Affinity Search Engine

-The anime recommendation system

# Context

This data set contains information on user preference data from 73,516 users on 12,294 anime. Each user is able to add anime to their completed list and give it a rating and this data set is a compilation of those ratings.

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).

Acknowledgements

Thanks to myanimelist.net API for providing anime data and user ratings.

Inspiration

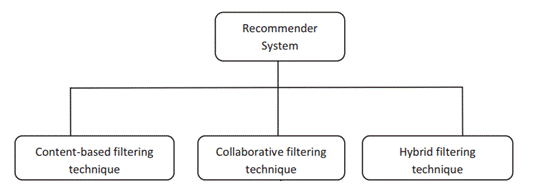
Building a better anime recommendation system based only on user viewing history.

Recommendation System:

Recommendation systems produce a ranked list of items on which a user might be interested, in the context of his current choice of an item.

* Subclass of information filtering system that seeks to predict the ‘rating’ or ‘preference’ that a user would give to them.
* Helps deciding in what to wear, what to buy, which stock to purchase etc.
* Applied in variety of applications like movies, TV series, Books, articles and so on.

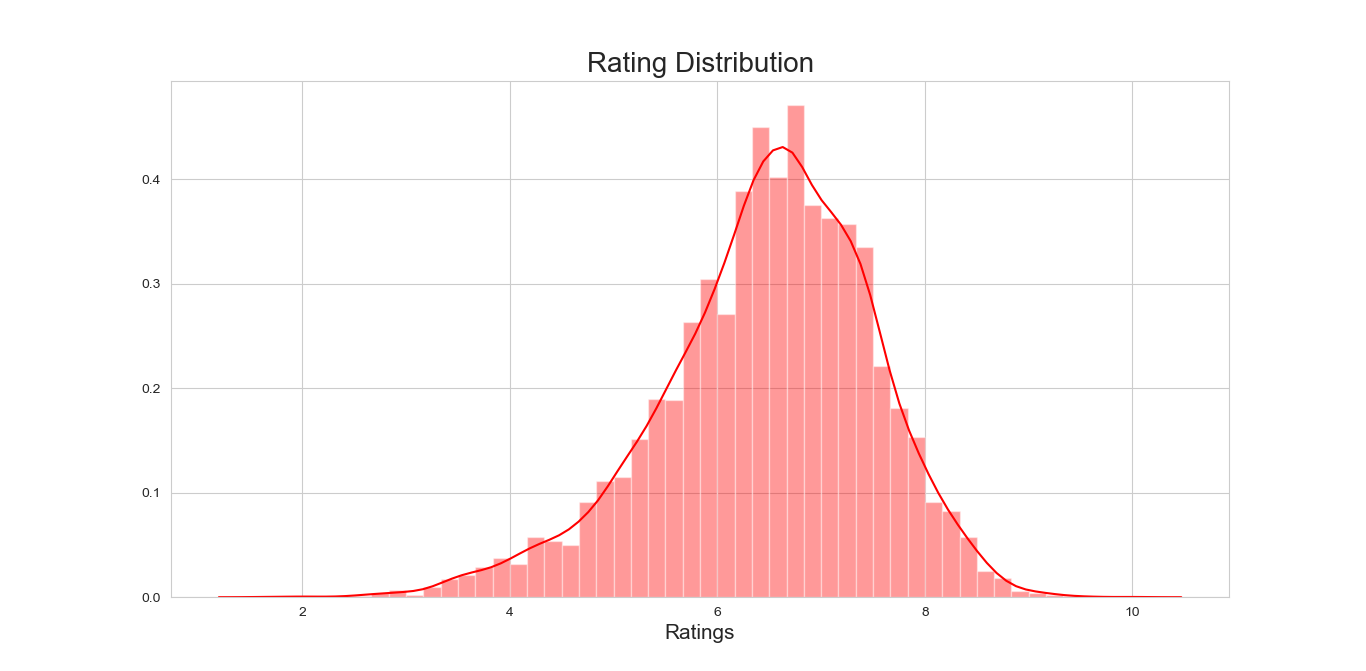
Recommendation systems has mainly two elements, Items and User.



For our Project,

Visualization:

Rating Distribution graph:



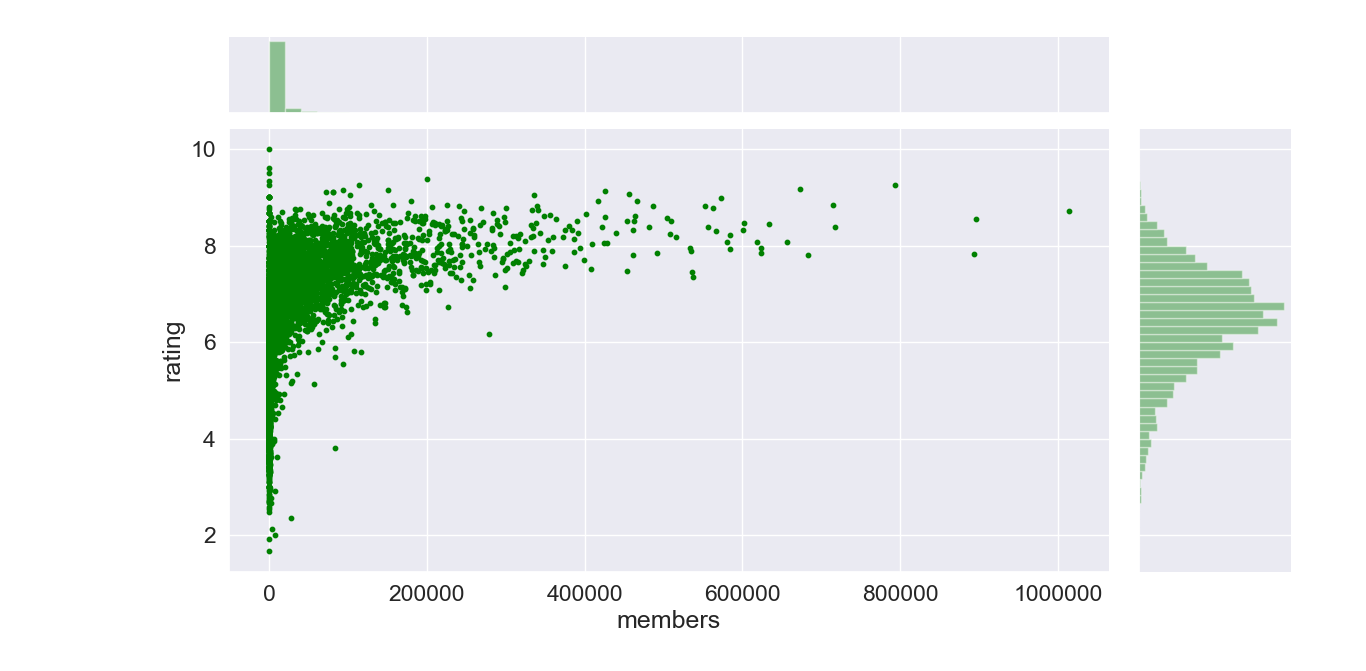
This Rating Distribution graph shows that the ratings of animes are normally distributed, where the mean avg rating is 7 approximately.

Tree map of Genres:

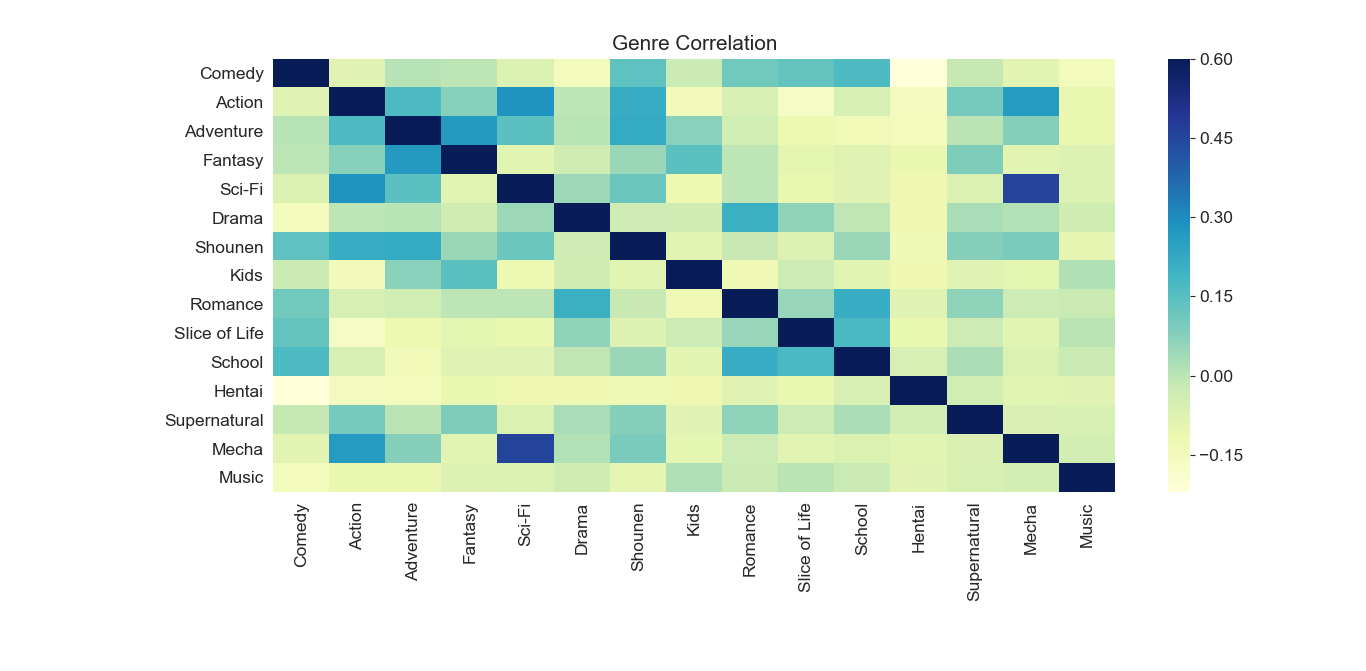


This shows top 20 genre types with counts. Bigger and darker the shapes, higher the freq. of the genre type and so the other shapes.

Joint plot:

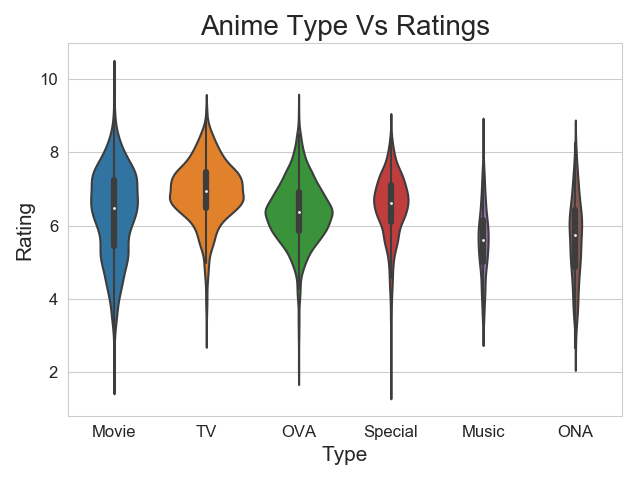


This is the joint plot of scatter plot of members Vs Anime ratings, on X axis we can see the scatter plot of members and on Y axis we can see the histogram of ratings, from this graph we can understand that **higher number of users given rating, higher chance of anime gets high ratings too.**

Genre Correlation:

This plot shows correlation between the genres. Darker is the color, more is the correlation and vice-versa, thus the plot shows the higher correlation between Romance and Comedy animes.

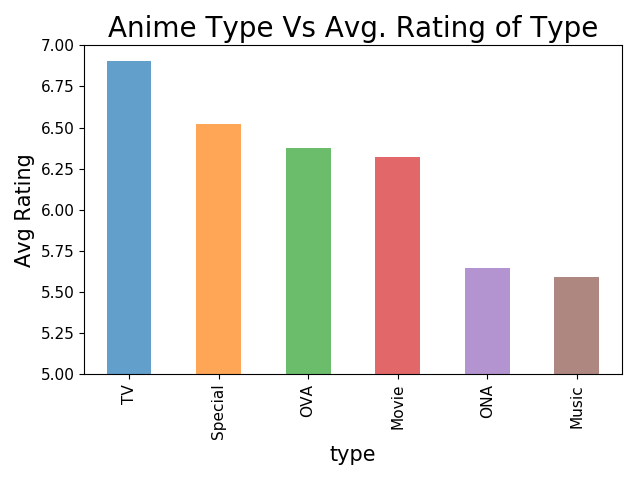
Violin plot:



This is the graph of Anime type Vs Anime rating, that shows the rectangular box in the center of each violin shape that states the interquartile range of rating of specific anime type. About 25% of the data of each rating lies on each sides of the white spot in between, which is mean of the ratings.

Each violin size is influenced by the sample size of anime type. Here we represent only the top six anime types based on the frequency of occurrence

Bar plot:



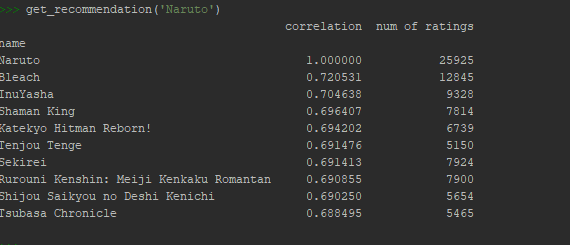
This bar plot shows top 6 avg rating of each anime type depending on frequency of occurrence, where the TV type has got the higher avg rating

* Content based Filtering

This recommends other movies, which are similar to that selected movies

Steps involved in Content based Filtering(Similar Rating & Genre):

* Firstly, both the rating and the anime datasets are merged to create a new data frame.
* The newly created data frame will also contain the columns namely “anime names”, “avg. rating”, “no. of rating”. Then the data frame is sorted down by the rating counts in the descending order to find the animes with more no. of ratings.
* Secondly, a function is created to recommend the Animes based on the ratings and genre of the specific anime that entered in the “my anime list” on the anime page.
* To assist the function above, we have created a pivot table with the index as name of anime and the columns as user ids and ratings, those values are stored in a Sparse matrix.
* This sparse matrix will show the correlation between the anime ratings, so that user will get the recommendation based on similar genre and similar rating as well.



Steps involved in Content based Filtering(Genre of the anime watched by the user):

Here we use Apriori algorithm, that fetch us the support average of 0.2

* From user data we select only login user data then from that we took the anime id
* From this anime id we took that other information(anime id, genre, anime name and rating).
* From that detail list we consider only genre and made list of unique genre list.
* We find occurrences of those genre and created dictionary of genre and there occurrences which also get store base on occurrences and out of that we selected only top 10 genres which occurred mostly.
* From this we got mostly watched genre. Which we pass to function
* Function, get list of genre, now here we took combination of those genres and 1st consider only pair of 2, and then in next iteration we consider pair of 3 until we got pair of 4.
* In this iteration we again find occurrences of those pairs.
* From this function we got combination of pair which mostly occurred.
* Using this list we find anime which are not watched by user and those anime get stored by rating and recommended top 10.



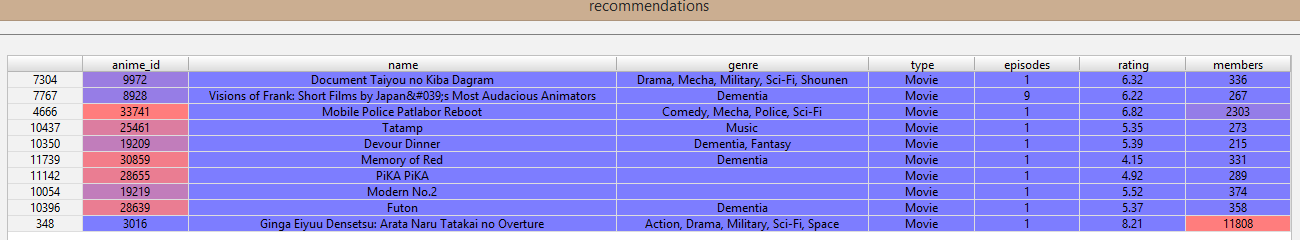
Collaborative Filtering technique

This recommends the items which are rated highly by the similar users.

* Item – Item type
* User - Item type

Steps involved in Item-Item Collaborative Filtering Based Recommendation System:

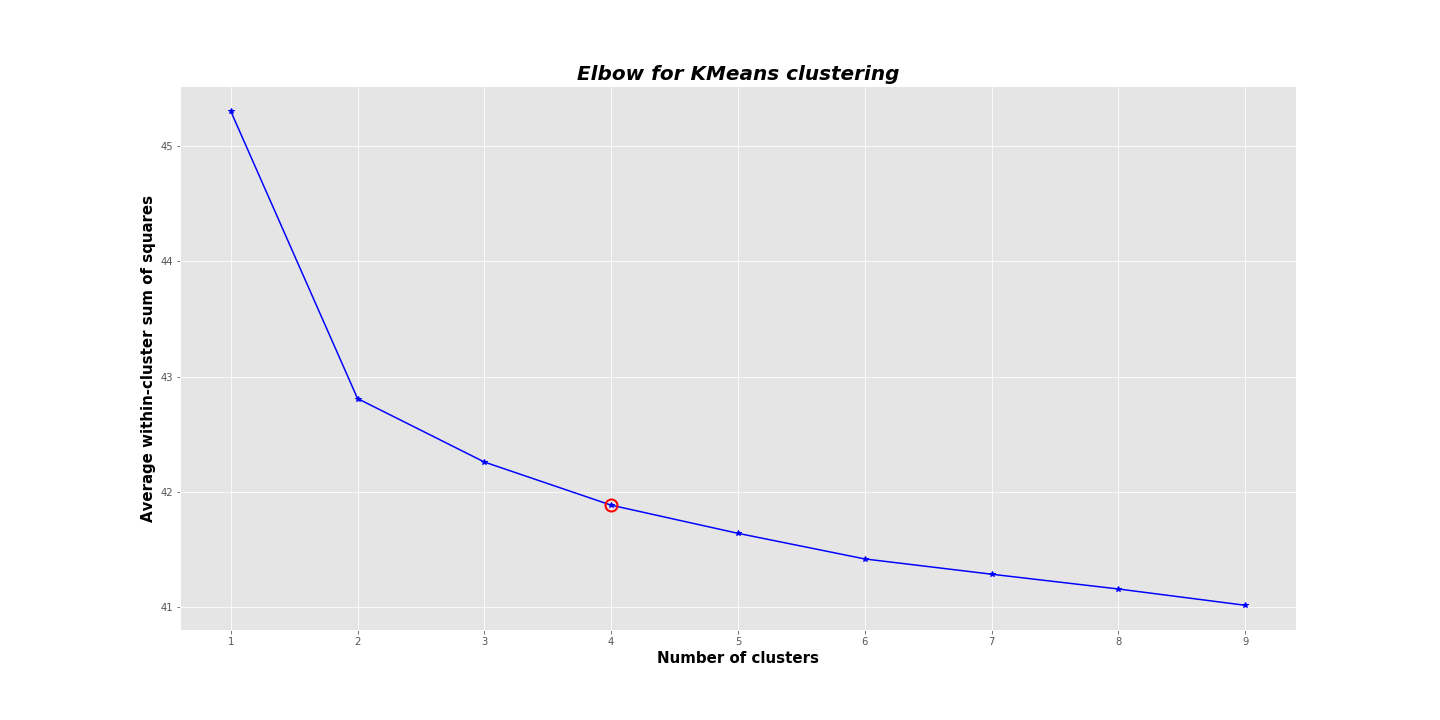
* From Anime dataset, we selected only Tv and Movie types (Keeping the hugeness of the data in mind).
* From rating dataset, we imputed 0 wherever the user didn't rated the anime.
* later we merged both anime and rating datasets on anime id column, and considered only user id, name of anime and its rating columns.
* then we created a pivot table with the index as name of the anime and columns with user ids and the ratings.
* Continued with normalizing the pivot table, the similarity between the animes are calculated using cosine similarity.
* with the calculated similarity score vector we got top similar animes and their scores for the anime which user has just watched.
* now by using this similar animes and their similarity scores we predicted the ratings of animes which user didn't rated.
* After predicting the ratings in sparse matrix for particular user
* we'll recommend animes to that user which he didn't watched yet as our final output of recommendations.



Steps involved in User-Item Collaborative Filtering Based Recommendation System using Clustering technique:

We have used the clustering technique to create the recommendation system.

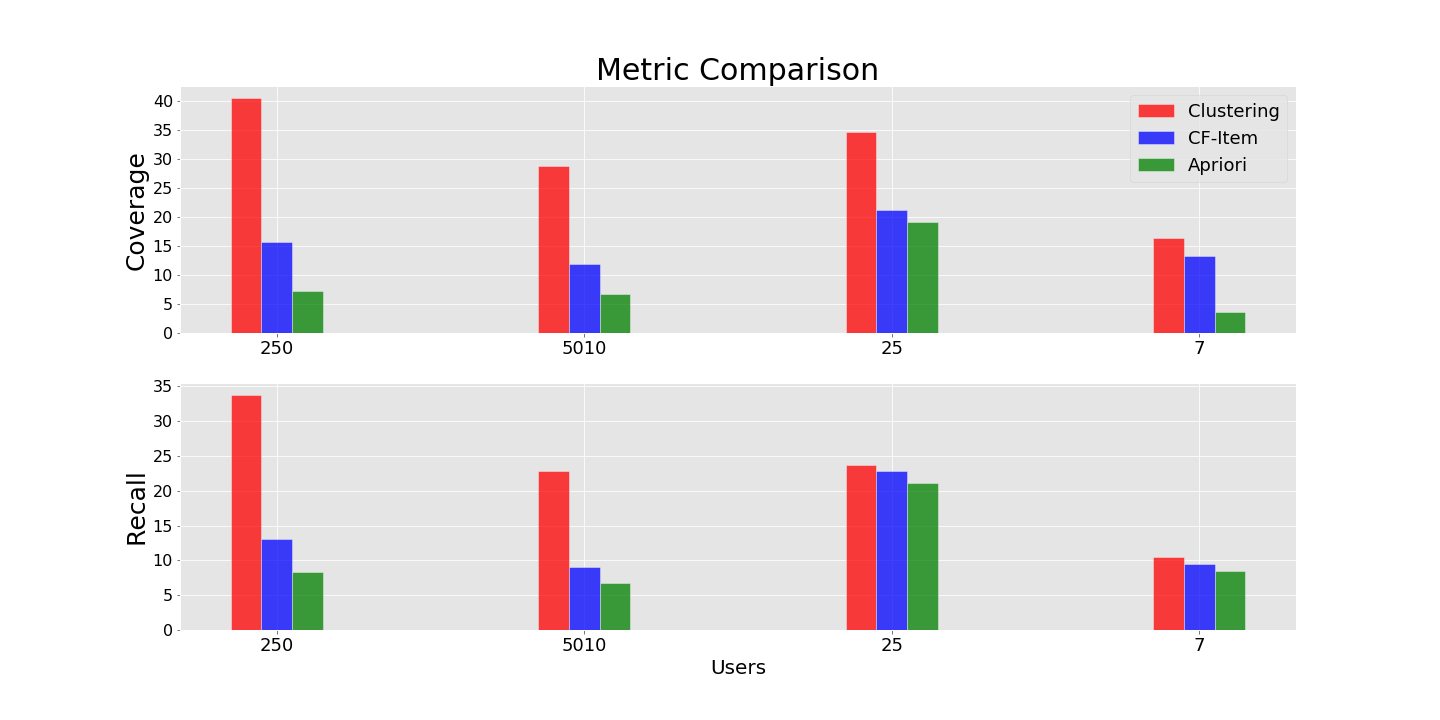
* The recommended movies need to be ones that are favourite among the mass. Hence, we choose to find the mean rating provided by each user and select only those animes for which user has rated higher than his own mean rating. The reason to do this is to remove the differences in preferences in rating of different animes by different users. Hence, we have imputed the -1 rating and NA values score with 0 for sake of mean computation.
* Since the data is huge and to save on running time, we have filtered first 20000 users only.
* Secondly, we merged the two datasets anime and user on anime\_id and got the dataset with 42L records.
* Then, we created a pivot table with index as user and a column ‘name of animes’ and values as ratings provided by each user for each anime. The dimensions of this pivot table are 20000 rows x 7835 columns.
* Later, we apply PCA on this pivot table. PCA converts the original variables to a new set of variables, which are linear combinations of the original set of variables. Our main goal is to reduce dimension of data for clustering i.e, from 7835 features to 600 components retaining 78% of information from our original dataset.
* Using the elbow curve, we selected the optimum clusters to be 4.
* Then we applied the KMeans algorithm on achieved principal components and predicted the cluster for each user. Assign these clusters to respective users to recommend animes further.
* Finally, we filtered the data based on user’s cluster and sorted the anime names based on average rating to recommend top animes.



Below is output for recommendation for cluster no. 0:



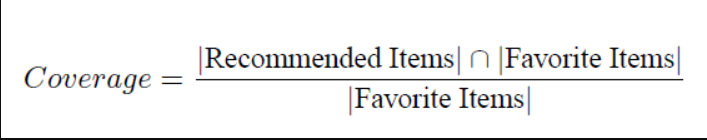
Metric Comparison:



The above graph shows comparison between coverage and recall calculated for 4 users namely- 250, 5010, 25 and 7. These 4 random users are taken from each cluster from Clustering technique and measures for all algorithms are compared above.

Coverage indicates how much the system could discover items desirable to user.

Coverage is ratio of no. of favorite items recommended to total favorite items of the user.



Recall indicates what % of Animes that user watched are actually recommended.

  |Recommended Items| ∩ |Watched Items|

   Recall = —————————————————

|Watched Items|

Favorite items for the user are the ones that are given rating by that user more than average of rating given by that user. Similarly watched items are all the items that are watched by user irrespective of his rating given.

**Conclusion**:

Based on above results plotted we can see that for current set of information clustering technique works best giving recommendation closer to user’s likeness.

Designs:

