy is predicated on the notion that if a user consistently rates or buys two goods together, they are likely to be of interest to other users who have previously shown an interest in one of the items. For instance, if a user has highly rated item A and another user has highly rated item A, it is likely that the second user will also be interested in item B. This method is frequently employed in recommendation systems to make recommendations for products that are relevant to a user's current interests.

- **User-User Collaborative Filtering**

A technique for predicting a user's interests based on the interests of other users is called user-user collaborative filtering. It is typical practice in recommendation systems to provide suggestions for products based on a user's prior behavior and the actions of other users who have their interests. This approach is referred to as "collaborative filtering" since it excludes things that are not likely to be of interest to the user and depends on input from a number of users to create predictions. The system initially recognizes a group of users who share the same interests as the target user in order to execute user-user collaborative filtering. Numerous methods may be used to do this, such as collaborative filtering algorithms that determine individuals' similarity score with other users based on their purchases or ratings. Following the discovery of the comparable users, the algorithm can use their ratings or purchases to infer what the target user is most likely to be interested in. For instance, the algorithm may suggest an item to the target user if numerous people with comparable ratings have given it high marks. User-user collaborative filtering is an effective strategy because it considers several users' preferences in addition to the target user's preferences. Compared to other approaches that just take the target user's interests into account, this can offer a more precise and comprehensive set of recommendations. However, because it necessitates examining the actions of several people in order to create predictions, it can also be more computationally demanding.

E. Cosine Similarity

Cosine Similarity measures the similarity between two vectors of an inner product space. Given the vectors $\vec{v}$ and $\vec{w}$, their cosine similarity is calculated by the formula:

$$\text{cosine}(\mathbf{v}, \mathbf{w}) = \frac{\mathbf{v} \cdot \mathbf{w}}{|\mathbf{v}||\mathbf{w}|} = \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

F. Evaluation Metrics

RMSE was used to evaluate the predictions of the ratings given by users in the recommendation system. RMSE shows how far predictions fall from measured true values using Euclidean distance. The formula for RMSE is as follows:

$$\text{RMSE} = \sqrt{\sum \frac{(y_{pred} - y_{ref})^2}{N}}$$

Where:

- y_{pred} is the predicted rating.
- y_{ref} is the actual rating.
- N is the number of observations.

Project Description

Our primary aim is to build a personalized recommendation system that can recommend anime to users based on their previous watch history or their interests. We also aim to group different anime together based on their genre, member count and type. We later compare the results produced by different methodologies used below:

A. Data Cleaning and Preprocessing

The data from the two files were checked to see if any rows contained null values. The Anime data file had 3 columns containing null values. The rows with such columns were removed from the data. The Rating file did not contain any null values.

The rating file contained a user rating of -1 which meant that the user has watched the anime but has not rated it. As we only required data of when a user has rated the anime, we removed any rows with -1 rating.

As the data was large, we removed the rows where the user rating was below the average rating of all the records in the data set. This helped with computation times.

Both datasets were finally merged and used for the rest of the project.

B. K-Means Clustering

A statistical method called principal component analysis (PCA) is used to examine the variation among several variables in a dataset. It is frequently used to decrease the number of dimensions in a dataset, which implies keeping as much information as possible while minimizing the number of variables. This can be helpful for training machine learning models on fewer variables or for displaying high-dimensional data. PCA is a method used to identify patterns in data, to put it simply. Before clustering PCA was done to reduce the number of dimensions required. We selected 3 components as those explained 95% of the variance in the dataset.

Next, clustering has been done on the reduced dataset using K-means clustering. The value k has been chosen using the elbow method that uses inertia as its metric. The figure 1 shows how the ideal number of clusters is 4. This result, however, was validated using another metric called the silhouette score that measures both inter and intra cluster distances, thus measuring/ giving a more accurate representation of the clustering performance. Figure 2 also communicates that the ideal number of clusters is 4. Hence, the reduced dataset has then been clustered into 4 groups.
The goal of performing clustering on our dataset was to recognize the different cluster characteristics as shown in Table 1 under the empirical results section.
In addition, we made use of word clouds to better visualize and learn what the most dominant genres in each cluster were. Figure 7 shows how 'Action' was the most dominant genre across three out of four clusters. For visual purposes, the clusters have been displayed on a two dimensional axis as shown in Figure 3.

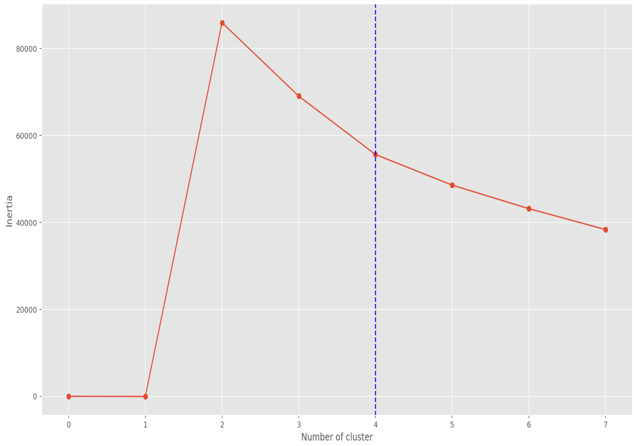

Fig 1: Number of clusters using Inertia

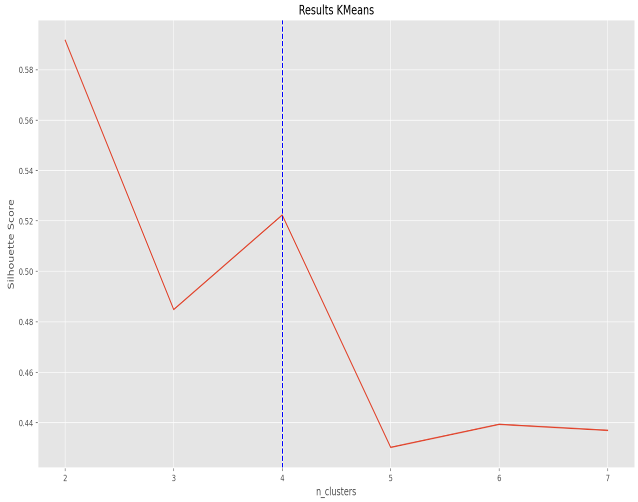

Fig 2: Number of clusters using Silhouette score

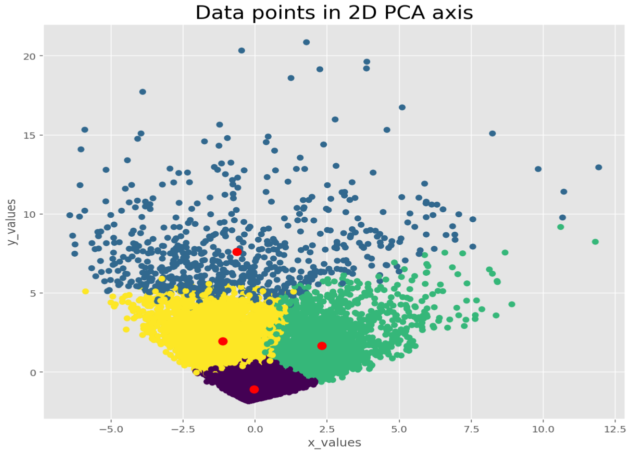

Fig 3: Data points in 2D PCA axis

C. Popularity Based Recommendation

This is the baseline performance and the most intuitive recommendation that we can find anywhere. These recommendations can be found when you are a new joiner and the provider doesn't have enough information about you. So it would be a safe bet to recommend to you what others like. All users get the same recommendation set. It's not personalized.

A weighted rating is calculated to rank the anime. The formula used to calculate the weighted rating is:

$$(\frac{v}{v+m} \times R) + (\frac{m}{v+m} \times C)$$

Where:

- R is the average rating for the item.
- v is the number of votes for the item.
- m is the minimum votes required to be listed in the popular items
- C is the average rating across the whole dataset.

Our project defined the ‘m’ value as anime in the top 95% percentile of total votes. Once the weighted rating was calculated, the anime were sorted based on this rating in descending order. The top anime are the better rated ones and are recommended to users.

D. Collaborative Filtering

Collaborative filtering is the earliest and most popular method for recommendation.It is a technique that recommends items that a user may be interested in based on the reactions of other users.

There are two methods to perform in collaborative filtering:

1. User-Based Collaborative Filtering

User-Based Collaborative Filtering is a technique for predicting products that a user would like based on ratings provided to that item by other users who have similar tastes as the target user.

2. Item-Based Collaborative Filtering

Item-based collaborative filtering is a form of recommendation system that calculates the similarity of items based on the ratings people have provided to them.

The steps taken to perform collaborative filtering are:

- A user-item matrix or pivot matrix was created based on the user-item ratings and normalized the ratings by subtracting mean ratings.
- Users that have not rated any anime were removed.
- Item-item and user-user similarity was calculated based on cosine similarity metrics.
- The top 10 anime and users that were similar were displayed as the result.
- Rating prediction function was created where the input was a user and anime and output was a prediction of what that user would rate the anime.

E. Surprise Library

Surprise is a Python scikit for building and analyzing recommender systems that deal with explicit rating data. KNNWithMeans and SVD algorithms were used to predict user ratings for different anime. GridSearchCV was used to find the best parameters for both algorithms using CV value as 3. The best train and test RMSE was found and compared to see which algorithm worked better for the data.

Empirical Results

A. Data

The dataset used is from Kaggle called “Anime Recommendations Database” which consists of data collected from 76,000 users at myanimelist.net

This data set includes details on user preferences for 12,294 anime across 73,516 users. This data set is a compilation of the ratings that users gave the anime they added to their completed lists.

The dataset consists of two files:

- Anime.csv: This file contains details about the anime like anime_id, name, genre, type, episodes, rating, members.
- Rating.csv: This file contains details about the rating each user has given an anime like user_id - anime_id, rating.

B. Environment

All the implementation is done on Google Colab Pro with HIGH-RAM (27GB CPU) and Jupyter Notebook.

C. Exploratory Data Analysis

To thoroughly analyze and comprehend the data is one of the most important tasks in any machine learning problem. To get a better understanding of our data, we visualized it using different python libraries like matplotlib and seaborn.

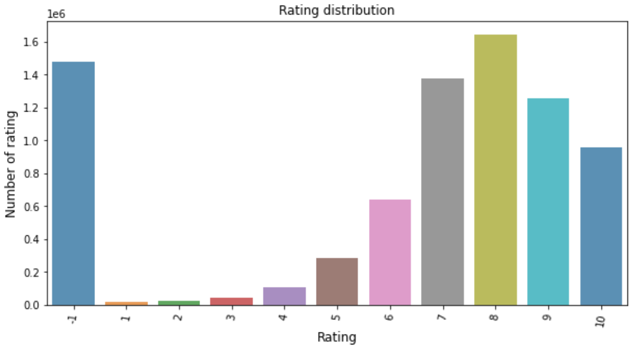

Fig 4: Distribution of ratings in Rating.csv

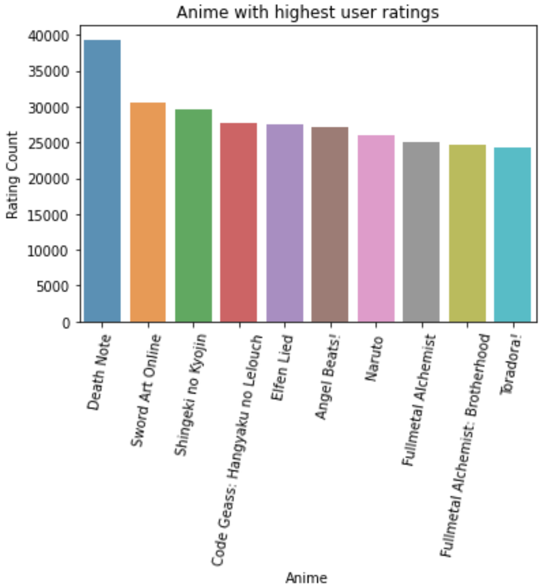

Fig 5: Top 10 anime with highest number of user ratings

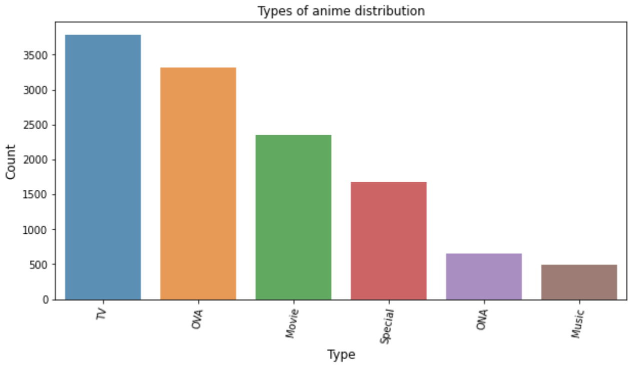

Fig 6: Distribution of anime types

D. Results

Clustering the anime based on their attributes gave us 4 clusters. The average episodes, rating, members and top genre of each cluster was found and is listed in Table 1.

Cluster	0	1	2	3
Average episodes	38.467	23.4	24.13	23.2
Average rating	7.86	8.0	8.0	8.06

Average members	668177	647913	561513	687956

Table 1: Clustering Results

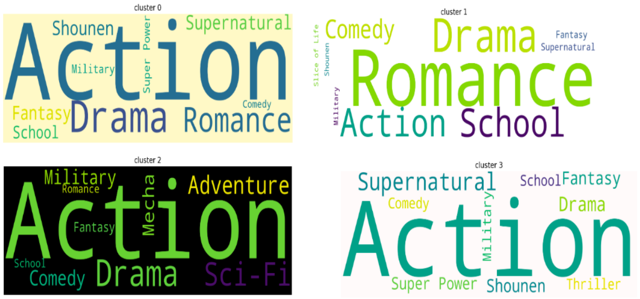

Fig 7: Dominant genre in each cluster using word cloud

After ranking the data based on their weighted rating, the data can be filtered to find different trending/ popular anime. The data was filtered to find the top romance-comedy genre anime and the top TV type anime shown in Fig 8 and 9..

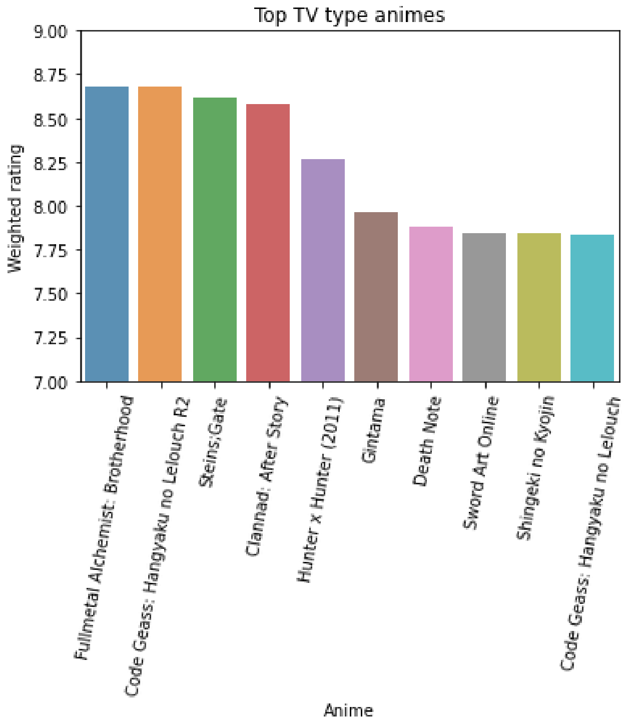

Fig 8: Top TV type anime

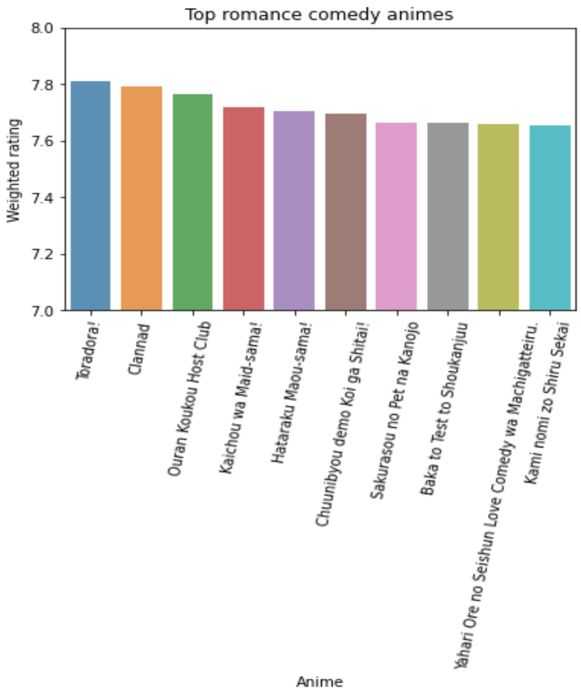

Fig 9: Top romance comedy genre anime

In collaborative filtering, recommendations for the anime "Naruto" were found using both User-User and Item-item CF and the top 10 similar anime were recommended. The user-user CF also gives the most similar users with their similarity score when a user_id is inputted. The top most similar anime to Naruto for each are shown in Table 2.

User-User CF	Item-Item CF
Fairy Tail	Bleach
Bleach	Dragon Ball Z
Naruto: Shippuuden Movie 1	Sword Art Online
The Last: Naruto the Movie	Fairy Tail
Naruto: Shippuuden Movie 3 - Hi no Ishi wo Tsugu Mono	Ao no Exorcist
JK to Inkou Kyoushi 4	Dragon Ball GT
Sword Art Online	Death Note

Naruto: Shippuuden Movie 2 - Kizuna	Dragon Ball
Kangoku: Injoku no Jikkentou	Soul Eater
Kinbaku no Yakata: Ryakudatsu	Shingeki no Kyojin

Table 2: Top 10 similar anime to Naruto using CF

The results of the best train and test RMSE values using SVD and KNNWithMeans and the best parameters found from GridSearchCV can be found in Table 3. We see that KNNWithMeans did better than SVD as the RMSE value is lower.

Algorithm	SVD	KNNWithMeans
Best Parameters	{'n_epochs': 4, 'lr_all': 0.005, 'reg_all': 0.2}	{'name': 'cosine', 'user_based': False}, 'k': 10}
Train RMSE	0.18106	0.02959
Test RMSE	0.1762	0.0257

Table 3: Surprise Library algorithm RMSE

Conclusion and Future Work

Including user demographic data in a user-user collaborative filtering system may enhance the precision of the suggestions. This is so that the algorithm can more accurately forecast what products they would enjoy. Demographic information can give extra context about the consumers and their interests. In general, by offering more context about the users and their interests, adding demographic information to a user-user collaborative filtering system can assist increase the relevance and accuracy of the suggestions. When using demographic data in this way, it's crucial to take questions of fairness and privacy into account. Another aspect would be to treat the ratings -1(indicating that a user has watched something but not rated it) such that their values were predicted by the recommendation system. This would help enhance the data. Adding onto data issues- a few anime had their genre attribute missing. These could be added in manually in order to gain more information.

We would also like to explore: (i) Building a UI such as a website to host our model on for real-time interaction. (ii) Neural Networks Architecture and Deep Learning models that use pre-trained network weights to obtain better results.

The link to the project repository on github can be accessed via: **https://github.com/saachi-c/Anime-Recommendation-System**

References

[1]https://www.kaggle.com/datasets/CooperUnion/anime-recommendations-database

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

[3]https://towardsdatascience.com/recommendation-systems-explained-a42fc60591ed

[4] Chen, L., Chen, G., and Wang, F. (2015). Recommender systems based on user reviews: the state of the art. User Modeling and User-Adapted Interaction, 25, 99-154.

[5]https://www.analyticsvidhya.com/blog/2022/02/introduction-to-collaborative-filtering/

[6] https://surpriselib.com/

[7] Thorat, P. B., Goudar, R. M., & Barve, S. (2015). Survey on collaborative filtering, content-based filtering and hybrid recommendation system. *International Journal of Computer Applications*, *110*(4), 31-36.

[8]https://www.geeksforgeeks.org/recommendation-system-in-python/

[9] Pandya, S., Shah, J., Joshi, N., Ghayvat, H., Mukhopadhyay, S. C., & Yap, M. H. (2016, November). A novel hybrid based recommendation system based on clustering and association mining. In *2016 10th international conference on sensing technology (ICST)* (pp. 1-6). IEEE.

[10]https://www.datarevenue.com/en-blog/building-a-production-ready-recommendation-system

[11] https://en.citizendium.org/wiki/Recommendation system