Report

Recommended System - Collaborative Filtering on Anime Dataset

Susritha Nelapati

May 26, 2020

1.INTRODUCTION:

1.1 Background

Anime is hand-drawn and computer animation originating from Japan. The word anime is the Japanese term for animation, which refers to all forms of animated media. Outside Japan, anime refers specifically to animation from Japan or as a Japanese-disseminated animation style often characterized by colorful graphics, vibrant characters and fantastical themes.Recommendation systems are a collection of algorithms used to recommend items to users based on information taken from the user. These systems have become ubiquitous can be commonly seen in online stores, movies databases and job finders. In this report, we will explore recommendation systems based on Collaborative Filtering and implement simple version of one using Python and the Pandas library. For instance, if two users are similar or are neighbors in terms of their interested movies, we can recommend a movie to the active user that her neighbor has already seen.

1.2 Problem

When a user want to watch a new series/ anime he /she would look for different sources in order to choose a anime to their interest so what if we can create a system where the user can get recommendation based on similar user interests.

1.3 Interest

As anime being one of the popluar interest among many users in the internet this recommendation engine helps the users to find their similar interest with respective to other users.

2. Data acquisition

Anime_id, name, genre,rating, user_id all of these values can be obtained from two Kaggle datasets Anime Dataset and Rating Dataset

2.1 Data Content

Anime.csv

- anime_id - myanimelist.net's unique id identifying an anime.
- name - full name of anime.
- genre - comma separated list of genres for this anime.
- type - movie, TV, OVA, etc.
- episodes - how many episodes in this show. (1 if movie).
- rating - average rating out of 10 for this anime.
- members - number of community members that are in this anime's "group".

Rating.csv

- user_id - non identifiable randomly generated user id.
- anime_id - the anime that this user has rated.
- rating - rating out of 10 this user has assigned (-1 if the user watched it but didn't assign a rating).

2.2 Data Cleaning

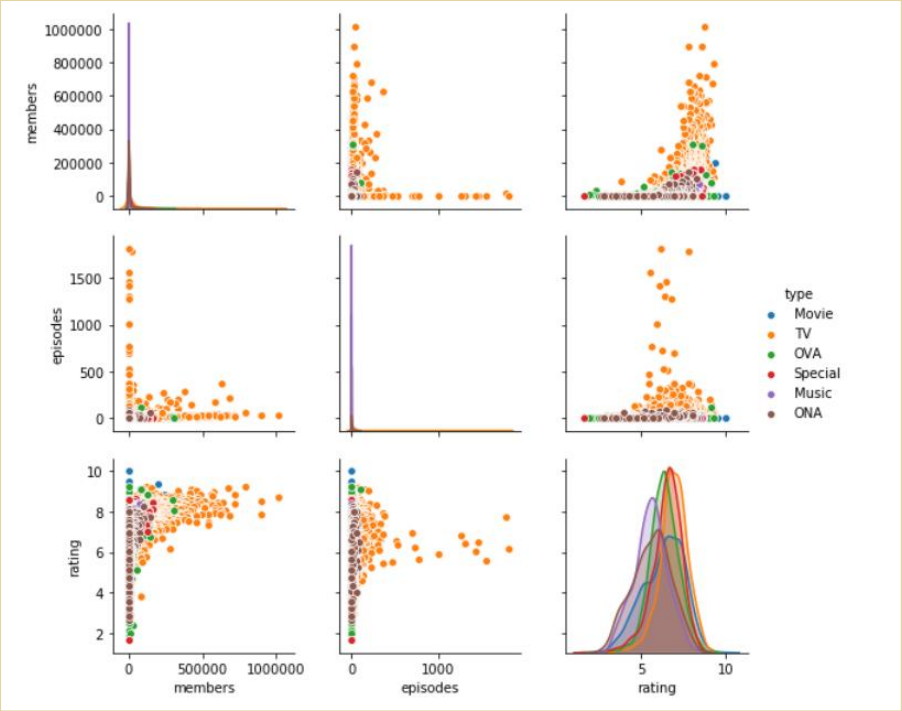
Before we start to build a recommendation system the dataset should be ready for collaborative filtering process that we further use. Though there are columns which are not use in the case of collaborative filtering we still keep them in the dataset for data visualization for better understanding and for better practise. As per the ratings dataset there are negative values for non rated anime. So we replace those values with NaN value. This step has to be done compulsory for not to get errors during calculating Pearson coefficient.

3. Methodology

In this part of the report we explain two parts one is data visualization where graphs or plots are used to just simply understand the data and how one term is dependent on other factors in a dataframe. And second part is where we solve the main problem using collaborative filtering and Pearson coefficient.

3.1 Data Visualization

A pairplot can be very helpful in comparing many factors at a same time. Type was chosen as determining factors in these graphs and they are represented with various colors for better understanding.



3.2 Recommendation system

Let's begin by learning little bit recommended systems and types before we get to know about collaborative filtering for better understanding nothing is wrong with gaining a little more knowledge is there?!!!.

Even though peoples' tastes may vary, they generally follow patterns. For example, if you've recently purchased a book on Machine Learning in Python and you've enjoyed reading it, it's very likely that you'll also enjoy reading a book on Data Visualization. People also tend to have similar tastes to those of the people they're close to in their lives. Recommendation systems try to capture these patterns and

similar behaviors, to help predict what else you might like. Recommendation systems are usually at play on many websites. For example, suggesting books on Amazon and movies on Netflix. In fact, everything on Netflix's website is driven by customer selection.

There are generally 2 main types of recommendation systems: Content-based and collaborative filtering. The main difference between each, can be summed up by the type of statement that a consumer might make. For instance, the main paradigm of a Content-based recommendation system is driven by the statement: "Show me more of the same of what I've liked before." Content-based systems try to figure out what a user's favorite aspects of an item are, and then make recommendations on items that share those aspects.

Collaborative filtering is based on a user saying, "Tell me what's popular among my neighbors because I might like it too."Collaborative filtering techniques find similar groups of users, and provide recommendations based on similar tastes within that group. In short, it assumes that a user might be interested in what similar users are interested in.

Collaborative filtering:

Now, time to start our work on recommendation systems.

The first technique we're going to take a look at is called Collaborative Filtering, which is also known as User-User Filtering. As hinted by its alternate name, this technique uses other users to recommend items to the input user. It attempts to find users that have similar preferences and opinions as the input and then recommends items that they have liked to the input. There are several methods of finding similar users (Even some making use of Machine Learning), and the one we will be using here is going to be based on the Pearson Correlation Function.

The process for creating a User Based recommendation system is as follows:

- Select a user with the movies the user has watched
- Based on his rating to movies, find the top X neighbors
- Get the watched movie record of the user for each neighbour.
- Calculate a similarity score using some formula
- Recommend the items with the highest score

The process begin by creating or choosing a particular user preferences i.e, the anime series the particular user watched and reviewed.

	name	rating
0	Kimi no Na wa.	9.5
1	Mononoke Hime	7.5
2	One Punch Man	9.0
3	Bungou Stray Dogs 2nd Season	9.2
4	Great Teacher Onizuka	8.0

Add anime_Id to input user

With the input complete, let's extract the input anime ID's from the anime dataframe and add them into it.

We can achieve this by first filtering out the rows that contain the input movies' title and then merging the subset with the input dataframe. We also drop unnecessary columns for the input to save memory space.

	anime_id	name	genre	type	episodes	rating	members
0	32281	Kimi no Na wa.	[Drama, Romance, School, Supernatural]	Movie	1	9.37	200630
23	30276	One Punch Man	[Action, Comedy, Parody, Sci-Fi, Seinen, Super...	TV	12	8.82	552458
24	164	Mononoke Hime	[Action, Adventure, Fantasy]	Movie	1	8.81	339556
30	245	Great Teacher Onizuka	[Comedy, Drama, School, Shounen, Slice of Life]	TV	43	8.77	268487
161	32867	Bungou Stray Dogs 2nd Season	[Mystery, Seinen, Supernatural]	TV	12	8.39	83641

The users who has seen the same movies

Now with the movie ID's in our input, we can now get the subset of users that have watched and reviewed the movies in our input.

	user_id	anime_id	rating
246	3	30276	NaN
338	5	245	10.0
822	7	245	8.0
1103	7	30276	8.0
1182	11	164	8.0

Similarity of users to input user

Next, we are going to compare all users (not really all !!!) to our specified user and find the one that is most similar.

we're going to find out how similar each user is to the input through the Pearson Correlation Coefficient. It is used to measure the strength of a linear association between two variables. The formula for finding this coefficient between sets X and Y with N values can be seen in the image below.

Why Pearson Correlation?

Pearson correlation is invariant to scaling, i.e. multiplying all elements by a nonzero constant or adding any constant to all elements. For example, if you have two vectors X and Y,then, Pearson(X, Y) == Pearson(X, 2 * Y + 3). This is a pretty important property in recommendation systems because for example two users might rate two series of items totally different in terms of absolute rates, but they would be similar users (i.e. with similar ideas) with similar rates in various scales .

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

The values given by the formula vary from r = -1 to r = 1, where 1 forms a direct correlation between the two entities (it means a perfect positive correlation) and -1 forms a perfect negative correlation.

In our case, a 1 means that the two users have similar tastes while a -1 means the opposite.

We will select a subset of users to iterate through. This limit is imposed because we don't want to waste too much time going through every single user.

```
[(784,
        user_id  anime_id  rating
  75639      784       164     9.0
  75647      784       245     8.0
  75980      784     30276     9.0
  76023      784     32281     9.0),
 (2243,
         user_id  anime_id  rating
  217638     2243       164     NaN
  217651     2243       245     NaN
  218376     2243     30276     NaN
  218385     2243     32281     NaN),
 (2819,
         user_id  anime_id  rating
  271457     2819       164     9.0
  271463     2819       245     9.0
  271679     2819     30276     9.0
  271685     2819     32281     9.0),
 (4437,
         user_id  anime_id  rating
  430654     4437       164     9.0
  430660     4437       245     8.0
  430963     4437     30276     6.0
  430995     4437     32281     6.0),
 (4821,
```

Now, we calculate the Pearson Correlation between input user and subset group, and store it in a dictionary, where the key is the user Id and the value is the coefficient

	similarityIndex	user_id
0	0.402355	784
1	NaN	2243
2	0.000000	2819
3	-0.577291	4437
4	0.309055	4821

The top x similar users to input user

Now let's get the top 50 users that are most similar to the input.

	similarityIndex	user_id
63	0.99714	28238
98	0.99714	41865
31	0.99714	16784
84	0.99714	37007
35	0.99714	17432

Now, let's start recommending movies to the input user.

Rating of selected users to all movies

We're going to do this by taking the weighted average of the ratings of the movies using the Pearson Correlation as the weight. But to do this, we first need to get the movies watched by the users in our pearsonDF from the ratings dataframe and then store their correlation in a new column called similarityIndex". This is achieved below by merging of these two tables.

	similarityIndex	user_id	anime_id	rating
0	0.99714	28238	6	9.0
1	0.99714	28238	19	9.0
2	0.99714	28238	20	8.0
3	0.99714	28238	30	9.0
4	0.99714	28238	31	8.0

Now all we need to do is simply multiply the movie rating by its weight (The similarity index), then sum up the new ratings and divide it by the sum of the weights.

We can easily do this by simply multiplying two columns, then grouping up the dataframe by anime_id and then dividing two columns:

It shows the idea of all similar users to candidate movies for the input user:

	similarityIndex	user_id	anime_id	rating	weightedRating
0	0.99714	28238	6	9.0	8.974257
1	0.99714	28238	19	9.0	8.974257
2	0.99714	28238	20	8.0	7.977117
3	0.99714	28238	30	9.0	8.974257
4	0.99714	28238	31	8.0	7.977117

4. Results:

The final recommendations options we received is:

	anime_id	name	genre	type	episodes	rating	members
11	28851	Koe no Katachi	[Drama, School, Shounen]	Movie	1	9.05	102733
467	2623	Flanders no Inu (Movie)	[Drama, Historical]	Movie	1	8.05	8505
534	996	Bishoujo Senshi Sailor Moon: Sailor Stars	[Adventure, Comedy, Drama, Fantasy, Magic, Rom...	TV	34	7.99	52586
1011	1395	Future GPX Cyber Formula Sin	[Drama, Sci-Fi, Shounen, Sports]	OVA	5	7.72	1070
1063	1087	Kimagure Orange☆Road	[Comedy, Drama, Romance, School, Shounen, Slic...	TV	48	7.70	16012
1968	3508	Genius Party	[Action, Dementia, Fantasy, Mecha, Music, Psyc...	Movie	7	7.39	18612
2427	17699	Toriko Movie: Bishokushin no Special Menu	[Action, Adventure, Fantasy, Shounen]	Movie	1	7.27	3077
2635	1079	Armitage III	[Action, Adventure, Mecha, Police, Romance, Sc...	OVA	4	7.20	14930
4013	6855	Da Nao Tiangong	[Adventure]	Movie	1	6.82	342
4683	12661	Light Lag	[Music]	Music	1	6.66	891

After calculating Pearson coefficient and constructing weighted average score we compare those anime id's with the anime the user can watch in the dataset and as result we will get the above dataframe as recommendations for user.

5. Discussion

Though the process itself is not complicated but it miraculous how the recommendation system works despite containing such a large amount of dataset, though one thing was quite disappointing was that the input dataset in I used was anime that I watched and I was expecting something that I might have watched will be recommended in the recommendation system but it was not, so it is some what clear that recommendation system or collaborative filtering is always not the best option though we have many other types of recommendation systems.

6. Conclusion:

In future the demand for recommend system is going to increase in tremendous rate since recommendations systems can be quite useful in time saving which we might waste while trying to find something to our interest. In order to search for interests for a single user we used a quite a large data set and we able to create a dataframe of the desired output.