6CCS3PRJ Final Year Project

Analysis and testing of different Recommendation Systems using Machine Learning Techniques

Final Project Report

Author: Adam Akram

Supervisor: Dr Senir Dinar

Student ID: 1842752

22/04/2022

Abstract

In today's world there is an abundance of data available to us through the advancement in technology and the progression of the internet. With all this data available to us, it is therefore important that we filter through this data and allow individuals to focus on data which is more relevant to them. This project aims to look at some recommendation systems and test these systems on a data-set associated with music streaming. Using a variety of evaluation metrics, we will be able to compare these systems and conclude which seems to be the most effective.

Originality Avowal

Acknowledgements

I would like to thank my supervisor Dr Senir Dinar, for his assistance throughout the year for the project. I am grateful for his advice and patience throughout the different stages of the project.

Contents

Chapter 1

Introduction

1.1 Project Motivation

The advancement of technology is occurring at an exponential rate. This has been stated in Moore's law, where it is said that the amount of transistors on a microchip doubles every two years, despite the cost of computers being halved [1]. Naturally, with more computing power available at a cheaper cost, computers will collect and process larger amounts of information. A key issue with increasing amounts of this information is that in most cases this information will be irrelevant to the individual. To filter through this information and only display what is relevant, we can use recommendation systems. .

Recommendation systems are used to determine the best allocation of information to individuals [2]. They typically seek to predict the preferences of users based on which they would give users these items of information. The main purpose of these systems is allowing users to primarily interact with information relevant to them as an individual..

For these recommendation systems, typically we are using machine learning algorithms. Machine learning can be defined as an application of artificial intelligence in which we facilitate systems to learn and improve from previous experience without being programmed directly [3]. The main objective for machine learning algorithms is to use data for themselves to come up with a solution. These algorithms are reliant on large amounts of data which have now become increasingly available to us. .

Looking at the amount of algorithms available to us, like with the large amounts of data we have, it is important to focus on recommendation systems that provide us with better recommendations, which this project is focused on.

1.2 Project Aims and Objectives

The main objective of the project is to measure how certain recommendation systems compare with each other using a singular data set. To measure the effectiveness of these recommendations, we will develop a few recommendation systems to use a variety of quantitative and qualitative evaluation techniques on. Once I have analysed these results, I will also look at the social and ethical implications of the recommendation systems. Finally, I will identify the strengths and weaknesses of these systems and highlight my opinion on them.

Chapter 2

Background

In this chapter, we will look at the relevant articles and information associated with recommendation systems. I will look at the evaluation techniques that other researchers have used to compare recommendation systems.

2.1 What are Recommendation Systems?

Recommendation systems can be described as tools used to navigate through large amounts of information in complex and large information spaces. This is done to determine the best allocation of information to individuals and what they would prefer to see [4]. They typically seek to predict the preferences of users; based on these preferences, they provide users with items which may be of interest to them.

2.2 How Does A Recommendation System Function?

Recommendation systems start by collecting relevant information from users to create a user profile. This feedback can be made up of the user's characteristics, how they behave, and the nature of the content they are accessing. It is imperative that the user profile is well constructed to provide the best recommendations for the system. The way in which feedback for a recommendation system is taken is through either implicit feedback where user preferences are predicted based on user activity, or explicit feedback which involves users inputting their preferences directly for items[5]. Information can be also established through hybrid feedback whereby we combine methods used in implicit and explicit feedback techniques.

Implicit feedback systems create the desired user profiles by examining information associated with user activity. Using this data, user preferences are predicted. A few examples of information included in this data includes purchase history, time spent on websites and artists the users follow. This way of creating profiles seems easier, as users would not need to be directly involved, although this means that the recommendations which are being provided to us are less accurate. [6].

Explicit feedback typically has a process whereby users are asked to provide their ratings for items directly. These ratings will then be used to construct the model, the more ratings that are provided, the better this model will be. This in comparison to implicit feedback requires more effort from the users. Despite requiring more effort, the recommendations which are provided from this system will be stronger and more authentic, since there will be no assumptions. Users can more accurately see how certain recommendations were made in the system and be more confident in them as a result. [6].

Hybrid feedback brings together elements of implicit and explicit feedback to mitigate issues which surround both systems. For example, in a hybrid system we can make it easier for users by only providing them with an option to enter ratings for items. The rest of the data needed to build user profiles can be taken through implicit feedback approaches such as analysing the user's activity. . Once all the data from the feedback stage has been collected, we can then apply this to our chosen algorithm depending on the type of data we have. After passing the data through the relevant algorithm we are given items which our recommendation system suggests that users may prefer. Figure 2.1 presents the key steps in this process.

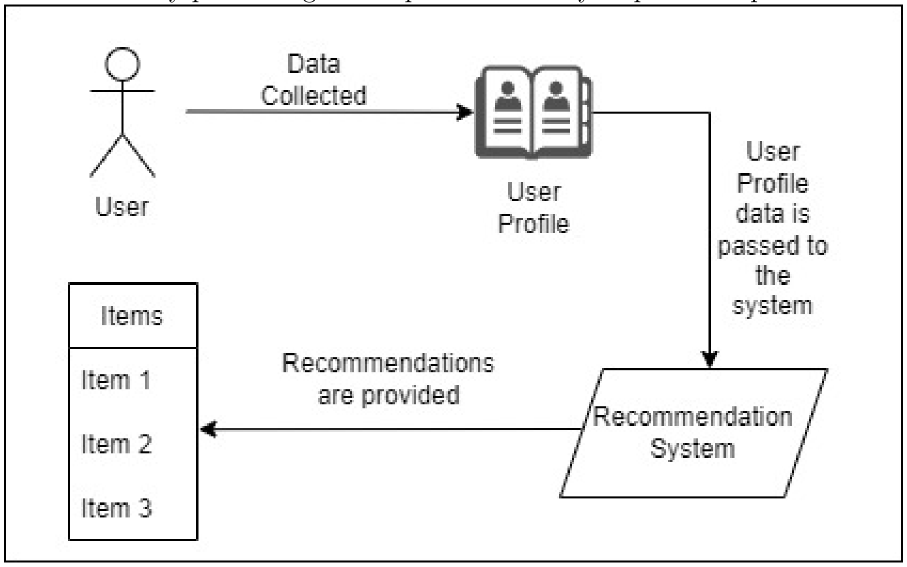


Figure 2.1: A summary of the different stages of the recommendation process

2.3 Different Types of Recommendation Systems

In this section of the report, we will look at the recommendation systems that are to be used for this project.

2.3.1 Collaborative Filtering

Collaborative filtering focuses on the premise that similar users will also have similar likes and interests (user-based collaborative filtering). Based on this collected data, we will be presented with recommendations. A short example of how this can be described is that if “User A” and “User B” were to have a purchase history that overlaps heavily, if “User A” were to buy a book, then the same book would be recommended to “User B” [7]. No information regarding the items which are being recommended are required for this method of recommendation [4].

Collaborative filtering can also work in reverse, by looking at similar items and recommending users for them. (Item Based Collaborative Filtering).

Finally another application of collaborative based filtering is based on clustering, (Cluster Based Collaborative Filtering)[9]. Clustering involves grouping similar items into groups rather than looking at them individually and providing our recommendations.

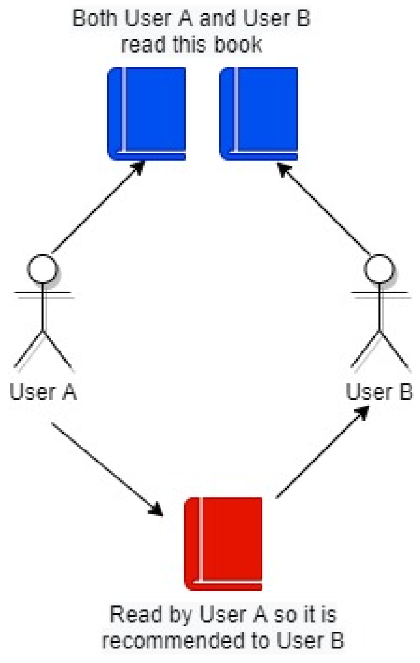

Both User A and User B read this book
User A
User B
Read by User A so it is recommended to User B

Figure 2.2: A representation of how user based collaborative filtering works

2.3.2 Content Based Filtering

Content Based Filtering works with the data that users provide individually opposed to a group of similar users with similar interests. Recommendations are provided using a explicit rating method done by users. With these ratings, we will be provided with what our system thinks is the best for the individual. When using this method with books for example, this system would collect data such as the "genre", "author" or "topic" of the book. User profiles may be derived based on user activity or specifically asked. Taking all these factors into account, the system will present the users with recommendations which it deems to be relevant.[7]

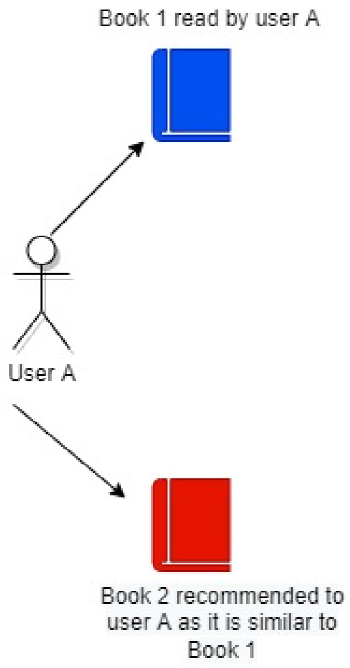


Figure 2.3: A representation of how Content Based filtering works

2.3.3 Demographic Based Filtering

This approach of filtering works towards categorising users based on their demographic attributes. Demographic based systems also take into consideration user opinions for items as a basis on what to set recommendations. This type of approach focusses on user-to-user correlations [8].

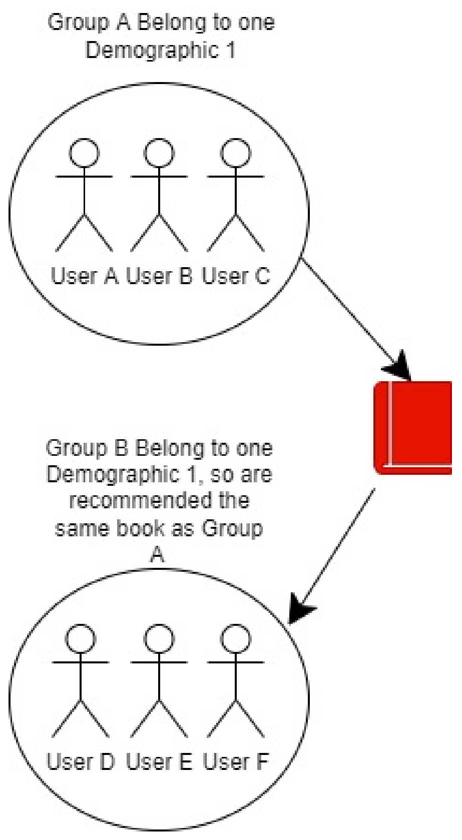

Group A Belong to one Demographic 1
User A
User B
User C
Group B Belong to one Demographic 1, so are recommended the same book as Group A
User D
User E
User F

Figure 2.4: A representation of how Demographic Based filtering works

2.3.4 Hybrid Approaches

Finally, another method that is used for recommendation is to combine different approaches into one. More specifically, we may use elements from collaborative filtering with some from content-based filtering. An instance of where this may be applied is if community knowledge already exists, along with detailed information associated with individuals. With this hybrid system we can take data from both these areas. In theory, this will provide us with the most accurate recommendations [7].

2.4 Similarity Measures

With recommendation systems, fundamentally we are looking at the similarity between two items. These are used as the basis for us creating our systems. There are a number of mathematical formulae that help us calculate this similarity. These techniques can be used to implement our recommendation systems.

Cosine Similarity

This is a measurement of the cosine of the angle between two non-zero vectors[9]. The calculation is centred on the orientation of the vectors instead of the magnitude. For instance, two cosine vectors aligned in the exact same orientation, have the highest similarity of 1. The following can be demonstrated in the formula below, where the similarity is looked at between two vectors, i and j:

$$\text{similarity(i,j)=cosine}(\vec{i} \cdot \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{||\vec{i}||_2 * ||\vec{j}||_2}$$

Pearson Correlation Coefficient

We can use the Pearson correlation coefficient as a method to examine the relationship between two continuous quantitative variables[10]. This measure considers the orientation of vectors, however, unlike cosine similarity, also takes into account the magnitude of the vectors too. The following can be seen in an example with vectors i and j:

$$\text{similarity(i,j)=PearsonCorrelation(i,j)} = \sum(i - \vec{i})(j - \vec{j})$$
$$\sqrt{\sum(i - \vec{i})^2 \sum(j - \vec{j})^2}$$

Chapter 3

Approach

3.1 Language

For the development of this project, the decision was made to use ”R” as the programming language for the recommendation systems and evaluation metrics. ”R” provides us with many facilities for carrying out machine learning operations specifically, such as classification and regression.

There were more packages in ”R” also in comparison to other languages that were focused around transforming messy data, for example ”dplyr” where data can be changed into a structured format.

3.2 Data

The data was taken from the website ”Kaggle” , an online platform where data scientists publish and use code provided by others. This code can be used to provide data to build machine learning models, more specifically recommendation systems for our project. Since there have been a number of studies on recommendation systems, using a platform such as ”Kaggle” will enable us to find data that is unique and has not yet been tested.

Chapter 4

Evaluation Methods

To compare our chosen recommendation systems, we have used a variety of different evaluation metrics such as prediction accuracy, ranking accuracy and classification accuracy to compare the results of different recommendation techniques

4.1 Predictive Accuracy

This can be defined as how closely the predicted values of a system match with the actual values of the target field within the uncertainty due to statistical fluctuations [11].

4.1.1 Root Mean Squared Error(RMSE)

This is a popular evaluation method for recommendation systems and works by calculating the square root of the arithmetic mean of squared observations[12]. In the context of this project, the arithmetic mean of the difference between the desired predictions and the actual value. The square root is then calculated from this result.This can be seen in the formula[13]:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(Predicted\ Value - Actual\ Value)^2}{Number\ of\ Predictions}}$$

When calculating the RMSE, if the value ends up being much higher than that of the training set, this likely means that the predictions provided are poor. The lower the RMSE, generally we have higher accuracy have in predictions.Additionally, we can increase the accuracy of RMSE by increasing the amount of observations.

4.1.2 Mean Absolute Error(MAE)

For Mean Absolute Error we are calculating the average of every difference between the actual value and the predicted value. Similar to RMSE, for MAE the lower the value, the more accurate our predicted values are. MAE can be represented in the following formula in context of our project[14]:

$$MAE = \frac{\sum_{i=1}^{n} abs(Predicted\ Value - Actual\ Value)}{Amount\ of\ Ratings}$$

4.2 Classification Accuracy

We can define this as the number of correct predictions that a system makes to the total number of input samples [15]. The terms true positive, false positive, true negative and false negative are used throughout classification accuracy metrics[16]. True positives are where our system correctly predicts a positive class, whereas true negatives are where our system correctly deduces a negative class. However, false positives refer to where our system incorrectly predicts a positive class, meaning false negatives happen when our system incorrectly predicts a negative class.

4.2.1 Precision

The precision is a measure of the ratio between the ”true positives” and the ”false positives”. Relating this to our project, we are checking to see if the system we are testing produces[16]. The closer our score is to 1, the better the precision, because a score of 1 indicates that there are no false positives in our recommendation system. Precision can be represented by this formula[16]:

$$Precision = \frac{True\ Positive}{False\ Positive + True\ Positive}$$

4.2.2 Recall

Recall identifies how accurately our model is able to distinguish ”True Positives”. Relating this to recommendation systems, for items we have been recommended, recall will tell us if these items are accurate based on our user.The closer the recall value is to 1, the lower the amount of ”False Negatives” there are in our recommendation system. This is shown in the formula[16]:

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

4.2.3 F1 Score

After considering the values obtained for both recall and precision, the f1 score takes both of these into account. The closer the value for our F score is to 1,the better. The F1 score may be a useful measure when we want to establish a balance between precision and recall, and can be used when our data has a large amount of "Actual Negatives" indicating an uneven class distribution[17] The following can be represented in this formula[18]:

$$F1\ Score = 2 \times \frac{Precision+Recall}{Precision+Recall}$$

4.3 Ranking Accuracy

This handles the varying levels of utility of recommended items with regards to their positions in the ranked list put forward by the user. [19].

4.3.1 Area Under the ROC Curve (AUC)

The ROC Curve

The Receiver Operating Characteristics Curve (ROC), looks at the rate of true positives against the rate of false positives. This curve will show us the sensitivity of the classifier model[20]. The true positive rate and the false positive rate can be defined as the following[16]:

$$True\ Positive\ Rate = \frac{True\ Positive}{True\ Positive+False\ Negative}$$

$$False\ Positive\ Rate = \frac{False\ Positive}{False\ Negative+True\ Negative}$$

AUC

This area under the ROC curve is calculated as the entire two-dimensional area which lies under the whole ROC curve. The value calculated from the AUC is between 0 and 1. Looking at this from the perspective of our project, if AUC is 1, we can assume that the predictions from our recommendation system are accurate 100% of the time. If AUC is 0 we can assume that the recommendations provided are entirely false.

Chapter 5

Experiments

5.1 Inspiration

Users of internet music streaming services such as Spotify, Apple Music and Last-FM have a number of ways to find new music. This may be in the form of new music art, artists, or specific music that they have not heard of previously. To increase user-engagement, these platforms already utilise various different recommendation tools. It is our goal to see how specific recommendation systems can provide Last-FM users with artist recommendations. This will be done using several quantitative and qualitative methods.

5.2 Data Set

The data set that was used to evaluate and analyse recommendation algorithms was taken from the "Last FM music recommendation data set" [21]. This data used information which corresponded to 1892 Last FM internet and radio system users. Associated with this data was also information regarding 17362 unique musical performers. In addition, there were also 11946 tags and genres.

5.3 Collaborative Based Filtering

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

Looking at 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.

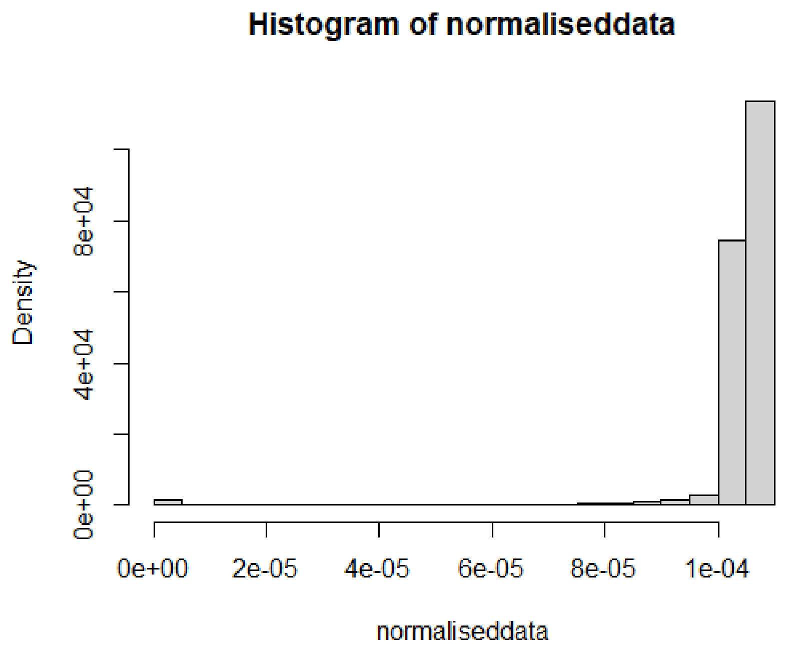


Figure 5.1: Representation of highly skewed data

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

We took a logarithm with base 10 of the weights in the first step. This was still insufficient as the weights with 1 would have a log value of 0. Therefore, with an additional step, we treated the following logarithmic weights with a linear model to obtain the constant value. After the transformations were completed, the constant value was 5.389e+00. This was then added to the logarithm value's weight. The final weights when plotted on a histogram showed a uniform distribution ranging from ratings 5.389 to 18.162.

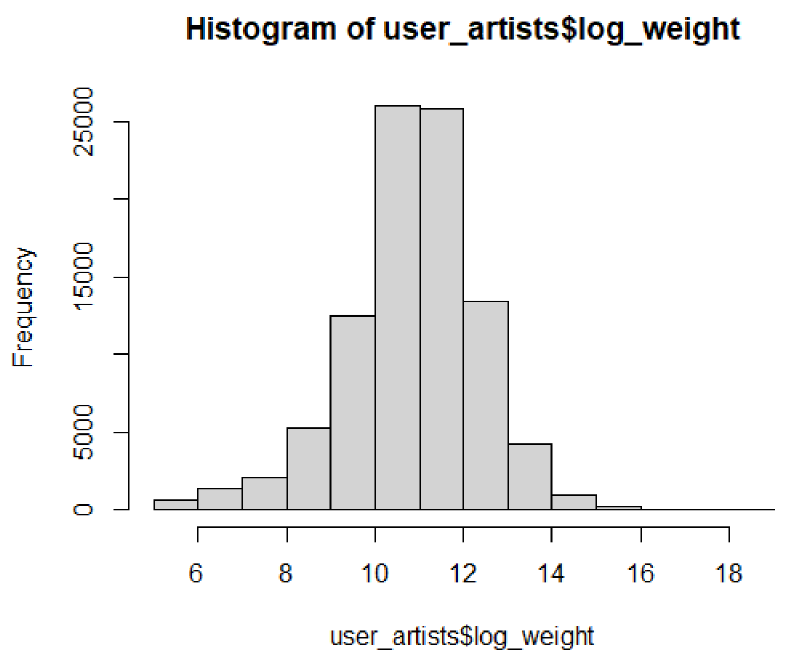


Figure 5.2: Representation of altered, unifrom sidtribution of data

Due to the data being too large and taking too long to process, 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 of these were to be chosen for our subset. We used the most popular artists for the data set. The most popular artists were given as those that are very frequently heard by users. The artists who were eliminated were those who were followed by simply two or fewer users. This clearly defines that most popular artists would be better for our modelling, as the recommendation of most popular artists would be of more use to Last FM. With such a subset, we are increasing the quality of data while reducing the quantity.

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

5.4 Content Based Filtering

For content based filtering the file "user_taggedartists.dat" was used. This contains a listing of each instance in which a Last FM user had assigned a musical genre label ('tag') to an artist. Although the file also contains the date (day, month and year) of the 'tagging', those attributes will be ignored for the purposes of this project.

The file 'user_taggedartists.dat' contains a total of 186,479 total instances of Last FM users who have 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 categorise several 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.

5.4.1 Selection of Final Tags

We aggregated the number of tags per user and selected only the top 200 tags which are listened by the largest amount of distinct users.

5.4.2 Selection of Final Artists

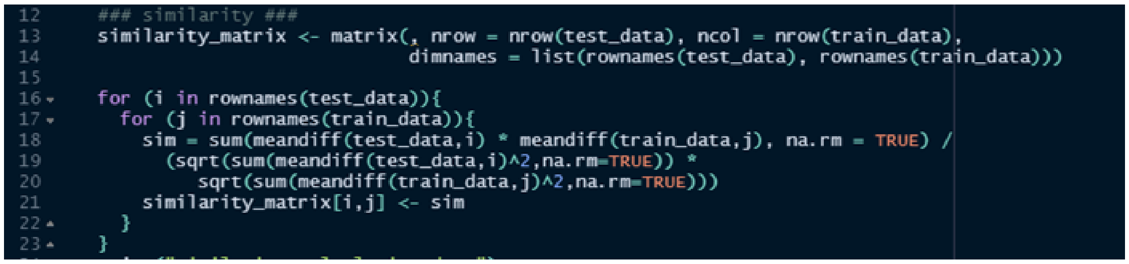
We only kept those artists who were used for our collaborative filtering recommendation system. This was done so that we could compare the two systems easily.

5.4.3 Creating an Artist - Genre Matrix

As part of our content-based recommendation system, we would like to be able to provide Last FM users with the ability to receive a "Top N" list of suggested artists who would likely be selected by the user. Furthermore, we'd like to be able to provide a user with a "Top N" list of recommended artists from a user selected musical genre. These goals were possible through the creation of an artist-genre matrix (ag_mat).

Previously we were able to reduce our "user_taggedartists.dat" file by limiting it to the top 200 genres ("tags"). We were also able to calculate the number of times each artist had been labelled with a genre tag. We now use the results of those calculations as the basis of an artist-genre matrix.

A section of the artist-genre matrix is shown below. The "rownames" within the matrix represent the unique Last FM's artist IDs (which are also present in Collaborative Filtering Matrix) while the "columnnames" represent the unique Last FM genre tag ID's we extracted from the tags.dat (Top 200 as explained above).



```r
### 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
  }
}
```

Figure 5.3: Representation of how the artist-genre matrix was created

The matrix then underwent binarization[9] whereby all words corresponded to an ID, for example, " Michael Jackson" would be written as " Michael-1 Jackson-2". From here, each word would be replaced by it's tag to give us a binary vector, in this case it would be "$< 1, 2 >$". This can then be further refined by providing each of our words with four possible slots and setting this slot to relate to a specific word. For example, $< 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1 >$. This method is only suitable with our data, as with longer sentences, the harder it is to maintain the correct order.

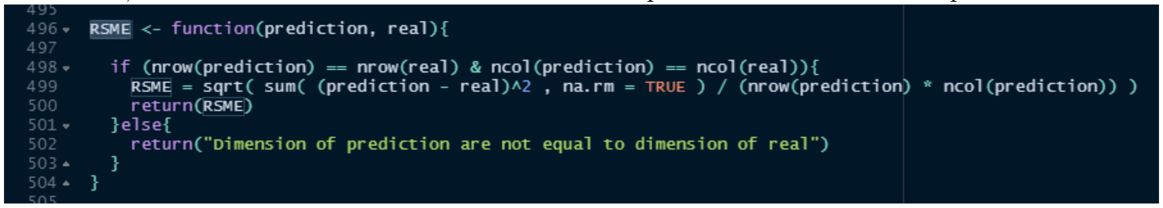
An artist-genre matrix was used to generate an artist similarity matrix. Despite this, an argument could be made for utilising the raw genre tagging counts for each artist as the basis of a similarity matrix. Additionally, we could also treat all such tags as binary indications of whether Last FM users 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 our artist similarity matrix.

5.5 Creation of Functions

5.5.1 Evaluation Functions

RMSE

This function works by taking our predicted values and real data as arguments. For each given data value, the mathematical function of RMSE is computed and returned to be printed.



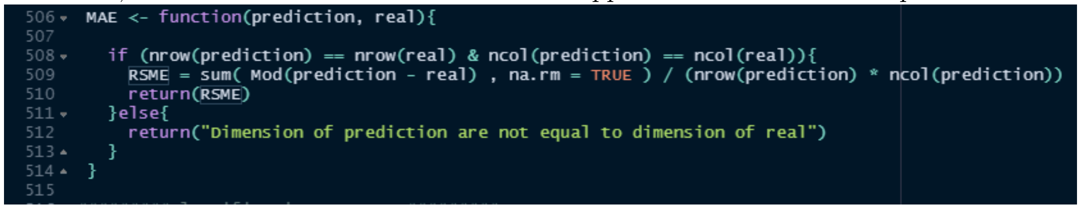
```r
RSME <- function(prediction, real){

  if (nrow(prediction) == nrow(real) & ncol(prediction) == ncol(real)){
    RSME = sqrt( sum( (prediction - real)^2 , na.rm = TRUE ) / (nrow(prediction) * ncol(prediction)) )
    return(RSME)
  }else{
    return("Dimension of prediction are not equal to dimension of real")
  }
}
```

Figure 5.4: RMSE function to evaluate data

MAE

In this function we also take our predicted values and real data as arguments. For each given data value, the mathematical function of MAE is applied and returned to be printed.



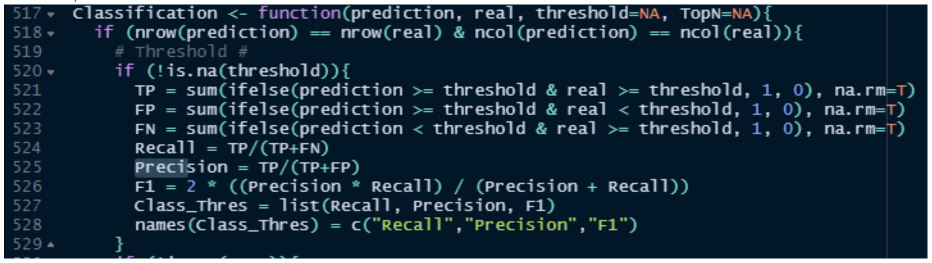
```r
MAE <- function(prediction, real){

  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")
  }
}
```

Figure 5.5: MAE function to evaluate data

Precision, Recall and F1 Evaluation

For our classification metrics, we have created a confusion matrix, a table that categorizes our predicted values against our actual values. After establishing this we can perform mathematical functions for our true positive(TP) , true negative(TN) and false negative(FN) values to work out precision, recall and our F1 score.

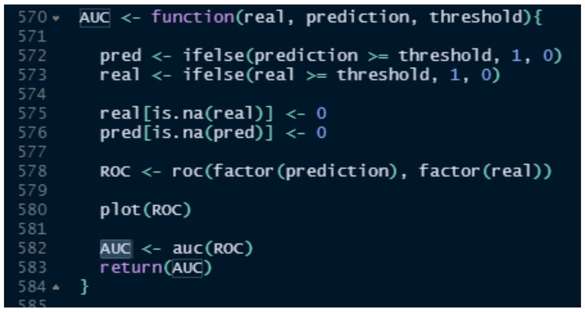


```r
Classification <- function(prediction, real, threshold=NA, TopN=NA){
  if (nrow(prediction) == nrow(real) & ncol(prediction) == ncol(real)){
    # Threshold #
    if (!is.na(threshold)){
      TP = sum(ifelse(prediction >= threshold & real >= threshold, 1, 0), na.rm=T)
      FP = sum(ifelse(prediction >= 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")
    }
```

Figure 5.6: Function used for all classification evaluation metrics

AUC

AUC calculates specificity and sensitivity when using a section of the data randomly sampled. The AUC function then increases the proportion of data it uses, and plots how sensitivity and specificity change.



```
AUC <- function(real, prediction, threshold){

  pred <- ifelse(prediction >= threshold, 1, 0)
  real <- ifelse(real >= threshold, 1, 0)

  real[is.na(real)] <- 0
  pred[is.na(pred)] <- 0

  ROC <- roc(factor(prediction), factor(real))

  plot(ROC)

  AUC <- auc(ROC)
  return(AUC)
}
```

Figure 5.7: Function used for calculating AUC

5.5.2 Recommendation Functions

Content Based Filtering

To apply content based filtering, the user's ratings are first converted into a matrix. Once this has been done, we look at the cosine angle of the vectors in the artists of the matrix. This data can be used to calculate a square matrix with dimensions which are equal to the amount of artists in the previous matrix. The score of each artist is computed as being the linear combination of all the cosine angles in relation to the user.

Collaborative Based Filtering

For collaborative based filtering, the user's ratings are also converted into a matrix. Once this has been done, we look at the cosine angle of the vectors in the rows(users) of the matrix. This data can be used to calculate a square matrix with dimensions which are equal to the amount of artists in the previous matrix. The score of each row is computed as being the linear combination of all the cosine angles in relation to the user.

Demographic Based Filtering

Finally to apply demographic based filtering, the user's attributes are again converted into a matrix. Once this has been done, we look at the cosine angle of the vectors in the artists of the matrix. This data can be used to calculate a square matrix with dimensions which are equal to the amount of artists in the previous matrix. The score of each user is used to provide recommendations. The rows and columns represent users. An individual user is compared with all other rows and columns. Through this, users with similar attributes are found and artist recommendations are provided that users in the demographic will like. Our demographic attributes are taken from the user id's in our data set. They are generated from the age and location of users.

Similarity Measure Used

When calculating similarity, both metrics of cosine similarity and Pearson correlation were used and implemented within different recommendation systems. Whilst creating functions utilising Pearson correlation, it was found that values produced for evaluation metrics were mostly 0 values. This was due to the fact that in Pearson correlation, the magnitude of the vectors are also considered in the computation, and because the differences between the magnitude and the angle of vectors was small.The reason for this in our data set was due to users only having listened to a few select artists; despite our data containing information regarding a large amount of artists.

Chapter 6

Legal, Social, Ethical and Professional Issues

- Recommendation systems are at the risk of hiding users from specific viewpoints. This would create things such as ”filter bubbles”. The creation of these “filter bubbles” would have a transformative effect on society as they would cause an effect on a normal functioning public debate. The impact of this can be seen with recent events and news that is against the vaccine, this has been linked to lower herd immunity [22].
- With recommendation systems in general there also becomes an issue when detecting sarcastic comments or writing. This kind of information may then be recommended and misrepresented as factual, leading to confusion.
- To use recommendation systems, we need to gather data associated to individuals. This data by default may be sensitive in nature so it is important that users have full confidence that they are aware of exactly how their information is used.
- With implicit data collection, as user's do not have a direct say in what data they would like to use to build their respective user profiles, data may be extracted from websites or other areas where they had no intention of being used. This can particularly be detrimental as depending on the country you are in; authorities may raise this as an issue of concern that you are linked to materials which are opposed to their regime.

6.1 British Computing Society code of Conduct and Code of Good Practice

This project was guided by the rules and professional standards which have been established by the British Computing Society. The algorithms that have been tested and analysed have been done so in the interest and public security, health, and the well-being of the surrounding environment.

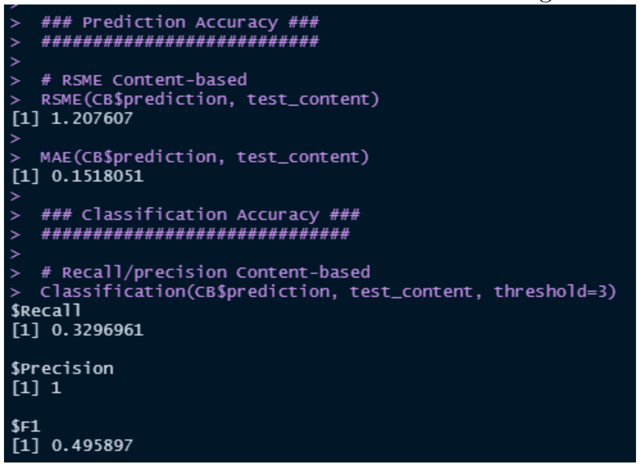
Chapter 7

Results/Evaluation

7.1 Quantitative Results

7.1.1 Collection of results

After executing the code, results were recorded on a table while the code was running. This example shows us the evaluation metrics for content based filtering.



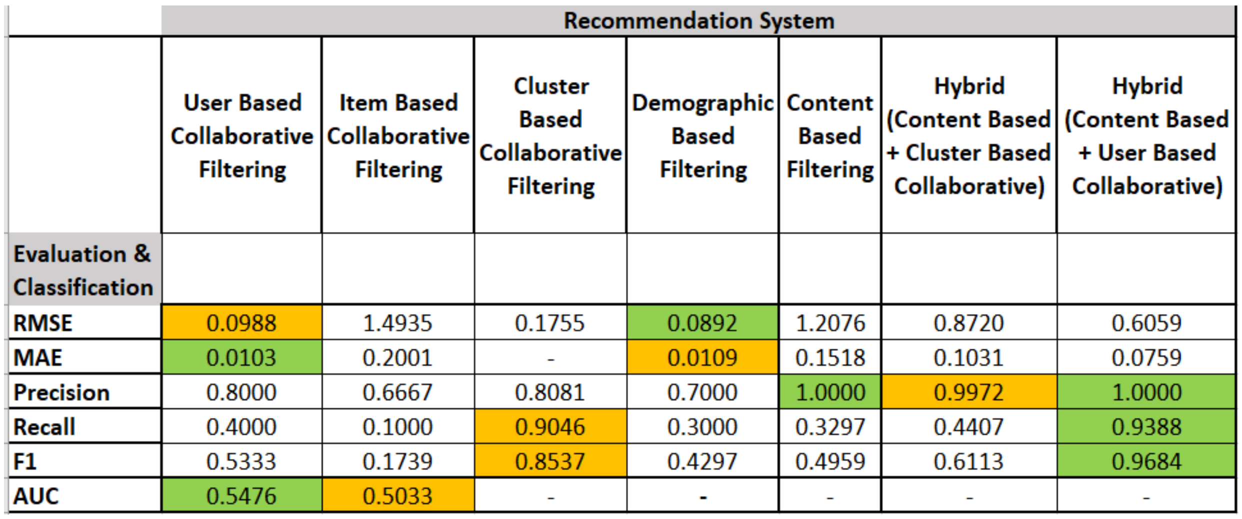
```
>  ### Prediction Accuracy ###
>  ###########################
>
>  # RSME Content-based
>  RSME(CB$prediction, test_content)
[1] 1.207607
>
>  MAE(CB$prediction, test_content)
[1] 0.1518051
>
>  ### Classification Accuracy ###
>  ###############################
>
>  # Recall/precision Content-based
>  Classification(CB$prediction, test_content, threshold=3)
$Recall
[1] 0.3296961

$Precision
[1] 1

$F1
[1] 0.495897
```

Figure 7.1: This shows how results were collected from R Studio.

7.1.2 Results Table



	Recommendation System						
	User Based Collaborative Filtering	Item Based Collaborative Filtering	Cluster Based Collaborative Filtering	Demographic Based Filtering	Content Based Filtering	Hybrid (Content Based + Cluster Based Collaborative)	Hybrid (Content Based + User Based Collaborative)
Evaluation & Classification							
RMSE	0.0988	1.4935	0.1755	0.0892	1.2076	0.8720	0.6059
MAE	0.0103	0.2001	-	0.0109	0.1518	0.1031	0.0759
Precision	0.8000	0.6667	0.8081	0.7000	1.0000	0.9972	1.0000
Recall	0.4000	0.1000	0.9046	0.3000	0.3297	0.4407	0.9388
F1	0.5333	0.1739	0.8537	0.4297	0.4959	0.6113	0.9684
AUC	0.5476	0.5033	-	-	-	-	-

Figure 7.2: This shows a collection of all results gathered as a result of testing. Highlighted in green are the best and in yellow are the second best results for a given metric.

7.1.3 Issues

In most of the cases the recommendation systems were using a section of the data to calculate the recommendations; however, in some cases, the multiplication of matrices for which rows and columns were not equal was not possible. This is why in some areas of the table like with AUC and MAE we are missing some results. Despite this impact, with all other evaluation data gathered, we still have a good representation of each recommendation system's performance.

7.1.4 Analysis of Results

The results gathered from our tests indicate that the hybrid approach of using ”Content-Based Filtering and User-Based Collaborative Filtering” provided the best classification accuracy. With this approach we were given with the best precision, recall and F1 score.

We can see that in terms of predictive accuracy, ”Demographic Based Filtering” produced the best results, with the lowest RMSE and MAE. This shows us that generally errors were the lowest in this approach.

Although AUC was only collected for ”User Based Collaborative Filtering” and ”Item Based Collaborative Filtering” we saw that ”User Based Collaborative Filtering” provided us with the best value for AUC. It would be unfair for us to conclude that ”User Based Collaborative Filtering” has the best ranking accuracy as we hadn't measured this in the other recommendation systems.

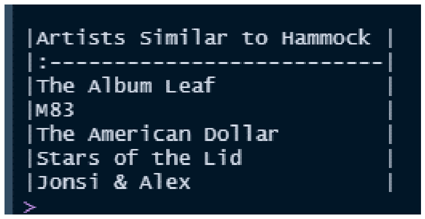
The worst performing system all-round seemed to be the implementation of Item Based collaborative filtering. This was due to the poor predictive accuracy, with RMSE and MAE being the highest. The classification accuracy was also quite poor, with lowest F1 score and recall values.

Taking all this into consideration we can summarise that the hybrid approach of using ”Content Based Filtering and User Based Collaborative Filtering” produced the best quantitative results, being superior in the most evaluation metrics overall.

7.2 Qualitative Results

7.2.1 List of 5 Artists similar to a provided Artist

The similarity matrix allows Last FM users to search for musical artists based on those they have specified. To give us an example of how this might work, we can randomly select an artist ID and output a list of 5 similar artists to be recommended for cluster-based collaborative filtering.

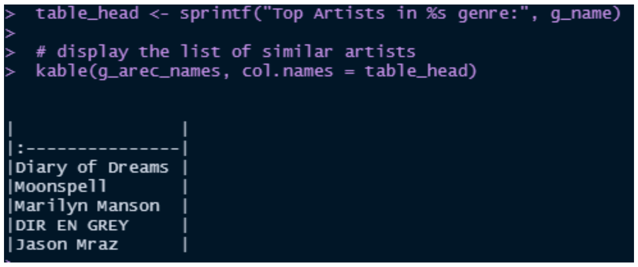


```
|Artists Similar to Hammock |
|:--------------------------|
|The Album Leaf             |
|M83                        |
|The American Dollar        |
|Stars of the Lid           |
|Jonsi & Alex               |
>
```

Figure 7.3: List of top 5 similar artists given based of a randomly selected artist for cluster based collaborative filtering

7.2.2 List of top 5 similar Artists to a provided Genre

To display a list of the top 5 similar list of musical artists based on the genre, we used the non-binary version of our artist-genre matrix. This matrix consisted of the counts of how often a given artist had been labelled as belonging to a specified genre. This metric can be interpreted as how strongly a Last FM user felt that an artist belonged to a genre. This means that we are able to use the tag counts to rank artists within the genres. The more times these artists are tagged within the genre label, the higher they will rank within the genre.



```
>  table_head <- sprintf("Top Artists in %s genre:", g_name)
>
>  # display the list of similar artists
>  kable(g_arec_names, col.names = table_head)


|               |
|:--------------|
|Diary of Dreams |
|Moonspell      |
|Marilyn Manson  |
|DIR EN GREY    |
|Jason Mraz     |
```

Figure 7.4: List of top 5 similar artists given based of a randomly selected genre, in this case it's the "Mossy genre", there were issues printing "g_name" which previously showed "Mossy genre"

7.3 Advantages and Disadvantages

7.3.1 Advantages

User Based Collaborative Filtering

- As seen in our results, using a user based system is it is most appropriate collaborative filtering. As with less users than items to be recommended, we are provided with the best recommendations.

Content Based Filtering

- For our data, comparisons between items are possible. The user gets recommendations based on similar types of music in the user's past playlist.
- Our content based recommendation system overcame the challenge of collaborative filtering systems. It can still provide us with recommendations, even if users do not give us ratings for new items. Furthermore, these systems have the capacity to adjust it's recommendations quickly if the user's preferences change. Content-based filtering techniques can inform us with explanations of how recommendations are generated to target users.

Demographic Based Filtering

- As recommendations are also provided on a user-user basis, our model was accurate in assuming that a particular demographic would like a specific artist. We can see that this is evident with low RMSE and MAE values.

7.3.2 Disadvantages:

User/Item Based Collaborative Filtering

- Cold start problem[23], referring to the lack of information available to a recommendation system. In our case, where a new user/item without any ratings is added, it is quite difficult to handle using our systems. This was a factor leading to item based collaborative filtering producing the worst results from the experiments.
- Due to us using a user's individual ratings to make a prediction; if we want to predict ”User A's” rating for Rock Music, the item-based approach will look at ”User A's” ratings, to provide related features to Rock Music. On the other hand, with the user-based approach, we look at similar users who might have intersecting but conflicting interests to ”User A”. This may result in a decrease in accuracy.

Demographic and Content Based Filtering

- Overspecialization [24] is one of the main concerns generally faced here. This refers to the lack of ability for our system to provide diverse recommendations, in our case a lack of diversification of artists.

Demographic Based Filtering

- In this specific data set, demographic-based filtering did not perform as well as it could have. This is because the information associated with the demographic groups was limited.
- This technique would be more effective when there is a high correlation between specific demographic attributes and user preferences. Our data set is not sufficient to see any correlation like this.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

Throughout this project we have looked at a number of different recommendation systems individually, as well as combining these systems to make hybrid recommendation systems. Quantitatively, we can conclude that combining our systems like with 'Content-Based Filtering and User-Based Collaborative Filtering' produced the best results. This combined approach allows us to mitigate the drawbacks of these systems had they been used individually; for example, overspecialization[24] with content based filtering and the cold start problem [23] for collaborative based filtering. As a result, it is safe to conclude that the more methods used to gather information regarding users, the more accurate a user profile will be, in turn providing us with better recommendations.

8.2 Future Work

There are numerous ways in which this project can be expanded. Firstly, there are other recommendation systems that exist apart from those which have been looked at in this project, for example, "Utility Based Filtering". Furthermore, in this study only two hybrid recommendation systems were examined, using different combinations of recommendation systems may have produced some interesting results.

With more time available, we could have tested recommendation systems across data sets with larger or smaller amounts of data to see if the results we gathered were universally similar, with hybrid systems performing the best.

As for our evaluation metrics, another metric for classification accuracy which we attempted to implement was NDCG, normalised discounted cumulative gain. This metric can still be seen in the source code.

Coding our systems in alternative languages could have been considered, because ”r” takes a while to compile code. This made debugging time consuming, which also impacted the addition of new features.

Appendix A

User Guide

A.1 Instructions

A.2 Walk-through:

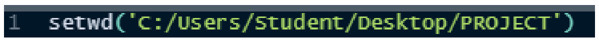
1. Install ”R Studio” or the ”R” extension for ”Visual Studio Code”
2. Check that all the specified packages have been installed:
   (a) mlr
   (b) recommenderlab
   (c) tidyverse
   (d) tidyr
   (e) dplyr
   (f) readr
   (g) knitr
   (h) AUC
   (i) doParallel
3. Make sure the relevant code and all data files are present in the folder:
   (a) Artists.dat
   (b) user_artists.dat
   (c) user_taggedartists.dat

(d) Recommendation.r

If any of the files a,b or c are missing, please download and extract the files from the data set linked below

4. Open the Recommendation.r file

5. On line 1, set the working directory to the location of the project folder on the machine:



```
setwd('C:/Users/Student/Desktop/PROJECT')
```

6. Click on the code and press Cntrl+A to select all lines

7. Press Cntrl+Enter to run the code

8. As the code is run, the output of the recommendation systems is printed, with a few graphs and histograms

9. This process takes long depending on the machine's RAM

Appendix B

Source Code

B.1 Declaration

B.2 Packages Used

B.3 Data Set

B.4 Code Inspiration

B.5 Source Code

Appendix C

Appendix C

C.1 Graphs and Charts

References

[1] C. Tardi, “Moore’s law explained,” Feb 2022 , Access-Date:29/2/2022. https://www.investopedia.com/terms/m/mooreslaw.asp#:~:text=Moore’s.

[2] B. Rocca, “Introduction to recommender systems,” Jun 2019, Access-Date:23/2/2022. https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada.

[3] J. Selig, “What is machine learning? a definition.,” Apr 2022 , Access-Date:12/4/2022. https://www.expert.ai/blog/machine-learning-definition/.

[4] R. Burke, A. Felfernig, and M. H. Göker, “Recommender systems: An overview,” *Ai Magazine*, vol. 32, no. 3, pp. 13–18, 2011.

[5] D. W. Oard, J. Kim, *et al.*, “Implicit feedback for recommender systems,” in *Proceedings of the AAAI workshop on recommender systems*, vol. 83, pp. 81–83, AAAI, 1998.

[6] J. Buder and C. Schwind, “Learning with personalized recommender systems: A psychological view,” *Computers in Human Behavior*, vol. 28, no. 1, pp. 207–216, 2012.

[7] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender systems: an introduction.* Cambridge University Press, 2010.

[8] M. Y. H. Al-Shamri, “User profiling approaches for demographic recommender systems,” *Knowledge-Based Systems*, vol. 100, pp. 175–187, 2016.

[9] D. DeepAI, “Cosine similarity,” May 2019, Access-Date:12/3/2022. https://deepai.org/machine-learning-glossary-and-terms/cosine-similarity.

[10] M. Lüthe, “Calculate similarity the most relevant metrics in a nutshell,” May 2021, Access-Date:9/2/2022. https://towardsdatascience.com/

calculate-similarity-the-most-relevant-metrics-in-a-nutshell-9a43564f533e, journal=Medium, publisher=Towards Data Science.

[11] I. IBM, "Predictive accuracy a," Mar 2021, Access-Date:12/2/2022. https://www.ibm.com/docs/en/db2/10.5?topic=concepts-predictive-accuracy.

[12] C. A. AI, "Root mean square error (rmse)," Sep 2021, Access-Date:11/1/2022. https://c3.ai/glossary/data-science/root-mean-square-error-rmse/.

[13] T. Chai and R. R. Draxler, "Root mean square error (rmse) or mean absolute error (mae)?– arguments against avoiding rmse in the literature," *Geoscientific model development*, vol. 7, no. 3, pp. 1247–1250, 2014.

[14] C. Sammut and G. I. Webb, eds., *Mean Absolute Error*, pp. 652–652. Springer US, 2010.

[15] N. Bu, T. Tsuji, and O. Fukuda, "Chapter three - emg-controlled human-robot interfaces: A hybrid motion and task modeling approach," in *Human Modelling for Bio-Inspired Robotics* (J. Ueda and Y. Kurita, eds.), pp. 75–109, 2017.

[16] G. Developers, "Classification: Precision and recall nbsp;—nbsp; machine learning crash course nbsp;—nbsp; google developers," Oct 2020, Access-Date:18/2/2022. https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall.

[17] K. P. Shung, "Accuracy, precision, recall or f1?," Apr 2020, Access-Date:11/1/2022. https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9.

[18] J. Brownlee, "How to calculate precision, recall, and f-measure for imbalanced classification," Aug 2020, Access-Date:11/1/2022. https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification.

[19] M. G. Campana and F. Delmastro, "Recommender systems for online and mobile social networks: A survey," *Online Social Networks and Media*, vol. 3-4, pp. 75–97, 2017.

[20] V. Dey, "Understanding the auc-roc curve in machine learning classification," Oct 2021, Access-Date:9/2/2022. https://analyticsindiamag.com/understanding-the-auc-roc-curve-in-machine-learning-classification/.

[21] K. Kerneler, "Starter: Last.fm music artist scrobbles 45e52454-b," Jun 2020, Access-Date:1/03/2022. https://www.kaggle.com/code/kerneler/

`starter-last-fm-music-artist-scrobbles-45e52454-b/data?select=lastfm_user_scrobbles.csv`.

[22] S. Milano, M. Taddeo, and L. Floridi, “Recommender systems and their ethical challenges,” *Ai & Society*, vol. 35, no. 4, pp. 957–967, 2020.

[23] B. Lika, K. Kolomvatsos, and S. Hadjiefthymiades, “Facing the cold start problem in recommender systems,” *Expert Systems with Applications*, vol. 41, no. 4, pp. 2065–2073, 2014.

[24] T. Kvifte, *Video Recommendations Based on Visual Features Extracted with Deep Learning.* PhD thesis, 06 2021.