Analysis of Apple Music Trends and Topics: Final Report

Introduction: Motivations and Background

Are the most popular songs trending on Apple Music in the United States basic (too mainstream, lacking creativity/diversity)? Are they mostly centered around the same topics and do they share a lot of the same language?

These are some of the questions we had in mind at the beginning of this project. Music is an important aspect of many peoples' lives, and we wanted to know what kinds of topics and words people are listening to the most. Apple Music has an incredible versatility in its data collection and analysis. The performance of individual artists and the ability to reach the top 100 play list in the United States are based on a number of different variables, including the total number of plays, average daily listeners, and number of purchases from the iTunes store. This play list, which essentially represents the most popular songs in real time, is updated on a regular basis and provides the ranking of each track, the artists involved, whether or not the track has changed rankings recently and, if so, how far up or down the track's ranking has changed.

Data

The data that we used is collected by a programmer and data scientist from the Netherlands. He created the website from which we got our data from for the purpose of having a "personal playground". The data that he collects from various sources, including Spotify, Itunes, and YouTube, is updated on a regular basis, and he often includes his own methods and calculations as well. The main link, which includes all of these different sources of information, is https://kworb.net/, while the link that we used specifically for the Apple Music charts is https://kworb.net/charts/apple_s/us.html. The data we used is in the form of a table, which includes information about the ranking of the song, whether that ranking has changed and by how much, and the artist and title of the song. Because this website is constantly changing and the information is updated on a regular basis, we chose to look at the top songs on Apple Music on April 26, 2022. Therefore, the methods we used to scrape our data from this site in our data wrangling RMarkdown will not work on any other day. This data demonstrates the common tastes in music among the American population at this time.

The data wrangling portion of the project was difficult given the nature of the data and the amount of it. Initially, we had planned to scrape the official Apple Music website, but this proved challenging due to the interactive nature of the website and the inability of the Apple Music API to work with the specific data we were looking for. The second challenge that we ran into involved linking the song titles and artists with the lyrics on a separate website. This was difficult to handle on a larger scale due to cases of inconsistent formatting and symbols. This stage of the data wrangling retrieving the lyrics of each song took a significant amount of time. Finally, finding the lyrics on each song's page was a long process due to the html layout of the webpage. Overall, the web scraping portion of the project was trying and involved researching several different strategies.

Here is a snapshot of our data source as a result of scraping the site:

```
songs <- read_csv("../Data/songs.csv", show_col_types = FALSE)
head(songs, 5)
```

```
## # A tibble: 5 x 5
##     Pos Artist   Title       link                    lyrics
##   <dbl> <chr>    <chr>       <chr>                   <chr>
## 1     1 Jack Ha~ First Class https://genius.com/Ja~ "Intro: Jack Harlow Mm Chor~
```

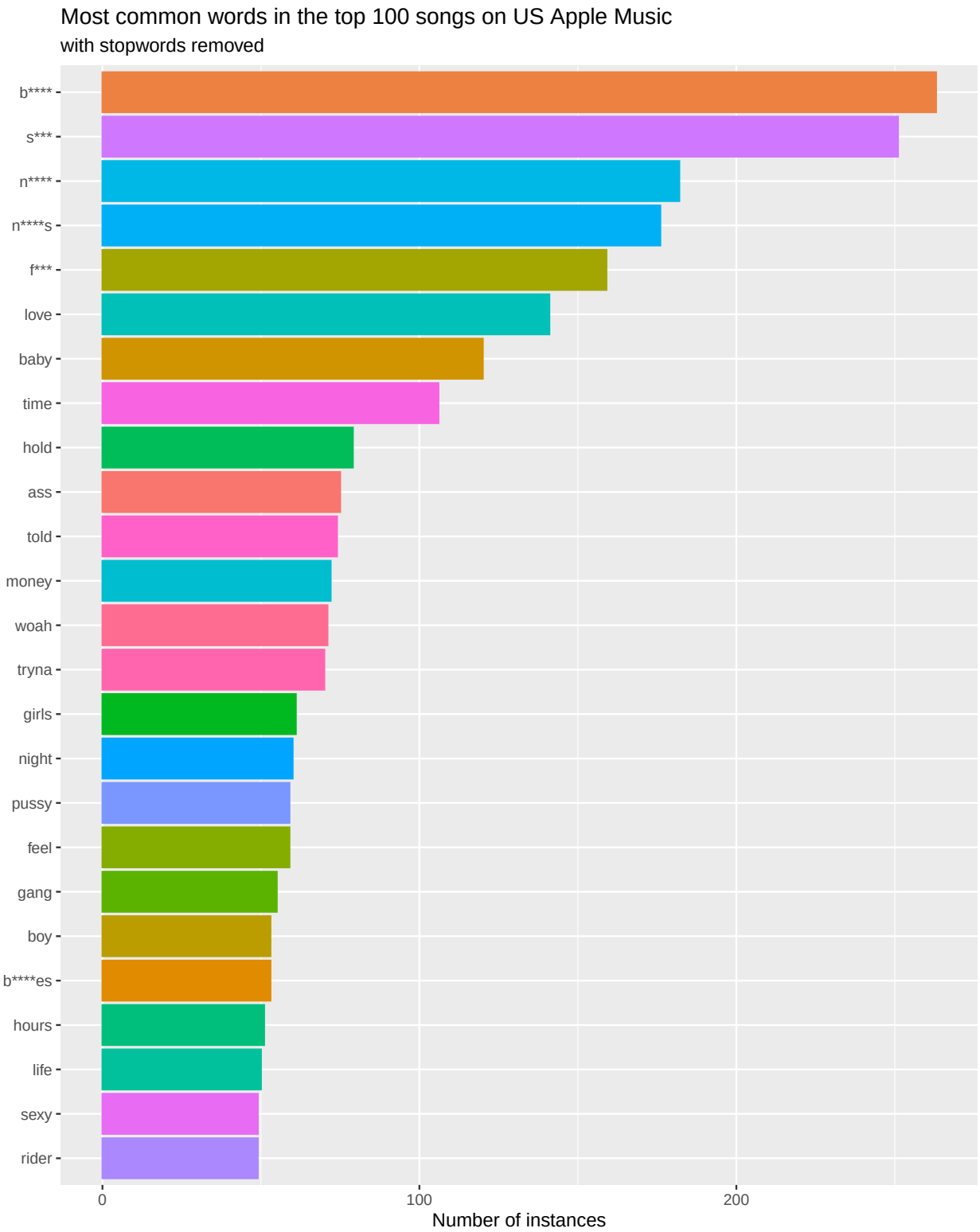
```
## 2     2 Harry S~ As It Was   https://genius.com/Ha~ "Intro Come on, Harry, we w~
## 3     3 Lil Baby In A Minute https://genius.com/Li~ "Intro (Damn, Kai, you goin~
## 4     4 Lil Durk What Happe~ https://genius.com/Li~ "Intro: King Von Banger Oh ~
## 5     5 Lil Baby Right On    https://genius.com/Li~ "Intro (ATL Jacob, ATL Jaco~
```

Methods

We first performed simple text analysis on the tokenized, censored words of the songs, all unigrams.

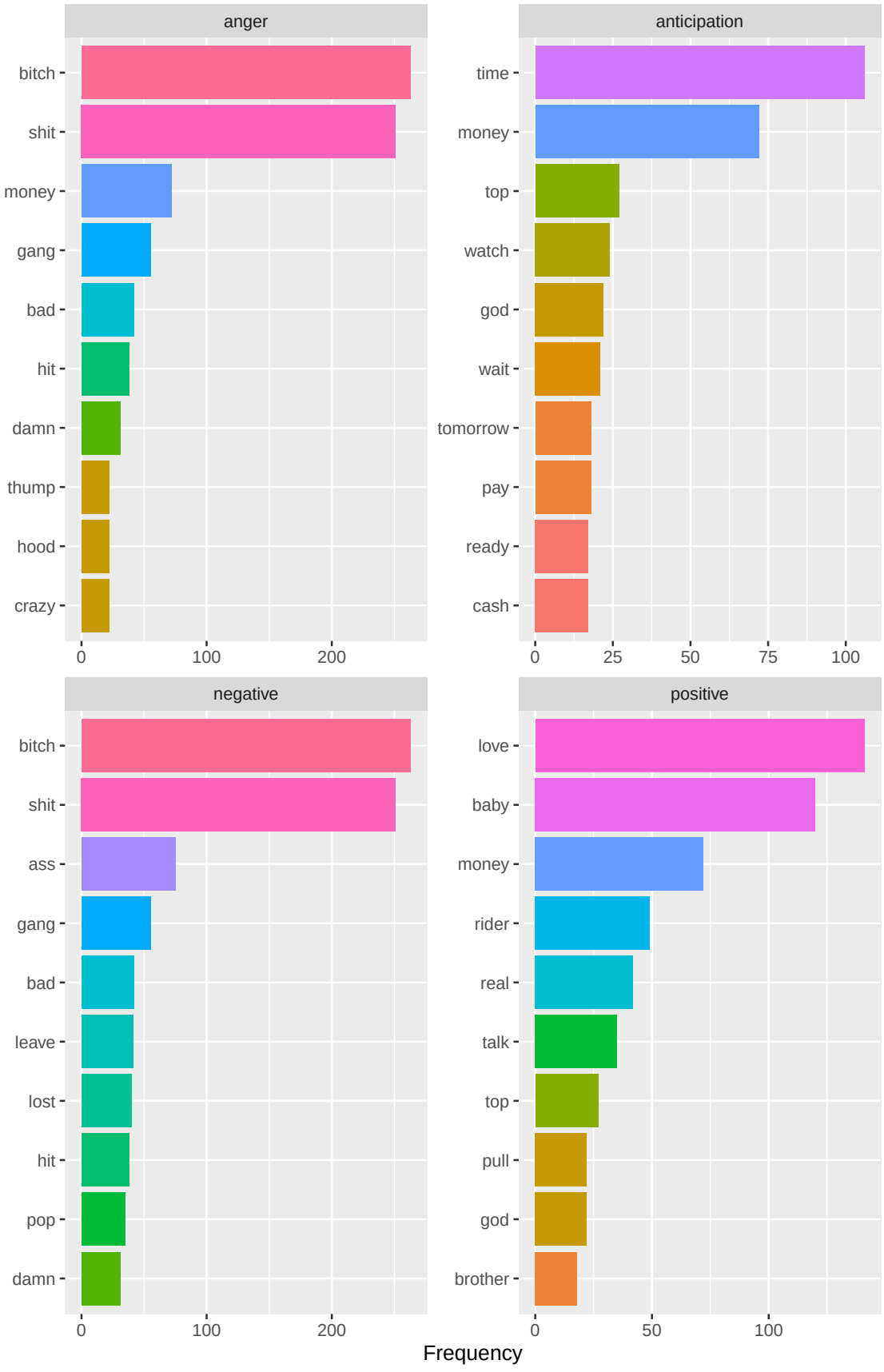
```
# Tokenize words to obtain unigrams
songs_words_all <- songs %>%
  unnest_tokens(output = word, input = lyrics) %>%
  select(Artist, Title, word)
```

```
# Plot bar graph of censored word frequencies
g <-
  ggplot(data = songs_words_censored, aes(x = reorder(word_censored, n), y = n,
              color = word_censored, fill = word_censored)) +
  geom_col() +
  coord_flip() +
  guides(color = "none", fill = "none") +
  labs(x = NULL,
       y = "Number of instances",
       title = "Most common words in the top 100 songs on US Apple Music",
       subtitle = "with stopwords removed")

g
```




```
# Apply nrc sentiment analysis to words
nrc_songs <- word_frequencies %>%
  inner_join(nrc_lexicon, by = "word") %>%
  filter(sentiment %in% c("anger", "negative",
```

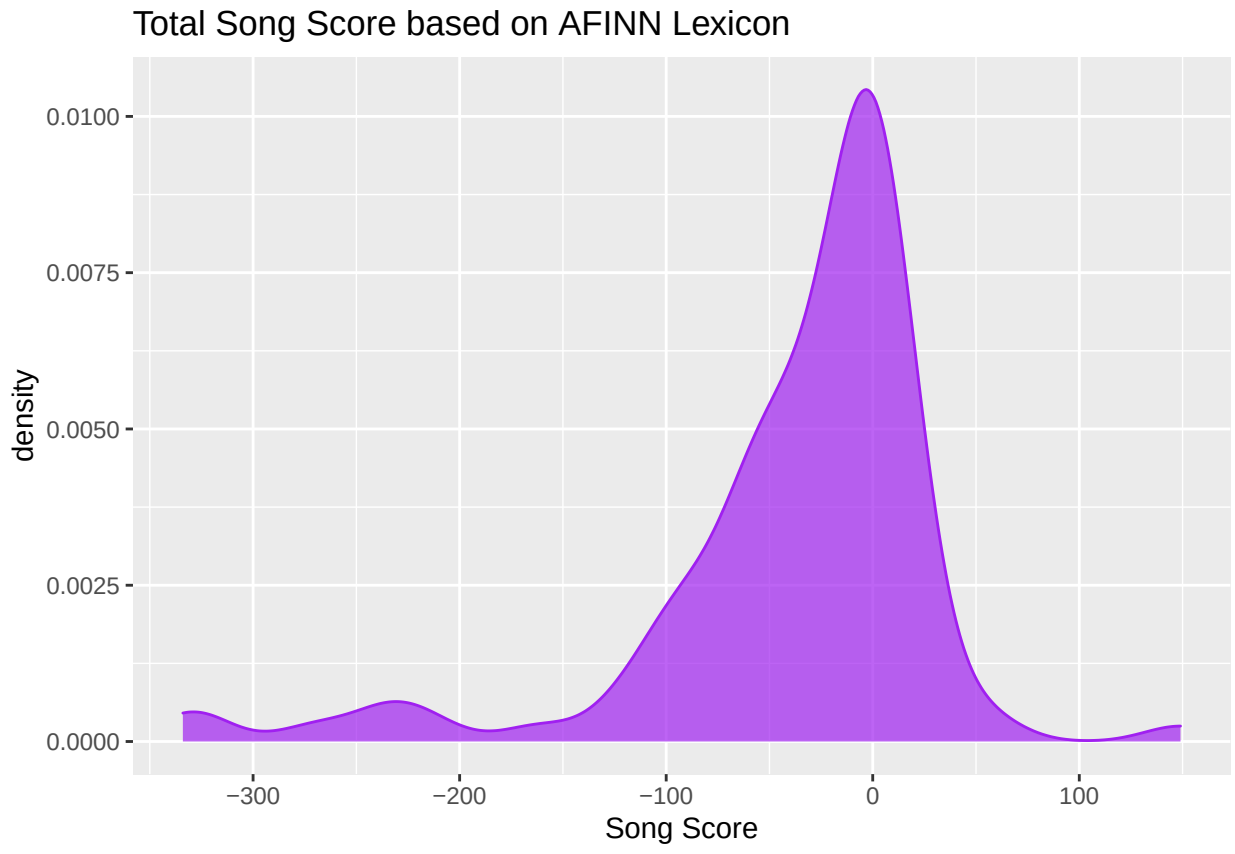
```
                          "positive",
                          "anticipation")) %>%
  arrange(sentiment, desc(n)) %>%
  group_by(sentiment) %>%
  slice(1:10)

ggplot(data = nrc_songs,
       aes(x = reorder(word, n),
           y = n,
           fill = as.factor(n))) +
  geom_col(show.legend = FALSE) +
  coord_flip() +
  facet_wrap( ~ sentiment, ncol = 2, scales = "free") +
  labs(x = NULL,
       y = "Frequency")
```


anger
bitch
shit
money
gang
bad
hit
damn
thump
hood
crazy
0
100
200
anticipation
time
money
top
watch
god
wait
tomorrow
pay
ready
cash
0
25
50
75
100
negative
bitch
shit
ass
gang
bad
leave
lost
hit
pop
damn
0
100
200
positive
love
baby
money
rider
real
talk
top
pull
god
brother
0
50
100
Frequency

```r
# Calculate overall AFINN score
songs_words_all %>%
  inner_join(afinn_lexicon, by = "word") %>%
  summarize(total_score = sum(value))
```

```
## # A tibble: 1 x 1
##   total_score
##         <dbl>
## 1       -3718
```

```r
# Calculate AFINN scores by poem
afinn_by_song <- songs_words_all %>%
  inner_join(afinn_lexicon, by = "word") %>%
  group_by(Title) %>%
  summarize(song_score = sum(value)) %>%
  arrange(song_score)

ggplot(data = afinn_by_song,
       aes(x = song_score)) +
  geom_density(color = "purple",
               fill = "purple",
               alpha = 0.7) +
  labs(title = "Total Song Score based on AFINN Lexicon",
       x = "Song Score")
```



For our element of going beyond, we explored latent Dirichlet allocation, which is a method that finds natural groups of topics even when obvious groupings are not present. This method is a powerful tool to find hidden

themes using the probabilities of words belonging to certain topics. Latent Dirichlet allocation was first introduced in 2000, and has since been applied to a wide variety of academic fields including population genetics and machine learning. We were initially interested in this method for topic modeling because we wanted to see whether the most popular songs on Apple Music in the United States shared any common themes.

```
# Choose top 15 words, group by topic
song_top_terms <- song_lda_td %>%
  group_by(topic) %>%
  slice_max(beta, n = 15) %>%
  ungroup() %>%
  arrange(topic, -beta)

# Pivot dataset to see words grouped by topic 1 or 2 (still ordered by decreasing beta value)
song_top_terms_pivoted <- song_top_terms %>%
  pivot_wider(names_from = topic, values_from = term_censored) %>%
  select(-beta)

song_top_terms_pivoted_top1 <- song_top_terms_pivoted %>%
  select(2)
song_top_terms_pivoted_top1 <- song_top_terms_pivoted_top1[1:15,]
kable(song_top_terms_pivoted_top1)
```

1
b****
love
girls
time
ass
baby
feel
play
block
ayy
mind
headshot
pussy
call
hold

```
song_top_terms_pivoted_top2 <- song_top_terms_pivoted %>%
  select(3)
song_top_terms_pivoted_top2 <- song_top_terms_pivoted_top2[16:30,]
kable(song_top_terms_pivoted_top2)
```

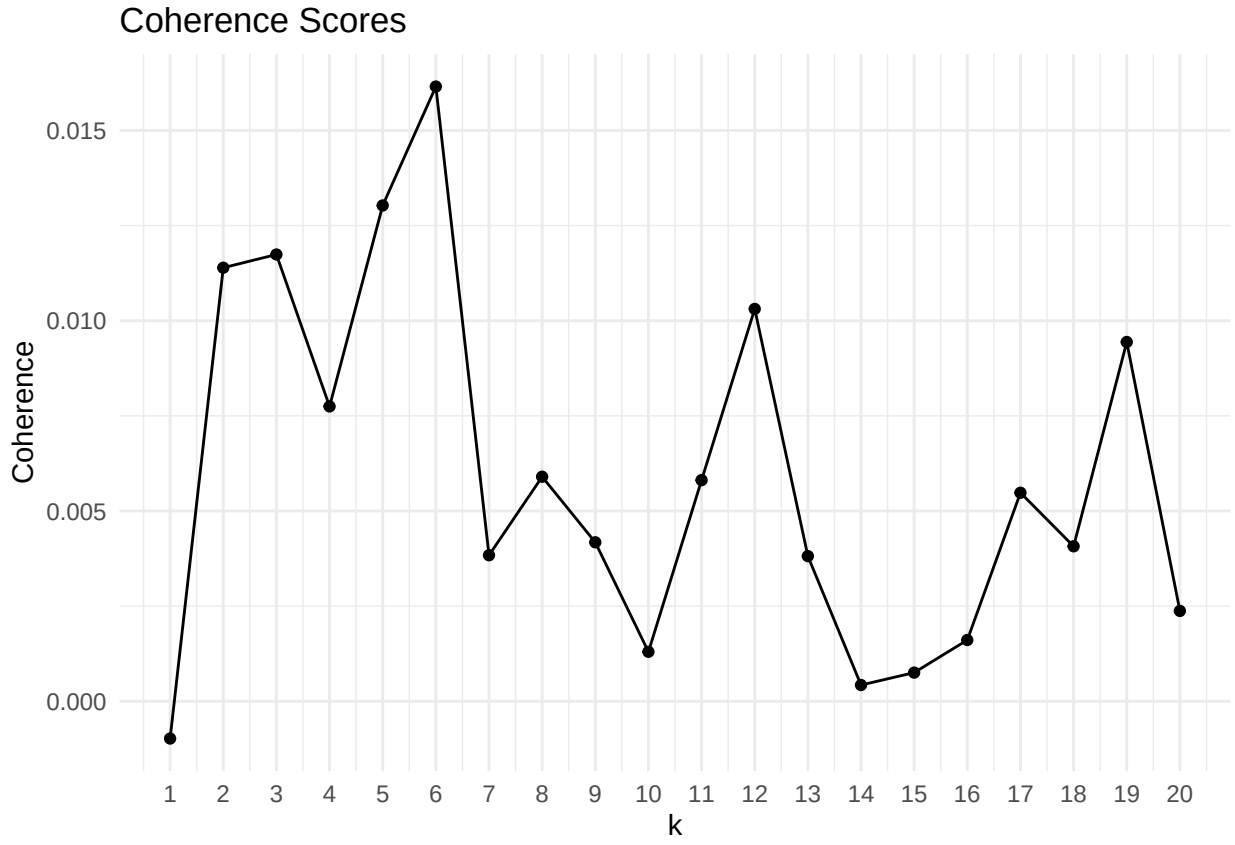
2
b****
baby
love
time
night
money
hours
sexy
gang
hold
gettin'
rider
pushin'
b****es
petty

We began with a coherence plot to check which number of topics most made sense for this model (similar to the use of an elbow plot for clustering). From this coherence plot which showed low coherence scores overall, we concluded that, while 2 and 5 topics had relatively higher coherence scores, latent Dirichlet allocation is not entirely appropriate for the dataset we were working with. Coherence scores tell us how well words fit into a given topic or grouping. Low coherence scores essentially showed us that the words among the most popular songs did not really share any common themes. We also chose to use k=2 groups because we determined that a greater number of topics yielded an even looser interpretation.

```
# Create DTM (document term matrix)
dtm <- textmineR::CreateDtm(songs_words$word,
                  songs_words$n,
                  ngram_window = c(1, 2))

# Explore basic frequency
tf <- TermDocFreq(dtm = dtm)

# Eliminate words appearing less than 2 times
vocabulary <- tf$term[ tf$term_freq >= 1 ]

dtm = dtm
```

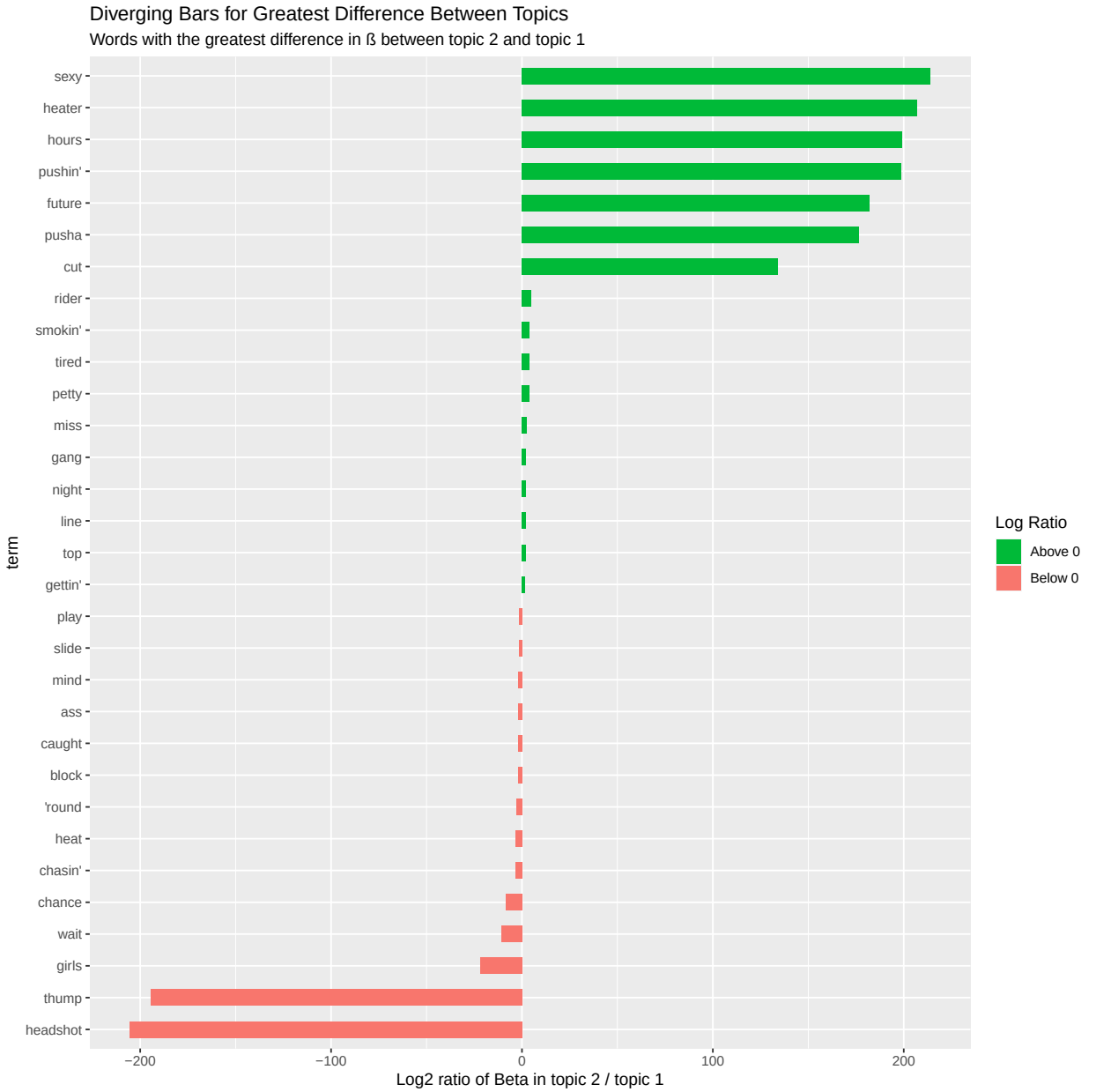
```
k_list <- seq(1, 20, by = 1)
set.seed(100)
model_dir <- paste0("models_", digest::digest(vocabulary, algo = "sha1"))
if (!dir.exists(model_dir)) dir.create(model_dir)
model_list <- TmParallelApply(X = k_list, FUN = function(k){
  filename = file.path(model_dir, paste0(k, "_topics.rda"))

  if (!file.exists(filename)) {
    m <- FitLdaModel(dtm = dtm, k = k, iterations = 500)
    m$k <- k
    m$coherence <- CalcProbCoherence(phi = m$phi, dtm = dtm, M = 5)
    save(m, file = filename)
  } else {
    load(filename)
  }

  m
```

```
}, export=c("dtm", "model_dir"))

# Model tuning, choose the best model
coherence_mat <- data.frame(k = sapply(model_list, function(x) nrow(x$phi)),
                            coherence = sapply(model_list, function(x) mean(x$coherence)),
                            stringsAsFactors = FALSE)

# Plot coherence scatterplot with connected points
ggplot(coherence_mat, aes(x = k, y = coherence)) +
  geom_point() +
  geom_line(group = 1)+
  ggtitle("Coherence Scores") + theme_minimal() +
  scale_x_continuous(breaks = seq(1,20,1)) + ylab("Coherence")
```




```
# Create chart of terms, log ratios of topic betas
beta_wide <- song_lda_td %>%
  mutate(topic = paste0("topic", topic)) %>%
  pivot_wider(names_from = topic, values_from = beta) %>%
  filter(topic1 > .001 | topic2 > .001) %>%
  mutate(log_ratio = log2(topic2 / topic1),
         type = ifelse(log_ratio < 0, "below", "above"))

# Filter log ratios since some are almost indiscernible, order by log ratios
beta_wide <- beta_wide %>% filter(log_ratio > 1.5 | log_ratio < -1.5)
beta_wide <- beta_wide[order(beta_wide$log_ratio), ]
beta_wide$term <- factor(beta_wide$term, levels = beta_wide$term)
```

```
# Create divergence plot for greatest differences in word beta values
diverge_graph <- ggplot(beta_wide, aes(x=term, y = log_ratio, label = log_ratio)) + geom_bar(stat='iden
  scale_fill_manual(name="Log Ratio",
                    labels = c("Above 0", "Below 0"),
                    values = c("above"="#00ba38", "below"="#f8766d")) +
  labs(subtitle="Words with the greatest difference in  between topic 2 and topic 1",
       title= "Diverging Bars for Greatest Difference Between Topics",
       y = "Log2 ratio of Beta in topic 2 / topic 1") +
  coord_flip()

diverge_graph
```



Results/Conclusions

To achieve an appropriate set of words to perform simple text analysis on, we censored quite a bit by using str_replace() inside mutate(). The top 5 most commonly used words based on our bar graph were all swear words, which holds obvious implications about the popular music that US listeners hear. There are more swear words than any other words, and given that swear words are normally used in vernacular to emphasize sentiment, they likely contribute to the overall impression of a line or a song but do not contain any subject content themselves. Thus, we might say that the top 100 songs in the US are "basic," since there are more filler than content words. The heavy presence of swear words also comes into play later, when we remove them from our datasets for further analysis. Their role of emphasizing sentiment rather than holding content becomes all the more important for sentiment analysis.

Our next components of simple text analysis involved implementing the NRC and AFINN lexicons. When using the NRC lexicon, we decided to create only 4 grouped plots as categorizations of the common words in lyrics. This is because when we did similar analysis on the Emily Dickinson poems, we allowed for a wider range of sentiments. However, those same sentiments did not apply well to the top 100 songs, since we saw that categories like joy contained words that did not easily fit. Thus, after cutting down to the sentiments that have logical groupings, we found that the lyrics consist of mostly negative words. Especially observing the two plots for negative and positive sentiment, we noticed that the x-axis of the negative one scaled to cover twice that of the positive one, indicating that the frequencies of negative words are much higher.

Our analysis with the AFINN lexicon tells the same story. We calculated a very negative total sentiment score for the words, which is certainly not any measure of center or average, but a more even distribution of word sentiments would result in a milder total score around 0. We also created a density plot of the total scores of the songs by the AFINN lexicon. The center of the plot laid reasonably below 0, and the distribution was left-skewed, meaning that there were more songs containing very negative words.

As for our LDA analysis, most of the coherence scores that we obtained from the data laid very low at around 0.0005, whereas LDA analysis performed on, for example, Harry Potter texts yields coherence scores of around 0.37 (Rafferty, 2018). Considering the incredibly low coherence scores that we started the analysis with, it is not surprising that our topics seemed to fit rather loosely around the word groups. This only shows that there is a need to tune the model, such as increasing k to achieve better results or having more songs to analyze (tang, 2019).

For the purpose of analysis of our specific data (from a certain day and a certain website, and thus limited), we disregarded the values and chose the highest coherence score of $k = 2$ as explained in Methods. We then peeked the words that fell within the two groups and drew out (with some difficulty) the topics which seemed to reside most coherently in the word groups, girls and money. Each topic also contained an assigned beta value ($P(word|topic)$) for every of its words — probability of word given a topic (tang, 2019). Using this, we calculated the greatest difference in the beta values of words between topic 1 and topic 2 and applied the log ratio of the two for symmetrical differences, thus yielding the probability ratio of the same word being in both topics. Although there were common words among these two topics, we decided not to remove them because this would detract from the overall interpretations of the topics. Additionally, we did not use bigrams, or phrases, in our analysis, is because we believed that it would be easier to analyze individual words. It is difficult to find databases with sentiments attached to a combination of words, especially those found in songs which often don't adhere to proper grammar.

Our second plot containing the diverging bars, then, illustrates the magnitudes of difference between the words of the two topics and their polarities relative to the topics found by LDA (Yan, 2020). We also chose to analyze only two topics due to the binary nature of the divergence graph. The further two words are from each other on the vertical axis and the longer their bars in opposite directions are, the less likely they are to belong to the same topic, and generally, the more different they are. By this logic, we see that "sexy" and "headshot" must be the two words that would be least likely to fit into the opposite's topic, and considering our topics of girls and money, the contradiction of these words make sense.

An interesting aspect of the divergence plot is that there are many more green bars (above 0, indicating that a word belongs to topic 1, girls) that are long than red ones. This means that more words in topic 1 are

less likely to be found in topic 2. It also dissolves the possibility that the words in corresponding, opposite locations on the vertical axis are necessarily "antonyms" in the context of the song lyrics. So, although the log function mentioned earlier allows for symmetrical and thus comparable magnitudes, we cannot say that the words themselves are found to be symmetrical to one another by nature.

Limitations

There are several limitations to this work; the first of which stems from the inherent nature of the data we worked with. Because the data changes on a daily basis, it is difficult to standardize a web scraping method that will work every time the data is updated. The connection between the artist and song title and the associated lyrics of the song is based on this web scraping method, which is why we chose to analyze the data on a single day, April 26, 2022. This makes it difficult to draw conclusions about common topics among the most popular songs over any length of time.

The second limitation is the language barrier. English is not the only language in the world, and it is certainly not the only language artists choose to work with. The analysis that we performed and the conclusions we made are based on the songs that are predominantly in English, which then leaves out a significant amount of data.

The third limitation involves the swear words prevalent in the content of these songs. Because these words are not included in the databases of sentiments and associated topics, swear words were not included in the text and LDA analysis beyond analyzing the frequency of words. This leaves out a significant portion of words that could potentially tell us a story about what emotions and topics are covered in popular songs in the United States, thus we would suggest text analysts to explore swear words as an addition to their lexicon databases. We understand that this can be a sensitive topic, but we believe that the words are commonly used in current culture and should be considered when performing text analysis. An artist's usage of swear words can be commonplace and out of habit or it may be intentional in order to carry strong sentiment. Clearly, it would be very difficult to accurately perform LDA analysis on swear words then, as there are countless contexts in which artists are using them. However, we still miss a considerable amount of valuable data by disregarding swear words.

Finally, because the data was collected by a programmer and data scientist unaffiliated with Apple Music, it is possible that the data does not exactly match what is portrayed on the official Apple Music website. Human error, technological mistakes, and inconsistent information sources could result in discrepancies between the two sites. The study could be improved by directly navigating the Apple Music site to work with the most up-to-date information possible.

References

Bouchet-Valat, Milan. SnowballC: Snowball Stemmers Based on the C "libstemmer" UTF-8 Library. 0.7.0, 2020. R-Packages, https://CRAN.R-project.org/package=SnowballC.

Grün, Bettina, et al. Topicmodels: Topic Models. 0.2-12, 2021. R-Packages, https://CRAN.R-project.org/package=topicmodels.

Jones, Tommy, et al. TextmineR: Functions for Text Mining and Topic Modeling. 3.0.5, 2021. R-Packages, https://CRAN.R-project.org/package=textmineR.

MarketMuse. "What Is Latent Dirichlet Allocation (LDA) - Latent Dirichlet Allocation (LDA) Definition from MarketMuse Blog." MarketMuse Blog, https://blog.marketmuse.com/glossary/latent-dirichlet-allocation-definition/. Accessed 3 May 2022.

Robinson, Julia Silge and David. 6 Topic Modeling | Text Mining with R. www.tidytextmining.com, https://www.tidytextmining.com/topicmodeling.html. Accessed 3 May 2022.

RPubs - A Practical Application of the TM Package. https://rpubs.com/tsholliger/301914. Accessed 3 May 2022.

tang, Farren. “Beginner’s Guide to LDA Topic Modelling with R.” Medium, 14 July 2019, https://towardsdatascience.com/beginners-guide-to-lda-topic-modelling-with-r-e57a5a8e7a25.

Yan, Qiushi. 6.1 Latent Dirichlet Allocation | Notes for “Text Mining with R: A Tidy Approach.” bookdown.org, https://bookdown.org/Maxine/tidy-text-mining/latent-dirichlet-allocation.html. Accessed 3 May 2022.

Rafferty, Greg. “LDA on the Texts of Harry Potter” Medium, 6 December 2018, https://towardsdatascience.com/basic-nlp-on-the-texts-of-harry-potter-topic-modeling-with-latent-dirichlet-allocation-f3c00f77b0f5. Accessed 5 May 2022.