RECOMMENDATIONS SYSTEMS FOR LASTFM

Introduction:

Web-based music streaming services such as Spotify, Wynk Music, **LastFM etc** provide their users with many opportunities to discover new music, whether in the form of specific pieces of music the user hadn't heard before or in the form of musical artists to which the user hasn't previously been exposed. These systems make use of sevaral **Recommendation Tools** as part of their efforts to further engage their user base. Given the widespread use of such methodologies for enabling the discovery of new music, today's web-based streaming environment offers ample opportunity for those interested in exploring both the methods typically used for constructing recommender systems and how such systems can effectively be applied to enable the discovery of novel content.

Objective:

The goal of this project is to provide artists recommendation to **LastFM** users based on several recommendation systems and choose the best recommendation system by evaluating different models based on quantitative and qualitative techniques.

Data Collection:

The data set to be used is comprised of music listening information for a set of **1,892 users** of the **Last.fm** online music system. A total of **17,632 distinct musical artists** are represented within the data set. **11946 Tags/genres** were included in this data.

Types of Recommendation Systems Developed:

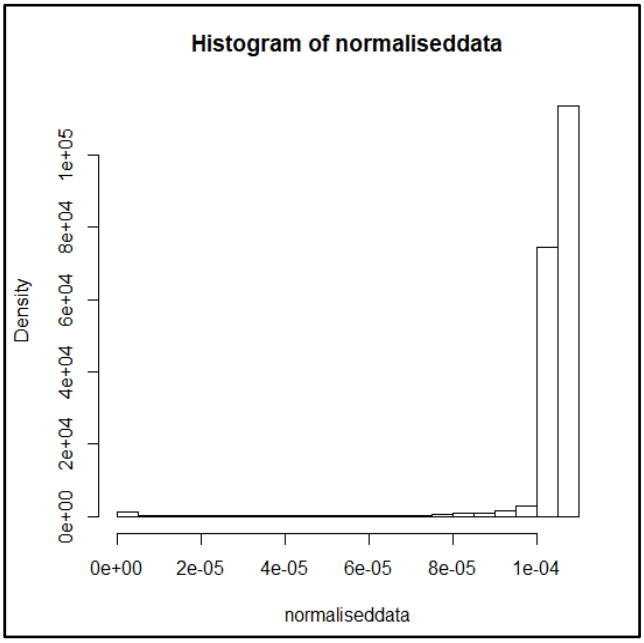
- Collaborative filtering recommendation system.
- Content Based Recommendation System.
- Hybrid Recommendation Systems.

1. Discussion of the transformations you use in step 1.

Collaborative Filtering:

For Collaborative filtering, we started with the given data Artists.dat. The data consisted of UserID, ArtistID and weight. Weight refers to the number of times the user has listened to a specific artist.

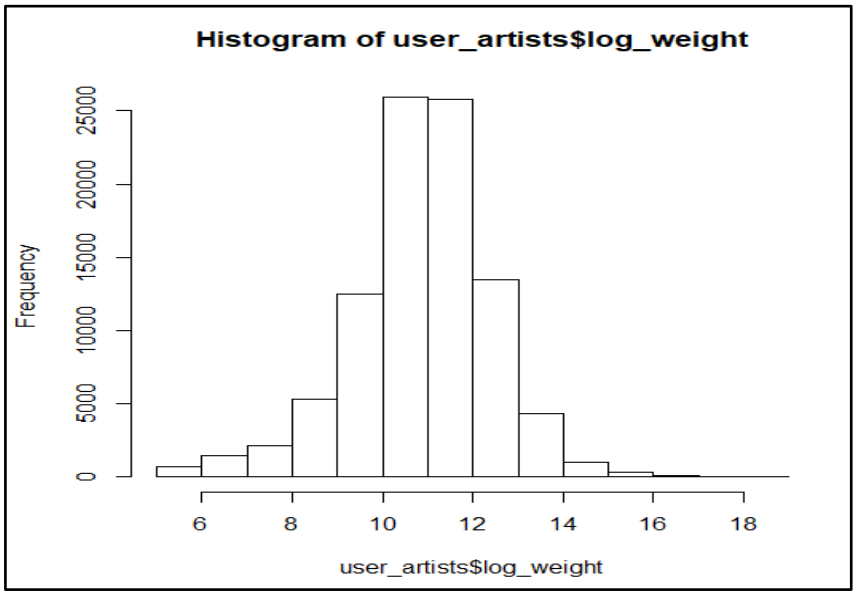
On further evaluation of the given raw data, we found that the minimum value of weight is 1 while maximum value is 352698. The average turning out to be 745.24. With such a huge range, we decided to further check the skewness of the weights. On plotting the data values on a normal function, we found high skewness in the data.



With such a high skewness, the recommendation matrix will not be right, as the available categories are wide, and the recommendation results would be inaccurate. Thus, to remove the skewness of the data, we did some mathematical transformation of the weights.

We took a logarithm with base 10 of the weights in the first step. This was not enough as the weights with 1 will have a log value of 0. Hence as a second step we treated the following log weights with linear model to obtain the intercept/constant value. After completing the transformations, the constant value was 5.389e+00. This was then added to the logarithm values weight, the resultant weights when plotted on a histogram showed a uniform distribution ranging from ratings **5.389 to 18.162**.

" user_artists$log_weight = log(user_artists$weight) + 5.389e+00 "



We had to subset the data as the given data was taking long to get processed, it was better for us to train our models on a subset of the entire data. The main question being, out of 17000+ artists, which one to be chosen for the subset. With some brainstorming, we used the MOST Popular artists for the dataset. We defined the most popular artists as the ones who are very frequently heard by user/users and have been heard by more than a specific number of users and not only followed by one or two users. This clearly defines that most popular artists would be better for our modeling as the most popular artists recommendation would be more beneficial for the company. With such a subset, we could capitalize the most sorted artists, thus in turn increasing the quality of data while reducing the quantity.

For our case, we filtered the artists with log weight greater than or equal to 10 and number of unique users heard to the artist as more than or equal to 5. With these constraints, we brought down the data set with 1943 unique artists and 1876 unique users.

CONTENT BASED RECOMMENDATIONS SYSTEMS:

User Tagged Artists and Tags Files were imported into R.

The user_taggedartists.dat file contains a listing of each instance in which a **last.fm** user has assigned a musical genre label (a.k.a., a "tag") to an artist. Though the file also contains the date (day, month, and year) of the "tagging", those attributes will be ignored for purposes of this project.

The file is found to contain a total of **186,479** total instances of **last.fm** users having applied a genre tag to an artist. All 1,892 users are represented within the file, and a total of **12,523 distinct artists** have been tagged with at least one genre name.

The **tags.dat** file contains a list of musical genres that **last.fm** users have used to categorize the various musical artists represented within the **last.fm** online music streaming platform. Each genre is assigned a unique identifier, or "tagID". A count of the unique tagID's reveals the presence of 11,946 distinct genre names:

Selection of Final Tags:

We aggregated the number of tags per user and we took only those **Top 200 tags** which are listened by the greatest number of distinct users.

Selection of Final Artists:

We only kept those **Artists which are used in the Collaborative Filter Recommendation Systems**. To get a match between both the RS.

Creating an Artist - Genre Matrix:

As part of our content-based recommendation efforts for this project, we would like to be able to provide last.fm users with the ability to receive a "Top N" list of suggested musical artists who are likely to be like an artist selected by the user. Furthermore, we'd like to be able to provide a user with a "Top N" list of suggested of artists for a user-selected musical genre. A path to the achievement of each of these objectives can be found through the creation of an artist-genre matrix.

Earlier were able to reduce our users-taggedartists data by limiting it to the top 200 genres as determined by the number of times each genre has been applied within the data set. Furthermore, we were able to calculate the number of times each artist had been labeled with a genre tag. We now use the results of those calculations as the basis of an artist-genre matrix.

A portion of the artist-genre matrix is shown below. The **row names** within the matrix represent the unique last.fm **artist ID's** (which are present in CF Matrix as well) while the **column names** represent the unique last.fm genre tag ID's we extracted from the **tags.dat (Top 200 as explained above)**.The Matrix was binarized.

Artist-genre matrix was used for purposes of generating an artist similarity matrix. Though an argument could be made in favor of making use of the raw genre tagging counts for each artist as the basis of a similarity matrix, an equally strong argument can be made in favor of treating all such tags as binary indications of whether **last.fm** users consider an artist to belong to a given genre. Therefore, we made use of a binary version of the artist-genre matrix for purposes of generating an artist similarity matrix.

2. Creation of Functions:

USER BASED CF – Pearson correlation

With the prepared dataset, we started modeling the UBCF. With the possibility to choose the similarity function among Pearson correlation and cosine, we defined the function for Pearson correlation using the mathematical formula

```
####Functions for user based CF using pearson correlation

meandiff <- function(data,i){
  data[i,] - mean(data[i,],na.rm=TRUE)
}

#userbased CF for multiple users
UserBasedCF_pearson <- function(train_data, test_data, N, NN, onlyNew=TRUE){

  ### similarity ###
  similarity_matrix <- matrix(, nrow = nrow(test_data), ncol = nrow(train_data),
                              dimnames = list(rownames(test_data), rownames(train_data)))

  for (i in rownames(test_data)){
    for (j in rownames(train_data)){
      sim = sum(meandiff(test_data,i) * meandiff(train_data,j), na.rm = TRUE) /
        (sqrt(sum(meandiff(test_data,i)^2,na.rm=TRUE)) *
           sqrt(sum(meandiff(train_data,j)^2,na.rm=TRUE)))
      similarity_matrix[i,j] <- sim
    }
  }
```

The above code snippet shows the Pearson correlation function definition for the user-based CF for multiple users. Similarly, Pearson correlation was defined for single user

ITEM BASED CF

After User Based CF, we started modeling the recommendation system based on item-item similarity or IBCF. For this, we again defined the similarity function with Pearson correlation and determined the similarity between the items/artists based on the log weight. With some relevant updates and modification in the previous Pearson correlation function, the new similarity function was defined for Item Based CF. The code snippet above shows the formulation of the same. For recommendation, the reference function was used to find the recommendation on based of item similarities.

```
#Item based using pearson correlation
meandiff2 <- function(data,i){
  data[,i] - mean(data[,i],na.rm=TRUE)
}


ItemBasedCF_pearson <- function(train_data, test_data, N, NN, onlyNew=TRUE){
  similarity_matrix = matrix(, ncol=ncol(test_data), nrow=ncol(train_data), dimnames = list(colnames(test_data), colnames(train_data)))
  for (i in colnames(test_data)){
    for (j in colnames(train_data)){
      sim = sum(meandiff2(test_data,i) * meandiff2(train_data,j), na.rm = TRUE) /
        (sqrt(sum(meandiff2(test_data,i)^2,na.rm=TRUE)) *
           sqrt(sum(meandiff2(train_data,j)^2,na.rm=TRUE)))
      similarity_matrix[i,j] <- sim
    }
  }
```

Evaluation Metrics - MAE

After defining the function for UBCF and IBCF, it was time to define the functions which could calculate the accuracy of the models. Not solely accuracy, but also the performance was gauged by evaluation metrics like RSME, MAE and classification accuracy.

The following code snippet shows the function definition for MAE

```
MAE <- function(prediction, real){

  if (nrow(prediction) == nrow(real) & ncol(prediction) == ncol(real)){
    RSME = sum( Mod(prediction - real) , na.rm = TRUE ) / (nrow(prediction) * ncol(prediction))
    return(RSME)
  }else{
    return("Dimension of prediction are not equal to dimension of real")
  }
}
```

Evaluation Metrics – F1

For defining the function for F1 metrics, we used the existing classification function and defined the F1 formula as the below snippet.

```
Classification <- function(prediction, real, threshold=NA, TopN=NA){
  if (nrow(prediction) == nrow(real) & ncol(prediction) == ncol(real)){
    # Threshold #
    if (!is.na(threshold)){
      TP = sum(ifelse(prediction >= threshold & real >= threshold, 1, 0), na.rm=T)
      FP = sum(ifelse(prediction >= threshold & real < threshold, 1, 0), na.rm=T)
      FN = sum(ifelse(prediction < threshold & real >= threshold, 1, 0), na.rm=T)
      Recall = TP/(TP+FN)
      Precision = TP/(TP+FP)
      F1 = 2 * ((Precision * Recall) / (Precision + Recall))
      Class_Thres = list(Recall, Precision, F1)
      names(Class_Thres) = c("Recall","Precision","F1")
    }
    if (!is.na(TopN)){
      TP = vector(, length = nrow(prediction))
      FP = vector(, length = nrow(prediction))
      FN = vector(, length = nrow(prediction))

      for (u in nrow(prediction)){
        threshold_pred = -sort(-prediction[u, ])[TopN]
        threshold_real = -sort(-real[u, ])[TopN]
```

3. Argumentation on using specific recommendation technique, evaluation systems and hybridization technique.

Recommendation technique:

Collaborative filtering:

We used collaborative filtering to start with as this is a very relevant solution for the given problem statement. This commonly used recommendation system fits well with the dataset we have. With around 1900 unique artists, we can recommend every type of user a different artist according to user-user and artist-artist similarity.

Within collaborative filtering, we used three different techniques like Userbased, Item based and Userbased techniques to check the results with different techniques with evaluation metrics and suggest the best one along them.

Content based:

In this Project as we deal with various Artists and genre of music, its highly likely that if there is a recommendation on Artists like a given artist or recommendation on genre based on genre /

artists becomes very helpful to the users in the real-world music websites. Hence, we choose the Content based Recommendation systems with Artists and Genre / Tags.

Hybrid Based:

In recommendation systems always, it is better to use a hybrid, by combining the two recommendation systems that give better accuracies and reduces sensitivity of the data.

In our case we used two hybrid techniques for comparison

Cluster based + Content Based:

In a website like LastFm the Clustering based Recommendation system helping in understanding various types of similar users and if we recommend content to each class of user defined by the cluster it would be helpful, so we tried this recommendation system in our case study.

Userbased + Content Based:

In this approach based on nearest neighbor of users the content would be recommended. For example, user listens to artist A and user B is close to user so recommend him the artists heard by user and the artists close to those artists. The algorithms calculate the better between these 2 and recommends finally the artists.

Evaluation Metrics:

We used different evaluation metrics such as prediction accuracy, ranking accuracy and classification accuracy to compare the results of different recommendation techniques.

RMSE, Recall, precision and F1 are the most commonly used evaluation metrics in recommendation systems. In order to evaluate prediction accuracy, Root mean square error and mean absolute error was calculated. Further, we wanted to evaluate product relevancy and thus we calculated the classification accuracy metrics Recall, Precision and F1. Finally, we also calculated ranking accuracy like AUC for userbased and item based collaborative filtering to evaluate based on the ranking of the user.

4. A final comparative discussion of the recommendation systems identifying the best one in step 3.

Please find the below results of several recommendation systems.

Own functions:

Evaluation & Classification	USER BASED CF	ITEM BASED CF	Cluster Based CF	Content Based CF	Hybrid (Cluster + Content based)	Hybrid (Content + Userbased collaborative)
RMSE	0.09884	1.4935	0.1762	1.20905	0.8892	0.602
MAE	0.01027	0.2001	-	0.153425	0.10588	0.07
Precision	0.8	0.666667	0.805081	1	0.99627	1
Recall	0.4	0.1	0.908086	0.33287	0.43274	0.9316
F1	0.53333	0.173913	0.853487	0.49948	0.60339	0.964
AUC	0.5476	0.5033	-	-	-	-

Recommenders lab:

Prediction accuracy	POPULAR	ITEM BASED CF	USER BASED CF	HYBRID(SVD)
RMSE	0.87536	0.93223	0.88048	0.86481
MSE	0.76626	0.86906	0.77529	0.74800
MAE	0.64196	0.72331	0.644638	0.63433

The above results show us that the RMSE and MAE values for User based is the best as it has lowest along with a good value for precision. This shows us that userbased collaborative filtering is the optimal quantitative results for this case Study. We can also say that User based is the best as we have more number of items than the users.

Below are the top 3 artists predictions for few users using userbased collaborative filtering

Users	V1	V2	V3
5	65	1375	81
8	306	466	913
12	930	718	1854
14	227	5446	2893
15	868	220	1412
20	603	2605	2677
23	1014	1713	58
30	298	681	294
31	1713	1019	1014

But for Qualitative results we found that the **Content Based** Recommendation systems was **second best** performing close to real life and recommending artists like the artists and recommending top artists for a genre as shown below:

Qualitative Results:

Display a List of 4 Recommended Artists like a Given Artist

The artist similarity matrix allows **Last.fm** users to find musical artists that are like one they specify. To simulate how this might work, we can randomly select an artist ID and display a list of 5 recommended similar artists:

```
|Artists Similar to Comeback Kid |
|:-------------------------------|
|Have Heart                      |
|Terror                          |
|No Use for a Name               |
|Propagandhi                     |
> |
```

Generate a Top N Artist List by Genre:

We made use of the non-binary version of the artist-genre matrix to provide a user with a "Top N" list of suggested of artists from a user-selected musical genre. Recall that the artist-genre matrix is comprised of counts of how often a given artist has been labeled as belonging to a genre. This metric can serve as a proxy for how strongly the **last.fm** user community feels that an artist belongs to a genre. Therefore, we can use the tag counts to rank artists within genres: The more often they've been tagged with a genre label, the higher they rank within the genre.

```
|Top Artists in mossy genre: |
|:---------------------------|
|Diary of Dreams             |
|Moonspell                   |
|Marilyn Manson              |
|Michael Jackson             |
|ABBA                        |
>
> ##########################################################################################
```

5. A discussion of advantages and disadvantages of the best recommendation system(s) and some thoughts for improvement.

Advantages:

Userbased:

- The main advantage of using user-based system is it is most appropriate collaborative technique when there are less users than items and gives good results.

Content Based:

- Comparison between the Items is possible. And User gets recommendation to similar types of music on his past playlist.
- Content based recommended system overcomes the challenges of collaborative filtering techniques. They have the ability to recommend even if there are no ratings provided by users for new items. It has the capacity to adjust its recommendations in a short span of time if the user preferences change. Content based filtering technique can also provide explanations on how recommendations are generated to users.

Disadvantages:

Userbased:

- **Cold start problem**, where a new user/item with no ratings is added, is extremely difficult to deal with using these methods. This issue is actually quite recurrent when designing recommendation systems so keep it in mind!
- As we use a user's own ratings to make a prediction. If we want to predict User1 rating for Rock Music, the item-based approach will look at User1 ratings for Related features to Rock Music, (thus very similar to the target Music). Meanwhile, with the user-based approach, we look at similar users who might have overlapping but different interests than User1 which may result in a decrease in accuracy.

Content Based:

- Overspecialization is one of the main concerns generally faced.