Random Chosen Artists clustering and patterns

Introduction

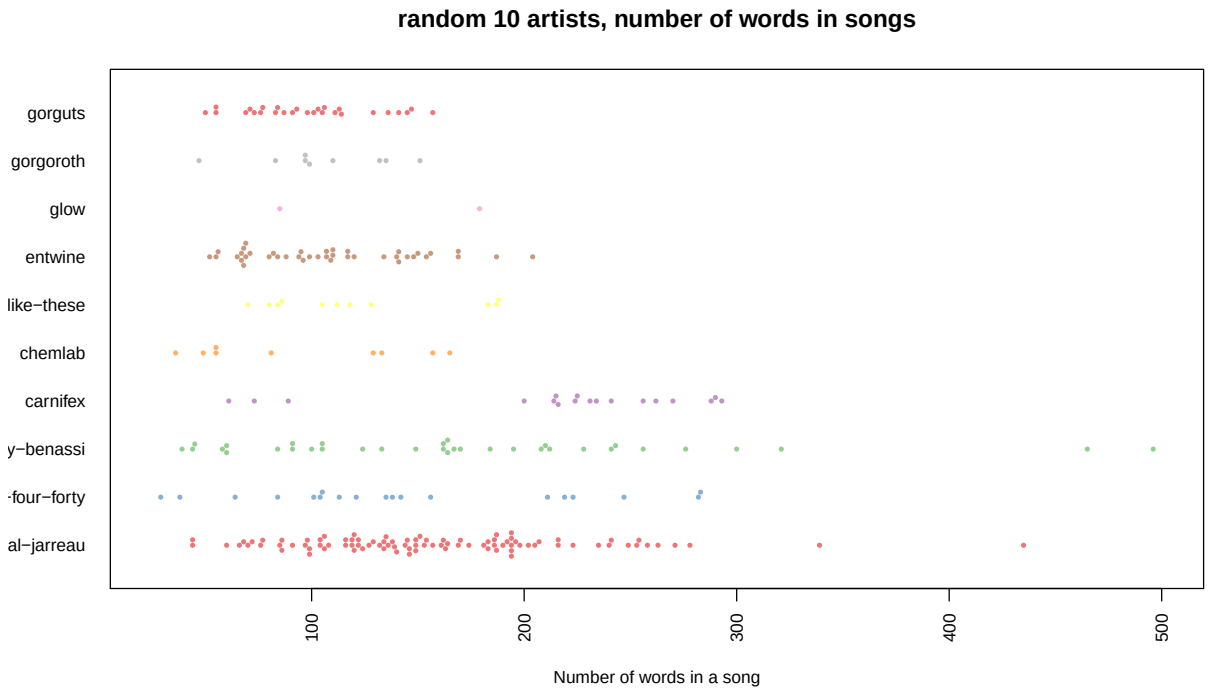
Artists clustering and patterns are interesting topics when we are going to do a recommendation system for users, the project is mostly focus on the clustering part. Due to the presentation limit, our topic is limited to random chosen artists instead of all artists.

load data

```
load('D:/Github/fall2019-proj1--hao871563506/output/processed_lyrics.RData')
# load packages
library("rvest")
library("tibble")
Sys.setenv(JAVA_HOME='C:\\Program Files\\Java\\jre1.8.0_221')
library("qdap")
library("sentimentr")
library("gplots")
library("dplyr")
library("tm")
library("syuzhet")
library("factoextra")
library("beeswarm")
library("scales")
library("RColorBrewer")
library("RANN")
library("tm")
library("topicmodels")
```

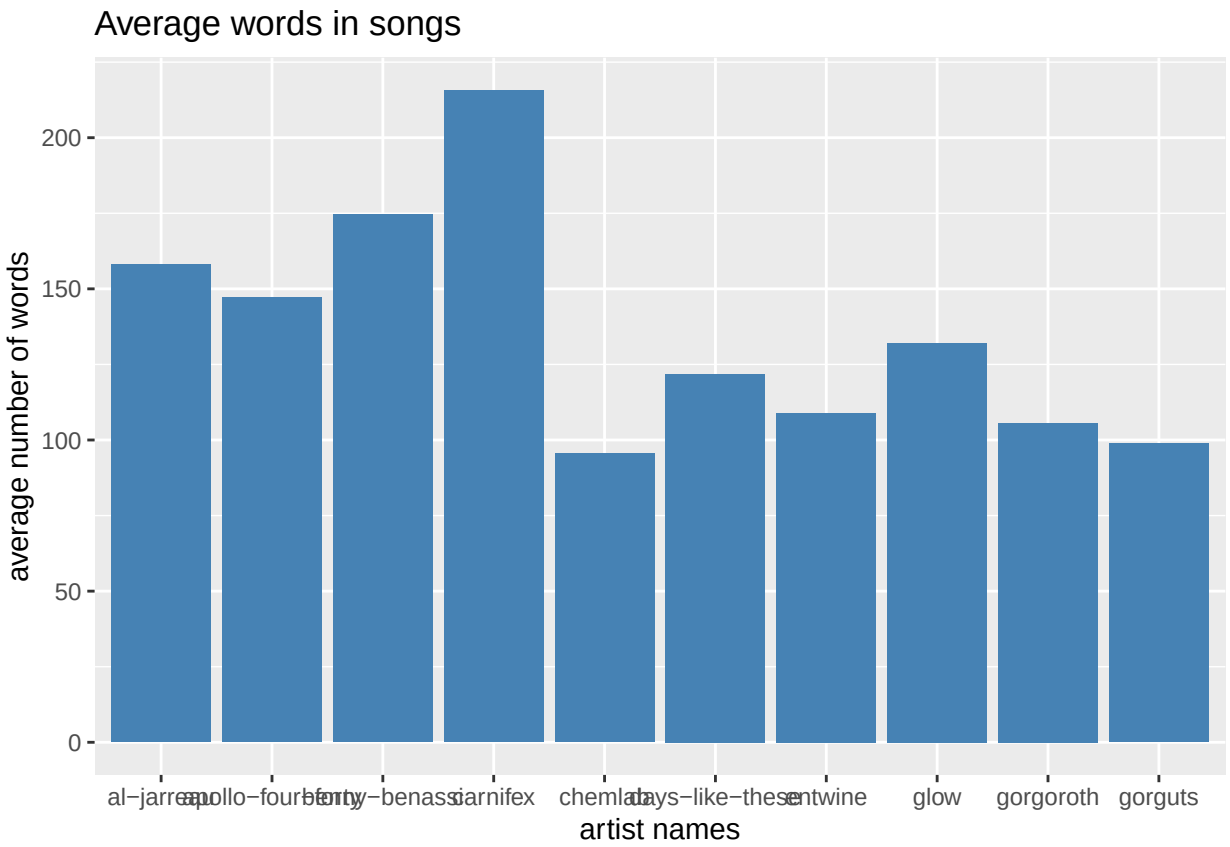
length of songs

```
artist<-unique(dt_lyrics %>% pull(artist))

dt_lyrics$wordscount <- sapply(strsplit(as.character(dt_lyrics$lyrics), " "), length)

set.seed(66)

artistramdomsampled <- artist[sample(1:length(artist), 10)]

dt_lyrics$x <- c(1:nrow(dt_lyrics))

dt_lyrics.random=filter(dt_lyrics, artist%in%artistramdomsampled)

beeswarm(wordscount~artist,
         data=dt_lyrics.random,
         horizontal = TRUE,
         pch=16, col=alpha(brewer.pal(9, "Set1"), 0.6),
         cex=0.55, cex.axis=0.8, cex.lab=0.8,
         las=2, xlab="Number of words in a song", ylab="",
         main="random 10 artists, number of words in songs")
```



I choose 10 artist at random. And for each of them, I count the number of words in each of their songs. And then plot this beeswarm plot. From the plot, x-axis is the word count of the songs, y-axis is the artist names. We can see that most songs has a count of words from around 80-200. While some songs are longer than normal, for example, we can see benassi has a song which has 500 word counts.

```
wordstotal <- aggregate(dt_lyrics.random$wordscount, by=list(artist=dt_lyrics.random$artist), FUN=sum)
songscount<- aggregate(x ~ artist, data = dt_lyrics.random, FUN = length)
wordsdf <- data.frame(artist = wordstotal$artist, songscount = songscount$x,
                      averagewords = wordstotal$x / songscount$x)
library(ggplot2)
ggplot(data=wordsdf, aes(x=artist, y=averagewords))+geom_bar(stat="identity",fill="steelblue")+
  ggtitle("Average words in songs") + xlab("artist names") + ylab("average number of words")
```



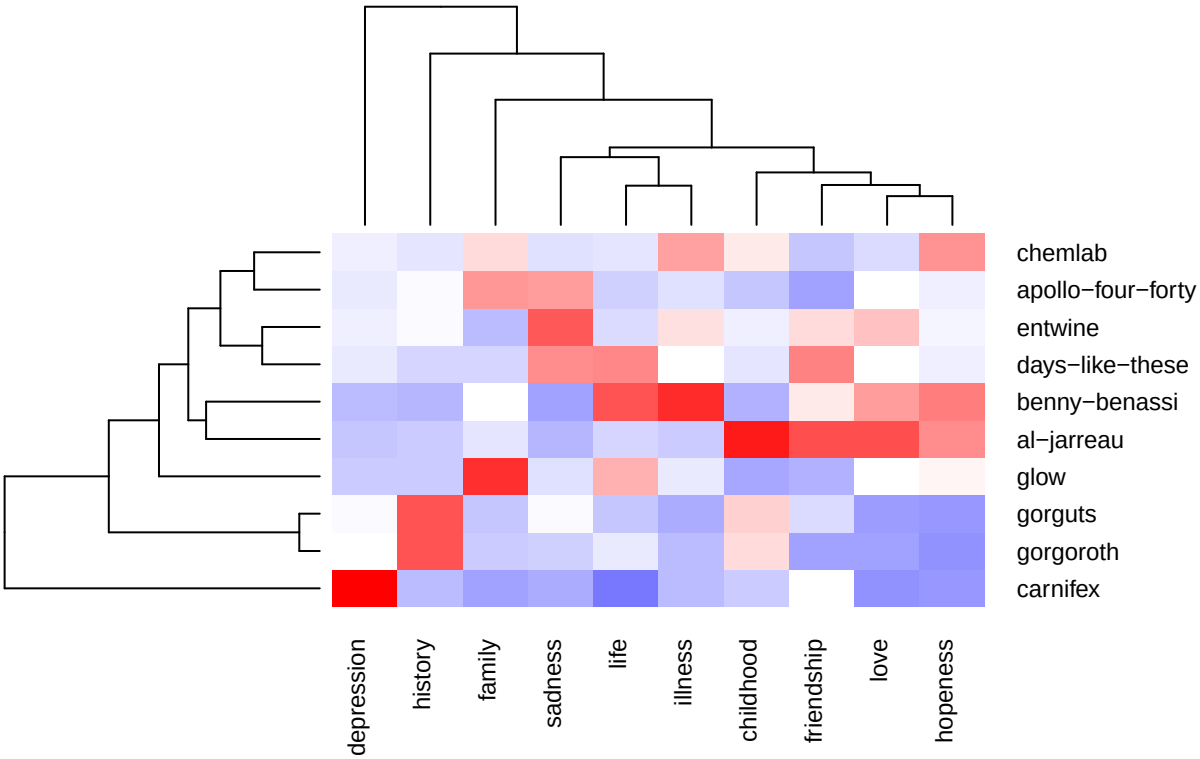
For the 10 random artists chosen, I calculate the average number of words in their songs. We can see Carnifex has an average of more than 200 words, while chemlab has an average of less than 100 words. Is interesting that they are so different, it is probably due to the personal style of the artist. And most of the artist has an average words around 100 to 150.

Topic modeling

```
docs <- Corpus(VectorSource(dt_lyrics.random$stemmedwords))
dtm <- DocumentTermMatrix(docs)
#convert rownames to filenames#convert rownames to filenames
rownames(dtm) <- paste(dt_lyrics.random$song, dt_lyrics.random$year,
                       dt_lyrics.random$artist, dt_lyrics.random$genre, sep="_")
rowTotals <- apply(dtm , 1, sum) #Find the sum of words in each Document
dtm  <- dtm[rowTotals> 0, ]
dt_lyrics.random=dt_lyrics.random[rowTotals>0, ]
burnin <- 4000
iter <- 2000
thin <- 500
seed <-list(2003,5,63,100001,765)
nstart <- 5
best <- TRUE
#Number of topics
k <- 10
#Run LDA using Gibbs sampling
ldaOut <-LDA(dtm, k, method="Gibbs", control=list(nstart=nstart,
                                                  seed = seed, best=best,
```

```
                                                    burnin = burnin, iter = iter,
                                                    thin=thin))
#write out results
#docs to topics
ldaOut.topics <- as.matrix(topics(ldaOut))
ldaOut.terms <- as.matrix(terms(ldaOut,20))

topicProbabilities <- as.data.frame(ldaOut@gamma)

terms.beta=ldaOut@beta
terms.beta=scale(terms.beta)
topics.terms=NULL
for(i in 1:k){
  topics.terms=rbind(topics.terms, ldaOut@terms[order(terms.beta[i,], decreasing = TRUE)[1:7]])
}
#ldaOut.terms
topics.hash=c("friendship","sadness","childhood","depression","history","life","illness","family","hope
dt_lyrics.random$ldatopic=as.vector(ldaOut.topics)
dt_lyrics.random$ldahash=topics.hash[ldaOut.topics]
colnames(topicProbabilities)=topics.hash
dt_lyrics.random.df=cbind(dt_lyrics.random, topicProbabilities)
```

```
par(mar=c(1,1,1,1))
topic.summary=tbl_df(dt_lyrics.random.df)%>%
              #filter(type%in%c("nomin", "inaug"), File%in%sel.comparison)%>%
              select(artist, friendship:love)%>%
              group_by(artist)%>%
              summarise_each(funs(mean))

topic.summary=as.data.frame(topic.summary)
rownames(topic.summary)=topic.summary[,1]
topic.plot=c(1, 2,3,4,5,6,7,8,9,10)


heatmap.2(as.matrix(topic.summary[,topic.plot+1]),
          scale = "column", key=F,
          col = bluered(100),
          cexRow = 0.9, cexCol = 0.9, margins = c(8, 8),
          trace = "none", density.info = "none")
```



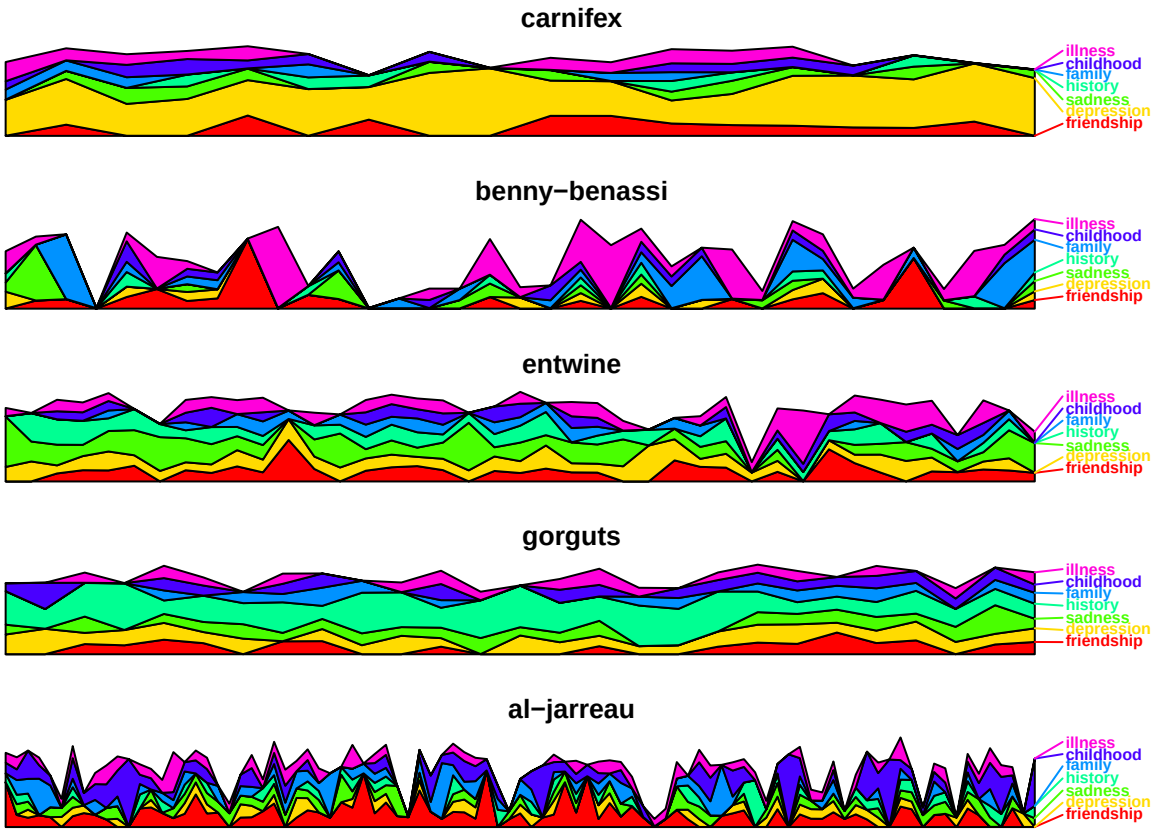
Above is the topic modeling. I manually define that there will be 10 topics as fellow: “friendship” ,“sadness”, “childhood”, “depression”, “history”, “life”, “illness”, “family”, “hopeness”, “love”, the topics are determined through the contents of each topic.
And for the heatmap above, We can see each artists preference in topics easily by the color. Red color means strong preference, while blue color means low preference of topics. We can see that artist ‘glow’ for example, has a strong preference in the topic of ‘family’ but not that much for ‘illness’, artist ‘gorguts’ and ‘gorgoroth’ both like the topic of ‘history’.
Besides these patterns, we can also see how clustering of the artist and clustering of the topics are going on, we can see certain artist has similiar preference, for example ‘gorguts’ and ‘gorgoroth’, they appears to be pretty similiar, while some are pretty different from each others, for example, ‘carnifex’ and ‘benny-benassi’, one likes the topic of ‘depression’, and the other likes ‘hopeness’, really opposite topics.
We can also see clustering of topics, for example,‘love’ and ‘hopeness’ are close to each othere in clustering while ‘depression’ and ‘hopeness’ are pretty far away from clustering. These clustering makes a lot of sense in the real-world actually, and is pretty amazing that this topic clustering can get this correctly.

Topics for artisits

```
plot.stacked <- function(
  x, y,
  order.method = "as.is",
  ylab="", xlab="",
  border = NULL, lwd=1,
  col=rainbow(length(y[1,])),
  ylim=NULL, stack.lab=colnames(y),
  ...
){
```

```
  if(sum(y < 0) > 0) error("y cannot contain negative numbers")

  if(is.null(border)) border <- par("fg")
  border <- as.vector(matrix(border, nrow=ncol(y), ncol=1))
  col <- as.vector(matrix(col, nrow=ncol(y), ncol=1))
  lwd <- as.vector(matrix(lwd, nrow=ncol(y), ncol=1))

  if(order.method == "max") {
    ord <- order(colSums(y), decreasing = T)
    y <- y[, ord]
    col <- col[ord]
    border <- border[ord]
    stack.lab <- stack.lab[ord]
  }

  if(order.method == "first") {
    ord <- order(apply(y, 2, function(x) min(which(x>0))))
    y <- y[, ord]
    col <- col[ord]
    border <- border[ord]
    stack.lab <- stack.lab[ord]
  }

  top.old <- x*0
  lab.y=0
  polys <- vector(mode="list", ncol(y))
  for(i in seq(polys)){
    top.new <- top.old + y[,i]
    polys[[i]] <- list(x=c(x, rev(x)), y=c(top.old, rev(top.new)))
    top.old <- top.new
    lab.y=c(lab.y, top.new[length(x)])
  }

  if(is.null(ylim)) ylim <- range(sapply(polys, function(x) range(x$y, na.rm=TRUE)), na.rm=TRUE)
  plot(x,y[,1], ylab=ylab, xlab=xlab, xlim=c(min(x), max(x)+diff(range(x))*0.1),
       ylim=ylim, t="n", ...)
  for(i in seq(polys)){
    polygon(polys[[i]], border=border[i], col=col[i], lwd=lwd[i])
  }

  lab.y.0=lab.y[-1]
  lab.y.1=(seq(0, max(rowSums(y))*0.95, len=ncol(y)+1))[-1]
  #print(lab.y.1)
  text(rep(max(x)+diff(range(x))*0.03, ncol(y)),
       lab.y.1, cex=0.8, font=2,
       stack.lab, col=col, adj=0)
  segments(rep(max(x), ncol(y)),
           lab.y.0,
           rep(max(x)+diff(range(x))*0.027, ncol(y)),
           lab.y.1,
           col=col)

}
```

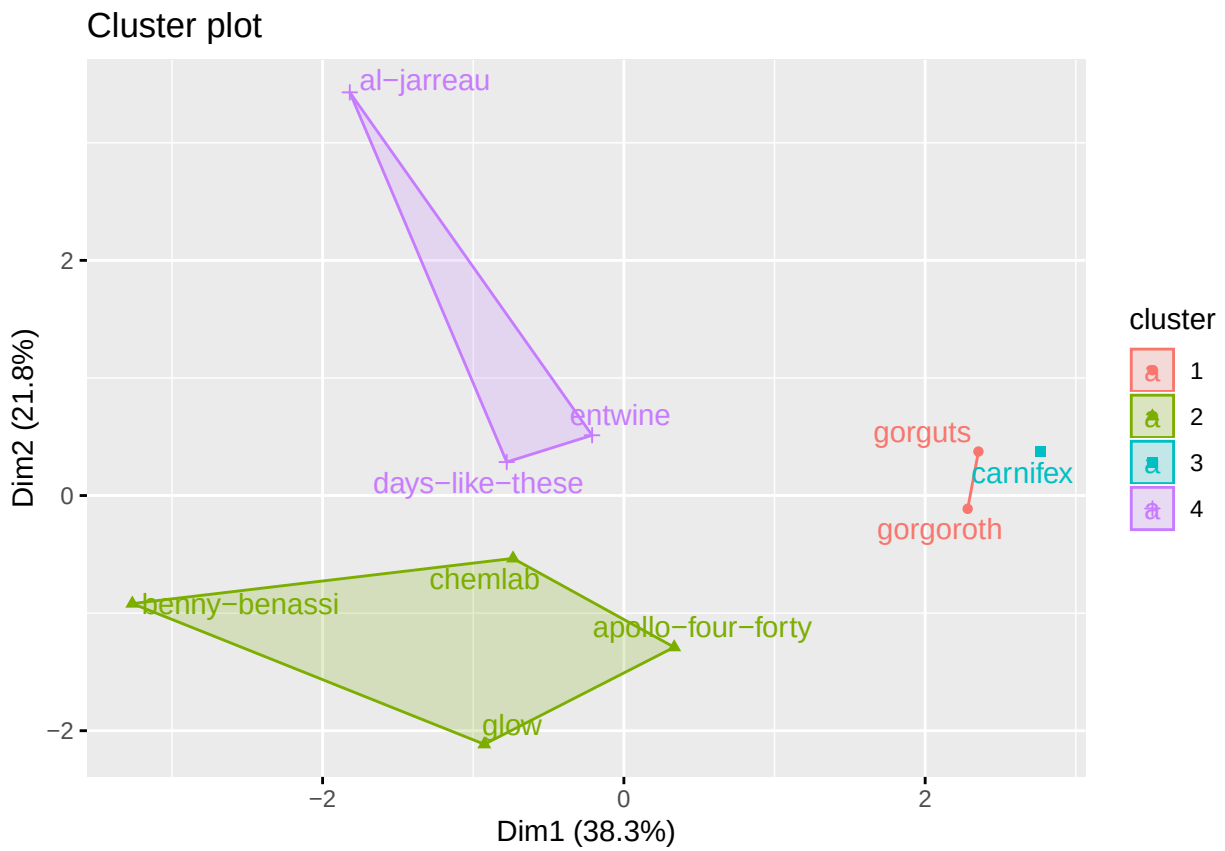
```
par(mfrow=c(5, 1), mar=c(1,1,2,0), bty="n", xaxt="n", yaxt="n")
topic.plot=c(1, 4, 2, 5, 8, 3, 7)
#print(topics.hash[topic.plot])
speech.df=tbl_df(dt_lyrics.random.df)%>%
  filter(artist=="carnifex")%>%
  select(id, friendship:love)
speech.df=as.matrix(speech.df)
speech.df[,-1]=replace(speech.df[,-1], speech.df[,-1]<1/15, 0.001)
#speech.df[,-1]=f.smooth.topic(x=speech.df[,1], y=speech.df[,-1])
plot.stacked(speech.df[,1], speech.df[,topic.plot+1],
             xlab="Sentences", ylab="Topic share", main="carnifex")


speech.df=tbl_df(dt_lyrics.random.df)%>%
  filter(artist=="benny-benassi")%>%
  select(id, friendship:love)
speech.df=as.matrix(speech.df)
speech.df[,-1]=replace(speech.df[,-1], speech.df[,-1]<1/15, 0.001)
#speech.df[,-1]=f.smooth.topic(x=speech.df[,1], y=speech.df[,-1])
plot.stacked(speech.df[,1], speech.df[,topic.plot+1],
             xlab="Sentences", ylab="Topic share", main="benny-benassi")


speech.df=tbl_df(dt_lyrics.random.df)%>%
  filter(artist=="entwine")%>%
  select(id, friendship:love)
speech.df=as.matrix(speech.df)
speech.df[,-1]=replace(speech.df[,-1], speech.df[,-1]<1/15, 0.001)
#speech.df[,-1]=f.smooth.topic(x=speech.df[,1], y=speech.df[,-1])
plot.stacked(speech.df[,1], speech.df[,topic.plot+1],
             xlab="Sentences", ylab="Topic share", main="entwine")

speech.df=tbl_df(dt_lyrics.random.df)%>%
  filter(artist=="gorguts")%>%
  select(id, friendship:love)
speech.df=as.matrix(speech.df)
speech.df[,-1]=replace(speech.df[,-1], speech.df[,-1]<1/15, 0.001)
#speech.df[,-1]=f.smooth.topic(x=speech.df[,1], y=speech.df[,-1])
plot.stacked(speech.df[,1], speech.df[,topic.plot+1],
             xlab="Sentences", ylab="Topic share", main="gorguts")

speech.df=tbl_df(dt_lyrics.random.df)%>%
  filter(artist=="al-jarreau")%>%
  select(id, friendship:love)
speech.df=as.matrix(speech.df)
speech.df[,-1]=replace(speech.df[,-1], speech.df[,-1]<1/15, 0.001)
#speech.df[,-1]=f.smooth.topic(x=speech.df[,1], y=speech.df[,-1])
plot.stacked(speech.df[,1], speech.df[,topic.plot+1],
             xlab="Sentences", ylab="Topic share", main="al-jarreau")
```



From the stacked plot above, we can see the artists' preference of topics easily. Here I just choose five artist to plot, we can see artist 'carnifex' has a strong preference on 'depression' topic, and 'al-jarreau' prefers 'friendship'. The plot is constant as the previous heatmap, they show the same patterns. For example, We can see 'al-jarreau' prefers 'friendship' both in this plot or the previous heatmap. Besides, we can also see how the topics of the artist are changing, we can see for exmaple, artist 'benny-benassi' prefers to say lots of 'illness' in some of his songs occasionally, while he alsp prefers to be cented on topic 'friendship' occasionally.

Clustering of artist

```
presid.summary=tbl_df(dt_lyrics.random.df)%>%
  select(artist, friendship:love)%>%
  group_by(artist)%>%
  summarise_each(funs(mean))
presid.summary=as.data.frame(presid.summary)
rownames(presid.summary)=as.character((presid.summary[,1]))
km.res=kmeans(scale(presid.summary[,-1]), iter.max=200,
              4)
fviz_cluster(km.res,
             stand=T, repel= TRUE,
             data = presid.summary[,-1],
             show.clust.cent=FALSE)
```



Above is A cluster plot, for the chosen 10 artists. I manually define it to be 4 clusters. And from the above result, we can see 'carnifex' himself is one cluster, while 'gorguts' and 'gorgoroth' are in one cluster. And, 'days-like-these', 'entwine', 'al-jarreau' belongs to one, and the biggest cluster is made from 'benny-benassi', 'chemlab', 'glow' and 'apollo-four-forty'.
Actually, we can understand why this cluster looks like this from the previous plots. For example, from the heatmap, we can see 'gorguts' and 'gorgoroth' are really similar in their preference, they both have a strong preference on topic 'history' while other artists do not have, so they are in one cluster. This clustering plot is a more precise clustering of artist than we have from the heatmap clustering trees.

wordcloud

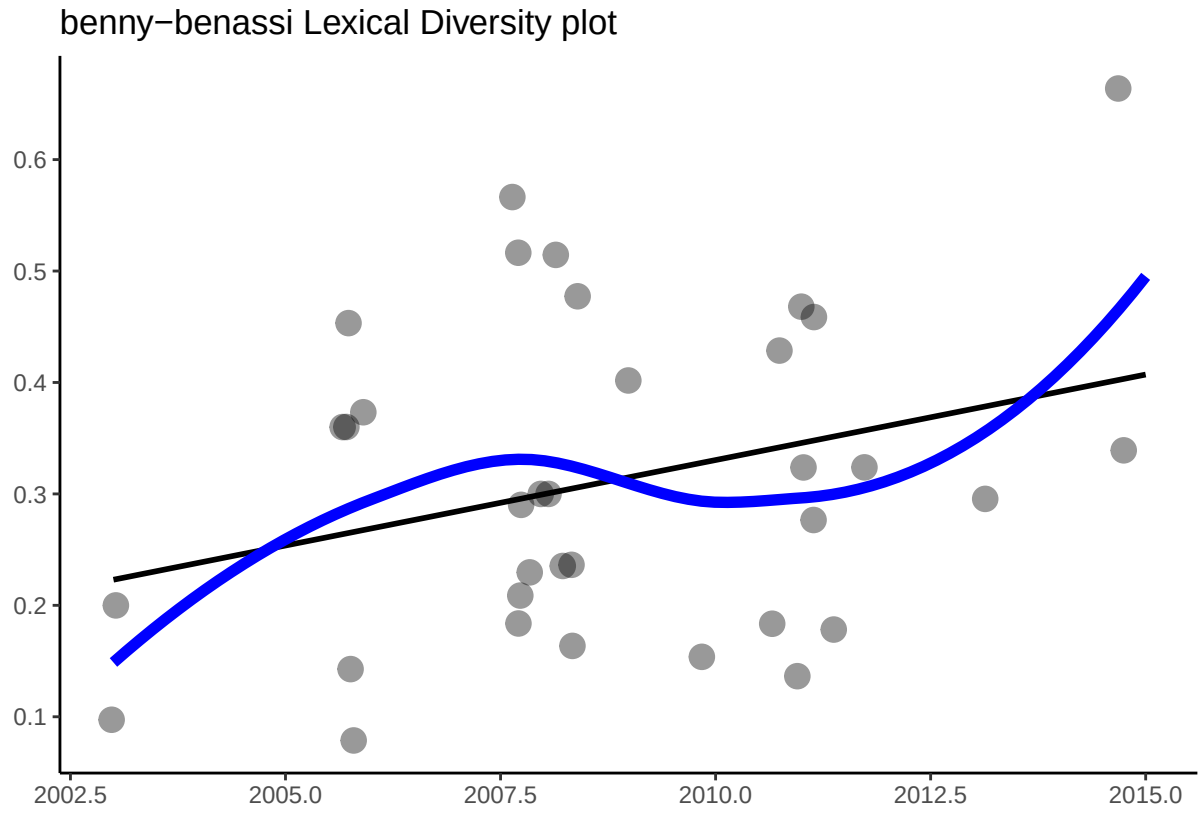
```
library(tm)
library(wordcloud)
library(RColorBrewer)
library(dplyr)
library(tidytext)
tdm.tidy=tidy(dtm)
tdm.overall=summarise(group_by(tdm.tidy, term), sum(count))
wordcloud(tdm.overall$term, tdm.overall$`sum(count)`,
          scale=c(5,0.5),
          max.words=100,
          min.freq=1,
          random.order=FALSE,
          rot.per=0.3,
          use.r.layout=T,
          random.color=FALSE,
```

```
          colors=brewer.pal(9,"Blues"))
```

Above is the wordcloud, is the stem words from all the songs in this set of random 10 chosen artists. The words with bigger and larger size, bascially means that it appreas more, because we have too much words, we can not place them all in one cloud, so this word cloud simply shows the words that have the most frequency. We can see 'love' has the most appearance among all the songs, which is actually the same as what we know as the real-world songs, most of them are talking about love. And then, 'time', and 'life', 'heart' also have a high frequency. Wordcloud is very straight forward, it is easy to understand, to be shared and are impactful.

```
lex_diversity_per_year <- dt_lyrics.random %>%
  filter(artist=="benny-benassi")%>%
  unnest_tokens(word, lyrics) %>%
  group_by(song,year) %>%
  summarise(lex_diversity = n_distinct(word)/n()) %>%
  arrange(desc(lex_diversity))

p1 <- lex_diversity_per_year %>%
  ggplot(aes(year, lex_diversity)) +
    geom_point( alpha = .4,
                size = 4,
                position = "jitter") +
    stat_smooth(color = "black", se = FALSE, method = "lm") +
    geom_smooth(aes(x = year, y = lex_diversity), se = FALSE,
                 color = "blue", lwd = 2) +
    ggtitle("benny-benassi Lexical Diversity plot") +
    xlab("") +
```

```
    ylab("") +
    theme_classic()

p1
```



Above is a lexical diversity plot for a chosen artist 'benny-benassi', x-axis is the year, y-axis is the lexical diversity . The more varied a vocabulary a text possesses, the higher its lexical diversity. Song Vocabulary is a representation of how many unique words are used in a song. So reading the above plot, it means that over the years, there is a upward trend in 'benny-benassi"s lyric diversity, so it means he is using more unique words in his lyrics.

Conclusion

Using the above methods, the clustering of artists can be correctly performed. Future effort will be put into how to determine topic not by manual, and this should improve the results a lot.