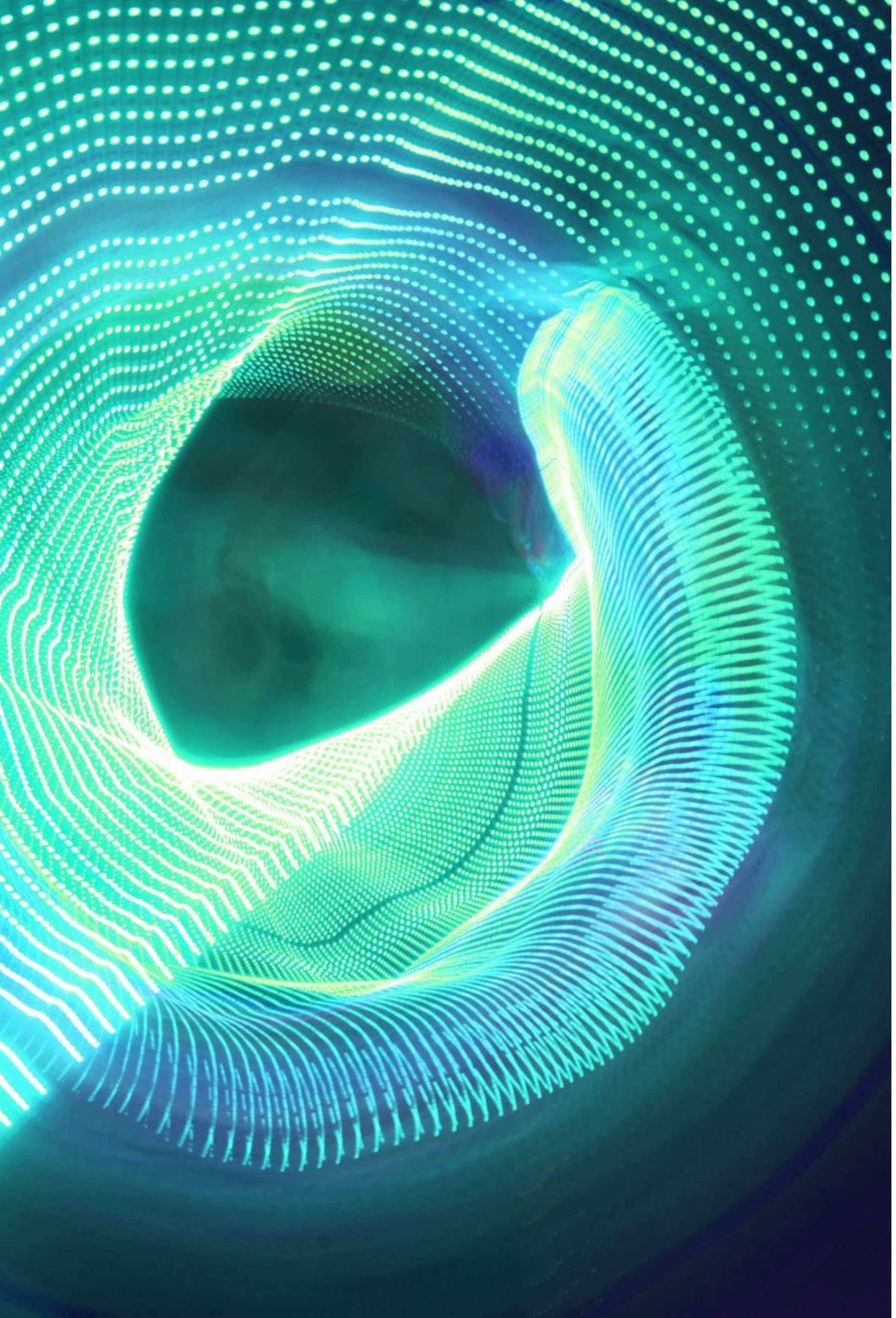
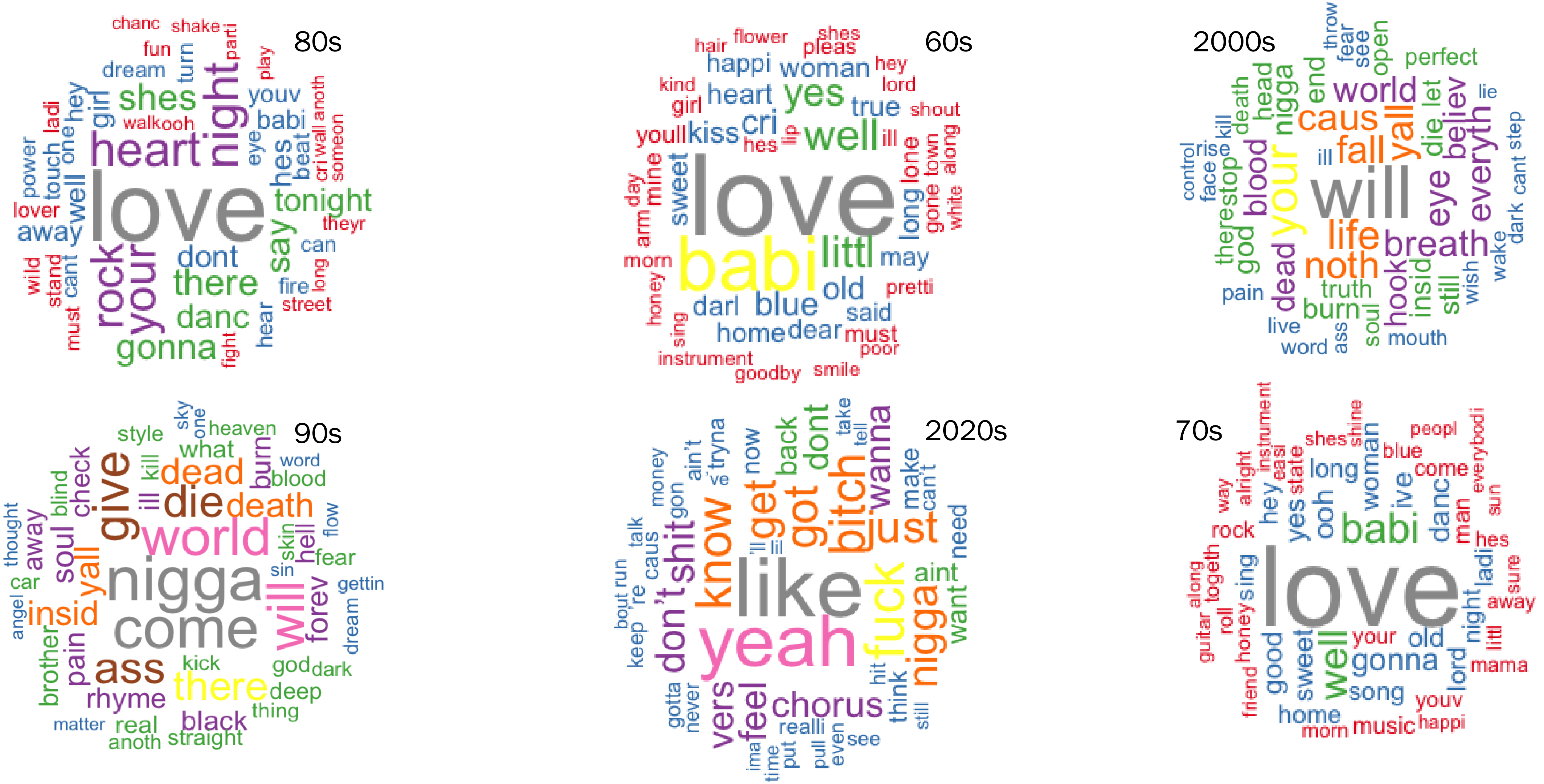


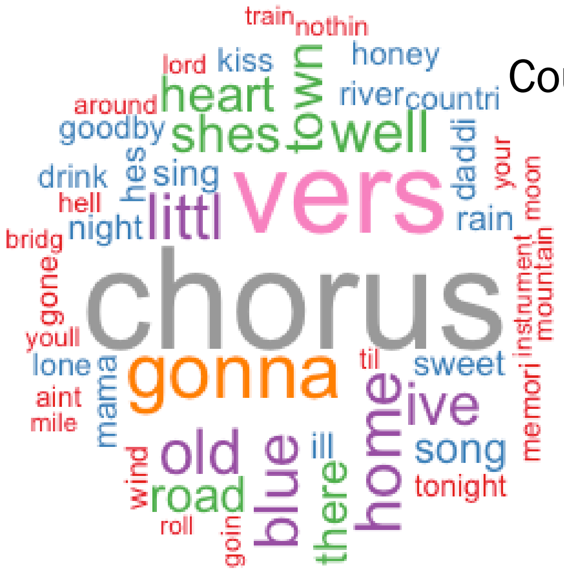
Natural Language Processing Analysis of Music Lyrics

STATISTICAL LEARNING 01:960:486

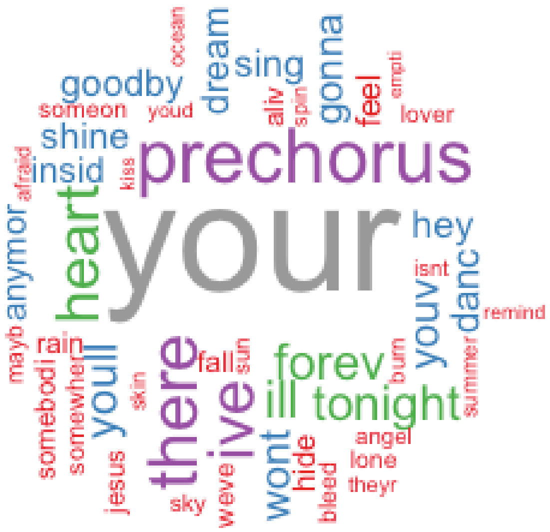


80s
love heart night your rock shes gonna danc there dont tonight say girl hes babi youv dream turn fun chanc shake parti play walkooh hey ladi one touch power lover away well eye beat cri wall anoth someon theyr can fire long street hear fight must stand cant wild
60s
love babi littl well yes cri kiss heart happi woman hey lord true shout ill youll girl kind hair flower shes pleas hes lip sweet mine day arm morn honey sing darl blue home dear old may said must poor pretti lone long gone town along white instrument goodby smile
2000s
will your life noth breath fall yall caus world nigga end head death kill rise control face stop blood there god dead pain live word ass soul burn truth mouth hook insid still wish wake dark cant step eye everyth believ die let lie ill throw fear see open perfect
90s
nigga come world will give die dead death yall insid ass there rhyme black real straight anoth matter brother pain angel car thought away soul blind check ill kill style sky one heaven what burn word blood skin sin hell fear flow gettin forev dream god dark deep thing kick
2020s
like yeah know fuck nigga just bitch got get wanna don't shit chorus vers feel back dont now tryna ain't money gon caus talk bout run keep 're 've 'll lil take tell make can't need aint want hit think still see even pull put time ima realli never gotta
70s
love babi well gonna your old song home sweet good music happi youv mama morn lord night ladi away sure hes man danc come blue peopl everybodi sun long woman ive ooh yes state easi hey alright way instrument shes shine rock along togeth roll guitar honey sing friend littl

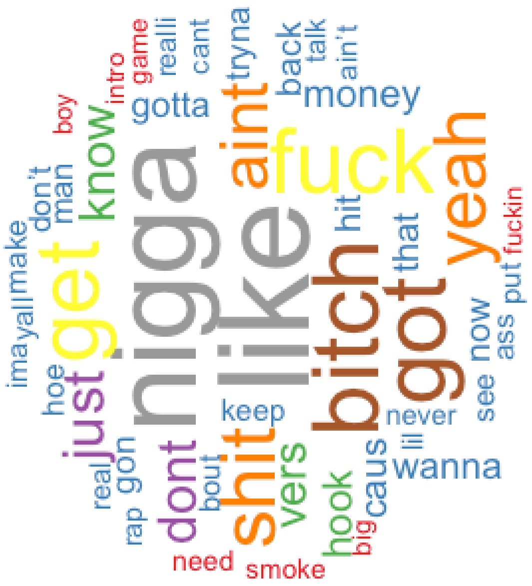
Country



Pop



Rap



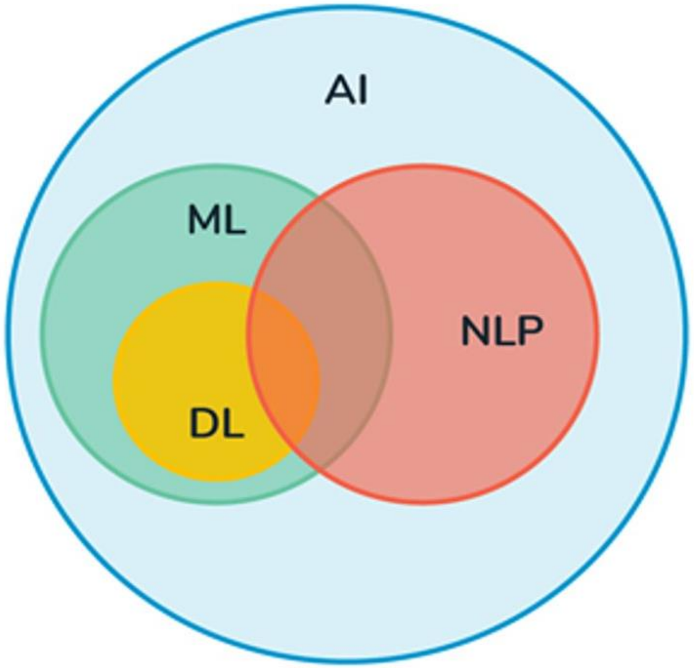
Rb

Rock

Misc

Motivation

We were curious to see if there was a relationship between music lyrics and decades, or music lyrics and genre. To handle the vast amount of lyrics text data that we had at our disposal, we decided to employ NLP techniques for our analysis.



The Music Lyrics Dataset

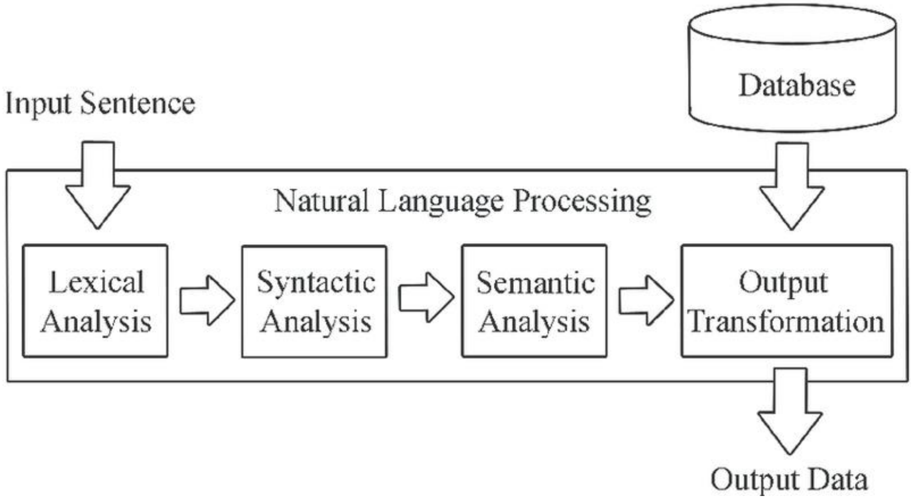
Tag (genre): rap, rb, rock, pop, misc, country

Year (decade): 1960, 1970, 1980, 2000, 2010, 2020

Column	Meaning
title	Title of the piece. Most entries are songs, but there are also some books, poems and even some other stuff
tag	Genre of the piece. Most non-music pieces are "misc", but not all. Some songs are also labeled as "misc"
artist	Person or group the piece is attributed to
year	Release year
views	Number of page views
features	Other artists that contributed
lyrics	Lyrics
id	Genius identifier
language_cld3	Lyrics language according to CLD3. Not reliable results are NaN
language_ft	Lyrics language according to FastText's langid. Values with low confidence (<0.5) are NaN
language	Combines language_cld3 and language_ft. Only has a non NaN entry if they both "agree"

https://www.kaggle.com/datasets/carlosgdcj/genius-song-lyrics-with-language-information

Clean the data & Corpus



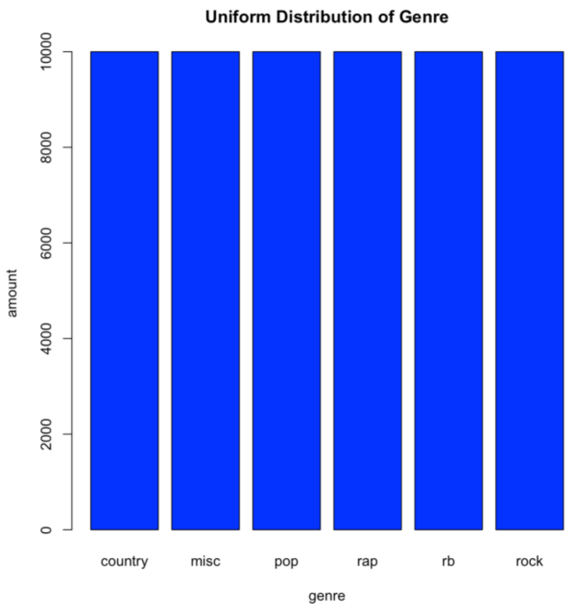
- Sample the data to 50,000 rows
- Remove stop words, punctuations, numbers, common words (the…) and unnecessary white space.
- Reduce words in their root or base forms

```
song.data1<-read.csv("/Users/tyralassiter/Desktop/Statistical Learning/songs_uniformbygenre.csv")
song.data1<-song.data1[,c(1,2,3,5,8)] # keeps id, title, lyrics, year, tag
genre<-song.data1$tag# create decade vector
song.data1$genre<-genre # save decade as column
table(song.data1$genre)
# clean data function using tm
clean <- function(corpus){
  corpus<-VCorpus(VectorSource(corpus))
  corpus<-tm_map(corpus,content_transformer(tolower))
  corpus<-tm_map(corpus,removeNumbers)
  corpus<-tm_map(corpus,removePunctuation)
  corpus<-tm_map(corpus,removeWords, stopwords('en')) # removes common words (the...)
  corpus<-tm_map(corpus,stemDocument) # loved->love
  corpus<-tm_map(corpus,stripWhitespace)
  return(corpus)
}
```

Uniform Distribution

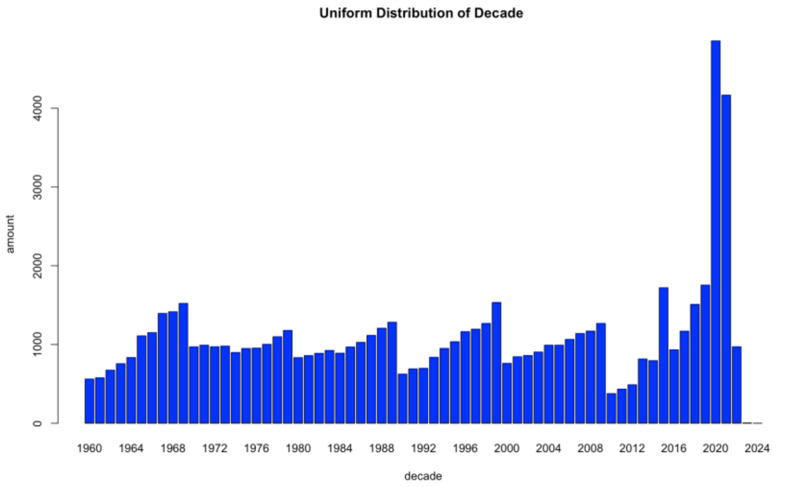
table(song.data1$genre)

country	misc	pop	rap	rb	rock
10000	10000	10000	10000	10000	10000



table(song.data$decade)

1960	1970	1980	1990	2000	2010	2020
10000	10000	10000	10000	10000	10000	10000



Since the original data had half of the sample from 2010, in order to make the results more accurate and fair. We uniformed lyrics by decade and genre.

Building the DTM

- A document term matrix is a matrix that describes the frequency of terms within our data.
- This allows us to perform statistical analysis on the relationship between lyrics and genre.
- With this in mind, we built a function that generated a DTM as a data frame.
- To do this, the function received two inputs:
  - Category: Genre (tag) | Corpus: A cleaned corpus of our Lyric data
- Our function then used the tm package to
  - clean the corpus
  - create a document term matrix
  - remove sparse terms
  - convert the document term matrix into a data frame.
- The function then returns that data frame and its category as output.

```r
# build document term matrix, even it out
generateDTM <- function(category,corpus){
  corpus<-clean(corpus)
  dtm<-DocumentTermMatrix(corpus)
  dtm<-removeSparseTerms(dtm,0.95) # removes sparse words
  df<-as.data.frame(as.matrix(dtm))
  df$category<-category
  return(df)
}
# running program, creating df
df<-generateDTM(song.data$decade,song.data$lyrics)
```

DTM Data Frame

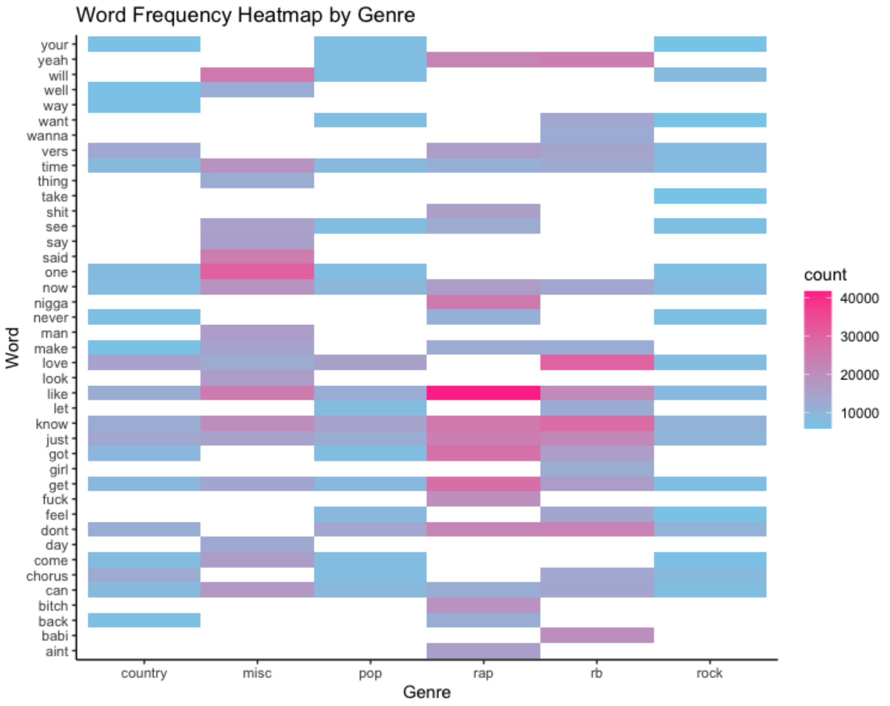
Upon converting DTM into a data frame, we are able to use that data frame for random forest and KNN models.

We can also use this data frame to view the data and create tables and graphs of the most frequent words

For example:

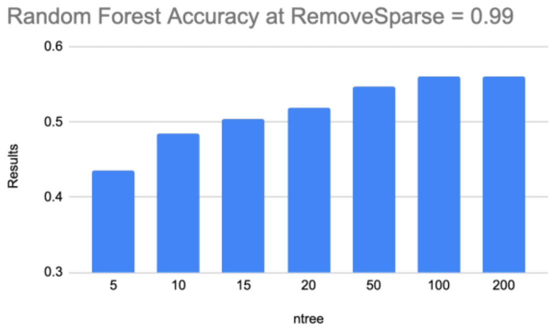
```r
```{r}
#rap
sort(colSums(df1[df1$category=='rap',-length(df1)]),decreasing = T)[1:10]
#pop
sort(colSums(df1[df1$category=='pop',-length(df1)]),decreasing = T)[1:10]
#misc
sort(colSums(df1[df1$category=='misc',-length(df1)]),decreasing = T)[1:10]
#rb
sort(colSums(df1[df1$category=='rb',-length(df1)]),decreasing = T)[1:10]
#rock
sort(colSums(df1[df1$category=='rock',-length(df1)]),decreasing = T)[1:10]
```
```

```
 like   get   got  know nigga  just  dont  yeah  fuck bitch
41730 27590 27119 25229 24870 24065 22750 22576 19713 18880
 love  know  dont  just  like   now   can  time  feel   get
15300 14778 13961 12252 12069  9425  9195  8877  8841  8795
  one  will  like  said  know   now  time   can  come   man
30852 25382 24709 24416 20117 18854 18645 17405 16003 15958
 love  know  yeah  dont  just  like  babi   got   get  vers
30487 28418 23945 22789 21820 21294 19874 16188 15985 14246
  know   dont   just   like   will chorus   vers    now   time   love
 10827  10248   9718   8792   8633   8608   8511   8469   8026   7695
```



Relationship Between Lyrics & Genre
Random Forest

- The best result we were able to achieve was 56.06% accuracy with RF.
- This was with Removing 1% of sparse terms and ntree = 200.
- We did see that when holding everything else constant, our accuracy increased as ntree increased.
- As we got to ntrees = 100 or higher, the code took an extremely long time to run and it became impractical to keep going.

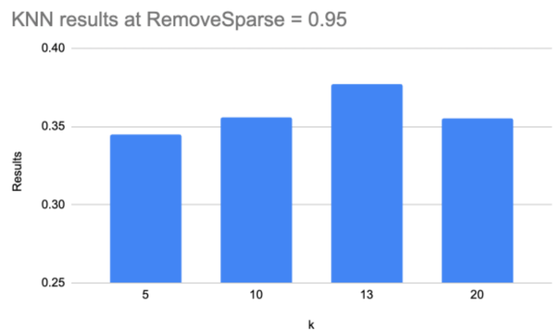


- That, and the fact that we saw very minimal improvement between ntrees=100 and ntrees=200 made us comfortable to stop where we did.
- We also know that increasing ntrees ad infinitum would have led to overfitting and a decrease in accuracy (bias-variance tradeoff).

Relationship Between Lyrics & Genre

K Nearest Neighborhood

- KNN overall was significantly worse at prediction than RF.
- This indicates that the data didn't cluster very well in multidimensional space.
- The best result we saw from KNN was the first run we did, which gave us 37.9% accuracy (recall a random guess would be about right about 16.7% of the time).
- This run was done after removing 5% of the sparse terms and choosing k = 13.
- KNN also took an extremely long time to run, limiting the amount of runs we were able to do.
- We did however see that 13 is probably close to the optimal number of neighbors to consider based on the graph below.



KNN Model vs. RF Model

Based on the findings mentioned in the prior slides, we came to the conclusion that the RF model predicted the relationship between lyrics and genre better.

The RF Model was quicker to run and yielded an accuracy of 56% compared to 37.9% from KNN

Despite the pitfalls of both models, we were still able to conclude that there is a relationship between lyrics and genre

rf.y

	country	misc	pop	rap	rb	rock
country	1818	134	503	57	316	241
misc	131	2031	262	172	110	287
pop	470	199	906	115	527	636
rap	57	69	78	2375	290	83
rb	253	57	323	541	1602	192
rock	486	234	646	89	296	1289

```
> sum(diag(rf.cm))/sum(rf.cm)
[1] 0.5606154
```

knn.y

	country	misc	pop	rap	rb	rock
country	627	985	517	124	233	583
misc	91	2374	142	127	76	183
pop	268	1133	506	123	307	516
rap	123	452	203	1602	337	235
rb	327	504	458	401	906	372
rock	217	1278	489	97	195	764

```
> sum(diag(knn.cm))/sum(knn.cm)
[1] 0.3792448
```

Conclusion and Future Direction

By comparing these two models, we noticed that RF was more successful with predicting genre and decade than KNN.

- RF's accuracy is the average of a handful of different classification trees.
- As for KNN, we learned that our data didn't cluster well and a lot of songs spanned multiple genres according to the word association we had already established.

It would be interesting to find data that clusters better than vague terms such as ‘decade’ or ‘genre’. Do individual artists have lyrical themes that bleed through over the course of their career? Could we create a matrix to find similar lyrics between artists? Would this then create a new, nuanced, and more naturally occurring clustering of genres easier to predict?