Task A - Recommendation Systems

Statistical Analysis of Big Data

Amanda Magzal 207608647

Project Overview

In this project, I develop a music recommendation system using two engines: (1) Item-based Collaborative Filtering, and (2) Association Rules.

This report includes four main sections. In section 1, I present, summarize and explore the data. In section 2, I present the `R` source code that was used to build the recommendation systems. In section 3, I evaluate and compare the performance (precision) of the two engines, and in section 4, I present the `R` source code that was used to create the Shiny app which can be found here.

1. Data Exploration

The data set contains information about users and the artists they have listened to on Last.FM in Germany. Each row represents a user and each column represents an artist.

Table 1: Data set

user	a.perfect.circle	abba	ac.dc	adam.green	aerosmith	afi	air
1	0	0	0	0	0	0	0
33	0	0	0	1	0	0	0
42	0	0	0	0	0	0	0
51	0	0	0	0	0	0	0
62	0	0	0	0	0	0	0
75	0	0	0	0	0	0	0

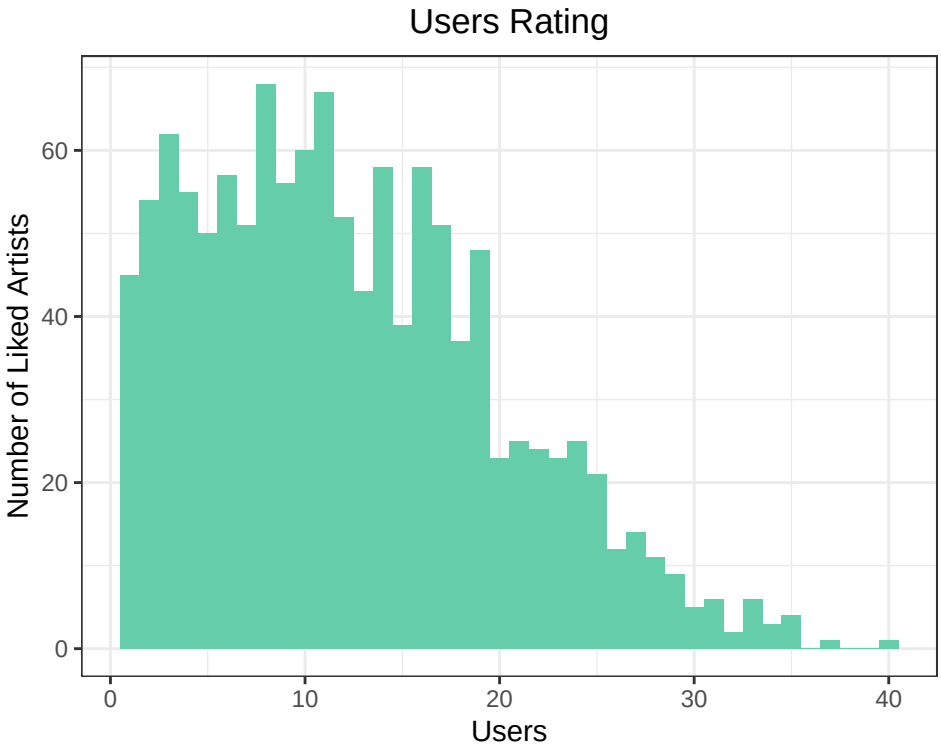
The data set includes 1257 users and 285 artists. There are 31 users who did not listen to any artist and therefore will be removed.

In addition, we do not need the users' information. Thus, the `user` column will be dropped.

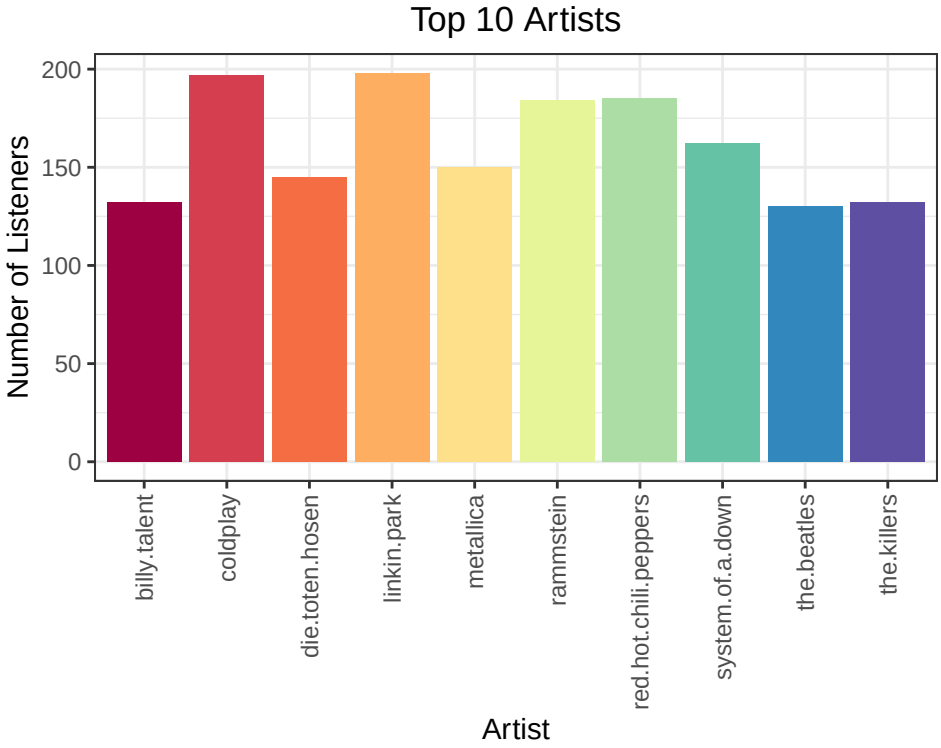
The following table and plot present the distribution of the number of artists that each user listened to.

Table 2: Summary Statistics for Number of Artists

Number of Artists
Min. : 1.00
1st Qu.: 6.00
Median :11.00
Mean :12.39
3rd Qu.:17.00
Max. :40.00



The top artists (highest number of listeners) are presented in the following plot.



2. Recommendation Engines

The data is divided into train and test data. The first 859 users are part of the train data, which is used to generate the recommendations. The remaining 357 users are part of the test data, which is later used to evaluate the performance of the engines.

```
train <- dat[1:859, ]
test <- dat[860:nrow(dat), ]
```

2.1 Item-based Collaborative Filtering

The item-based Collaborative Filtering algorithm recommends items based on how similar they are to other items. Similarity, in this case, is calculated using the *Jaccard index*.

Let X, Y be the sets of the items with a 1 in user profiles u_a and u_b, respectively. Hence,

$$\text{sim}_{Jaccard}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

In R, the *Jaccard index* is calculated using the following function:

```
Jaccard <- function(X, Y){
  intersect <- sum(X == Y & X == 1)
  union <- sum(X) + sum(Y) - intersect
  return(intersect/union)
}
```

Generating Recommendations

1. Using the `Jaccard` function, calculate the similarity for each pair of artists and store the result.

```
dat.similarity  <- matrix(NA, nrow = ncol(dat), ncol = ncol(dat),
                          dimnames = list(colnames(dat), colnames(dat)))

# Loop through the columns
for(i in 1:ncol(dat)){
  # Loop through the columns for each column
  for(j in 1:ncol(dat)){
    # Fill in placeholder with Jaccard similarities
    dat.similarity[i, j] <- Jaccard(as.matrix(dat[i]), as.matrix(dat[j]))
  }
}

dat.similarity <- as.data.frame(dat.similarity)
```

2. Find the top 5 neighbors for each artist in the following manner:

- Loop through all the artists.
- Sort the similarity matrix for the artist so that the most similar are first.
- Take the top 6 (first will always be the same artist) and put them into the placeholder.

```
dat.neighbours <- matrix(NA, nrow = ncol(dat.similarity), ncol = 6,
                         dimnames = list(colnames(dat.similarity)))

for(i in 1:ncol(dat)){
  dat.neighbours[i,] <- (t(head(n = 6,
                                rownames(dat.similarity[order(dat.similarity[,i],
                                                              decreasing = TRUE),][i]))))
}

dat.neighbours <- as.data.frame(dat.neighbours[, 2:ncol(dat.neighbours)])

colnames(dat.neighbours) <- c('Option 1', 'Option 2', 'Option 3', 'Option 4', 'Option 5')
```

The recommendations data frame is shown in table 3.

Table 3: Collaborative Filtering Recommendations

	Option 1	Option 2	Option 3	Option 4	Option 5
a.perfect.circle	tool	dredg	deftones	porcupine.tree	nine.inch.nails
abba	madonna	robbie.williams	elvis.presley	michael.jackson	queen
ac.dc	iron.maiden	metallica	red.hot.chili.peppers	the.offspring	die.toten.hosen
adam.green	the.libertines	the.strokes	babyshambles	franz.ferdinand	radiohead
aerosmith	led.zeppelin	u2	lenny.kravitz	ac.dc	the.rolling.stones
afi	funeral.for.a.friend	rise.against	fall.out.boy	anti.flag	sum.41

2.2 Association Rules

The Association Rules algorithm recommends items based on the *confidence* of the rules generated from the items.

Let X and Y be itemsets (in this case - artists). Hence, the confidence of the rule $X \to Y$ is

$$c(X \to Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

Where $\sigma(I)$ is the support count of itemset I.

In R, the support count of a rule is calculated using the following function:

```
Support <- function(X, Y){
  return(sum(X == Y & X == 1))
}
```

Generating Recommendations

1. Calculate the support count for each individual artist.

```
sup.count <- as.data.frame(colSums(dat))
colnames(sup.count) <- 'count'
```

2. Calculate the confidence for each possible rule and store the result.

```
conf <- data.frame(expand.grid(artist1 = colnames(dat), artist2 = colnames(dat),
                               confidence = NA))

for(i in 1:nrow(conf)){
  sup <- Support(as.matrix(dat[conf$artist1[i]]), as.matrix(dat[conf$artist2[i]]))
  conf$confidence[i] <- sup / sup.count[conf$artist1[i],]
}
```

3. Find the top 5 recommendations for each artist in the following manner:

- Select the rules with the highest confidence.
- In case of ties (rules with equal confidence), select the first 5 artists.

```
# Sort the dataframe
dat.top5 <- conf[order(conf$artist1, -conf$confidence),]

  # Filter rows with the same artist
dat.top5 <- dat.top5 %>% filter(artist1 != artist2) %>%
  # Select top 5 recommendations for each artist
  group_by(artist1) %>% top_n(5, confidence) %>% slice(1:5) %>%
  # Arrange the dataframe
  select(-confidence) %>% mutate(V = c('V1', 'V2', 'V3', 'V4', 'V5')) %>%
  pivot_wider(names_from = V, values_from = artist2)

row.names(dat.top5) <- dat.top5$artist1
dat.top5 <- dat.top5 %>% ungroup() %>% select(-artist1)

colnames(dat.top5) <- c('Option 1', 'Option 2', 'Option 3', 'Option 4', 'Option 5')
```

The recommendations data frame is shown in table 4.

Table 4: Association Rules Recommendations

Option 1	Option 2	Option 3	Option 4	Option 5
tool	system.of.a.down	dredg	incubus	metallica
the.beatles	madonna	coldplay	queen	michael.jackson
red.hot.chili.peppers	metallica	rammstein	die.toten.hosen	the.beatles
coldplay	radiohead	foo.fighters	the.beatles	the.kooks
metallica	red.hot.chili.peppers	ac.dc	jack.johnson	linkin.park
rise.against	billy.talent	system.of.a.down	fall.out.boy	linkin.park

3. Algorithms Evaluation and Comparison

In this section I perform a simulation to evaluate and compare the performance of the recommendation engines, using the *precision* measure which is calculated as follows:

$$Precision = \frac{correctly\ recommended\ items}{total\ recommended\ items}$$

In `R`, the precision measure is calculated using the following function:

```
Precision <- function(rec, actual){
  N <- length(rec)
  d <- 0
  for(i in 1:5){
    if(actual[, rec[1, i]] == 1)
      d <- d + 1
  }
  return(d/N)
}
```

For the two developed recommendation engines, I perform the following steps:

- For all users in the test data:
  1. Sample one artist from those rated by the user.
  2. Generate the top 5 recommendations for the user given the sampled artist.
  3. Calculate the precision measure for this user.
- Calculate *mean precision* over all users in the test data.

These steps are implemented using the `Iteration` function below.

```
Iteration <- function(df){

  p <- numeric()

  for(i in 1:nrow(test)){

    # Select a user
    user <- test[i, ]

    # Sample 1 artist from those rated by the user
    x <- user[which(user == 1)]
    artist <- sample(x, 1)

    # Generate the top 5 recommendations given an artist
    rec <- df[df$X == colnames(artist),]

    # Calculate precision for the recommendation
    p[i] <- Precision(rec, user)

  }
  return(mean(p))
}
```

The procedure described above is repeated 20 times as follows:

```
set.seed(123)

CF.res <- AR.res <- numeric()

# Perform the simulation 20 times
for(k in 1:20){
  CF.res[k] <- Iteration(CF.df)
  AR.res[k] <- Iteration(AR.df)
}

# Calculate the mean precision
CF.precision <- mean(CF.res)
AR.precision <- mean(AR.res)
```

The results are shown in table 5.

Table 5: Precision Measure

	Collaborative Filtering	Association Rules
Precision	0.194	0.1921

We can see that both engines had similar results.

4. R Shiny App

The app can be found here.

Preparations

```
# Get data frame
dat <- read.csv("lastfm-matrix-germany.csv")
dat <- dat[, 2:ncol(dat)] %>% filter_all(any_vars(. != 0))

# Get artists names - used to fill the combo box
artists <- colnames(dat)

# Get CF recommendations
CF.df <- read.csv("CF.csv")

# Get AR recommendations
AR.df <- read.csv("AR.csv")
```

UI

Alongside some text, the UI includes a combo box to select an artist and two tables of recommendations, one for each engine used.

```
ui <- fluidPage(

  theme = shinytheme('darkly'),

  tags$h1('Music Recommendation System'),
  tags$em('Amanda Magzal'),
  tags$hr(),
  tags$p('This system recommends artists you might like based on the artist you select,
         using two different recommendation engines.'),

  tags$hr(),

  fluidRow(
      column(4, offset = 0.5,
             # combo box to select an artist
             selectInput(inputId = 'artist',
                         label = 'Select an artist',
                         choices = artists),
      )
  ),

  tags$hr(),

  # CF algorithm
  fluidRow(
      column(6, offset = 0.5,
             tags$h3('Item-Based Collaborative Filtering'),
             tags$br(),
             # output table of recommended artists
             tags$strong('Top 5 recommendations'),
             tableOutput('CF.rec.table'))
  ),

  tags$hr(),

  # AR algorithm
  fluidRow(
      column(6, offset = 0.5,
             tags$h3('Association Rules'),
             tags$br(),
             # output table of recommended artists
             tags$strong('Top 5 recommendations'),
             tableOutput('AR.rec.table'))
  )
)
```

Server

The server function includes the creation of the two tables by selecting the relevant recommendations for the artist selected in the combo box, from the recommendations dataframes.

```
server <- function(input, output) {

  output$CF.rec.table <- renderTable({
      CF.df %>% filter(X == input$artist) %>% select(Option.1:Option.5)
  }, align = 'c', colnames = FALSE)

  output$AR.rec.table <- renderTable({
      AR.df %>% filter(X == input$artist) %>% select(Option.1:Option.5)
  }, align = 'c', colnames = FALSE)

}
```