**Report**

Recommended System - Collaborative Filtering on Anime Dataset

Susritha Nelapati

May 26, 2020

**1.INTRODUCTION:**

* 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**

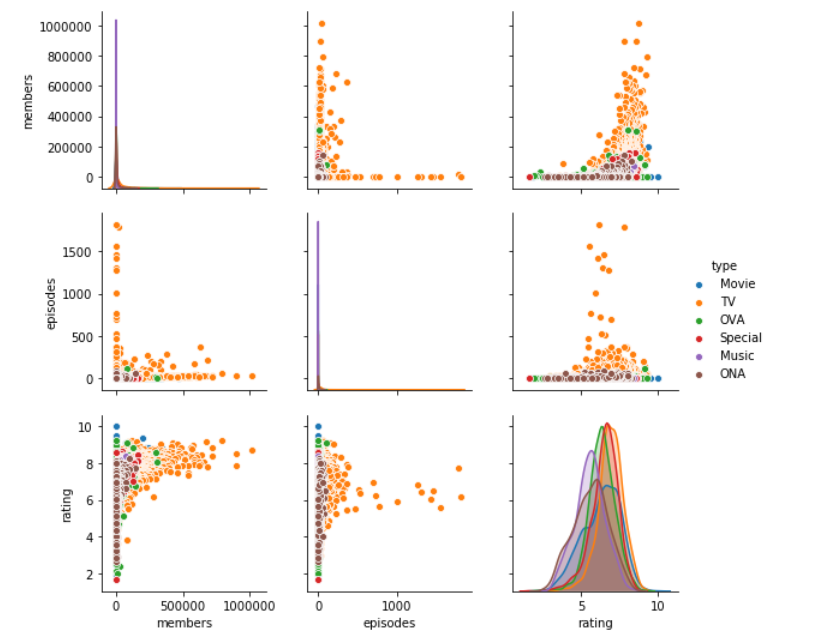
Before we start to build a recommendation system the dataset should be ready for collaborative filtering process that we further use. Though there are columns which are not use in the case of collaborative filtering we still keep them in the dataset for data visualization for better understanding and 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.

1. **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.

* 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.



* 1. **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:**

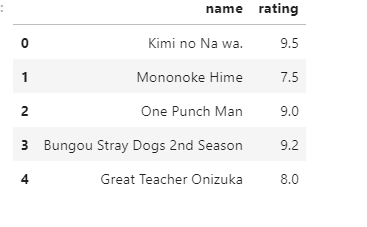
Now, time to start our work on recommendation systems.

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

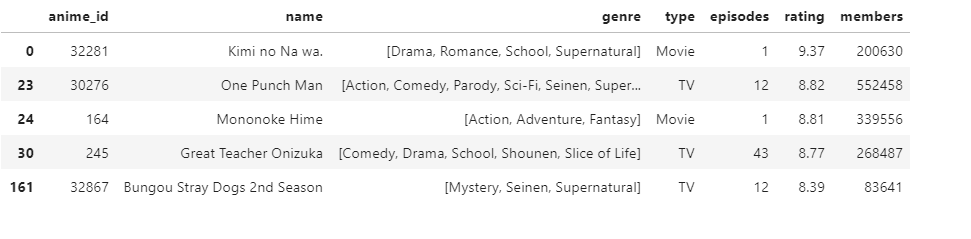
The process begin by creating or choosing a particular user preferences i.e, the anime series the particular user watched and reviewed.



Add anime\_Id to input user

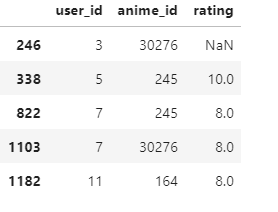
With the input complete, let's extract the input anime ID's from the anime dataframe and add them into it.

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.



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.



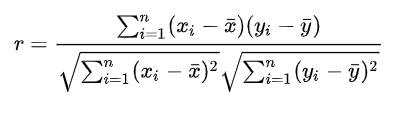
Similarity of users to input user

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?

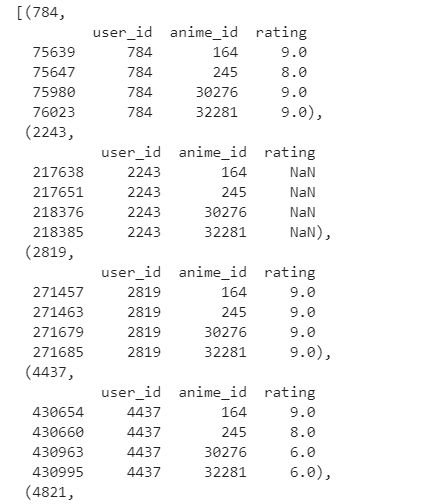
Pearson correlation is invariant to scaling, i.e. multiplying all elements by a nonzero constant or adding any constant to all elements. For example, if you have two vectors X and Y,then, Pearson(X, Y) == Pearson(X, 2 \* Y + 3). This is a pretty important property in recommendation systems because for 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 .



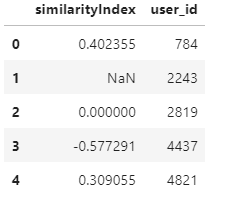
The values given by the formula vary from r = -1 to r = 1, where 1 forms a direct correlation between the two entities (it means a perfect positive correlation) and -1 forms a perfect negative correlation.

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.

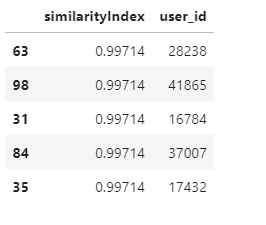


Now, we calculate the Pearson Correlation between input user and subset group, and store it in a dictionary, where the key is the user Id and the value is the coefficient



The top x similar users to input user

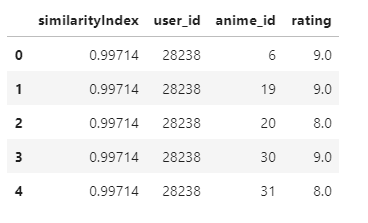
Now let's get the top 50 users that are most similar to the input.



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

Rating of selected users to all movies

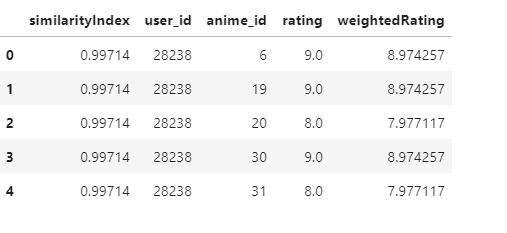
We're going to do this by taking the weighted average of the ratings of the movies using the Pearson Correlation as the weight. But to do this, we first need to get the movies watched by the users in our pearsonDF from the ratings dataframe and then store their correlation in a new column called similarityIndex".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



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:



1. **Results:**

The final recommendations options we received is:



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.

1. **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.

1. **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.