Indian Institute of Technology Kanpur

Department of Mathematics and Statistics

MTH522A

Building An Anime Recommendation System

Author:
Shreya Pramanik
Arkonil Dhar

Supervisor:
Dr. Amit Mitra

April, 2022

Abstract

Recommendation system is an extensive class of web applications that seeks to predict the 'rating' or the 'preference' an user might give to an item. These are simple algorithms to filter relevant items from a huge pool of information base. Some real time applications of recommendation systems are like Amazon, Flipkart where they use recommendation engines to suggest products customers might like. In our project we aimed to build an *Anime Recommendation System*. With a growing anime fanbase around the world numerous streaming websites racing to catch up the market using sophisticated recommendation engines. We worked with the user data and rating data from one of the most popular anime rating website *MyAnimeList*. Taking Collaborative Filtering approach we sought to find similarity among users and predict ratings accordingly. We used MAP@K score for measuring performance of our rating prediction.

Contents

1 Introduction

Anime is a hand-drawn and computer-generated animation originating from Japan. Since 1980s this animation style has seen a rise in international success due to foreign dubbing and subtitles and increasing distribution through streaming services. Some of the major anime studios *Studio Ghibli, MAPPA, Toho CoMix Wave Films* has produced some of the highest grossing films *Spirited Away, Your name, Howl's Moving Castle* directed by Hayao Miazaki and Makoto Shinkai. With more access of internet streaming service around the world now anime movies and series are not only limited to TV shows. We can now enjoy Anime in many paid online platforms such as Netflix, Disney+, Hulu, Crunchyroll.

Problem Statement

With the rise of online streaming services these websites are trying to boost their revenue and increase the time spent on the website. As there are many such options available, a site needs to provide a good user friendly environment to retain the viewership. So this websites are trying to recommend the viewer shows of their liking based on the genre of shows they've already watched or what other viewers are watching. Otherwise one may scroll thousands of animes never to find content of their liking. Several anime rating websites come handy for this purpose. One of such prominent rating website is *MyAnimeList* as IMDb for movie ratings.

Proposed Methodology

- We use MyAnimeList anime data and user ratings data.
- Visualize and preprocess the data.
- Build a sparse matrix of anime and users.
- Use Collaborative Filtering to build recommendation system.

2 Data Description

We have used the anime and user rating data from Kaggle [Data Link]. The data is scrapped from MyAnimeList. The Dataset contains information on user preference data from 325,772 users on 17,562 anime with a total of 109 Million rating data. Each user is able to add anime to their completed list and give it a rating. This dataset is a compilation of these ratings. We have used `anime.csv` and `animelist.csv` for our recommendation system.

Description of anime.csv:

This file contains general information such as genre, stats, studios etc for every anime. It has following columns:

- **name** — full name of the anime.

- **score** — average score of the anime given from all users in MyAnimelist database.
- **genres** — comma seperated list of genres such as Action, Adventure, Comedy, Drama, Sci-Fi, Space for the anime.
- **type** — medium of streaming TV, movie, OVA, etc.
- **episodes** — how many episodes are on the show.
- **studios** — comma separated list of studios.
- **rating** — age rate (e.g. R - 17+ (violence and profanity)).
- **score-1** — number of users who rated 1.
- **score-2** — number of users who rated 2.
- **score-3** — number of users who rated 3.
- **score-4** — number of users who rated 4.
- **score-5** — number of users who rated 5.
- **score-6** — number of users who rated 6.
- **score-7** — number of users who rated 7.
- **score-8** — number of users who rated 8.
- **score-9** — number of users who rated 9.
- **score-10** — number of users who rated 10.

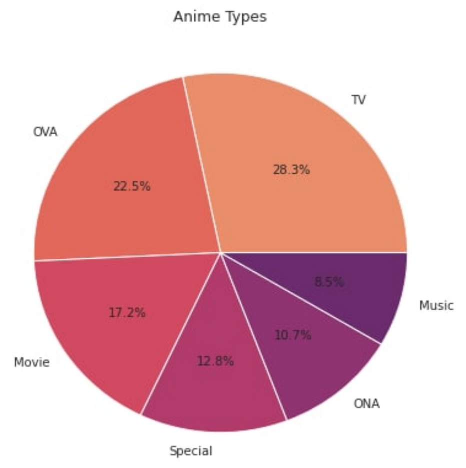
Description of animelist.csv:

This file has the list of all animes registered by the user with respective score. This dataset contains 109 Million row, 17.562 different animes and 325.772 different users. It has the following columns:

- **anime ID** — MyAnemlist's unique ID identifying an anime.
- **user ID** — non identifiable randomly generated user id.
- **rating** — score between 1 to 10 given by the user. 0 if the user didn't assign a score.

3 Visualization and Preprocessing of the Dataset

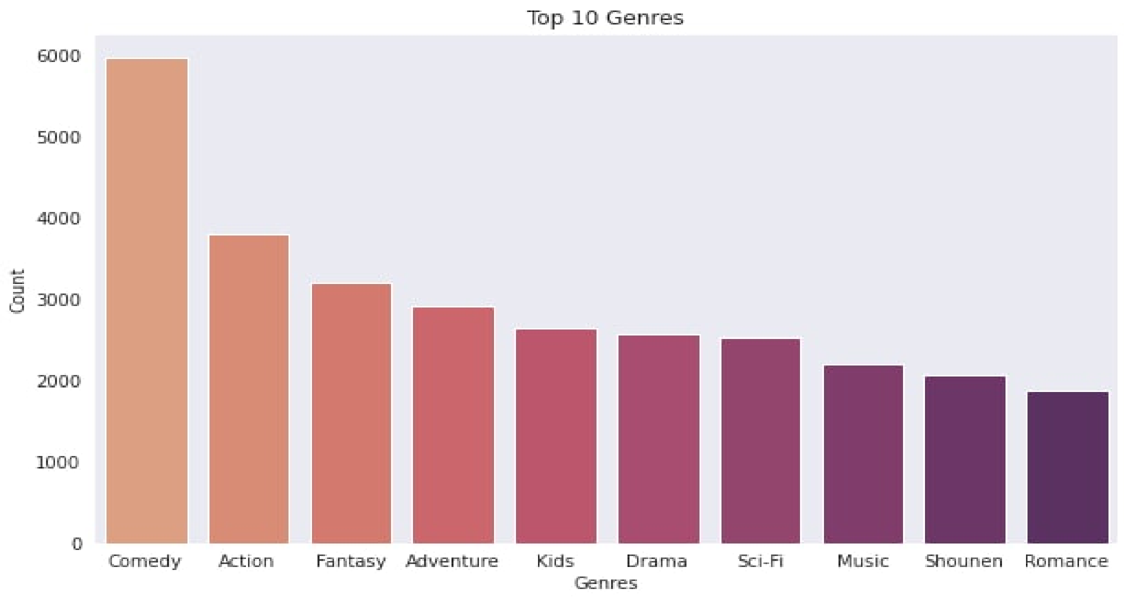
Now we try to gain insights about medium of streaming of the animes from the plot below:



Insights:

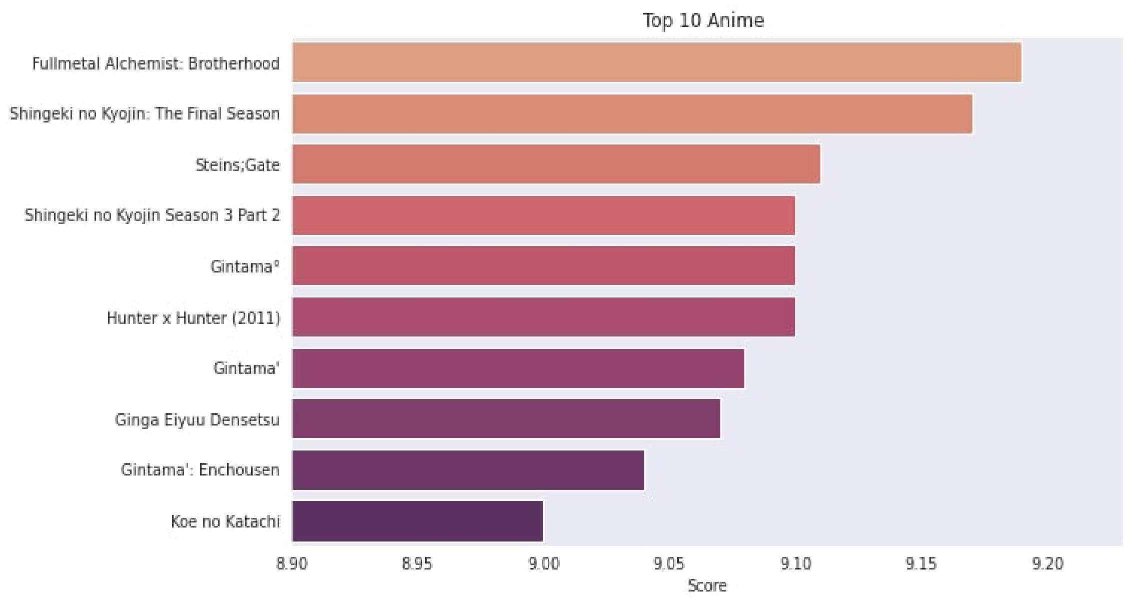
- 28.3% animes were wired on TV followed by 22.5 % through OVA and 17.2 % through movie.
- 10.7 % were streamed through ONA which is less than OVA(22.5 %).

Also We observe 43 genres in our dataset. Based on that we plot top 10 anime genres based on score and top 10 animes based on rating count.



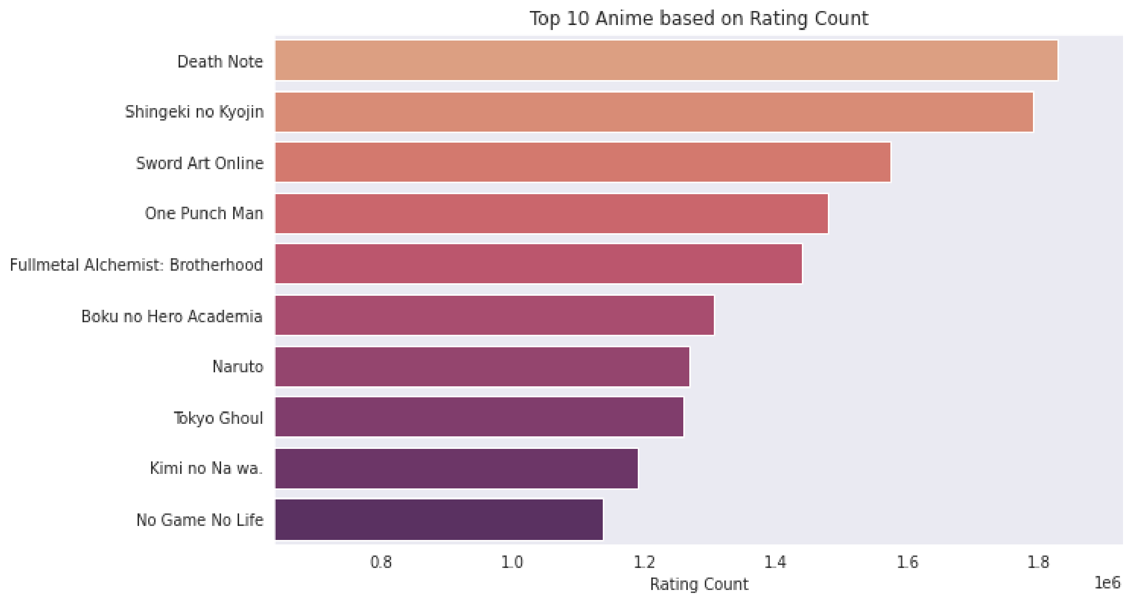
Insights:

- Comedy is the most popular genre followed Action and Fantasy.



Insights:

- Fullmental Alchemist:Brotherhood is enthroned as the most popular anime based on average score.



Insights:

- Death Note takes the glory of top anime based on rating count.

For ratings data from `animelist.csv` we discard those which user is planning to watch and also consider the ones with rating greater than 0(i.e. 1-10). After this we were left with a total of 61,626,739 rating data.

4 Building Recommendation System

Collaborative Filtering

Collaborative filtering is the process of filtering out contents a user might like based on taste of similar users. In this type of recommendation system, we classify the users into the class of similar types and recommend each user according to the preference of its class.

So we need to search for answer of the followings:

- How to determine similarity between users or items.
- Once we find which users are similar, immediate question arises how to determine the rating that a user would give to an item based on the ratings of similar users.
- Finally how to measure the accuracy of the rating we calculate.

To answer the first question one approach is to calculate similarity using angle between users. Here we consider cosine similarity metric to serve our purpose.

Cosine Similarity using KNN

Cosine similarity is a metric to measure cosine of the angle between two vectors projected in a multi-dimensional space. If the angle between the vectors is increased, then the similarity decreases, and if the angle is zero, then the users are very similar. The cosine similarity is advantageous because even if the two similar users are far apart by the Euclidean distance, chances are they may still be oriented closer together.

Now that we know how to find similar users we look into how we can predict the ratings based on those user ratings. For that we sort users of the training data according to their cosine similarities and take average ratings of the first k most similar users for each anime.

Let the user training vectors are $\underset{\sim}{u}_1, \underset{\sim}{u}_2, ..., \underset{\sim}{u}_n$ and a user test observation is $\underset{\sim}{u}^*$. We define cosine similarity for the user as:

$$c_i = \frac{\underset{\sim}{u}_i^T \underset{\sim}{u}^*}{||\underset{\sim}{u}_i|| \, ||\underset{\sim}{u}^*||}$$

Then we sort these cosine similarities and denote them as $c_{(1)}, c_{(2)}, ..., c_{(n)}$ and the corresponding user data as $\underset{\sim}{u}_{(i)}$. Now we take the first k most similar users and predict the rating using:

$$\hat{r}_j^* = \frac{1}{k} \sum_{l=n-k+1}^{n} u_{(l)}^{(j)} \text{ , for } j^{th} \text{ anime.}$$

We can also use cosine similarities as weights and predict final rating by taking weighted average of all the users as follows:

$$\hat{r}_j^* = \frac{1}{N(j)} (c_1 \underset{\sim}{\mu}_1 + c_2 \underset{\sim}{\mu}_2 + ... + c_n \underset{\sim}{\mu}_n)$$

where $N(j)$ is the number of available ratings for the j$^{\text{th}}$ anime. In this project We have used the weighted average as the predicted ratings.

Now we can recommend those animes for which the predicted ratings are the highest. We can choose K highest rated animes to recommend to the user.

Mean Average Precision at K (MAP@K)

Finally to measure the performance of the rating prediction we used MAP@K score.

In a binary classification problem the precision and recall are defined as,

$$P = \frac{\text{Number of correct predicted positives}}{\text{Number of predicted positives}}$$
$$R = \frac{\text{Number of correct predicted positives}}{\text{Number of actual positives}}$$

In case of recommender systems, we defined those as,

$$P = \frac{\text{Number of reccomended items that are actually relevant}}{\text{Number of items recommended}}$$
$$R = \frac{\text{Number of reccomended items that are actually relevant}}{\text{Number of relevant items}}$$

Here by "relevant" we mean those items that are rated or watched by the user. We can use the precision and recall described above and calculate the f-score. But one drawback of doing so is that it does not take into account the ordering of the predicted ratings.

One way to overcome this is to calculate the precision at each k and take the average of those $P(k)$'s for which the k^{th} item is relevant. That is, if the recommendation system recommends 10 top most rated items then for each $k = 1(1)10$ we calculate the precision for the set of first k recommended items, if the k^{th} item is relevant and take their average.

Hence, the formula for Average Precision at K (AP@K) is,

$$\begin{aligned} AP@K &= \frac{1}{m}\sum_{k=1}^{K}\left\{P(k) \text{ if the } k^{th} \text{ item is relevant}\right\} \\ &= \frac{1}{m}\sum_{k=1}^{K} P(k) * rel(k) \end{aligned}$$

where, m is the total number of relevant items and,

$$rel(k) = \begin{cases} 1, & \text{if the } k^{th} \text{ item is relevant} \\ 0, & \text{otherwise} \end{cases}$$

Now if we take the the average of the AP@K values for all the test users then we get the Mean Average Precision at K (MAP@K), i.e.,

$$MAP@K = \frac{\text{AP@K}}{\text{Size of the test data}}$$

5 Results

To assess performance of our model we take three sets of data: users who have done at least 1 rating, users who have done more than 50 ratings and users who have done more than 100 ratings. The MAP@K for the datasets are obtained as 0.718, 0.745, 0.753 respectively. So we can conclude that our model performance improves with the increase in number of ratings by an user.

6 References

- Wikipedia
- Anime Recommendation Engine using Content and Collaborative Filtering
- Build Recommendation Engine - Collaborative Filtering
- Mean Average Precision MAP for Recommender Systems