Project by Satyam Jha

RECOMMENDATION SYSTEM

UNSUPERVISED LEARNING - ANIME DATASET

Link: https://github.com/Satyamjha991/Anime-Reccomendation

Introduction

Every streaming content has its own viewers and each content has its rating. Viewers leave some good ratings for the content if they like it. But where does it apply? Viewers can spend hours scrolling through hundreds, sometimes thousands of anime's, never finding any content they like. Businesses need to provide suggestions based on their likes and needs in order to create a better streaming environment that boosts revenue and increases the time spent on a website.

With the ability of machine learning, we explore the dataset to create a recommendation system with the ability to help someone to find what to watch next.

What is a recommendation system?

Recommender systems are machine learning systems that help users discover new products and services. Every time you shop online, a recommendation system is guiding you towards the most likely product you might purchase.

Recommender systems are an essential feature in our digital world, as users are often overwhelmed by choice and need help finding what they're looking for. This leads to happier customers and, of course, more sales. Recommender systems are like salesmen who know, based on your history and preferences, what you like.

What is anime?

Anime is a hand-drawn computer animation originating from Japan that has drawn a cult following around the world. The animation industry consists of more than 430 companies. Pokemon and Doremon are some of the most popular anime show that has come to Indian television. Over recent years, the popularity of anime and its comic strip counterpart manga has grown considerably.

One of the main reasons why anime has stood the test of time and grown in popularity across the world is due to its unique ability to grow with its viewers. The famous anime expert, Takamasa Sakurai, claims that the genre has been widely accepted due to its unconventional nature, "Japanese anime broke the convention that anime is something that kids watch". Overseas fans of anime claim that they enjoy the intensity of the storylines with the endings being difficult to predict as anime is often targeted at adult audiences.

Below, you can see my step-by-step guide to a recommendation system for anime. I have downloaded anime and user ratings from: https://github.com/Hernan4444/MyAnimeList-Database

About Dataset

- ➔ Ratings are given by users to the anime that they have watched completely.
- ➔ Information about the anime like genre, stats, studio, etc.
- ➔ Synopsis of the anime as per myanimelist.com

In order to build a recommendation engine, we have to understand our dataset.

Challenges when Building a Recommendation System

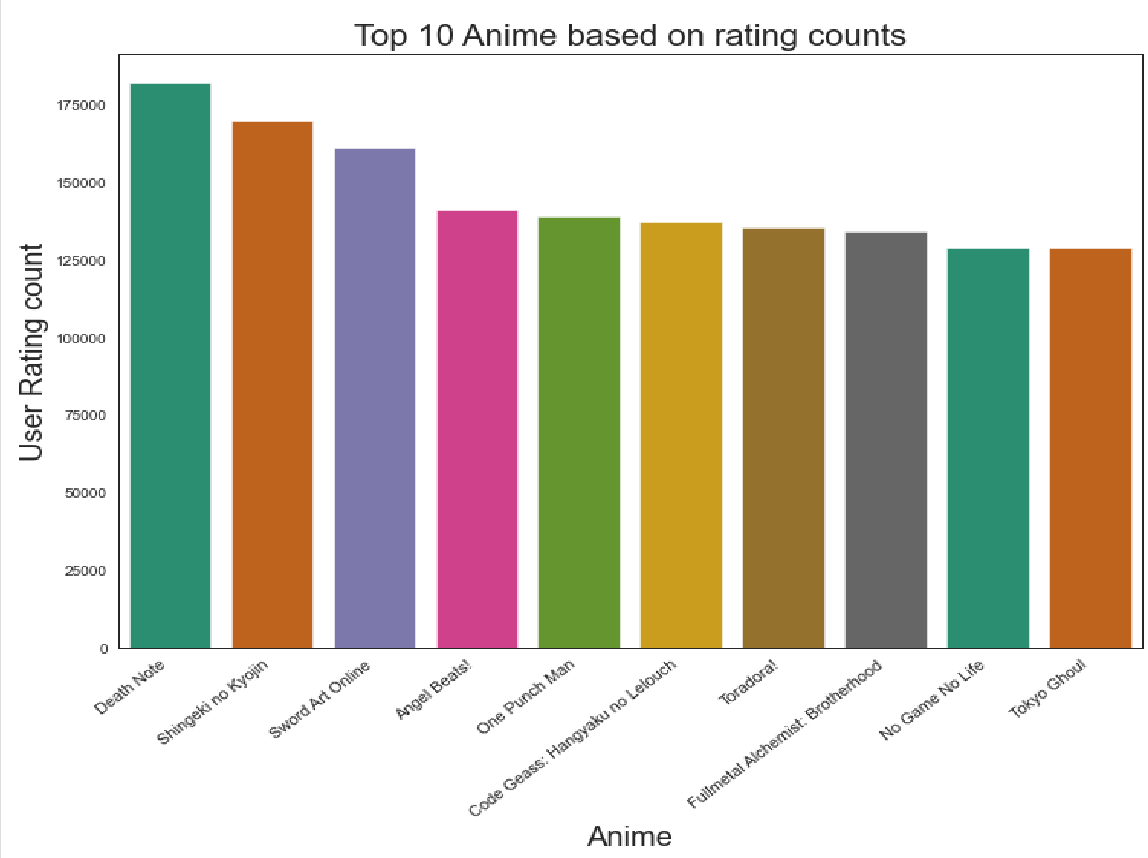
- Data Sparsity: There are lots of products to recommend to many users and it is unlikely that a user will ever try out a large fraction of products. Instead, probably a few items are demanded by many users, but many only by a few.
- Cold Start: We need to be able to give recommendations to users about which we only have scarce data (if at all).
- Accurate, but diverse predictions: We want to give useful recommendations in the sense that they match the user's preferences, but also that the recommendation contains some novelty for the user.
- Scalability: We need to be able to give recommendations on the spot even though there might be millions of users and items which we have to analyze carefully.
- User interface: Users want to know why they get particular recommendations.
- Vulnerability to attacks: We do not want our recommendation system to be abused for promoting or inhibiting particular items.
- Temporal resolution: Tastes and preferences do not remain the same over time.
- Evaluation: Evaluation is difficult and might differ from algorithm to algorithm.

This dataset contains information about 17,562 anime and the preference from 57,633,278 different users.

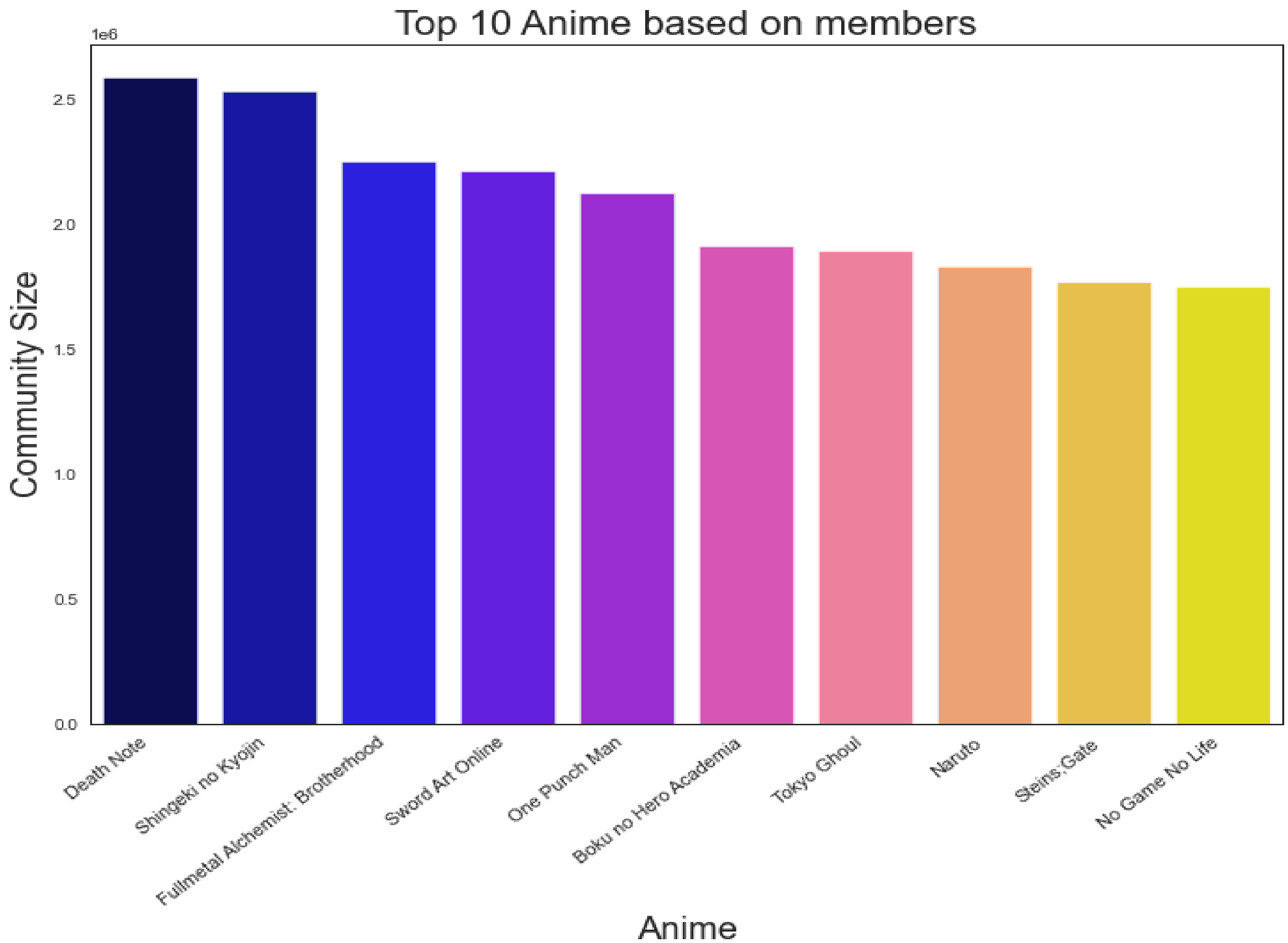
Analysis Results

➔ Top anime based on:

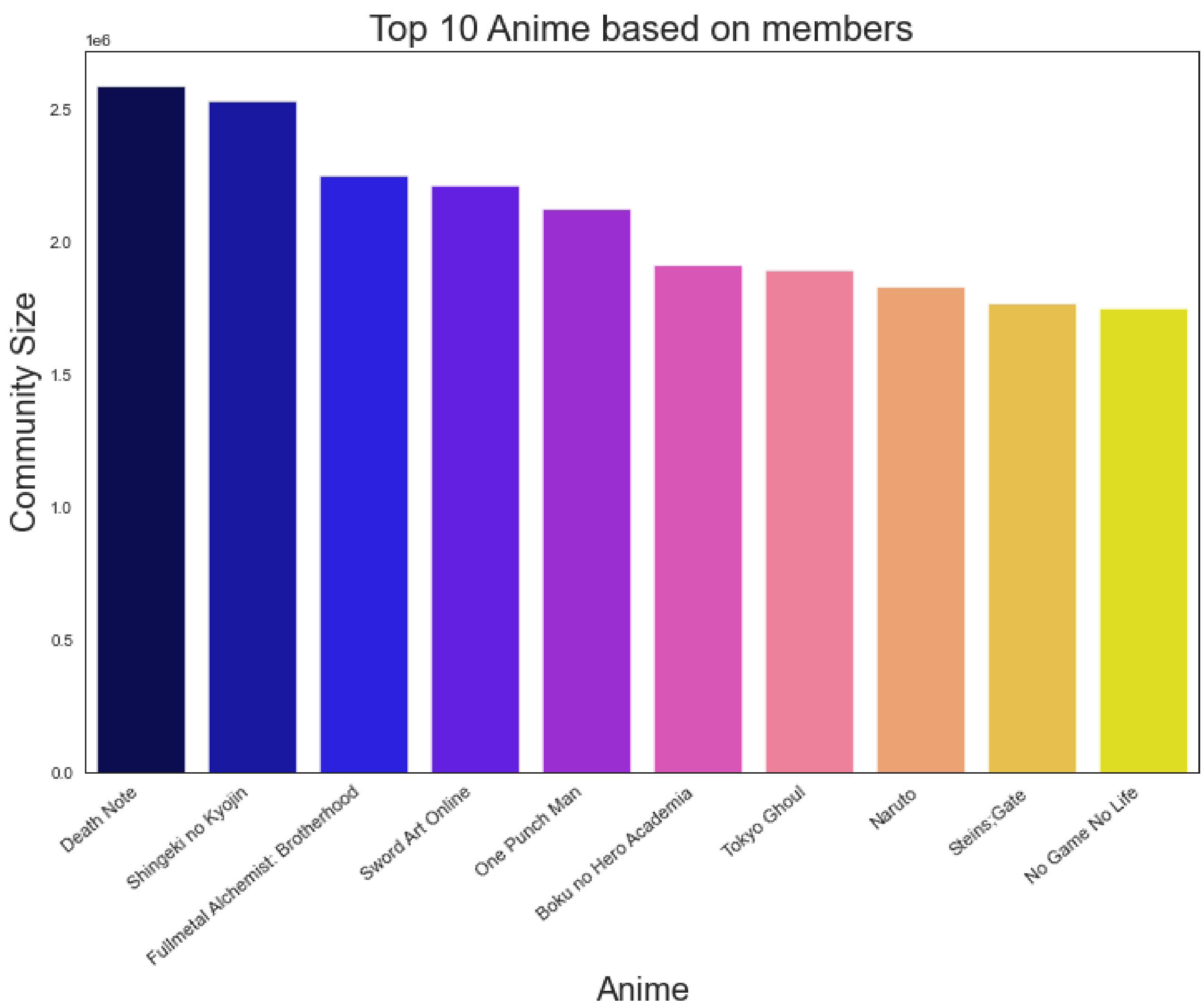
- Rating Counts: DeathNote Wears the Crown



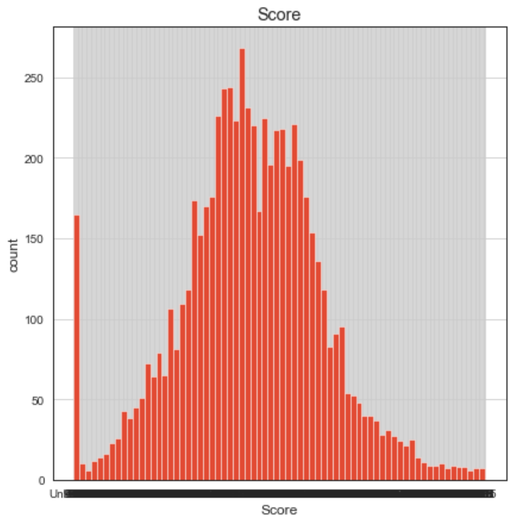
- Members count: DeathNote again

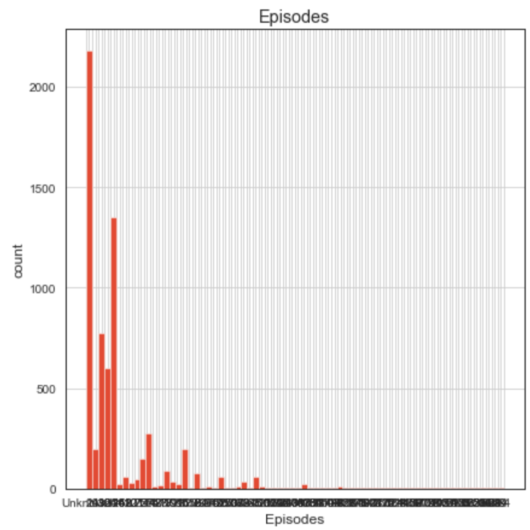


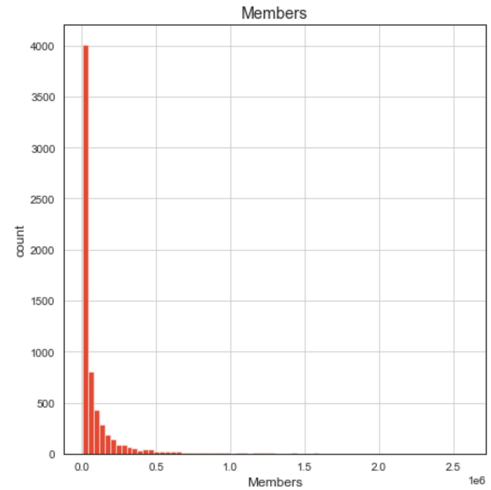
- Loved by community:

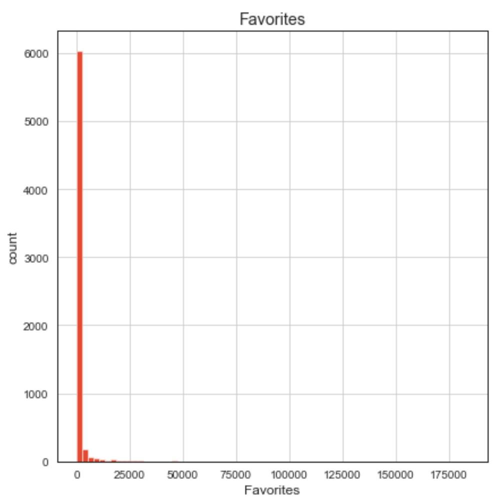


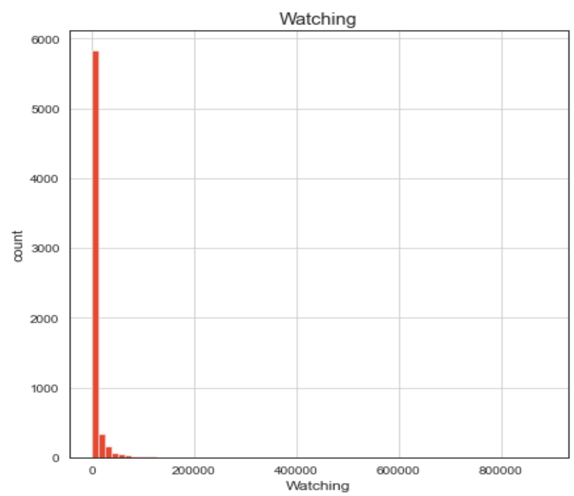
Data Distribution:

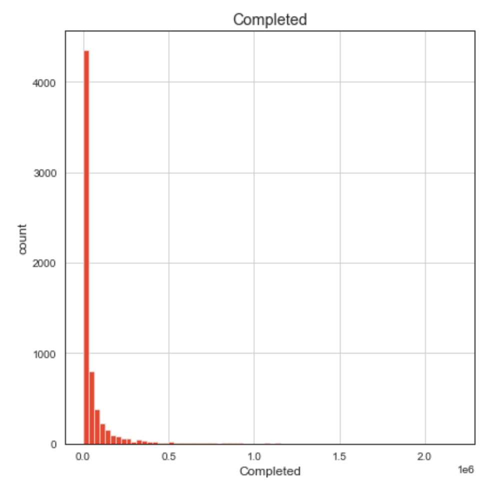




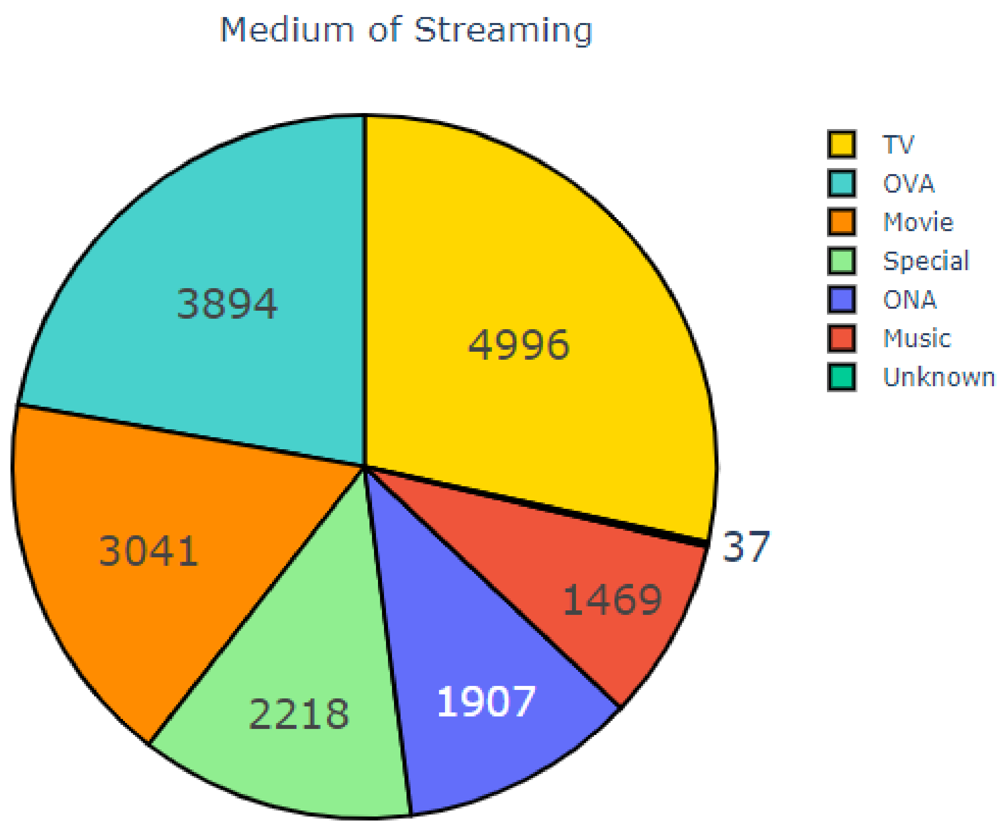








Medium of streaming



Insights:

- Most of the ratings are spread between 4-7.
- The mode of the distribution is around 7.5-8.0
- We don't have nan for the Members, Favorites, Watching, Completed, Plan to Watch.
- We have nan values as Unknown for Score and Episodes.
- 28.4% of the anime were aired on TV followed by 13.5% through Movie.
- 22.2% of anime are streamed as OVA which is greater than Movies(17.3%).
- rest are distributed in Special(12.6%), ONA(10.9%), Music(8.36%)

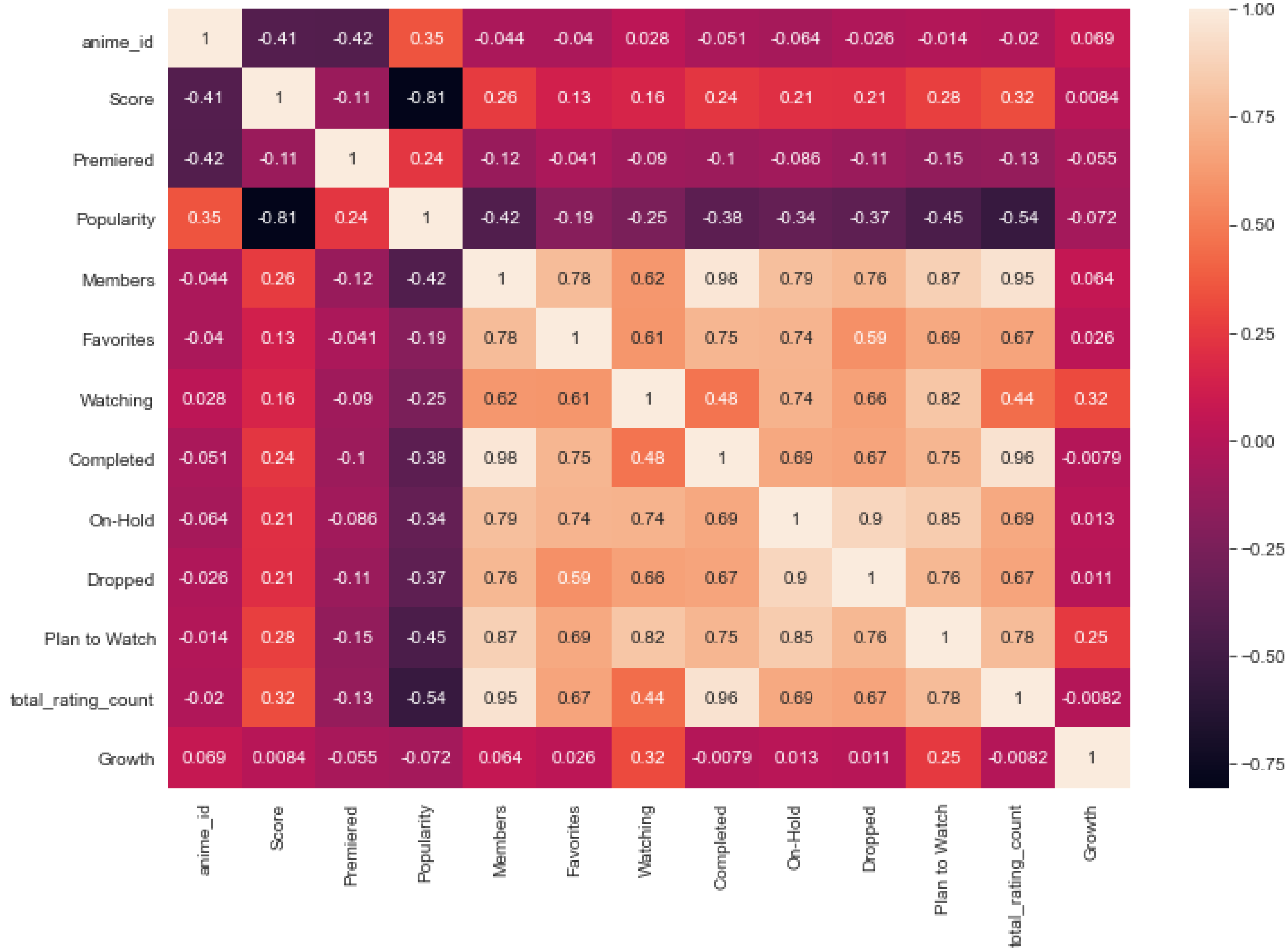
Features:

- Title Data: MAL_ID, Name, English Name, Japanese Name
- Categorical Data: Type, Genders, Episodes, Producers, Licensors, Studios, Source, Rating
- Numerical: Popularity, Members, Watching, Completed, On-Hold, Dropped, Plan to Watch, Score-i

In the Dataset, the nan values are written as "Unknown". For an unknown score, we have given 0 and for Episodes, we have given 13. For the premiere, we have calculated the unknown values from the Aired data and subtracted them with 2021 to get the feature containing how old the anime is (premiered).

We have created 2 new features 1) Growth=members/premiered 2) total rating counts

Relation between the features



Hence we have dropped Features accordingly and combined a few.

Simple Recommendation System

Approach:

- The Simple Recommender offers generalized recommendations to every user based on movie popularity and (sometimes) genre.
- The basic idea behind this recommender is that movies that are more popular and more critically acclaimed will have a higher probability of being liked by the average audience.
- This model does not give personalized recommendations based on the user.

What we are actually doing:

- The implementation of this model is extremely trivial.
- All we have to do is sort our movies based on ratings and popularity and display the top movies of our list.
- As an added step, we can pass in a genre argument to get the top movies of a particular genre.
- I will build our overall Top 250 Chart and will define a function to build charts for a particular genre. Let's begin!
- I use the Score to come up with our Top Anime Chart.
- I will use IMDB's weighted rating formula to construct my chart. Mathematically, it is represented as follows:

$$Weighted\ Rating(WR) = \left(\frac{v}{v+m} . R\right) + \left(\frac{m}{v+m} . C\right)$$

where,

- v is the number of votes/members/likers for the movie
- m is the minimum votes required to be listed in the chart
- R is the average Score of the data
- C is the mean vote across the whole Data

We build the chart on

- Members RMSE: 0.416884263572696
- Favourites RMSE: 0.34965154792583325
- Total Votes RMSE: 0.4005204588883558

Hence we conclude that the best result is obtained by a rating based on favorites

```
: build_chart_on_favourites('School').head(10)[["Name","English name","Score"]]
```

:

	Name	English name	Score
10	Koe no Katachi	A Silent Voice	9.00
15	Kimi no Na wa.	Your Name.	8.96
19	Haikyuu!!: Karasuno Koukou vs. Shiratorizawa G...	Haikyu!! 3rd Season	8.87
32	Shigatsu wa Kimi no Uso	Your Lie in April	8.74
35	Kaguya-sama wa Kokurasetai?: Tensai-tachi no R...	Kaguya-sama:Love is War Season 2	8.74
37	Haikyuu!! Second Season	Haikyu!! 2nd Season	8.73
41	Code Geass: Hangyaku no Lelouch	Code Geass:Lelouch of the Rebellion	8.72
43	Great Teacher Onizuka	Great Teacher Onizuka	8.70
45	Seishun Buta Yarou wa Yumemiru Shoujo no Yume ...	Rascal Does Not Dream of a Dreaming Girl	8.68
53	Suzumiya Haruhi no Shoushitsu	The Disappearance of Haruhi Suzumiya	8.65

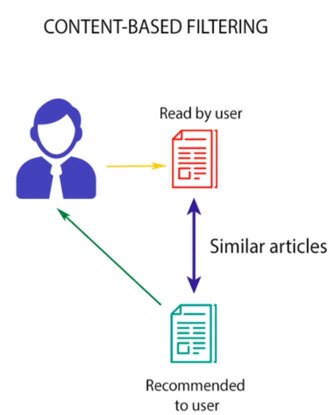
Content-based recommendation system

This algorithm recommends products that are similar to the ones that a user has liked in the past. The content of each item is represented as a set of descriptors or terms, typically the words that occur in a document.A content-based recommender works with data that the user provides, either explicitly (rating) or implicitly (clicking on a link). Based on that data, a user profile is generated, which is then used to make suggestions to the user. As the user provides more inputs or takes actions on the recommendations, the engine becomes more and more accurate

We do not have a quantitative metric to judge our machine's performance so this will have to be done qualitatively. We got many symbols found in anime_title hence we perform the necessary cleaning.

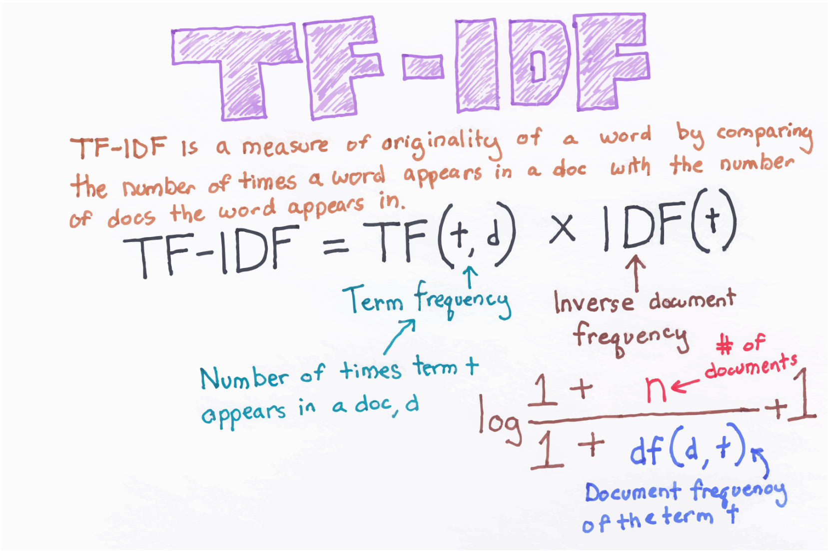
Here, there are various approaches to solve this problem and I have applied them keeping in mind their usage in the industry. They are broadly based on 2 categories:

1) User-Based: Takes user_id to get the recommendation
2) Item Based: Give recommendation based on the search



We have got the title cleaned and neat. Now it's time for the ultimate TFIDF to recommend us the next anime

- Term Frequency (TF) and Inverse Document Frequency (IDF)



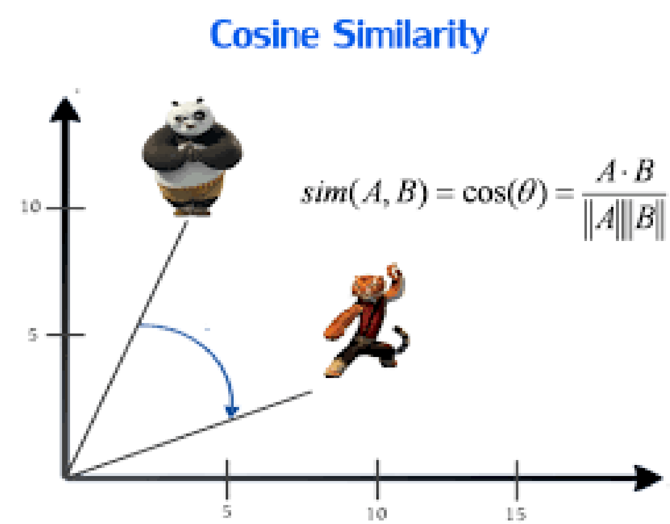
It is certain that "the" will occur more frequently than "Titans" but the relative importance of analytics is higher than the search query point of view. In such cases, TF-IDF weighting negates the effect of high-frequency words in determining the importance of an item (document).

We use cosine similarity to find the relation

The cosine similarity metric measures the cosine of the angle between two n-dimensional vectors projected in a multi-dimensional space. The Cosine similarity of the two documents will range from **0 to 1**. If the Cosine similarity score is **1**, it means two vectors have the same orientation. The value closer to 0 indicates that the two documents have less similarity.

The mathematical equation of Cosine similarity between two non-zero vectors is:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2}\sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$



The **Cosine Similarity** is a better metric than ***Euclidean distance*** because if the two texts document far apart by *Euclidean* distance, there are still chances that they are close to each other in terms of their context.

ITEM BASED: Way 0

- ➔ We apply TF-IDF to the genre to make the matrix.
- ➔ Find the cosine similarities between them.
- ➔ Get top 30 recommendations based on the anime we search

```
pd.DataFrame(get_rec('Naruto: Shippuuden').head(10))
```

	Anime name	Rating
0	Naruto	7.91
1	Boruto: Jump Festa 2016 Special	6.22
2	Dragon Ball Kai (2014)	7.69
3	Dragon Ball Super	7.42
4	Dragon Ball Z Movie 15: Fukkatsu no "F"	7.11
5	Dragon Ball Z: Summer Vacation Special	6.62
6	Dragon Ball GT: Gokuu Gaiden! Yuuki no Akashi ...	6.54
7	Dragon Ball Z: Atsumare! Gokuu World	6.48
8	Naruto: Shippuuden Movie 6 - Road to Ninja	7.67
9	Naruto: Shippuuden - Sunny Side Battle	7.43

: **ITEM-BASED: WAY 1**

In the Previous result:

- We see that for Naruto, our system is able to identify it is from Naruto Franchise and Shounen anime and subsequently recommend anime from this franchise and other Shounen anime as its top recommendations.
- But unfortunately, that is all this system can do at the moment.
- This is not of much use to most people as it doesn't take into considerations very important features such as Type of anime, Studio, Source, and Synopsis which determine the rating and the popularity of an anime.
- Someone who liked Naruto likes it more because of the Source and would hate Boruto Forever and every other substandard anime in the Naruto Franchise.
- Therefore, we are going to use much more suggestive metadata than Overview and Tagline.
- In the next subsection, we will build a more sophisticated recommender that takes genre, keywords, Studio, Synopsis etc., into consideration.
- To build our standard metadata-based content recommender, we will need to merge our current dataset with other features.

We use the following data to make a Combined Feature:

- Genders
- Type
- Producers
- Licensors
- Studios
- Source
- Synopsis
- English Name

Result:

```
: pd.DataFrame(get_rec('Naruto: Shippuuden').head(10))
```

:

	Anime name	Rating
0	Naruto	7.91
1	Boruto: Naruto Next Generations	5.81
2	Boruto: Naruto the Movie - Naruto ga Hokage ni...	7.40
3	Naruto: Shippuuden Movie 6 - Road to Ninja	7.67
4	Naruto: Honoo no Chuunin Shiken! Naruto vs. Ko...	7.16
5	Naruto: Shippuuden Movie 1	7.29
6	Naruto: Shippuuden Movie 2 - Kizuna	7.29
7	Naruto: Dai Katsugeki!! Yuki Hime Shinobu Houj...	6.87
8	Naruto: Shippuuden Movie 4 - The Lost Tower	7.42
9	Naruto SD: Rock Lee no Seishun Full-Power Ninden	7.14

- I am much more satisfied with the results I get this time around. The recommendations seem to have recognized other Naruto Anime (due to the high weightage given to the word "Naruto") and put them as top recommendations.
- I enjoyed watching Naruo as well as some of the movies on the list including Naruto: Shippuuden Movie 6 - Road to Ninja and Naruto: Shippuuden Movie 2 - Kizuna.

ITEM-BASED: WAY 2

Add Popularity and Ratings

- One thing that we notice about our recommendation system is that it recommends anime regardless of ratings and popularity. It is true that Naruto and Boruto has a lot of similar characters as compared to Naruto Shippuden but
- It was a terrible anime that shouldn't be recommended to anyone.
- Therefore, we will add a mechanism to remove bad anime and return anime that are popular and have had a good critical response.
- I will take the top 30 anime based on similarity scores and calculate the vote of the 50th percentile movie. Then, using this as the value of *m*, we will calculate the weighted rating of each movie using IMDB's formula as we did in the Simple Recommender section.
- Unfortunately, Boruto does not disappear from our recommendation list.
- This is probably due to the fact that it is rated a 7.76, which is only slightly above average on myanimelist.
- It certainly doesn't deserve a 7 when amazing anime like Bleach has only a 7.8.
- However, there is nothing much we can do about this.

Result:

	Name	Score	Genders	English name	wr
659	Naruto	7.91	Action, Adventure, Comedy, Super Power, Martia...	Naruto	7.829251
818	Bleach	7.80	Action, Adventure, Comedy, Super Power, Supern...	Bleach	7.672378
893	The Last: Naruto the Movie	7.76	Action, Super Power, Romance, Martial Arts, Sh...	The Last:Naruto the Movie	7.570269
1107	Naruto: Shippuuden Movie 6 - Road to Ninja	7.67	Action, Adventure, Super Power, Martial Arts, ...	Road to Ninja:Naruto the Movie	7.474668
1566	Boruto: Naruto the Movie	7.50	Action, Comedy, Martial Arts, Shounen, Super P...	Boruto:Naruto the Movie	7.381423
1701	Naruto: Shippuuden Movie 5 - Blood Prison	7.46	Action, Adventure, Martial Arts, Super Power, ...	Unknown	7.325430
1830	Naruto: Shippuuden Movie 4 - The Lost Tower	7.42	Action, Comedy, Martial Arts, Shounen, Super P...	Unknown	7.309718
2104	Naruto: Shippuuden Movie 3 - Hi no Ishi wo Tsu...	7.35	Action, Comedy, Martial Arts, Shounen, Super P...	Unknown	7.266627
2394	Naruto: Shippuuden Movie 1	7.29	Action, Adventure, Comedy, Fantasy, Shounen	Naruto:Shippuden the Movie	7.242666
2384	Naruto: Shippuuden Movie 2 - Kizuna	7.29	Action, Martial Arts, Shounen, Supernatural	Naruto:Shippuden the Movie 2 -Bonds-	7.238291

Improvement:

We can of course experiment on this engine by trying out different weights for our features (Studio, Source, genres), limiting the number of keywords that can be used in the soup, weighing genres based on their frequency, only showing anime with the same Durations, etc.

Drawback:

A major drawback of this algorithm is that it is limited to recommending items that are of the same type. It will never recommend products that the user has not bought or liked in the past. So if a user has watched or liked only action movies in the past, the system will recommend only action movies. It's a very narrow way of building an engine.
To improve on this type of system, we need an algorithm that can recommend items not just based on the content, but the behavior of users as well.

	Anime name	Rating
0	Shingeki no Kyojin Movie 2: Jiyuu no Tsubasa	7.74
1	Shingeki no Kyojin Season 2	8.45
2	Shingeki no Kyojin: Ano Hi Kara	7.14
3	Shingeki no Kyojin Movie 1: Guren no Yumiya	7.64
4	Shingeki no Kyojin Season 3	8.59
5	Shingeki no Kyojin: Chronicle	7.68
6	Shingeki no Kyojin Season 2 Movie: Kakusei no ...	7.76
7	Shingeki no Kyojin Season 3 Part 2	9.10
8	Shingeki no Kyojin: The Final Season	9.17
9	Shingeki! Kyojin Chuugakkou	7.05

USER BASED

This technique attempts to figure out what a user's favorite aspects of an item is and then recommends items that present those aspects.

Steps Involved:

1. Take input the user_id and get the data containing all the movies completed and rated by him.
2. We're going to start by learning the input's preferences, so we get the subset f movies that the input has watched from the Dataframe containing genres defined with binary values.
3. We'll only need the actual genre, source and type table, so we clean this up by resetting the index and dropping the anime_id, Name, etc.
4. Building Lawrence's Profile: To do this, we're going to turn each genre into weights, by multiplying user movie ratings by the user_genres_df table. And then sum up the resulting table by column. This operation is actually a dot product between a matrix and a vector.
5. We have the weights for all his preferences. This is known as the User Profile. We can now recommend movies that satisfy. We start by editing the original anime_genre_data frame that contains all movies and their genres columns.
6. Let's delete irrelevant columns from the anime_genre_data frame that contains all anime and distinctive columns of genres.
7. With the user's profile and the complete list of anime and their genres in hand, we're going to take the weighted average of every anime based on his profile and recommend the top 250+n anime that match his preference and apply the weighted rating to get the best. (value of n is given by the user)

Results for users 2 and 44:

```
find_rec(55,15)
```

anime_id	Name	Score
5114	Fullmetal Alchemist: Brotherhood	9.19
33486	Boku no Hero Academia 2nd Season	8.33
31933	JoJo no Kimyou na Bouken Part 4: Diamond wa Ku...	8.51
37450	Seishun Buta Yarou wa Bunny Girl Senpai no Yum...	8.38
33	Kenpuu Denki Berserk	8.49
36456	Boku no Hero Academia 3rd Season	8.25
30503	Noragami Aragoto	8.22
121	Fullmetal Alchemist	8.17
392	Yuu☆Yuu☆Hakusho	8.45
31964	Boku no Hero Academia	8.11
6	Trigun	8.24
1735	Naruto: Shippuuden	8.16
45	Rurouni Kenshin: Meiji Kenkaku Romantan	8.31
20507	Noragami	8.01
38408	Boku no Hero Academia 4th Season	8.06

```
find_rec(44,15)
```

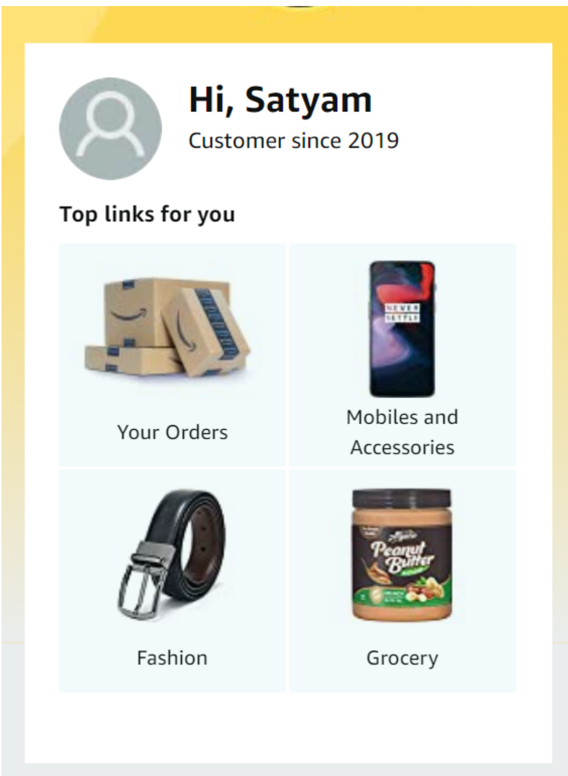
anime_id	Name	Score
38524	Shingeki no Kyojin Season 3 Part 2	9.10
35760	Shingeki no Kyojin Season 3	8.59
877	Nana	8.46
813	Dragon Ball Z	8.16
45	Rurouni Kenshin: Meiji Kenkaku Romantan	8.31
1735	Naruto: Shippuuden	8.16
1698	Nodame Cantabile	8.32
170	Slam Dunk	8.53
16706	Kami nomi zo Shiru Sekai: Megami-hen	8.08
1482	D.Gray-man	8.05
14075	Zetsuen no Tempest	7.99
1559	Shijou Saikyou no Deshi Kenichi	8.11
22145	Kuroshitsuji: Book of Circus	8.13
10080	Kami nomi zo Shiru Sekai II	7.94
3091	xxxHOLiC Kei	8.25

Drawback: It Doesn’t take into account the rating which is a very important aspect to understand a user.

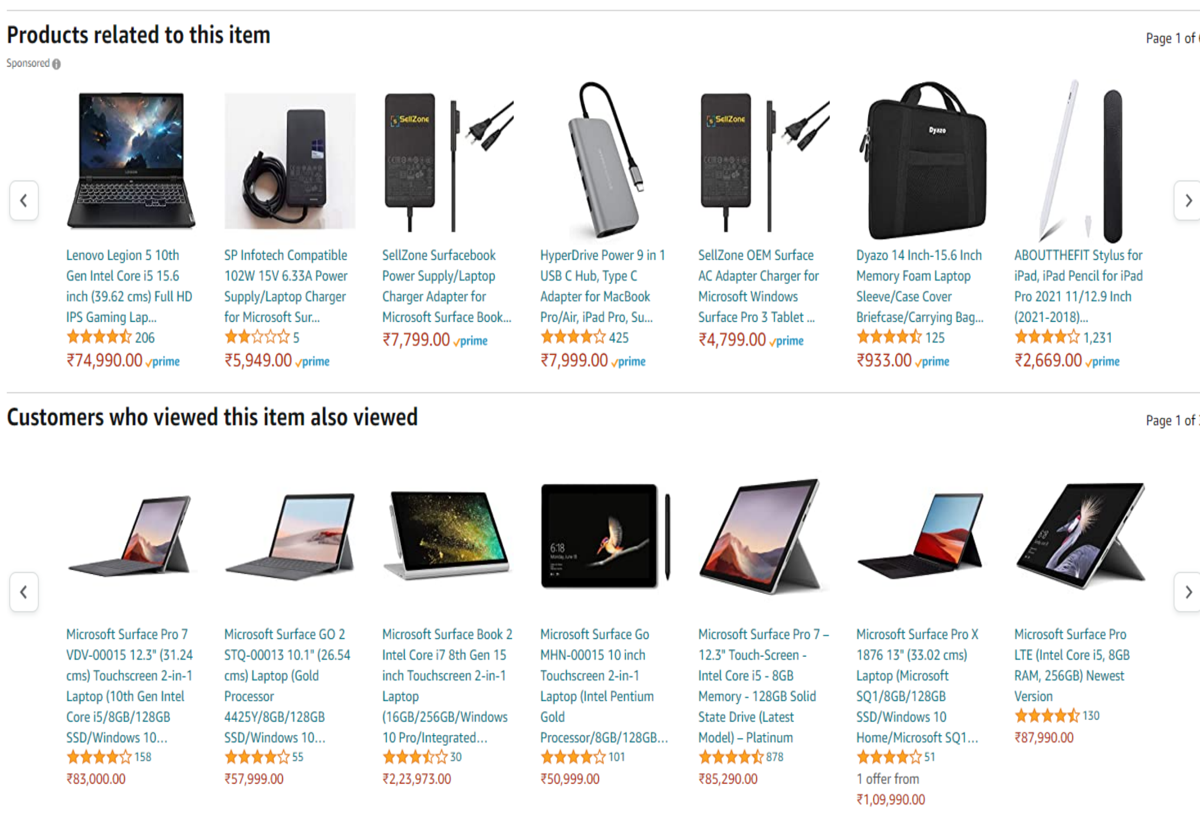
Collaborative Filtering Recommendation System

Collaborative filtering is a technique that can filter out items that a user might like on the basis of reactions by similar users. It works by searching a large group of people and finding a smaller set of users with tastes similar to a particular user.

1) **User-User Collaborative filtering-** This algorithm first finds the similarity score between users. Based on this similarity score, it then picks out the most similar users and recommends products that these similar users have liked or bought previously.



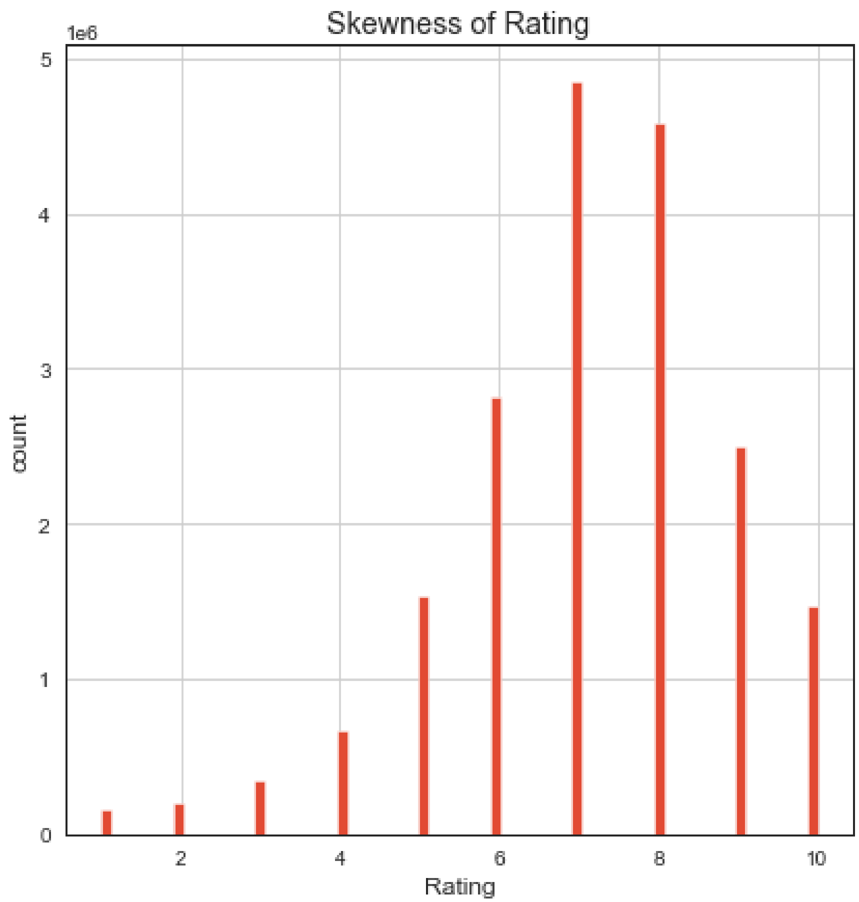
2) **Item-Item collaborative filtering:** In this algorithm, we compute the similarity between each pair of items. So in our case we will find the similarity between each movie pair and based on that, we will recommend similar movies which are liked by the users in the past.



Item Based:

- There are users who have rated only once, even if they have rated it 5, it can't be considered a valuable record for the recommendation. So I have considered minimum of 500 ratings by the user as the threshold value.

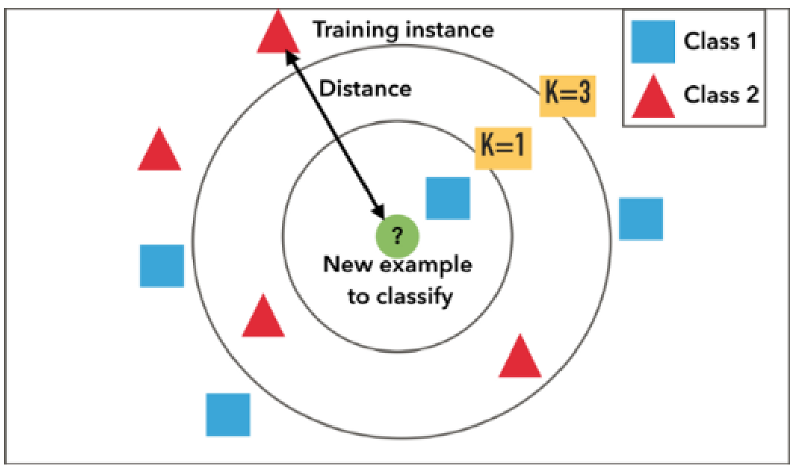
- **Distribution of ratings**



➔ To implement **item-based collaborative filtering**, KNN is a perfect go-to model and also a very good baseline for recommender system development. But what is the KNN? **KNN** is a **non-parametric, lazy** learning method. It uses a database in which the data points are separated into several clusters to make inferences for new samples.

➔ First, we need to transform the data frame of ratings into a proper format that can be consumed by a KNN model. We want the data to be in an `m x n` array, where `m` is the number of movies and `n` is the number of users. To reshape the data frame of ratings, we'll `pivot` the data frame to the wide format with movies as rows and users as columns. Then we'll fill the missing observations with `0`s since we're going to be performing linear algebra operations (calculating distances between vectors).

➔ KNN does not make any assumptions on the underlying data distribution but it relies on **item feature similarity**. When KNN makes an inference about an anime, KNN will calculate the "distance" between the target anime and every other anime in its database, then it ranks its distances and returns the top K nearest neighbor anime as the most similar recommendation systems.

➔ 

➔ KNN's performance will suffer from [the curse of dimensionality]() if it uses "Euclidean distance" in its objective function. **Euclidean distance** is unhelpful in high dimensions because all vectors are almost equidistant to the search query vector (target movie's features). Instead, we will use **cosine similarity** for the nearest neighbor search.

➔ Result: **We take the anime name and n and find top n who are similar to this one.**

```
get_rec('Shingeki no Kyojin',15)
```

```
Recommendations for Shingeki no Kyojin
```

	Name	Score
0	Fullmetal Alchemist: Brotherhood	9.19
2	Steins;Gate	9.11
4	Shingeki no Kyojin Season 3 Part 2	9.10
3	Hunter x Hunter (2011)	9.10
10	Koe no Katachi	9.00
15	Kimi no Na wa.	8.96
14	Clannad: After Story	8.96
17	Code Geass: Hangyaku no Lelouch R2	8.91
19	Haikyuu!!: Karasuno Koukou vs. Shiratorizawa G...	8.87
21	Mob Psycho 100 II	8.84
22	Sen to Chihiro no Kamikakushi	8.83
28	Cowboy Bebop	8.78
27	Monogatari Series: Second Season	8.78
32	Shigatsu wa Kimi no Uso	8.74
34	Made in Abyss	8.74

```
get_rec('Fullmetal Alchemist: Brotherhood',15)
```

```
Recommendations for Fullmetal Alchemist: Brotherh
```

	Name	Score
2	Steins;Gate	9.11
4	Shingeki no Kyojin Season 3 Part 2	9.10
3	Hunter x Hunter (2011)	9.10
10	Koe no Katachi	9.00
14	Clannad: After Story	8.96
12	Gintama	8.96
15	Kimi no Na wa.	8.96
17	Code Geass: Hangyaku no Lelouch R2	8.91
19	Haikyuu!!: Karasuno Koukou vs. Shiratorizawa G...	8.87
21	Mob Psycho 100 II	8.84
22	Sen to Chihiro no Kamikakushi	8.83
27	Monogatari Series: Second Season	8.78
28	Cowboy Bebop	8.78
34	Made in Abyss	8.74
32	Shigatsu wa Kimi no Uso	8.74

User-Based:

- ➔ First, we will create a user-item matrix that can be used to calculate the similarity between users and items
- ➔ There will be situations where the n similar users that we found are not equally similar to the target user U. The top 3 of them might be very similar, and the rest might not be as similar to U as the top 3. In that case, we could consider an approach where the rating of the most similar user matters more than the second most similar user and so on. The weighted average can help us achieve that.
- ➔ With the similarity factor S for each user similar to the target user U, you can calculate the weighted average using this formula:

$$R_U = \left(\sum_{u=1}^{n} R_u * S_u\right) / \left(\sum_{u=1}^{n} S_u\right)$$

- ➔ In the above formula, every rating is multiplied by the similarity factor of the user who gave the rating. The final predicted rating by user U will be equal to the sum of the weighted ratings divided by the sum of the weights.

Result for random user: RMSE for top 100: 2.624

```
finding_anime_for_user(2,250).sort_values(by="Score",ascending=False).head(15)
```

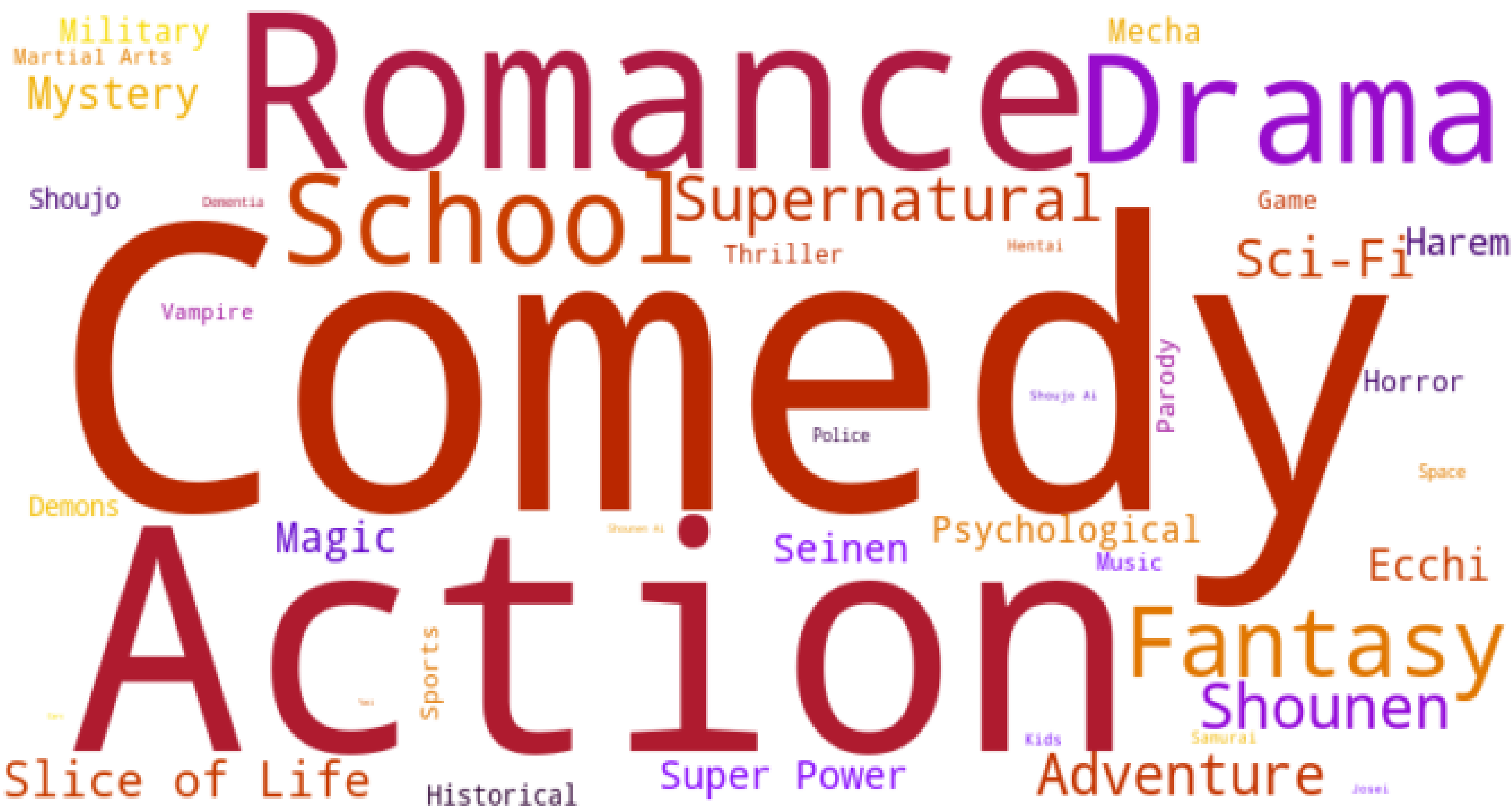
	Name	Score	Genders	English name	rating
1	3-gatsu no Lion 2nd Season	9.00	['Drama', 'Game', 'Seinen', 'Slice of Life']	March Comes In Like A Lion 2nd Season	2.745754
2	Owarimonogatari 2nd Season	8.93	['Mystery', 'Comedy', 'Supernatural', 'Vampire']	Owarimonogatari Second Season	2.403281
5	Sen to Chihiro no Kamikakushi	8.83	['Adventure', 'Supernatural', 'Drama']	Spirited Away	6.137339
8	Cowboy Bebop	8.78	['Action', 'Adventure', 'Comedy', 'Drama', 'Sc...	Cowboy Bebop	5.020926
9	Monster	8.76	['Drama', 'Horror', 'Mystery', 'Police', 'Psyc...	Monster	2.888621
10	Mushishi Zoku Shou 2nd Season	8.76	['Adventure', 'Fantasy', 'Historical', 'Myster...	NaN	2.291598
12	Kaguya-sama wa Kokurasetai?: Tensai-tachi no R...	8.74	['Comedy', 'Psychological', 'Romance', 'School...	Kaguya-sama:Love is War Season 2	3.056126
11	Made in Abyss	8.74	['Sci-Fi', 'Adventure', 'Mystery', 'Drama', 'F...	Made in Abyss	4.639688
14	Rurouni Kenshin: Meiji Kenkaku Romantan - Tsui...	8.73	['Action', 'Historical', 'Drama', 'Romance', '...	Samurai X:Trust and Betrayal	2.767258
15	Mushishi Zoku Shou	8.72	['Adventure', 'Slice of Life', 'Mystery', 'His...	MUSHI-SHI -Next Passage-	2.622428
16	Mononoke Hime	8.72	['Action', 'Adventure', 'Fantasy']	Princess Mononoke	5.184796
19	Howl no Ugoku Shiro	8.67	['Adventure', 'Drama', 'Fantasy', 'Romance']	Howl's Moving Castle	4.895557
21	Natsume Yuujinchou Shi	8.67	['Slice of Life', 'Demons', 'Supernatural', 'D...	Natsume's Book of Friends Season 4	2.807075
23	Yakusoku no Neverland	8.65	['Sci-Fi', 'Mystery', 'Horror', 'Psychological...	The Promised Neverland	3.971846
25	Violet Evergarden	8.64	['Slice of Life', 'Drama', 'Fantasy']	Violet Evergarden	4.408113

Drawbacks:

- ➔ In practice, many commercial recommender systems are based on large datasets. As a result, the user-item matrix used for collaborative filtering could be extremely large and sparse, which brings about challenges in the performance of the recommendation. One typical problem caused by the data sparsity is the cold start problem. As collaborative filtering methods recommend items based on users' past preferences, new users will need to rate a sufficient number of items to enable the system to capture their preferences accurately and thus provides reliable recommendations.
- ➔ Similarly, new items also have the same problem. When new items are added to the system, they need to be rated by a substantial number of users before they could be recommended to users who have similar tastes to the ones who rated them. The new item problem does not affect content-based recommendations, because the recommendation of an item is based on its discrete set of descriptive qualities rather than its ratings.

For Future:

- Creating a hybrid recommendation system that combines content-based filtering and collaborative filtering could potentially take advantage of both the representation of the content as well as the similarities among users and negates the limitations.
- Testing the model with the appropriate algorithm.


Military
Martial Arts
Mystery
Romance
Mecha
Drama
Shoujo
Dementia
School
Supernatural
Game
Thriller
Hentai
Sci-Fi
Harem
Comedy
Vampire
Parody
Shoujo Ai
Horror
Police
Space
Demons
Magic
Seinen
Psychological
Music
Ecchi
Action
Fantasy
Sports
Shounen
Kids
Samurai
Slice of Life
Historical
Super Power
Adventure
Josei