A PROJECT REPORT ON

Spotify Analysis

BY

Group No. : 15

Dev Barbhaya

Shatrughna Chaurasia

Akash Rawat

Aditya Mishra

Under the esteemed Guidance of

Prof. Dootika Vats

Acknowledgement

First and foremost, we would like to thank Prof. Dootika Vats, our academic and project instructor for MTH208A Data Science, for her guidance and patience throughout this project. The illuminating ideas from each meeting, although at certain stage made the project seem endless, turned out to be the most fun and rewarding part of this exploration. It has been a great learning experience and has also provided us with a practical insight of the theoretical knowledge gathered during the course lecture.

CONTENTS

“WITHOUT MUSIC, LIFE WOULD BE A MISTAKE.”

— **Friedrich Nietzsche (Twilight of the Idols)**

Introduction

These days we hear the music all the time. We wake up to the alarm of our favourite music, It motivates our workout, keeps us company on our commutes. Doesn’t matter what kind of music it is, music itself has the ability to affect our moods and our body, make us dance, feel horror, relives the memories with our loved ones. Music is an empathy-based language; there’s a reason why it’s often been compared to the songs from birds.

Music is the derivative of mathematical composition. It follows certain pattern, that’s why when we hear our favourite song, it relives the memory and the mood of the time when we first started liking that song that song. There are certain patterns that affects the 6 basic emotions in a human being. That are,

Sad - Sad music is thought to cause us to experience sadness, which is considered an unpleasant emotion. **K** notes are prevalent in sad music. Suspensions can also make things sound sad. Sad songs generally have **slow tempo** (less BPM). Alltogether, it induces a sense of suffering. But, studies have shown that we may actually feel positive emotions when we listen to sad music.

Joy- These are pleasant songs that brightens up our face and lightens up our mood. **A, B, G, H** notes comprise of joyous music. These songs are energetic, they are **up** tempo (mildly high BPM) , with very positive **valence .**

Angry -These are songs that provides a sense of aggression to the listener. They can contain violent song lyrics to increase negative emotions and thoughts that can lead to aggression. **B Major** is majorly used in angry music to invoke uncontrolled passions such as Anger, Jealousy, Fury, Despair, Burdened with negative energy. **Loudness** is high , along with **negative valence**

Fear - Scary movies seems be to nothing without audio effects . The squeaking of doors gives us chills. The quietness and sudden increase of tempo , startle us. **E flat minor** is the blackest of keys, having “feelings of anxiety and the soul's deepest distress.” In fear music loudness varies a lot , and **valence** is mostly **negative** .

Disgust – It is music that is aversive, repulsive and/or toxic . It the hardest emotion to represent in music . It’s **valence** is **negative** , and

Surprise - Uncertainity and surprise always enhance our enjoyment of music . **Tempo** changes drastically throughout the song , with generally a **positive valence .**

Problem statement

Our project aims at analyzing the overall nature of how music affect the way we think, feel, and behave. It may be identity development, emotional mimicry, memories etc. through the use of certain notes, keys, it's loudness, speechiness, valence and tempo. And also finding the emotional and empathic traits , mood , likings , daily routine of the user by extracting user's Spotify data like his favourite artist , most played track , total duration of songs played , lyrics of the tracks (by words classified as per emotions) etc. .

Music Attributes

The dataset gathered in the Spotify_API.R code of this project contains the song attributes. Before performing data analysis it is necessary to understand those features individually. The song attributes in the dataset are explained below:

Tempo: It is the speed of a particular track, how fast the contents in a track is played , basically calculated by beats per minute (BPM) . For e.g. A Rap track will have a very high pace and thus a very high tempo .

Energy: The intensity and impact that a track can generate in a person or audience is the energy of the track . It is typically measured between 0 to 1 . More energetic the track is, more closer it's value to 1 . For e.g. The tracks played in the EDM night at Udghosh, can easily be valued near 1 .

Danceability: How nice it is to dance in your room to your favourite Spotify playlist. Dancebility of a track describes just that. That how often will it make the listener to tap their feet and dance . Tracks are rated from 0 to 1 in terms of danceability. Higher the value more suitable the song is for dancing. For e.g. Daler mehendi songs always sets up the mood for dance parties.

Loudness: Remember the loud Heavy metal rock songs that one of our elder cousins played to annoy us all. Loudness measures how loud a song is (NO BRAINER) . It's values are averaged across the entire track and ranges from -60 to 0 DB. Higher the value, the louder the song.

Valence: There is definitely a reason why the soundtrack for a horror film is nothing like the one for a Romcom. The positive (e.g. happy, cheerful, euphoric) or negative (e.g. sad, depressed, angry) emotions a track can invoke is generally measured by it's valence . It is measured in the scale of 0 to 1. More positive the impact of a track is , more closed it is to 1.

Liveness: It detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides a strong likelihood that the track is live. For e.g. Asees kaur concert at IIT Kanpur was easily above 0.8 .

Acousticness: Songs with higher acousticness are more likely to use acoustic and non-electronic instruments. It is a confidence measure from 0.0 to 1.0 . 1.0 represents high confidence the track is acoustic.

Speechiness: It detects how often the spoken words are used in an audio file. The more the lyrics or words , the closer to 1.0 the attribute value. Above 0.66 it describe tracks that are probably made entirely of spoken words. Between 0.33 and 0.66 describe tracks that may contain mixture of music and speech .Values below 0.33 most likely represent that majority of that recording is instrumental .

Mode: Songs can be classified as major and minor. 1.0 represents major mode and 0 represents minor.

Key: Key is the pitch, notes or scale of song that forms the basis of a song. There are 12 keys that ranges from 0 to 11.

Data type	**Whats the Data**
	Playlist gives a collection of song by one or more artist compiled by the user. A summary of the playlists created or saved, and any songs saved, including: 1) Name of playlist. 2) Date the playlist was last modified.

Data type	Whats the Data
Playlist	3) Names of songs contained within the playlist. 4) Names of artists for each song. 5) Names of albums or episodes (if podcasts). 6) Local track name, if the user uploaded locally saved audio to be played on Spotify service. 7) Any descriptions added by the user to the playlist. 8) The number of followers the playlist has.
-------------	-------------
Streaming History (audio, video, and podcasts)	**StreamHistory** gives a list of tracks, streamcounts(one stream count is when user played track for 30 seconds or more) etc. listened to by the user in the past year, including: Date and time of when the stream ended . Name of artist for each stream (e.g. the artist name if a music track). Name of tracks listened by the user and msplayed on a track. A list of items (e.g. songs, videos, and podcasts) listened to or watched in the past year, including: 1) Date and time of when the stream ended in UTC format (Coordinated Universal Time zone). 2) Name of "creator" for each stream (e.g. the artist name if a music track). 3) Name of items listened to or watched (e.g. title of music track or name of video). 4) “msPlayed”- Stands for how many mili-seconds the track was listened to.
-------------	-------------
Your Library	Library is **everything user has saved on Spotify**—his/her playlists, the streaming radio stations user followed, the songs downloaded, and the artists and albums that comprise user’s saved stuff. A summary (at the point of the date of the request) of the content saved in Your Library (songs, episodes, shows, artists, and albums), including: 1) Entity names. 2) Album & Show names. 3) Creators. 4) Item’s Uniform Resource Identifiers (URIs).
-------------	-------------
Search queries	SearchQueries gives information on what user has search in his/her spotify account. A list of searches made, including: 1) The date and time the search was made. 2) Type of device/platform used (such as iOS, desktop). 3) Search Query shows what the user typed in the search field.

Data type	**Whats the Data**
	4) Search interaction URIs shows the list of Uniform Resource Identifiers (URI) of the search results the user interacted with.
-------------	-------------
Follow	Follow gives the name of artist’s account that user follow. This includes (where available) at the point of the request: 1) The number of followers the account has. 2) The number of other accounts this account is following. 3) The number of other accounts this account is blocking.
-------------	-------------
User Data	UserData is collection of user's personal data such as what songs they have played and what playlists they have made etc.. This includes (where available): 1) Spotify username. 2) Email address. 3) Country. 4) Created from Facebook - This shows true if the account was created via Facebook. 5) Facebook user ID - This is included if the user has turned on Facebook data processing and linked their Spotify account by signing in using Facebook log-in or created their Spotify account via Facebook. 6) Preferred locale. 7) Date of birth. 8) Gender. 9) Postal code. 10) Postal address. 11) Mobile number. 12) Mobile brand. 13) Account creation - This is the user’s date of registration. 14) Family Plan.
-------------	-------------
Inferences	Inferences has all the data about user’s listening habits in a file called inferences.json. We draw certain inferences about your interests and preferences based on your usage of the Spotify service and using data obtained from our advertisers and other advertising partners. This includes a list of market segments with which you are currently associated. Depending on your settings, this data may be used to serve interest-based advertising to you within the Spotify service.

- **Interesting questions to ask:**
- How song attributes are related to song lyrics?
- Does playback activity of a user shows any pattern?Is it cyclic?
- Which country has most no of upcoming artists in past years?
- Do lyrics also convey emotions or is it only song attributes?

Extraction of Data

Our project data is mainly extracted from SPOTIFY API and Genius API.

Challenges faced while extracting data :

- The developer account we made for twitter took 9 hrs to get authenticated.
- It took many attempts to enter correct scope and URI's for oAuth authentication of API keys.
- For twitter oAuth authentication we got this message many times

"Using direct authentication"

Error in check_twitter_oauth() : OAuth authentication error:

This most likely means that you have incorrectly called setup_twitter_oauth()'

Then we took help from William cheng over 'Stackoverflow' to resolve the authentication issue .

a API Tokens

Spotify API_token – BQAIa7roF-dAsNA_Os_McuRn48qxAsQIv9MEPhQxdtjeVbSt-sJKoLvFMpv5bM6e2JquJmvlKDrU

Steps :

First, we created a Spotify user account.

Then, we went to the Dashboard page at the Spotify Developer website and, logged in. Accepted the Developer Terms of Service and that completed the account set up.

We created and registered a new application to generate valid credentials. Which we will need these credentials later to perform API calls. Any application can request data from Spotify Web API endpoints and many endpoints are open and will return data without requiring registration. However, if your application seeks access to a user's personal data (profile, playlists, etc.) it must be registered.

Then we installed Node.js, to create a project folder for our application .

Next, we asked to get authorize access to:

After that, we started the process of authenticating. By entering the client ID, scopes, and redirect URI(the first call).

Entering authorization code returned by the first call and the client secret key. We got access token and also a refresh token.

After both calls were completed, we had authorized the app for access, the application had the 'access_token' it needs to retrieve the user data from the Web API.

After third call we got a new access token that we can issue when the previous has expired.

Twitter API_token –1555192646402211846-KnAbM7uyWdIJOolqafBDe40FruT53t

Steps :

We applied for a standard developer account, which enables access to tweets from the last week as well as tweets in real time.

And got it approved by Twitter(which took 9 hrs).

Then we opened Twitter app dashboard and went to "Keys and Tokens" page.

We clicked 'Create' under the "Access token & access token secret" section

And we generated API keys and bearer token for twitter.

Genius API_token – uYt322MvKqvgma0zdWAC3j283DYTCgp68anP0v0JiMDDi5DVeWzKnmeFKVHEPP7X

Steps :

We signed up for a Genius account, which is required to gain API access.

Then we went to https://genius.com/api-clients, where we clicked the button that says "New API Client."

After clicking "New API Client," we filled out a form about the "App" that we need the Genius API for.

Then we were given a series of API Keys: a "Client ID" and a "Client Secret."

To generate your "Client Access Token," which is the API key that we used, we clicked "Generate Access Token".

Websites we used:

Spotify newsroom– To scrape name of the artist , their track-name ,album-name , duration of track , no. of streams , upcoming artists , playback activity.

As Spotify newsroom has one of the largest collection of songs data ,and data is not biased because large number of users,works with spotify api, so we found spotify newsroom reliable for our project data.

URLs : https://newsroom.spotify.com/2021-12-01/what-the-world-streamed-most- in-2021/

https://newsroom.spotify.com/2020-03-09/36-new-artists-around-the-world-that-are-on-spotifys-radar/

Data we got: We used https://newsroom.spotify.com/2021-12-01/what-the-world-streamed-most-in-2021/ to scrape the **most streamed artists in different zones of the world** across Spotify. The data we got was text data, and contained 40 elements, but it was a mix of artist and tracks ,to sort them out we used (newsroom_spotify[1:5]) and (data.frame(most_streamed_global here we used the information of first five rows looking at the website) to get a dataframe of most streamed artists in different zones of the world. And (newsroom_spotify[6:10]) , (data.frame(most_streamed_US) here we used the information of next five rows looking at the website) to get a dataframe of most streamed artists in USA. We got top 5 most streamed artists in different zones of the world and top 5 in USA. We discarded rest of the rows of data, as they were titles or albums, and we wanted artist names only.

Data we got: We used https://newsroom.spotify.com/2020-03-09/36-new-artists-around-the-world-that-are-on-spotifys-radar/ to scrape the **upcoming artist names** across Spotify. The data we got was text data. And we stored them in a list. The list we got contained 35 observations under 2 variables namely X1 and X2 where X1 corresponds to **Artist's country** and X2 corresponds to **Artist name**. Then we created a separate list of artist by their country ,by looking at row of their respective countries and then slicing the list to obtain many list which contained artist from their respective countries.

S.no.	Country	No. of upcoming artists
1	Argentina	1
2	Australia	1
3	Austria	2
4	Brazil	1
5	Columbia	1
6	France	4
7	Germany	1
8	India	3
9	Indonesia	2
10	Japan	3
11	Mexico	1
12	Netherlands	1
13	Panama	1
14	Philippines	2
15	South Africa	1

S.no.	Country	No. of upcoming artists
16	Spain	5
17	Taiwan	1
18	UAE & Lebanon	1
19	UK	1
20	US	1

Here, is a table of countries with the no. of upcoming artist from that country.

Udiscovermusic –

It has abundant of data which it collects from sources like analytics providers, social networks such as Facebook and Twitter, search information providers. And also it provided a clean list of all the artist, so to do less work in sorting and cleaning of data.

URLs : https://www.udiscovermusic.com/artists-a-z/

Data we got: We used https://www.udiscovermusic.com/artists-a-z/ to scrape the **artist names** across Spotify. The data we got was text data. And we stored them in a list. The list we got contained 564 observations under variable name "Artist".

Chartmasters – To get a list of most streamed artists across Spotify and most followed artists across Spotify.

Data compiled on chartmasters was well organized and reliable.

URLs : https://chartmasters.org/most-streamed-artists-ever-on-spotify/

https://chartmasters.org/spotify-most-followed-artists/

Data we got: We used https://chartmasters.org/most-streamed-artists-ever-on-spotify/ to scrape data of **most streamed artists across Spotify**. Data we got was numerical and text data. As we needed only the artist names, so we sliced the list (chartmasters[[]][]), and got a list containing only artist names. The list we got contained 100 observations under variable name "Artist".

We used https://chartmasters.org/spotify-most-followed-artists/ to scrape data of **most followed artists across spotify**. Data we got was numerical and text data. As we needed only the artist names, so we sliced the list (chartmasters2[[]][]), and got a list containing only artist names. The list we got contained 200 observations under variable name "Artist".

After getting the data , we merged all the objects which contained 723 observations which provides artist information. The list obtained was saved in **Artist_list.Rdata** in our github repository.

Library used:

jsonlite : A Simple and Robust JSON Parser and Generator for R. A reasonably fast JSON parser and generator, optimized for statistical data and the web. Offers simple, flexible tools for working with JSON in R, and is particularly powerful for building pipelines and interacting with a web API.

Functions used : fromJSON() – It is used to read of JSON file .

rvest : It helps you scrape (or harvest) data from web pages .

Functions used : read_html() reads in all the HTML for the page by requesting data from the computer server that contains it. It takes website URL as argument. It returns a list object that contains the tree-like structure

html_table() returns the data in tibbles and gives us a dataframe. To process the data in meaningful manner.

html_text() to extract on text data.

html_elements()

dplyr : It provides many tools for the manipulation of data in R. The dplyr package is part of the tidyverse environment. It contains functions like arrange(), filter(), mutate(), rename() , select() .

Functions used : mutate() adds new variables and preserves existing ones.

filter() extracts rows of our data by a logical condition.

arrange() orders data sets according to a certain column of our data.

group_by() takes an existing tbl(subclass of data.frame

in tibble) and converts it into a grouped tbl where operations are performed "by group"

summarize() reduces a data frame to a summary of just one vector or value.

tidyverse: The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

Ggplot2: It is used to visualize the data given to it. To make the data easy to understand. It contains functions like aes(), geom_col(), ggtitle(), ggplot(), ggtitle() etc..

Functions used: ggplot() creates a coordinate system that we can add layers to.

ggplot(aes(x = date, y = hours, group = artistName))

aes() provides a way that its inputs are quoted to be evaluated in the context of the data. This makes it easy to work with variables.

geom_col() used to create bar charts of the data given, here heights of the bars represents values in the data.

geom_col(aes(fill = minutesListened))

scale_fill_gradient() provides a way to fill lines,bar, boxes or whatever we are representing our data with different colors. So, it can be easy to differentiate between low-high or sparsely-dense regions plotted.

scale_fill_gradient(low = "blue", high = "red")

labs adds labels to x and y axis.

labs(x= "Artist", y= "Minutes of music playback")

ggtitle() gives title to the plot.

ggtitle("Most Heard Artist", "> 3 hours listened to")

Knitr: The R package knitr is a general-purpose literate programming engine, with lightweight API's designed to give users full control of the output without heavy coding work.

spotifyr : It is an R wrapper for pulling track audio features and other information from Spotify's Web API in bulk. By automatically batching API requests, it allows you to enter an artist's name and retrieve their entire discography in seconds, along with Spotify's audio features and track/album popularity metrics.

geniusr : To scrape lyrics of a particular track, to find it's wordliness and speechiness, valence, temp, liveness, instrumentalness, energy, danceability, acousticness.

Functions used : get_spotify_access_token() to get spotify user token.

get_spotify_authorization_code() to get spotify authorization code.

get_user_playlists() returns a dataframe of playlists for a given Spotify username.

get_albums() to get albums from an artist using their ID .

get_tracks() to get Spotify catalog information for a single track identified by its unique Spotify ID.

get_track_audio_features() to get audio feature information for a single track identified by its unique Spotify ID.

twitterR : twitteR is an R package which provides access to the Twitter API. Most functionality of the API is supported, with a bias towards API calls that are more useful in data analysis as opposed to daily interaction. Our initial plan was to also extract the comments and reaction of people on a track by extracting comments with that track tag. Later we dropped this idea as we wanted to have a more scientific way to check impact of song on human emotions. So, we didn't use this library.

Personal Spotify data

First, we loaded the libraries (spotifyr), (ggplot2 – for plotting) that will be used.

We found that the history of the user contained variables like “endTime”, “artistName”, “trackName”,“msPlayed”.”searchQueries”,“playlist” , “inferences”, “identifiers”, “Follow”,“userData”, “Streamhistory”.

StreamHistory gives a list of tracks, streamcounts(one stream count is when user played track for 30 seconds or more) etc. listened to by the user in the past year, including:

Date and time of when the stream ended .

Name of artist for each stream (e.g. the artist name if a music track).

Name of tracks listened by the user and msplayed on a track.

We only used **streamhistory** data of user’s spotify account, as we were making analysis on user’s account history, and **streamhistory** provided us with all the data we needed, such as songs played with artist name, no.of streams , total time played over different periods of time etc.

Endtime denotes the time when the stream is ended.

ArtistName denotes the name of the artist for a track.

Msplayed how many mili-seconds the track was listened to.

SearchQueries gives information on what user has search in his/her spotify account.

Playlist gives a collection of song by one or more artist compiled by the user.

Inferences has all the data about user’s listening habits in a file called inferences.json.

Follow gives the name of artist’s account that user follow.

UserData collection of user's personal data such as what songs they have played and what playlists they have made etc..

By using user's **streamhistory** data we can make different analysis. Some of them are:

Which artist's songs are most played by user over a period of time?

Using the user's **streamhistory** data, we can find the artist of the respective songs in playlist. and then plot a bar graph of artist name vs no.of songs of that artist in user's playlist .

Which language songs, user plays most frequently?

We can define a time-period and extracting data from **streamhistory,** extract the language used in a particular song, to plot bar graph or heat map of the most frequently used language in user's id over the specified period of time.

On what dates did user listened to more or less music on Spotify?

First, we can define hours, minutes and seconds(variables). Then, by observing the behavior of the user's activity on Spotify throughout the year, per week for example, playback activity by time and time of day, and then plot it. Like we plotted a monthly graph which showed the amount of time the user has listened the music and at what time user listened, Playback activity by time and time of the day.

.

What were the artists you listened to the most on your Spotify?

We can set a minimum of playing hours and find out from there which artists you listened to the most. For example, we plotted a bar graph of the most listened to artist over a period of time by the user and the no. of minutes an artist track is played . Ritviz and Anuv jain were heard the most by the user accounting over 800 minutes. Listener might be used to lo-fi and acoustic track

At what time have you had the most playback activity on your Spotify?

By using spotifyr data(**streamhistory**) we can view by the hour of the day inclusive, the activity log of user's complete history with a heatmap, line graph to observe how user's habit has changed over time.

We can also create a bar chart to take a closer look in detail at the times of the day when there is a record of the highest activity on user's account.

After plotting we will be able to observe the specific time in which there are more accumulated minutes of reproduction.

What days of the week do user has the most playback activity on your Spotify?

We can answer this question by creating a heatmap by visualizing the relationship between day of the week and time of day, of the variables defined.

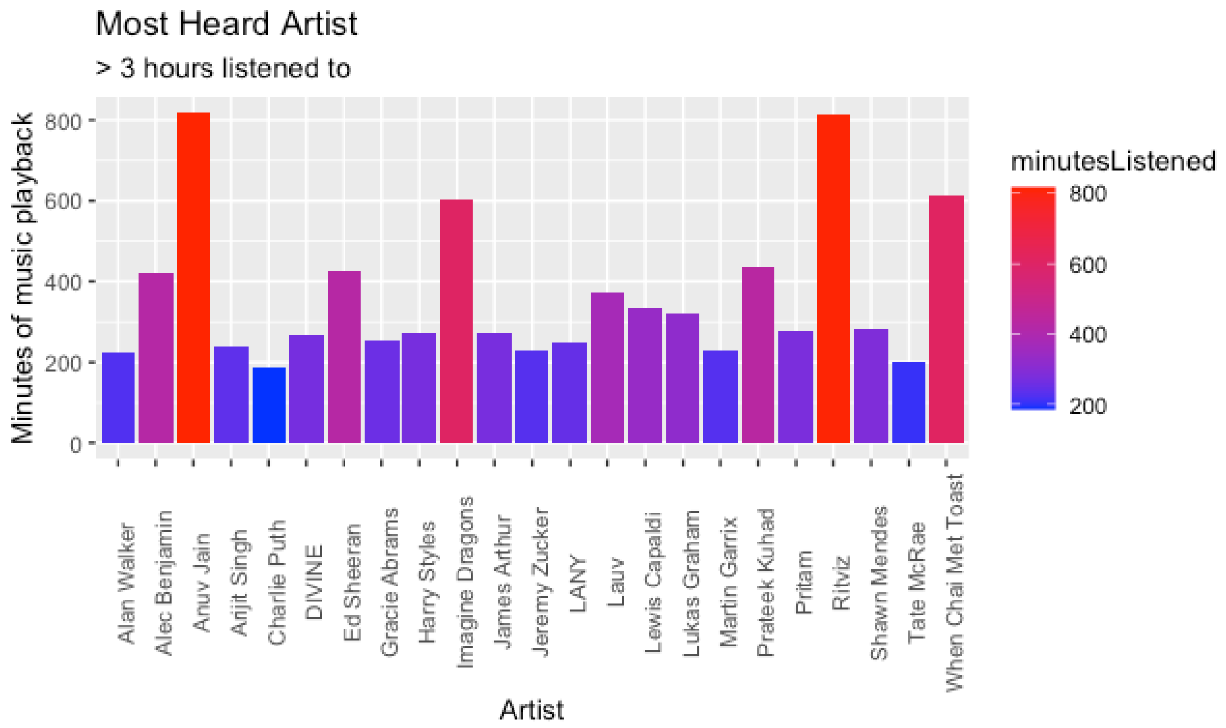
We will be able to observe which day (s) of the week mark the trend of greater activity in user's account.

Another way in which we can take a look at the details previously obtained could be by creating a line chart.

We can also plot graph where it shows which is the day when user has the least playback activity.

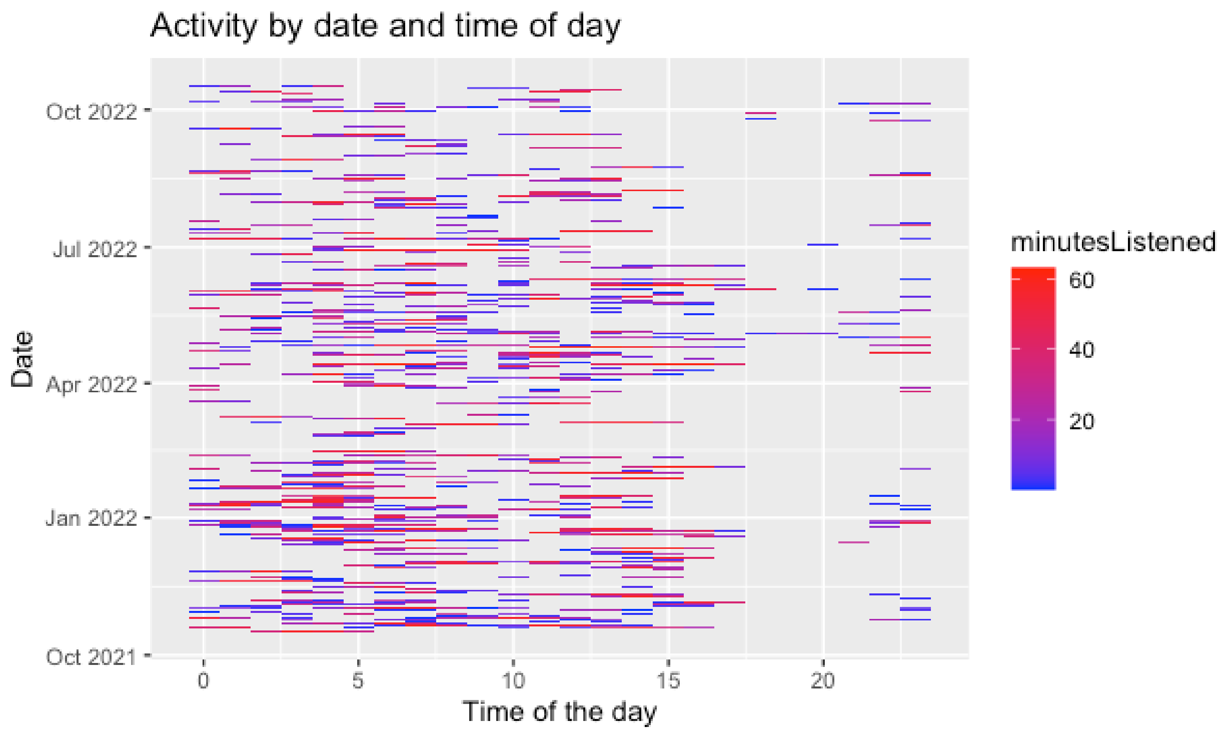
For example, we plotted line-chart of playback activity of user on weekdays and weekend and found that 4 am on weekend and 7 am on weekdays.

Visualization and inferences



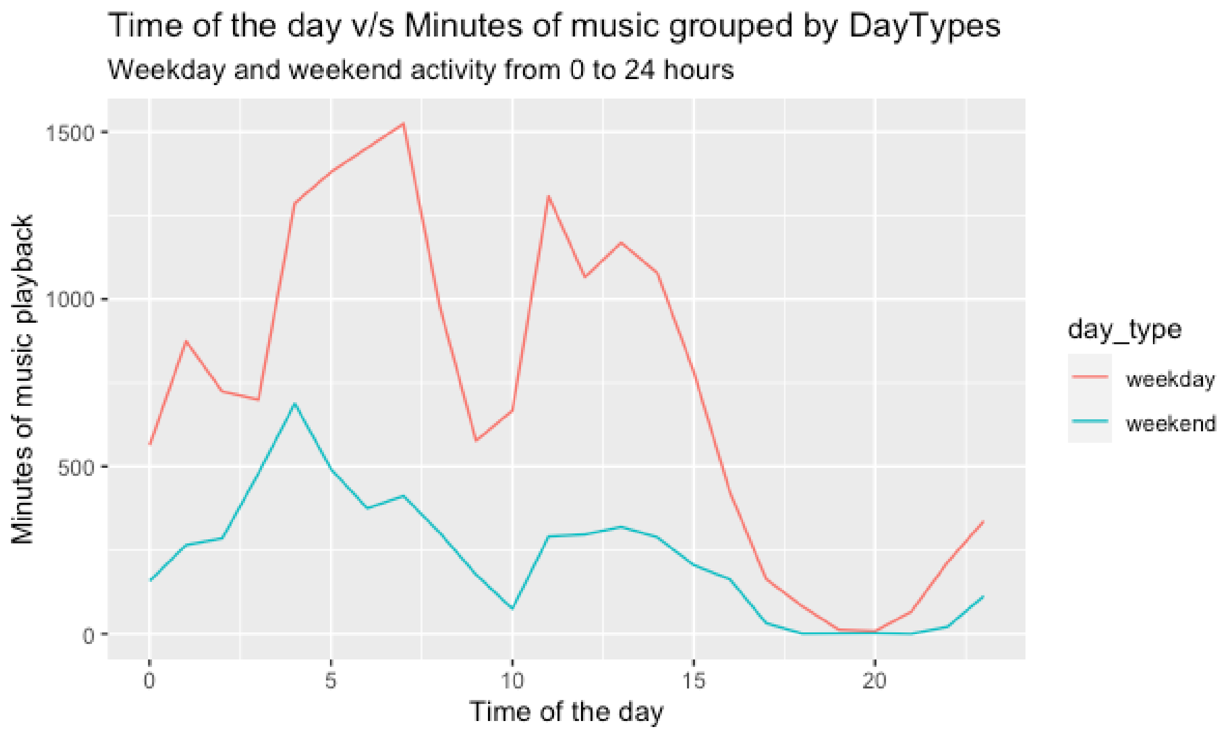
As the title(ggtitle()) suggests the above graph represents most heard artist. By plotting bar chart(geom_col()) of artist against minutes of music playback(aes(x = artistName, y = minutesListened)).

We can conclude that by the length of the bar user's favourite artists are **Anuv Jain** and **Ritviz**(over **800** mts) or as can be seen by "red" bar color(scale_fill_gradient(low = "blue", high = "red")).



Above plot is visualization of "Activity by time of the day" using line heatmap(geom_tile()) where x-axis(labs(x= "Time of the day", y= "Date")) represent "time of the day" and y-axis represents "date". Tiles ("specific date and specific time") with high density of streaming is colored "red" (scale_fill_gradient(low = "blue", high = "red")) compared to "blue" with less minutes of streaming by the user within that hour.

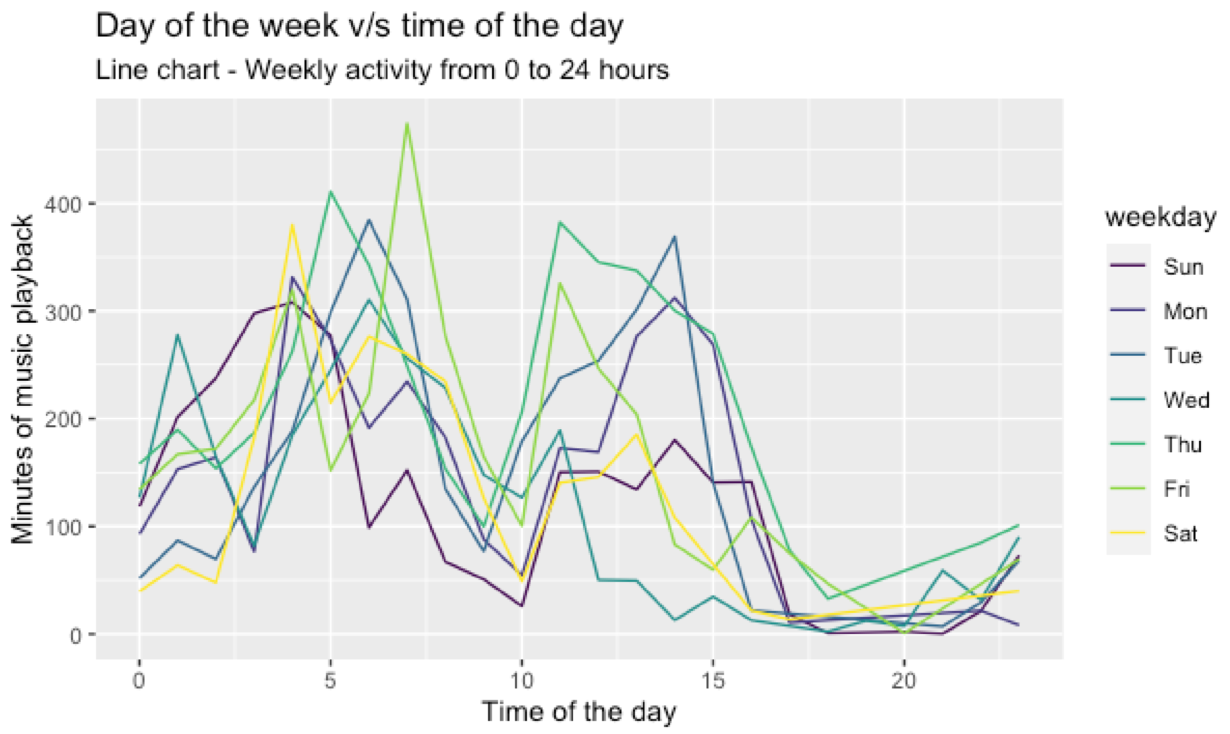
We can see at **0400-0500**hrs(tile **5** in along x-axis) y-axis is most congested. User hears songs between **0400**hrs and **0500**hrs at most days of the month. User might be a morning person or can be a night-owl(prevalent in IIT). Between **1700**hrs to **2100**hrs density of tiles are less, which shows less playback activity by the user during this period of time in a day.



Above plot is visualization of “comparison of total duration of songs played in weekend and weekdays” using line chart(geom_line()) where x-axis(labs(x= "Time of the day", y= "Minutes of music playback")) represent “time of the day” and y-axis represents “Minutes of music playback”. Playback time duration at particular time of the day on weekend and weekdays are represented by blue and red line respectively.

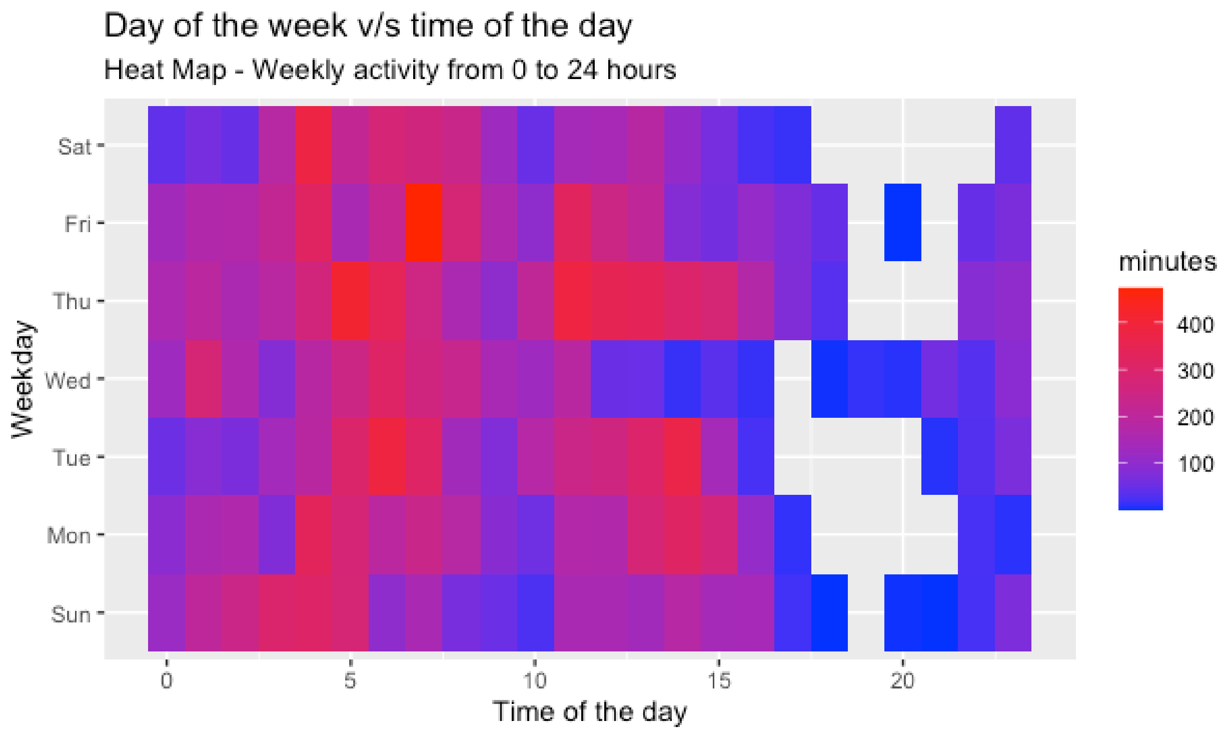
We can see at around **0730**hrs there is a peak in red line, we can conclude that

At **0730**hrs the user listen to song most, on weekdays. At around **0400**hrs there is a peak in blue line, we can conclude that at 0400hrs the user listen to song most, on weekends At round **2000**hrs user’s playback activity minimizes.



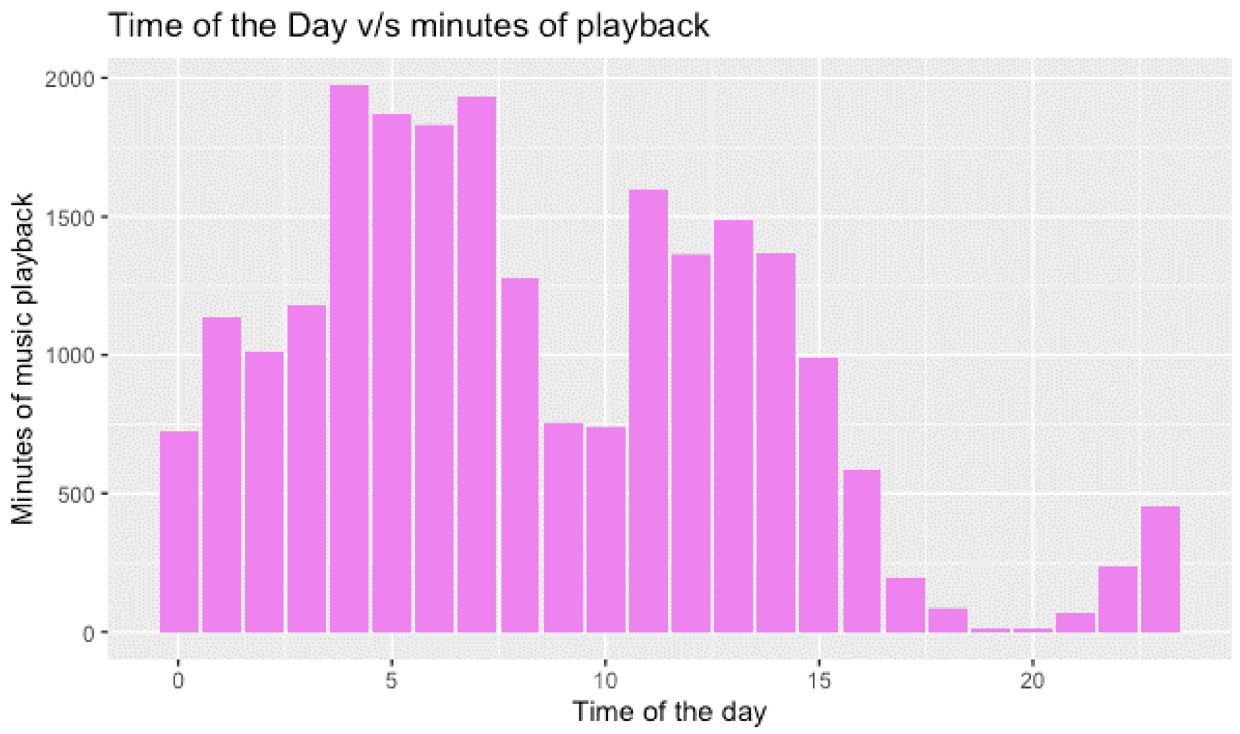
It is a line chart(geom_line()) of day of the week against time of the day (ggtitle("ggtitle("Day of the week v/s time of the day", "Line chart - Weekly activity from 0 to 24 hours"))representing playback time, where different colors are used to denote playback activity at different time of the day(aes(x = hour, y = minutes, color = day_type)) of different weekdays. Color of weekdays line are selected by default, and are denoted at the right side of the plot.

Peak of most of the days are around **0500** to **0730**hrs, between this time user listen to song more every day against other time period of the day. And as we speculated from the previous charts user's playback activity is lowest around **2000**hrs as we can see line reaching closer to x axis everyday.



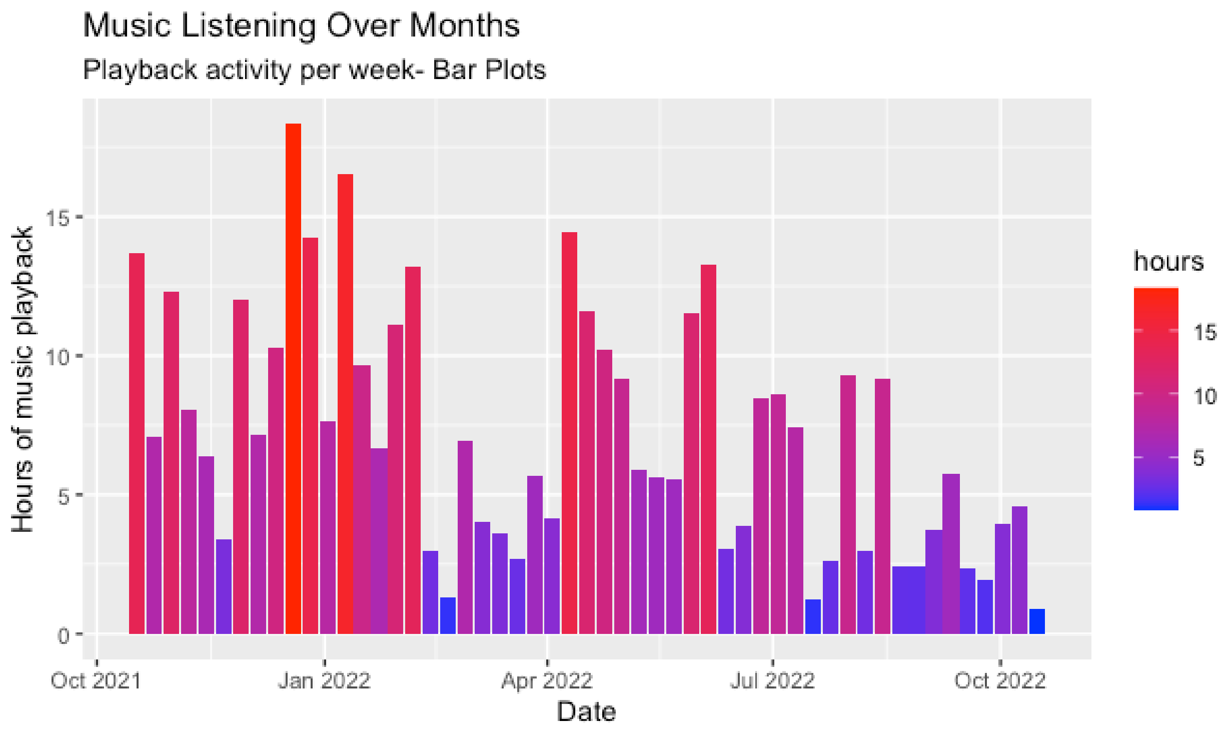
It is a heatmap(geom_line()) of day of the week against time of the day (ggtitle("Day of the week v/s time of the day", "Heat Map - Weekly activity from 0 to 24 hours")representing playback time, where different colors are used to denote playback activity at different time of the day(aes(x = hour, weekday, fill = minutes)of different weekdays. Color of blocks are selected by default, and are denoted at the right side of the plot. More minutes of playback in a block is closer to **red** and less minutes of playback are closer to **blue**.

Most of fully **red** colored blocks are around **0500** to **0730**hrs, between this time user listen to song more every day against other time period of the day. And as we speculated from the previous charts user's playback activity is lowest around **2000**hrs as we can see blocks closer to **blue** and **no** blocks representing less and no playback respectively.



As the title(ggtitle("Time of the Day v/s minutes of playback")) suggests the above graph represents relation between time of the day and minutes of playback. By plotting bar chart(geom_col()) of Time of the day against minutes of music playback(aes(x = hour, y = minutesListened, group = date)). Here group = date selects data of the interval of date provided and adds music activity of a particular time over a period of days.

We can conclude that by the length of the bar is highest at 0400 to 0800 between this time user listen to song more every day against other time period of the day. And as always user's playback activity is lowest around **1900 - 2100**hrs as we can see length of bar at that interval is shortest or close to none representing less and no playback activity.



The title(Music Listening Over Months", "Playback activity per week- Bar Plots- Line Plot) suggests the above graph represents minutes of playback activity over months. By plotting bar chart(geom_col()) of weeks against minutes of music playback(labs(x= "Date", y= "Hours of music playback")).And colors of bars represent amount of activity in that particular week with high playback activity closer to **red** and low playback activity closer to **blue**(aes(fill = hours)+ scale_fill_gradient(low = "blue", high = "red")).

Here, graph tells the cyclic nature of the user playback activity time all year round.

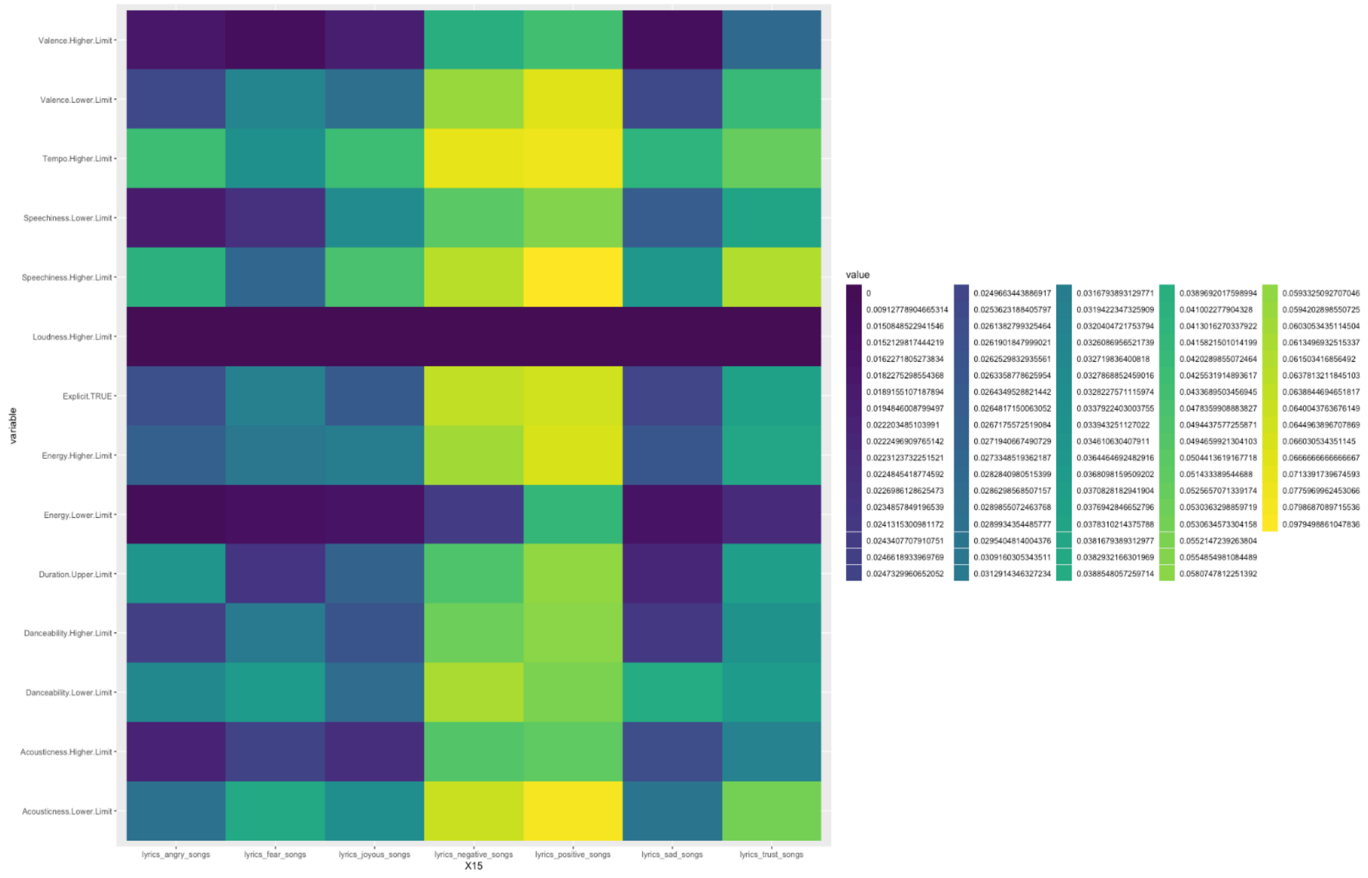
WordPlots

Lyrics analysis: Analysis of lyrics of songs with different components like high acousticness, low tempo, high danceability, high valence etc. were made, using libraries like “stats”, “ds4psy”, “quanteda”. After slicing different components for eg. Acousticness 0.01 from Artist list and using function (text_to_words(), which splits a string of text x (consisting of one or more character strings) into a vector of its constituting words).

Heatmap of word category and lyric correlation.

Graph below shows the heatmap(geom_tile()) of word category such as Valence high limit, Valence low limit, tempo high limit, Speechiness high limit, Loudness high limit etc. correlation with lyrics types such as lyrics of angry song, fear song, joyous song, negative song, trust song etc.. With yellow color representing highest value for correlation and violet for lowest value of correlation (scale_fill_viridis(discrete = TRUE)).

Library used- ggplot2, viridis.



Word plots with frequency word plot of song lyrics of all artist

Below are the words frequency bar graphs(barplot()) and word plot of all the tracks which had a specific criteria, like **acousticness 0.01**, **instrumentalness 0.995**, **valence 0.0**, **tempo 225.0** etc..

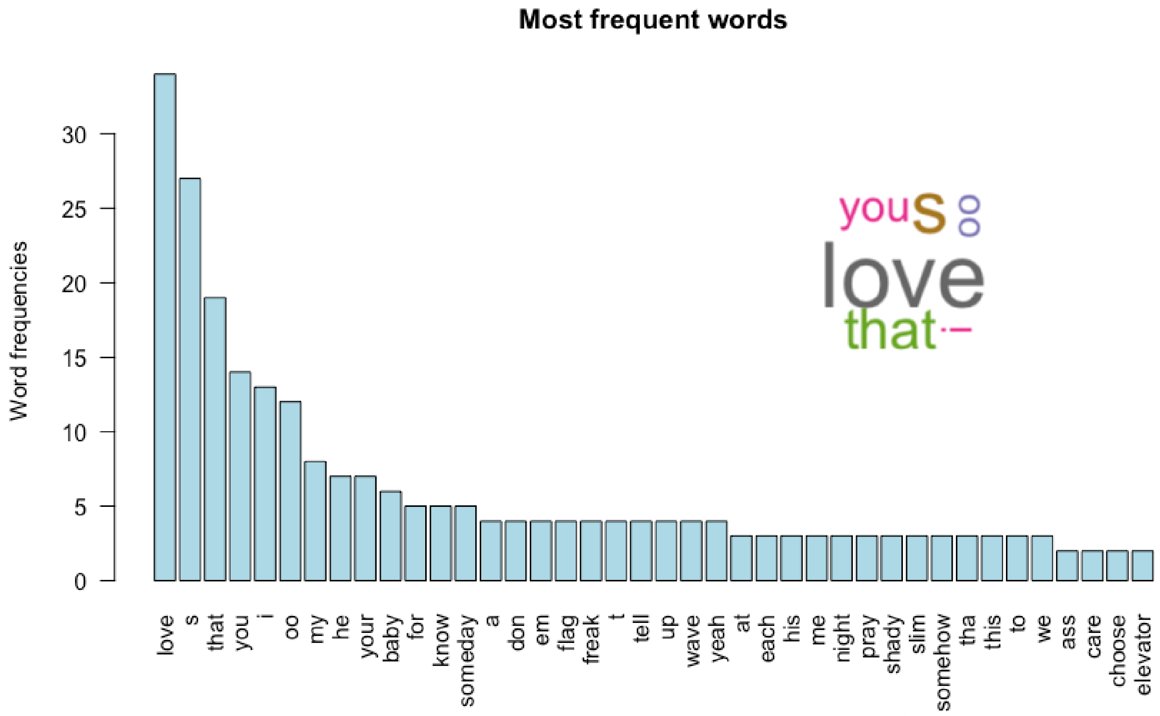
Library used are wordcloud, RColorBrewer, ggplot2, dplyr, tidyverse, graphics.

For word plot we took a little help from this idea -

http://www.sthda.com/english/wiki/text-mining-and-word-cloud-fundamentals-in-r-5-simple-steps-you-should-know

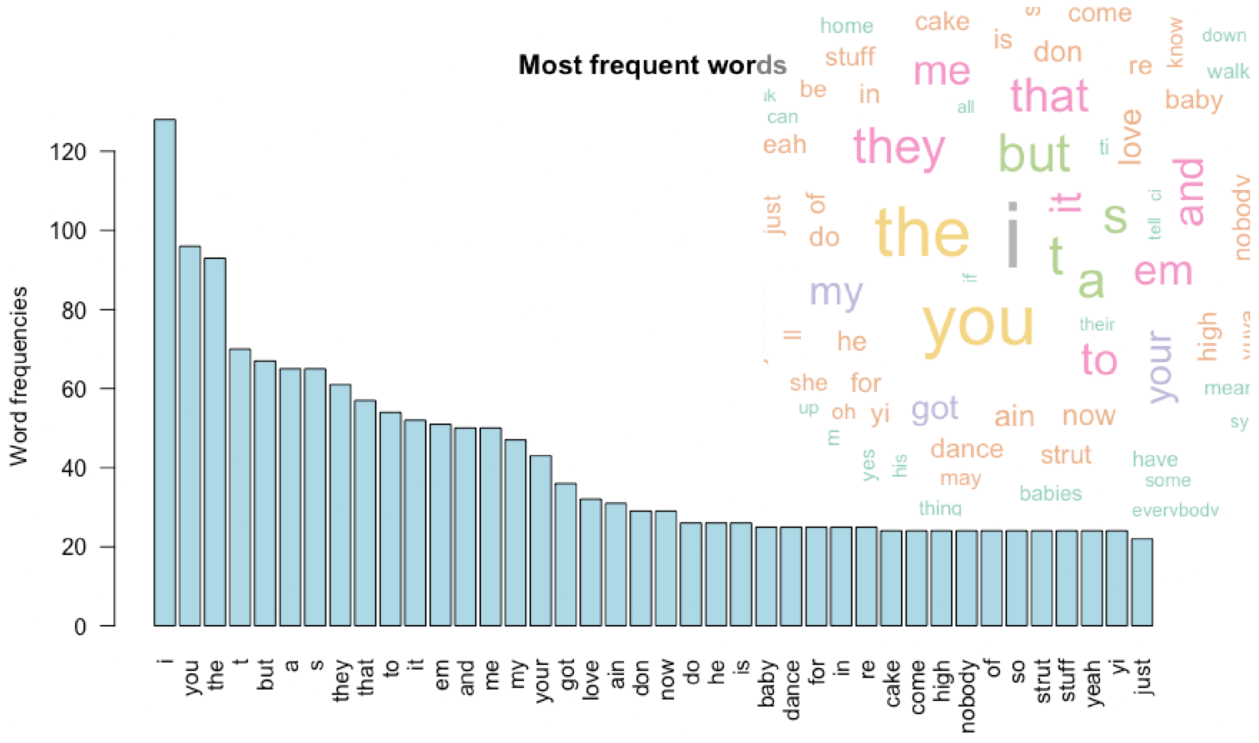
Acousticness 0.01

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having acousticness 0.01 , means songs which are produced with high use of electrical equipments. And we found out that lowest acousticness songs contain "love", "s", "that".



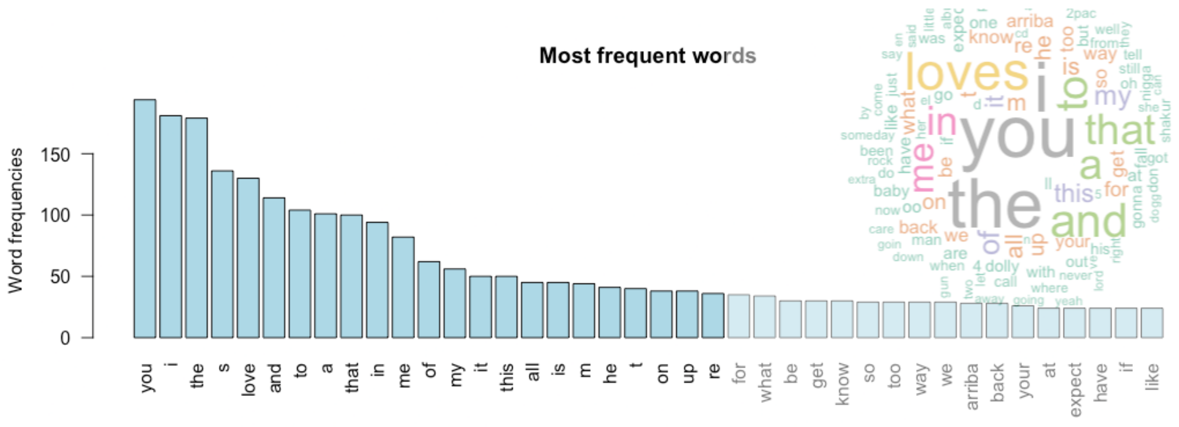
Acousticness 0.995

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having acousticness 0.995 , means songs which are produced without any use of electrical equipments. And we found out that highest acousticness songs contain "I", "you", "the".



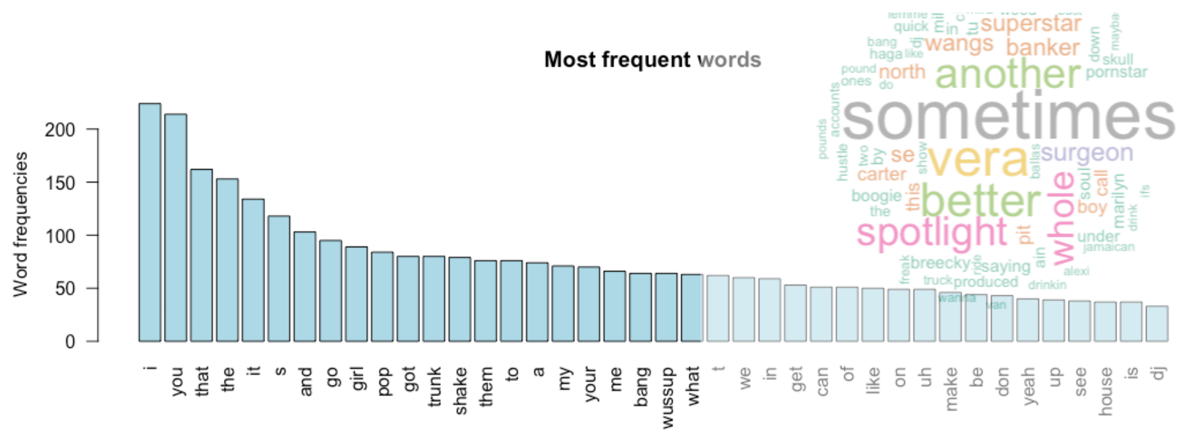
Danceability 0.01

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having danceability 0.01 , means which are least danceable. And we found out that least danceable songs contain “you”, “I”, “the”.



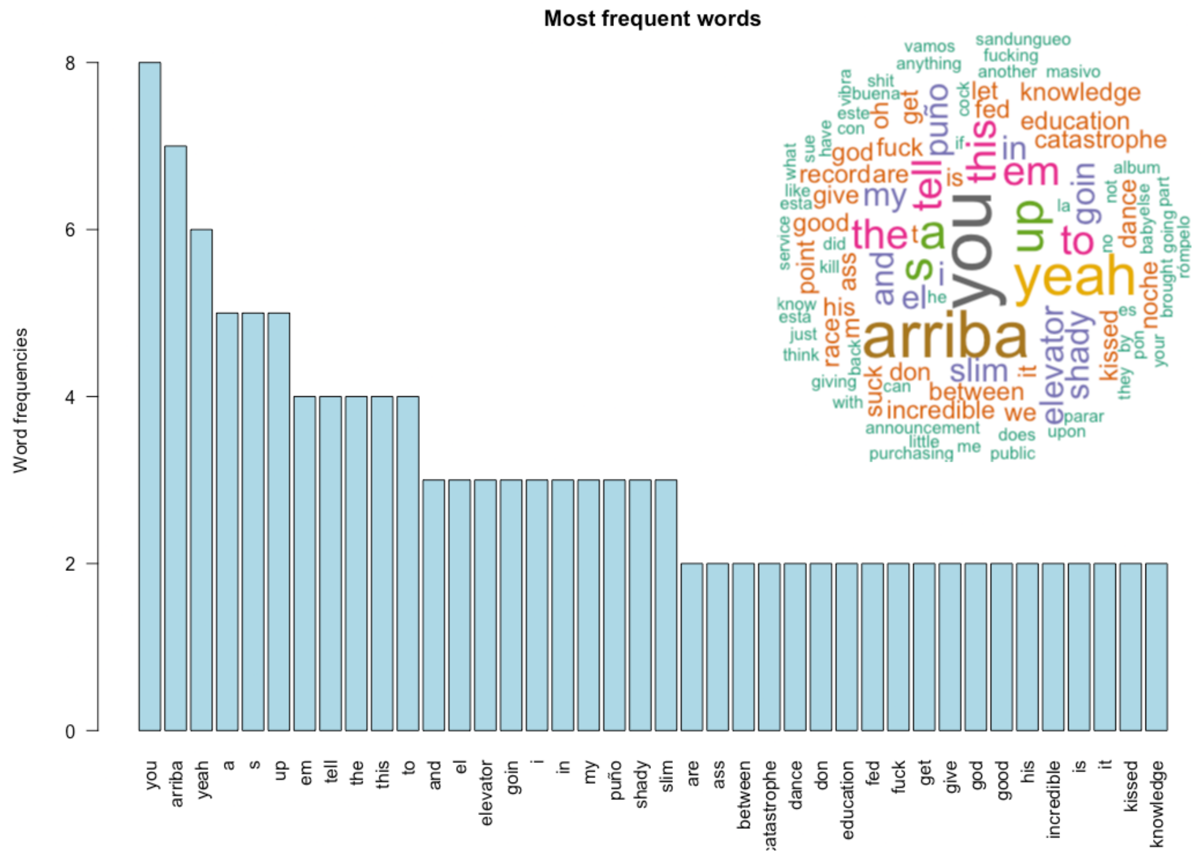
Danceability 0.97

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having danceability 0.97 , means which are most danceable. And we found out that most danceable songs contain “you”, “I”, “that”.



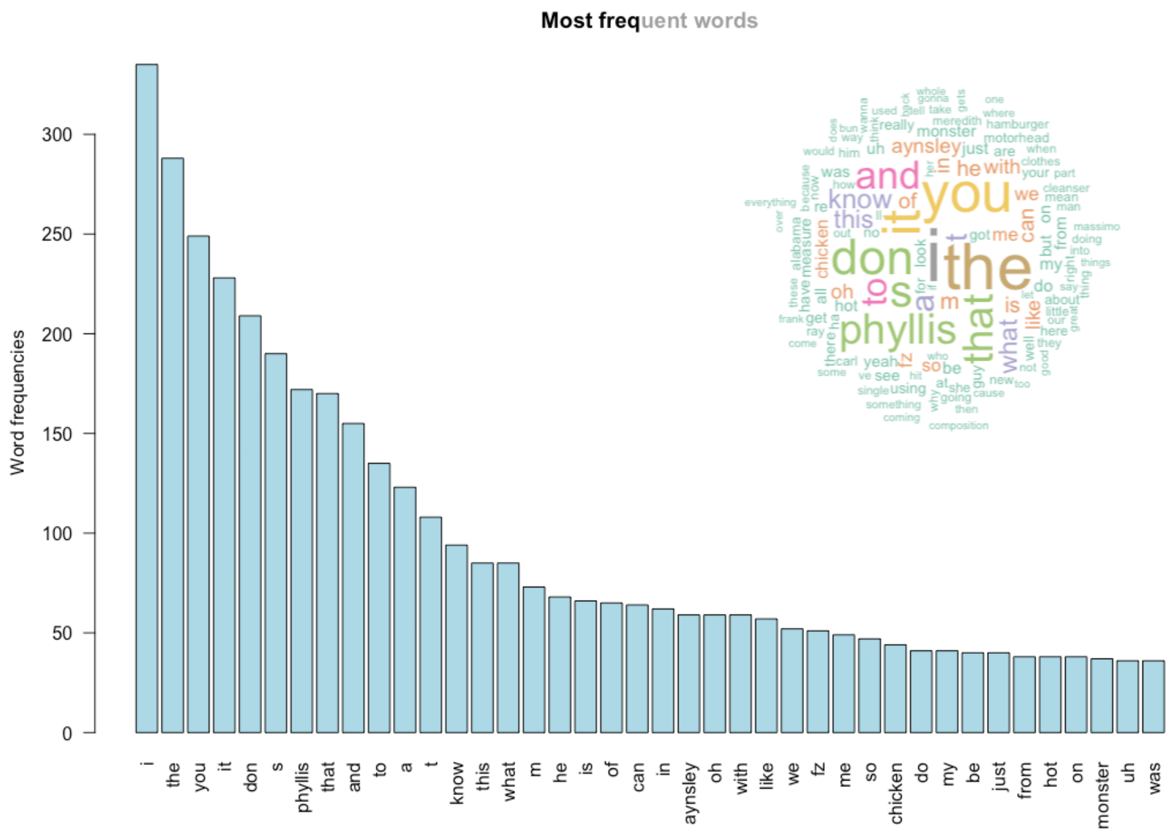
Duration 5000

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having duration 5000 , means which are of lowest duration. And we found out that lowest duration songs contain “you”, “Aribba”, “yeah”.



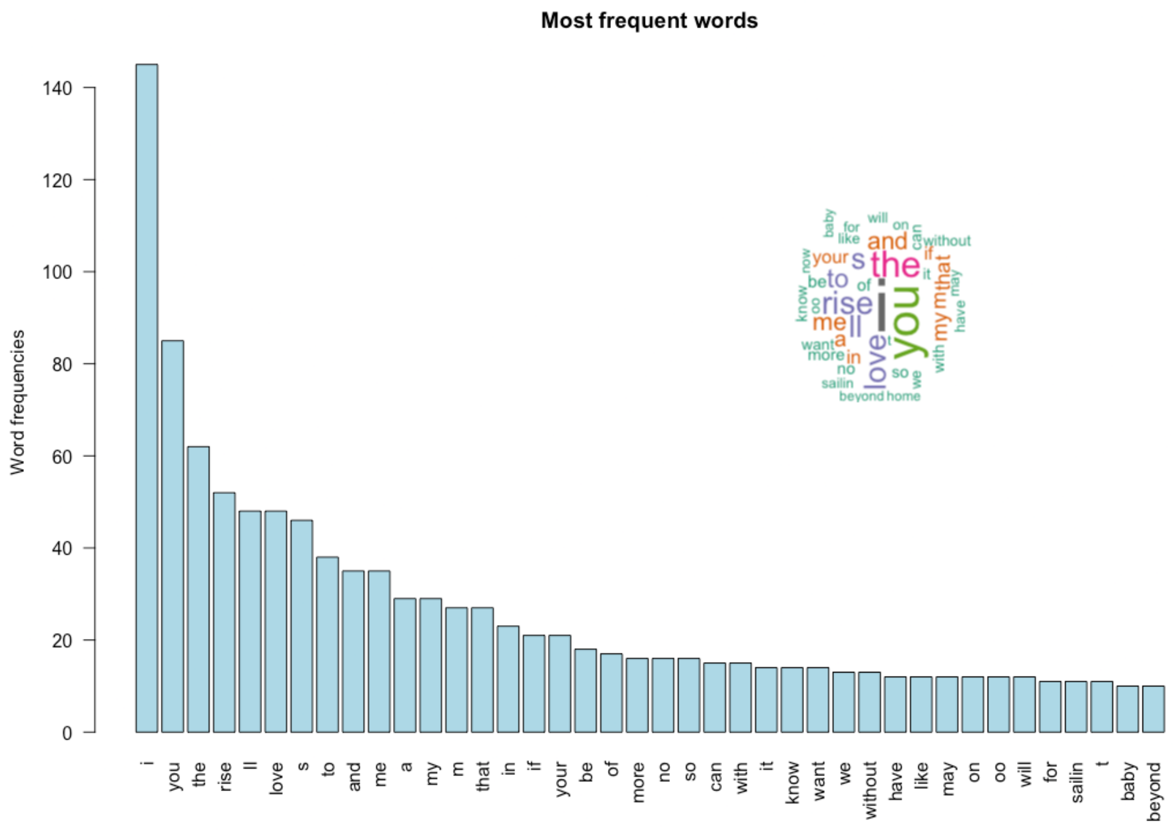
Duration 2000000

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having duration 2000000 , means which are of highest duration. And we found out that highest duration songs contain “you”, “I”, “the”.

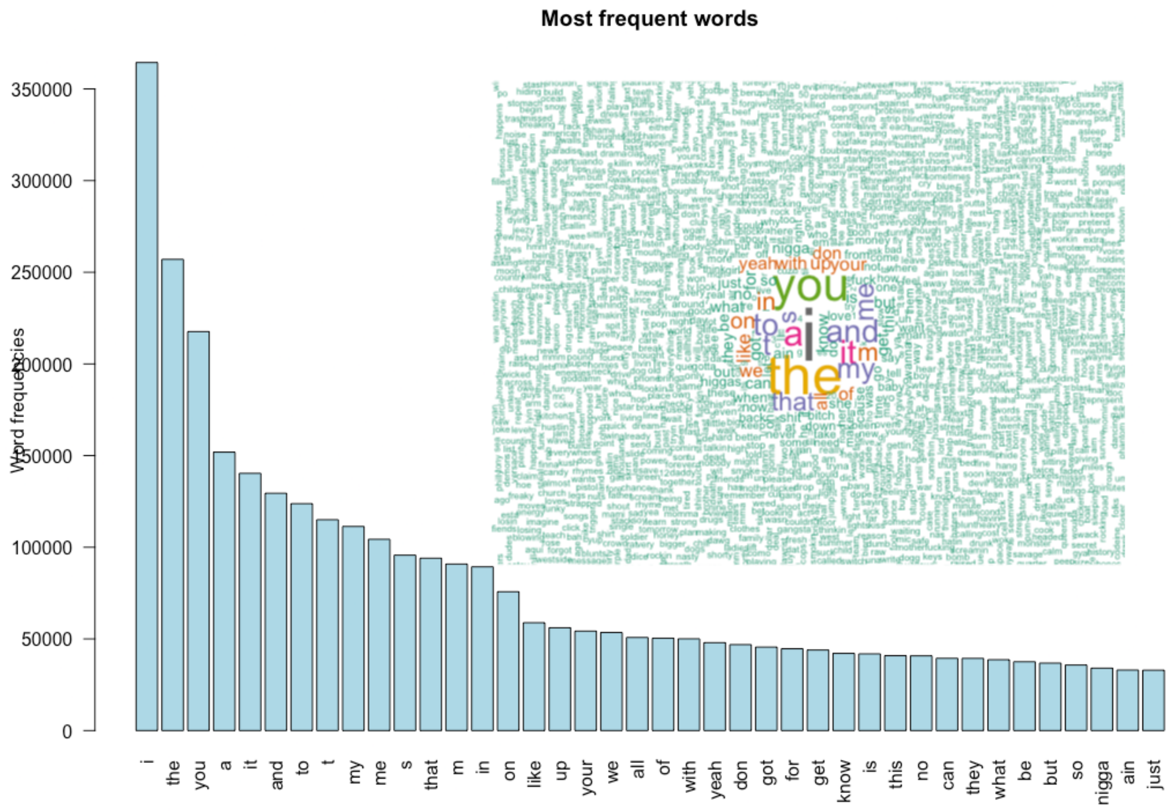


Energy 0.00

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having energy 0.00 , means which are of lowest intensity on user. And we found out that lowest energy songs contain “you”, “I”, “the”.

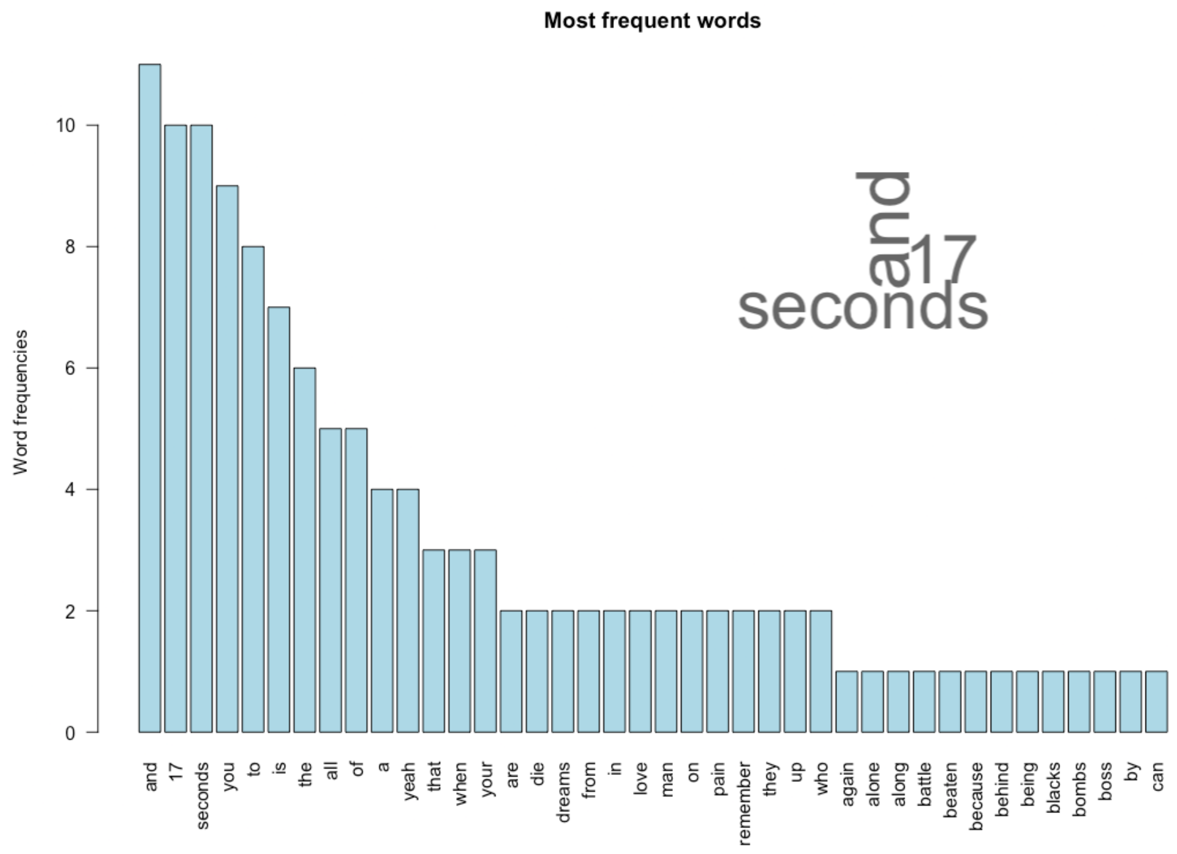


Explicit TRUE



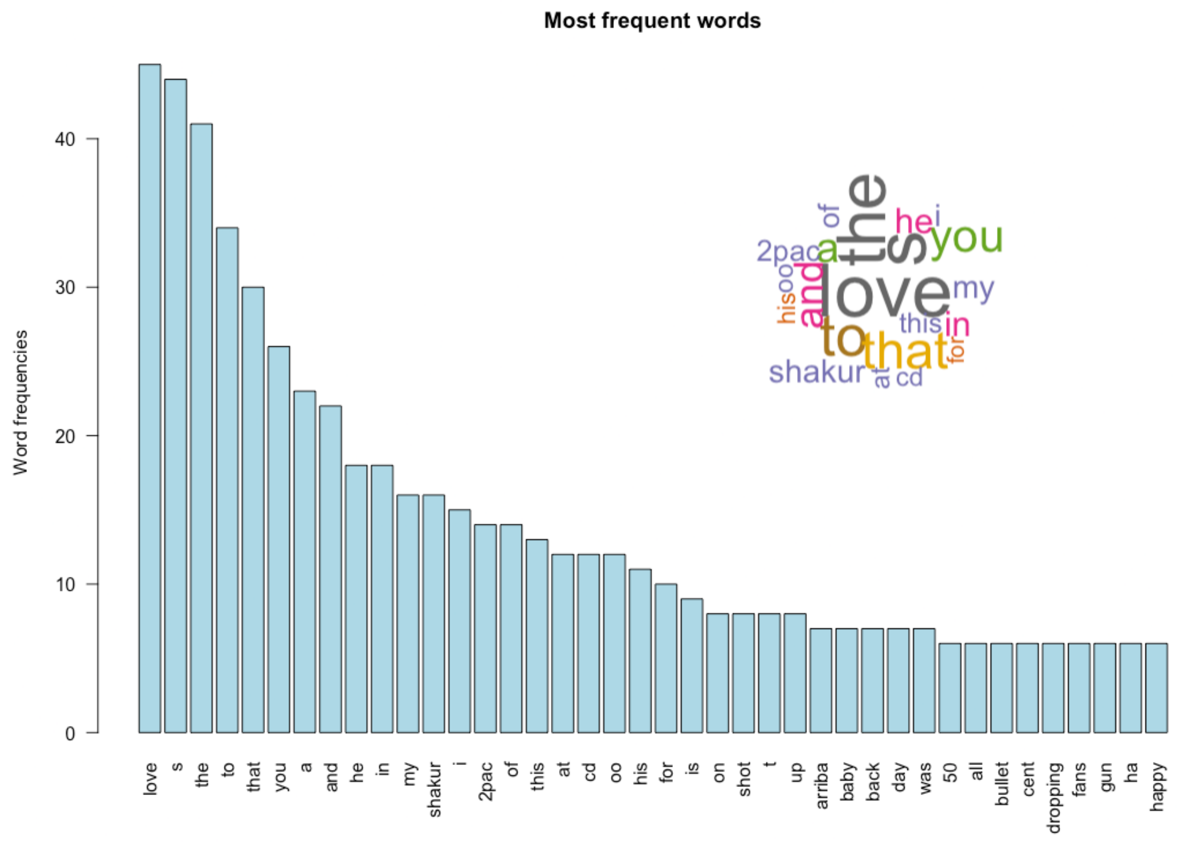
Instrumentalness 0.995

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having instrumentalness 0.995 , means songs in which instrumentals are in majority of the part with less lyrics. And we found out that high instrumental songs contain "and", "17", "seconds". Surprisingly, in the list "17" is third most popular.



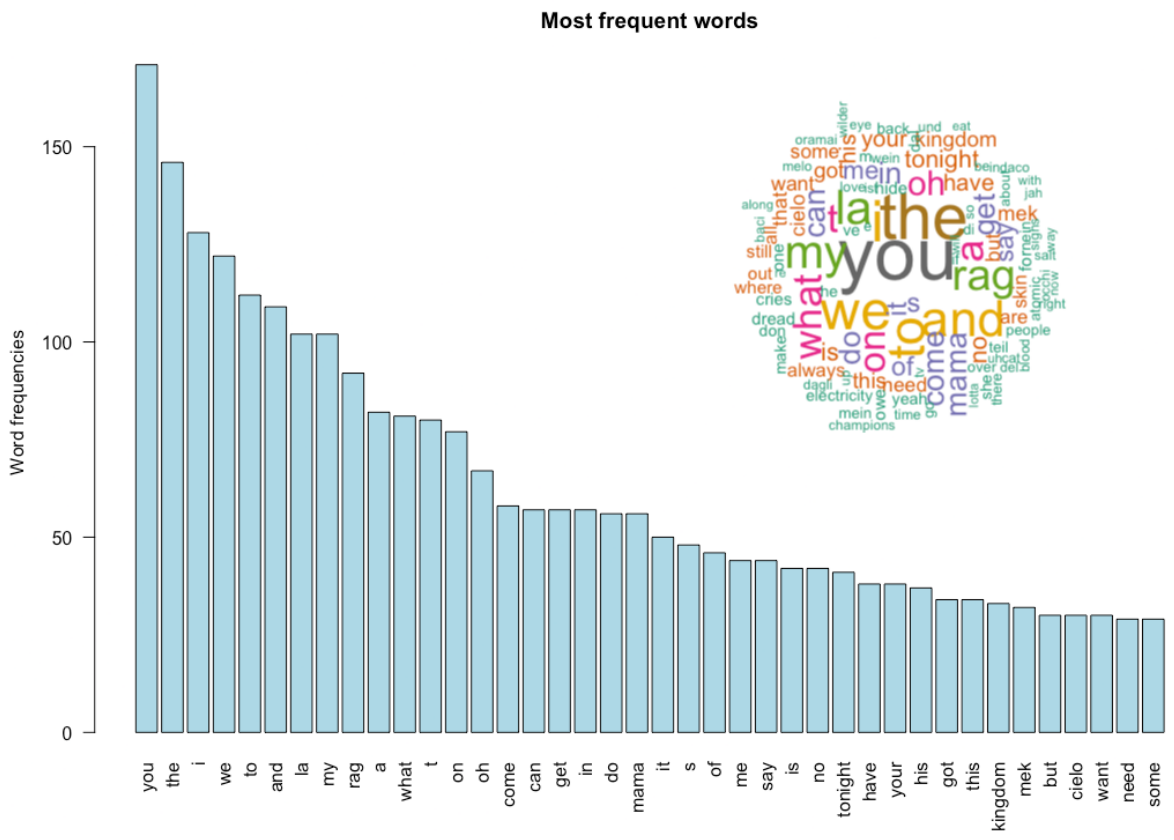
Liveness 0.0

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having liveness 0.0 , means where no audience or live nature is involved, basically recorded in a studio with no component of human crowd or noise. And we found out that lowest live nature songs contain “love”, “s”, “the”. Surprisingly, word “s” is in the list.



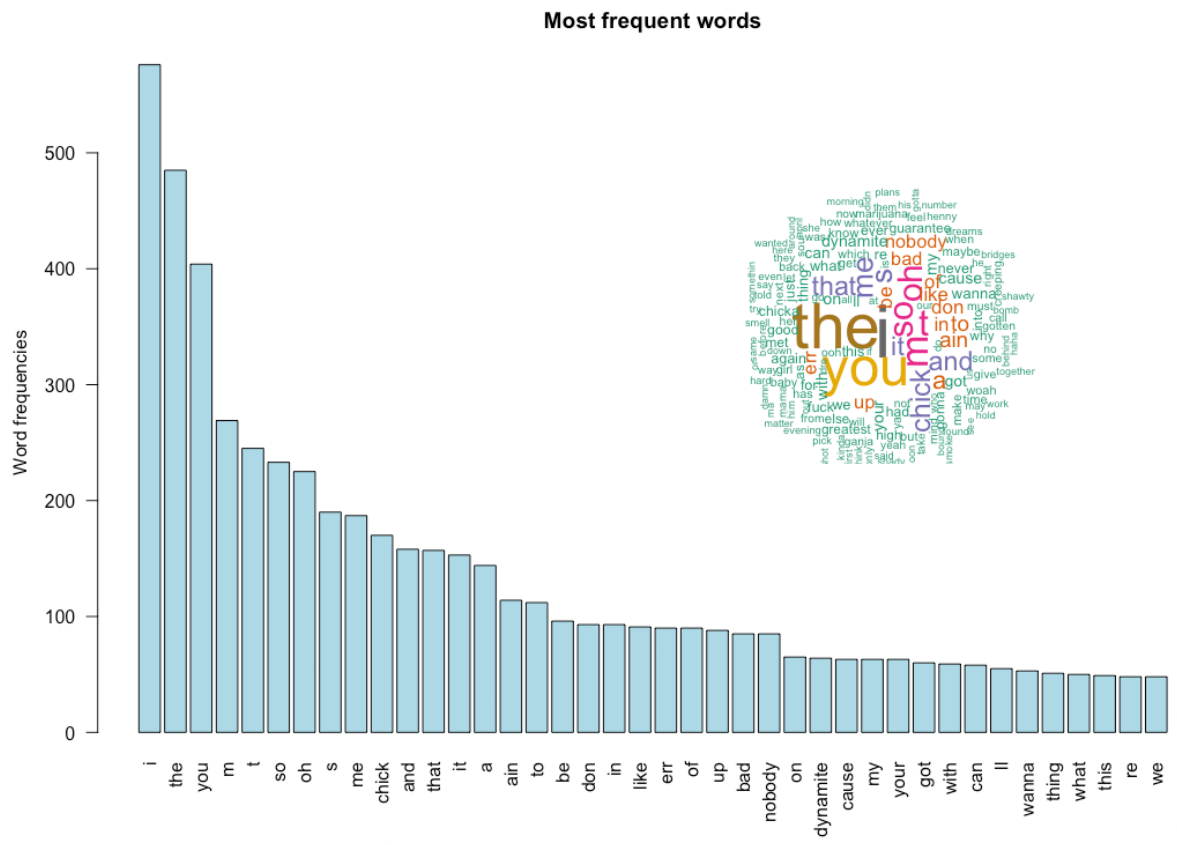
Liveness 0.997

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having liveness 0.997 , means songs which were performed and recorded live with major component of human crowd or noise. And we found out that highest live nature songs contain "you", "the", "I".



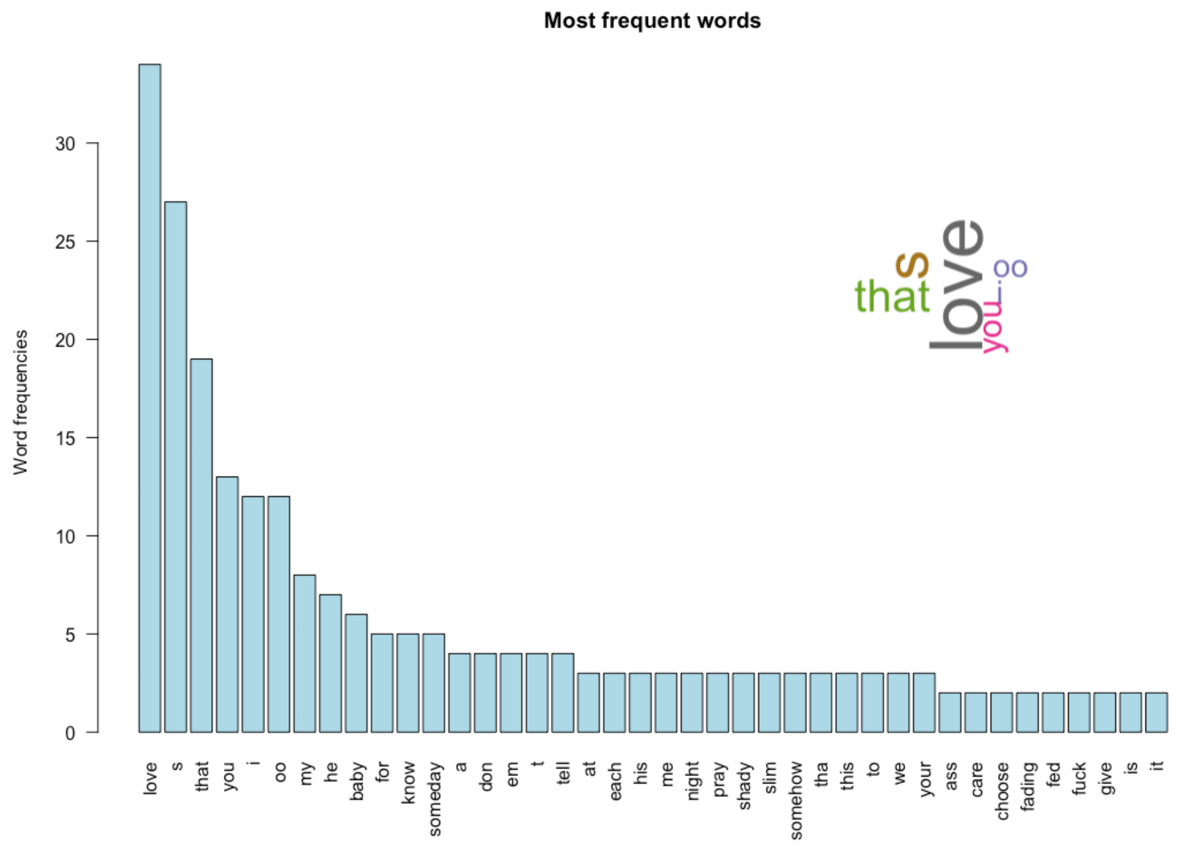
Loudness 0.0

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having loudness 0.0 , means where loudness is close to none , basically with low decibel(dB). And we found out that quietest nature songs contain "I", "you", "the".



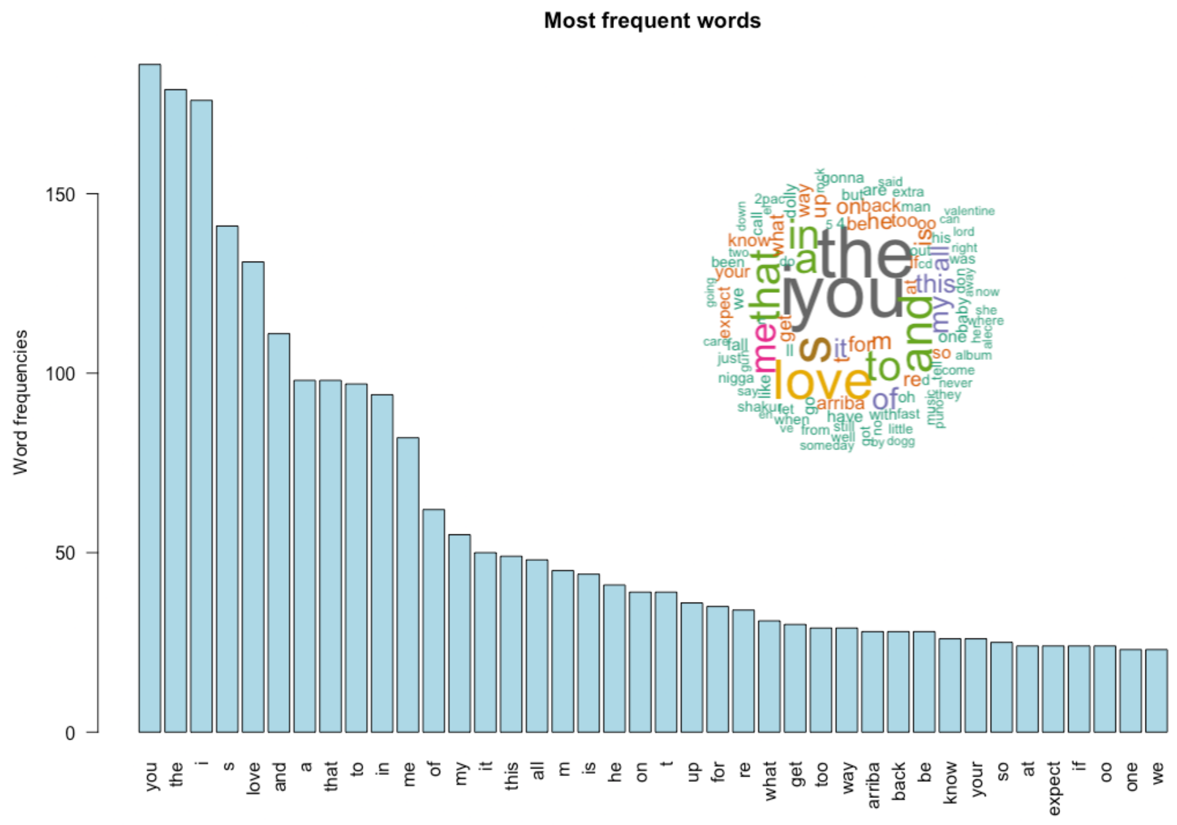
Loudness 60.0

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having loudness 60.0 , means where loudness is very high , basically with very high decibel(dB). For eg. Heavy metal songs, rock songs etc.. And we found out that loudest nature songs contain "love", "s", "that". Surprisingly, here also word "s" is involved.



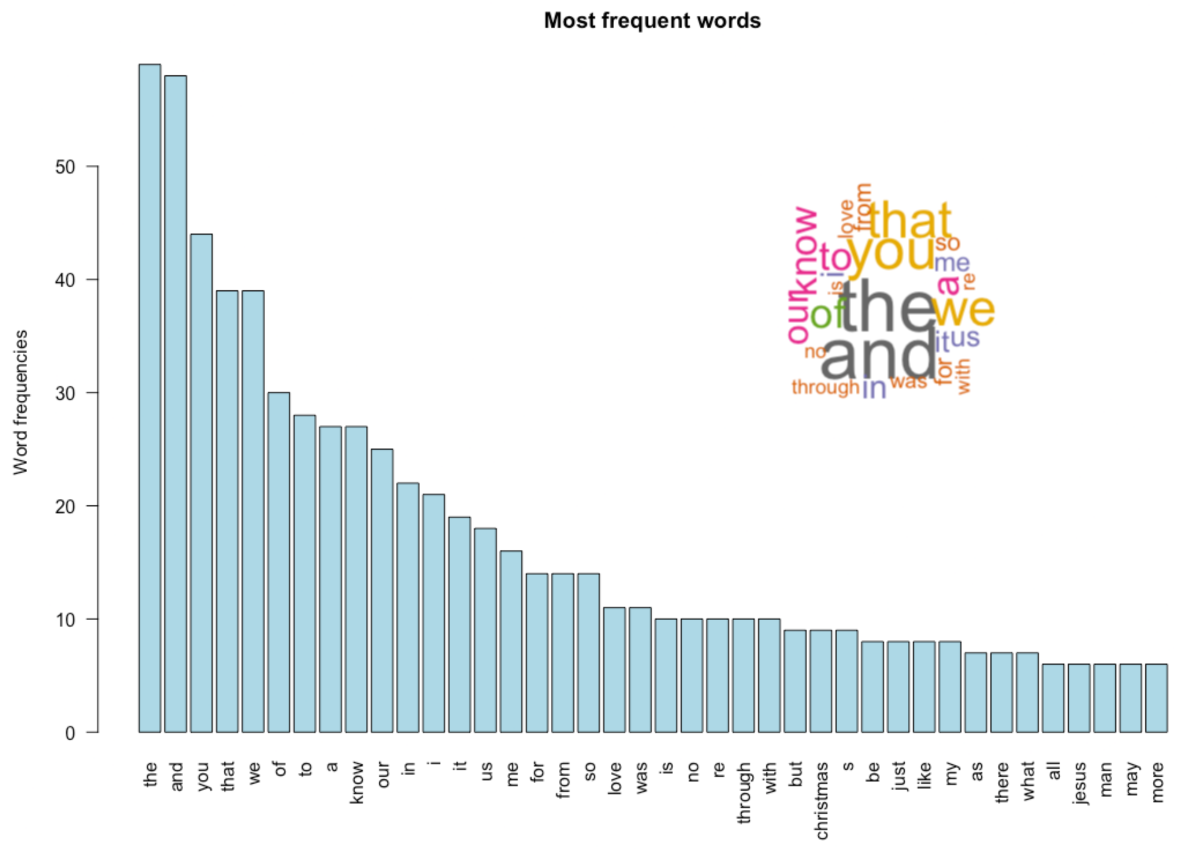
Speechiness 0.0

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having speechiness 0.0 , means where there is very low or no human or other creatures voice involved. And we found out that speechless nature songs contain "I", "you", "the".



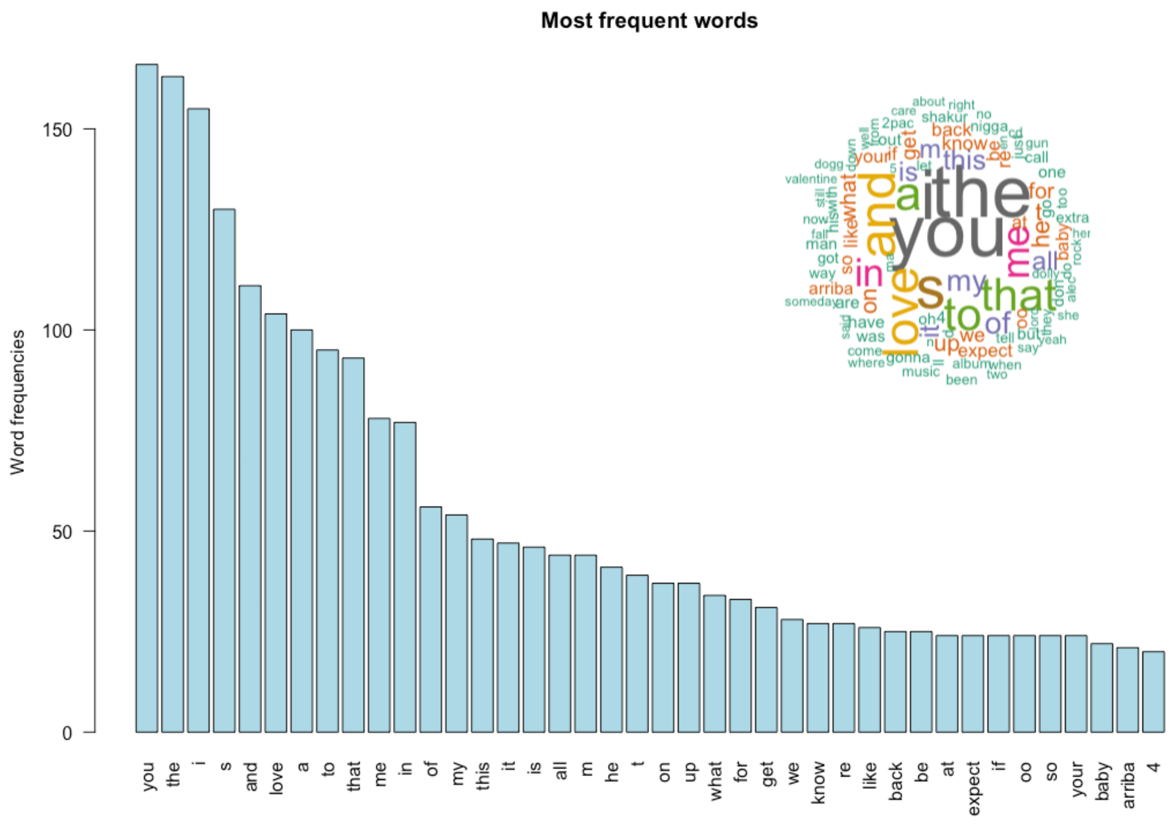
Speechiness 0.96

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having speechiness 0.96 , meaning for most part of the song human voice or speech is their. For eg. A Podcast. And we found out that songs with high speechiness contain "the", "and", "you".



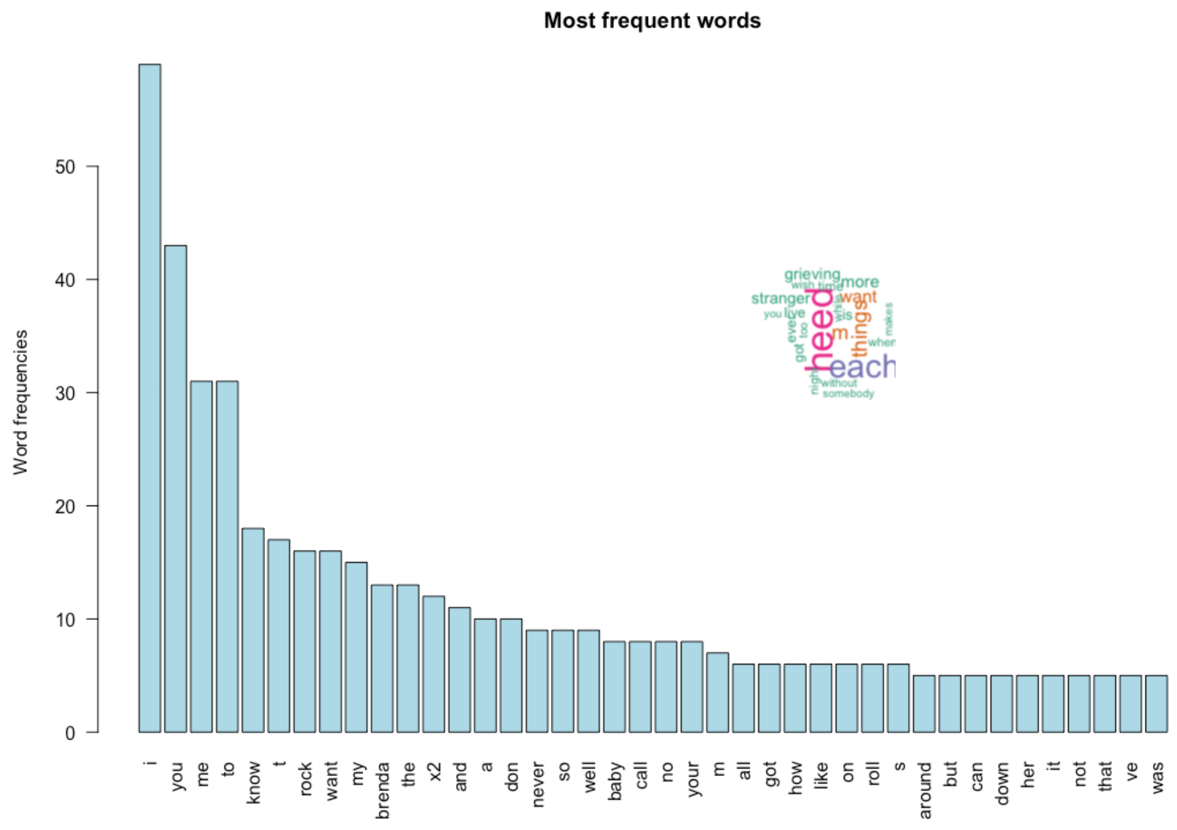
Tempo 0.0

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having tempo 0.0 basically the song is very slow. For eg. Ghazals. And we found out that speechless nature songs contain "you", "the", "I".



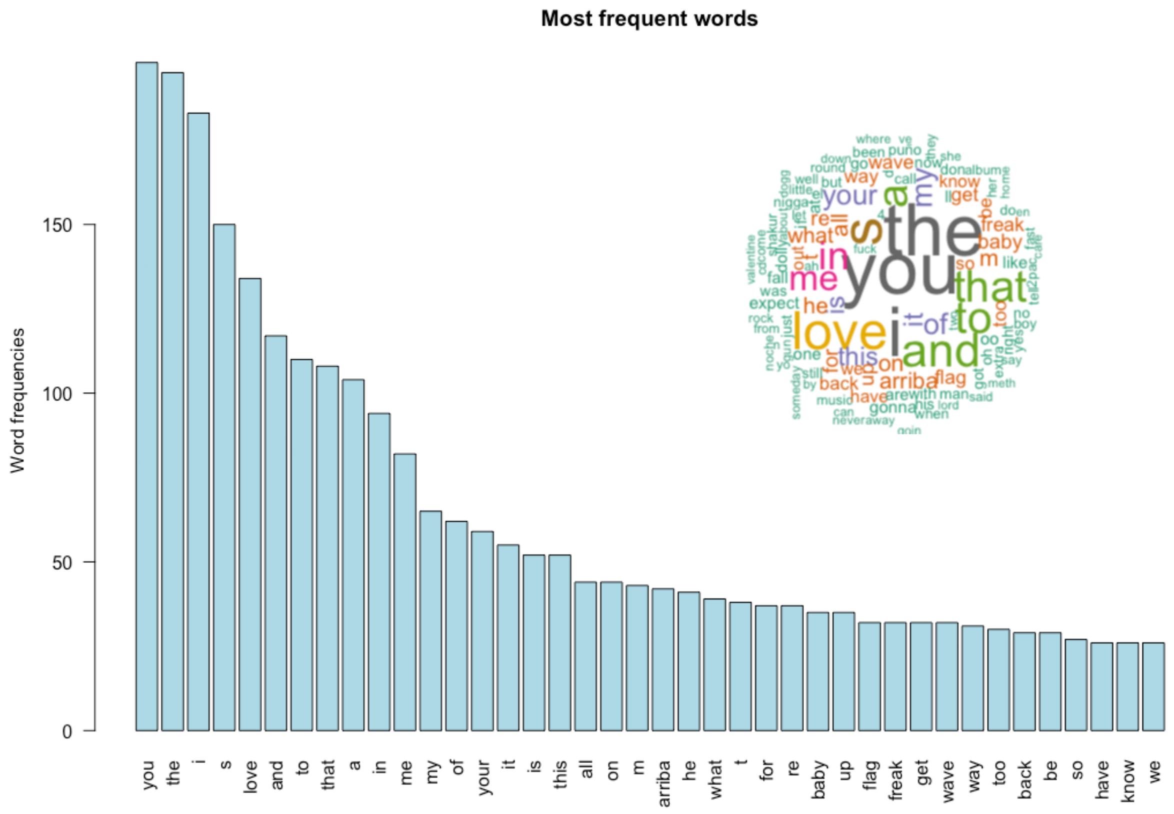
Tempo 225.0

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having tempo 225.0 basically the song which are very fast. For eg. Rap songs. And we found out that high tempo nature songs contain "I", "you", "me".



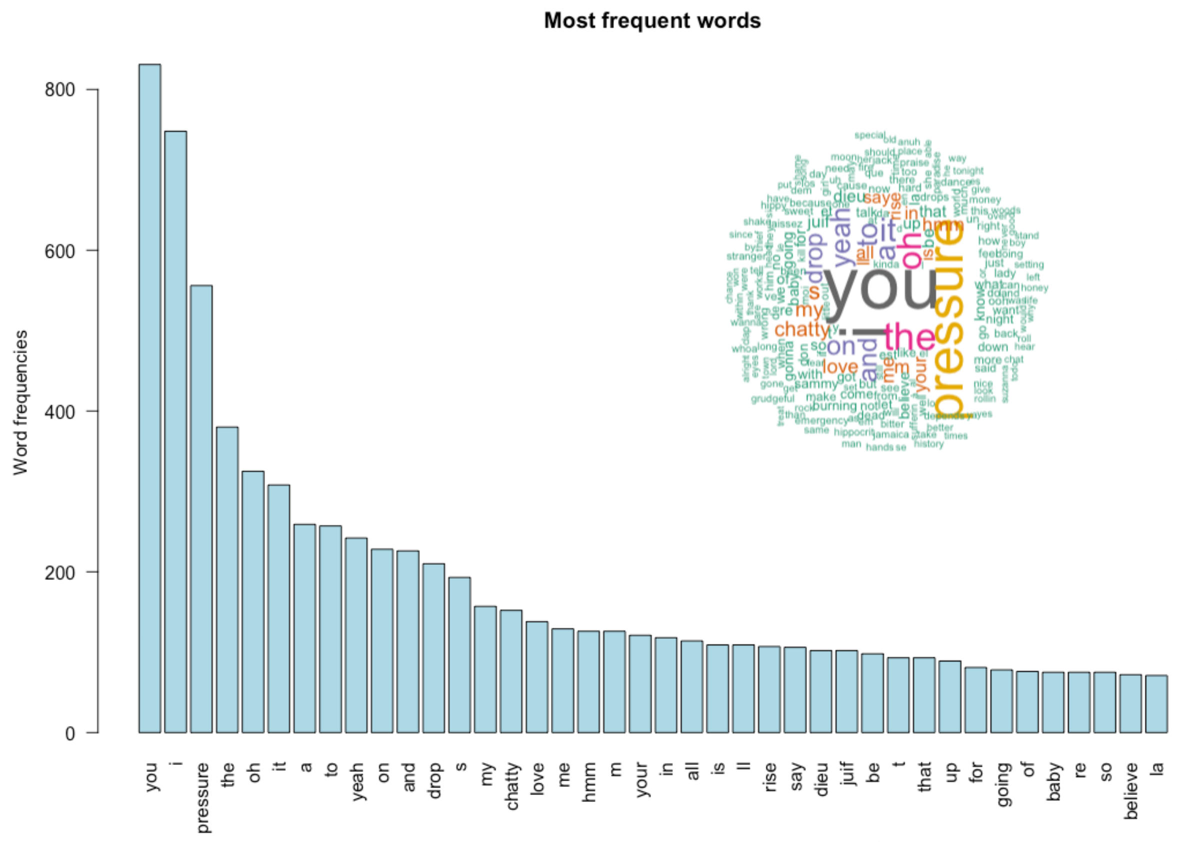
Valence 0.0

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having valence 0.0 basically the songs which are very negative in nature. They can be fearful, jealously, agony songs. And we found out that negative nature songs contain "you", "the", "I".



Valence 0.98

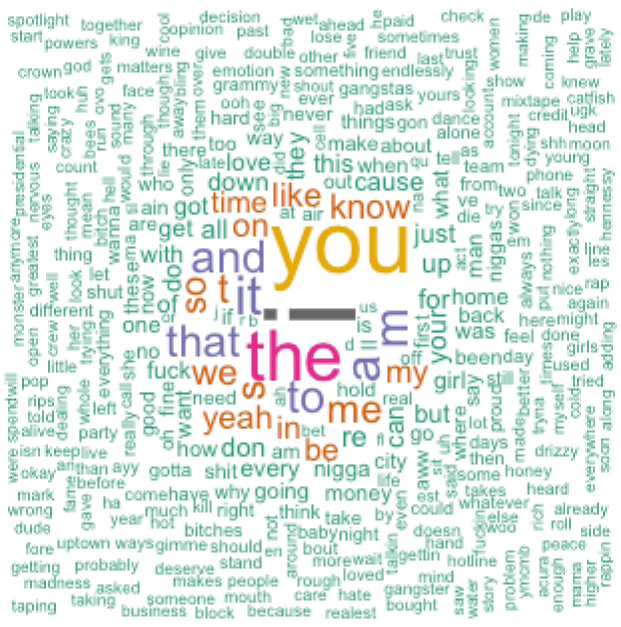
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having valence 0.98 basically the songs which brings positive energy. That make us feel good, joyous. And we found out that positive nature songs contain "you", "I", "pressure".

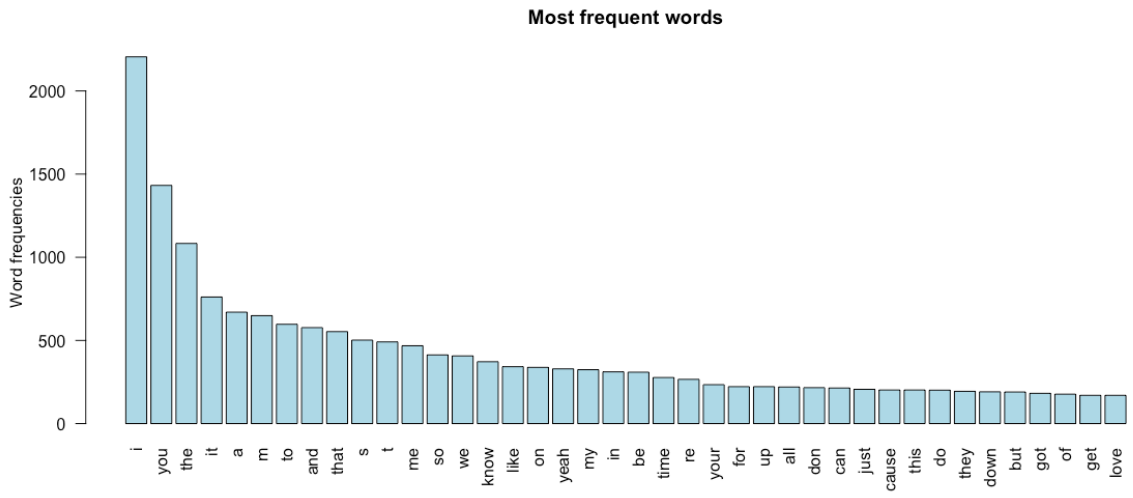

Most frequent words
Word frequencies
0
200
400
600
800
you
i
pressure
the
oh
it
a
to
yeah
on
and
drop
s
my
chatty
love
me
hmm
m
your
in
all
is
ll
rise
say
dieu
juif
be
t
that
up
for
going
of
baby
re
so
believe
la

Word plots with frequency word plot of upcoming artists

Acousticness 0.01

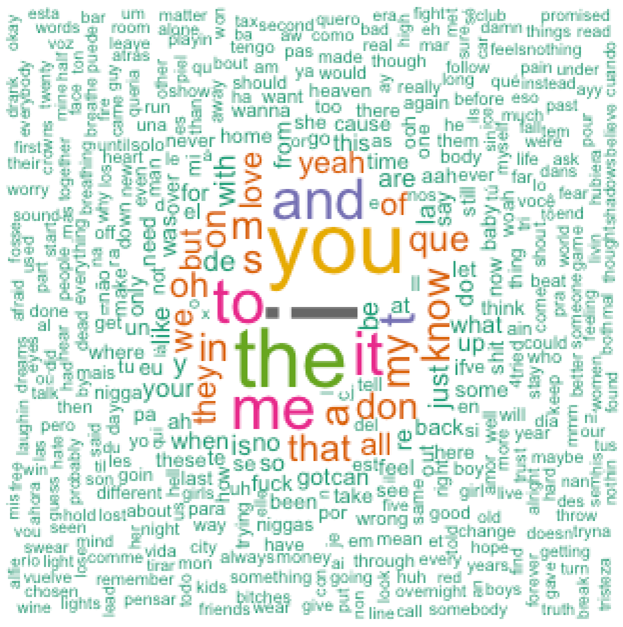
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having acousticness 0.01 , means songs which are produced with high use of electrical equipments. And we found out that lowest acousticness songs contain "i", "you", "the".

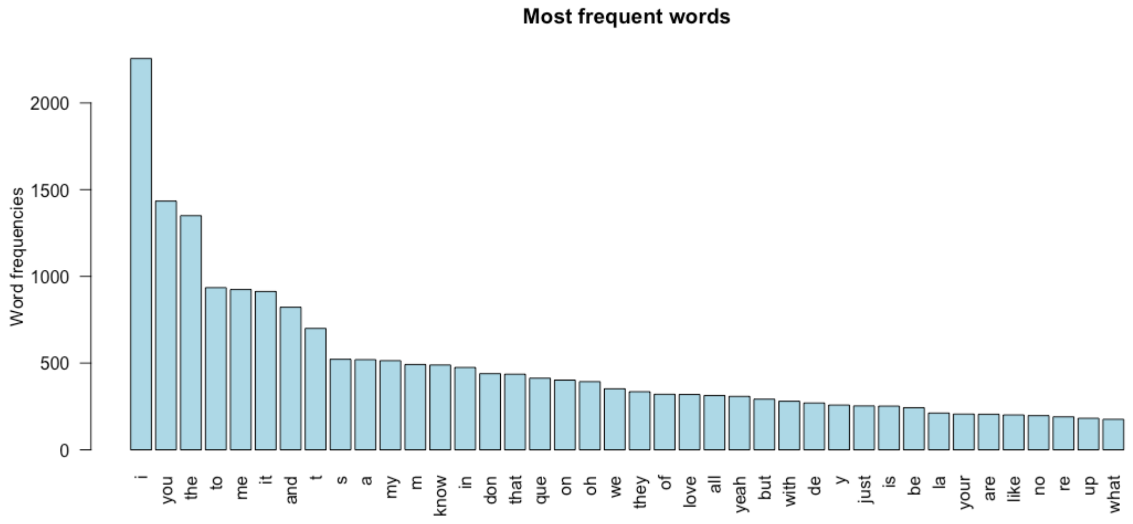




Acousticness 0.75

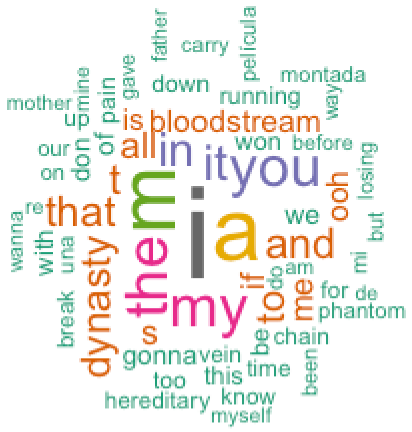
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having acousticness above 0.995 , means songs which are produced with less use of electrical equipments. And we found out that highest acousticness songs contain "I", "you", "the".

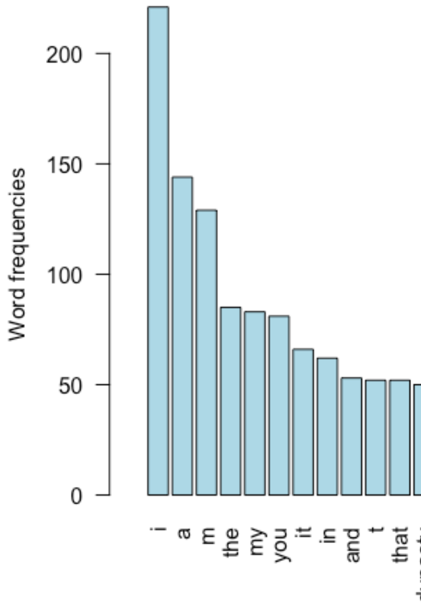




Danceability 0.03

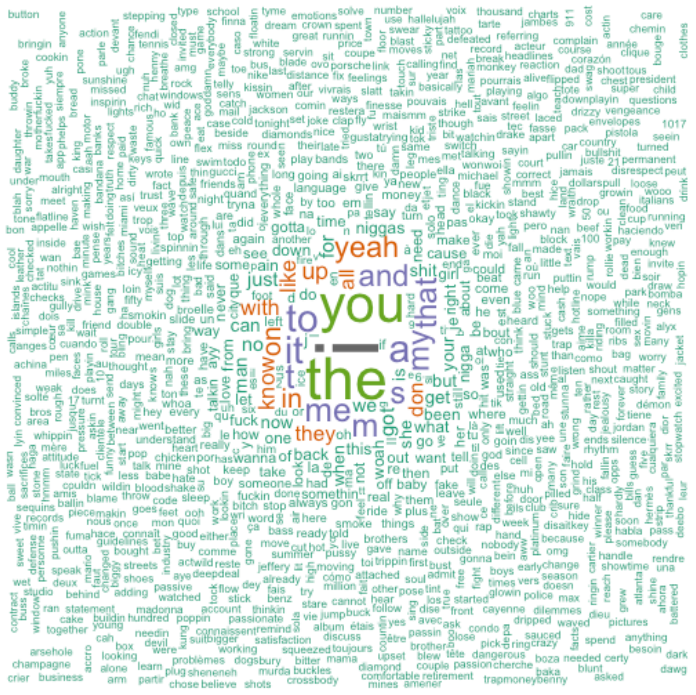
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having danceability below 0.03 , means which are least danceable. And we found out that least danceable songs contain "a", "I", "am".

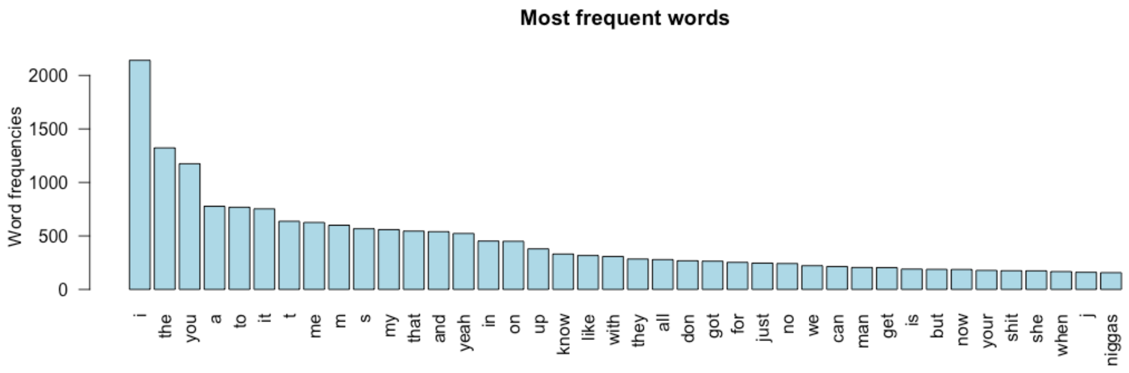




Danceability 0.8

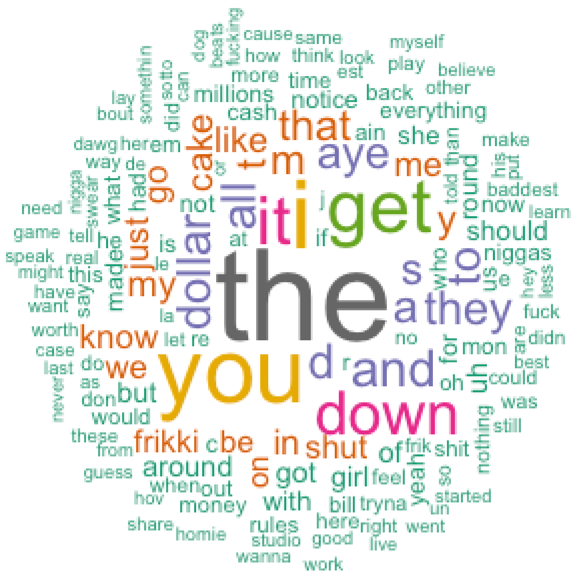
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having danceability above 0.8 , means which are most danceable. And we found out that most danceable songs contain “you”, “I”, “the”.

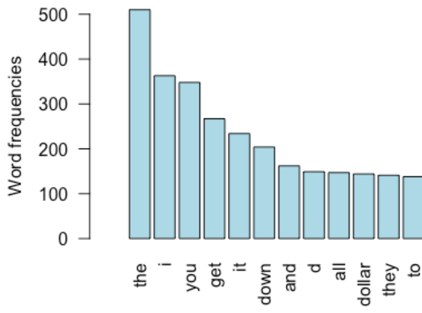




Duration 400000

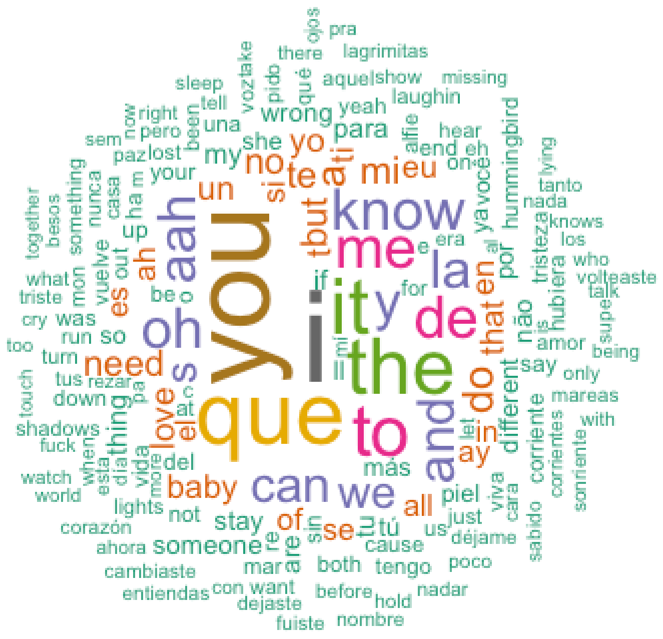
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having duration 400000 , means which are of low duration. And we found out that moderately low duration songs contain "you", "the", "you".

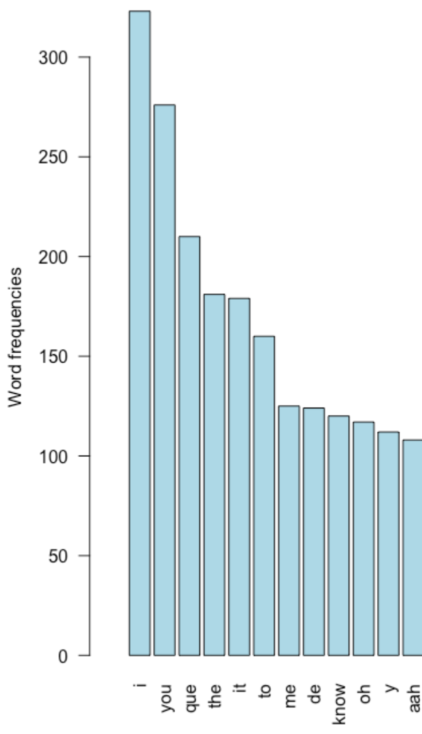




Energy 0.02

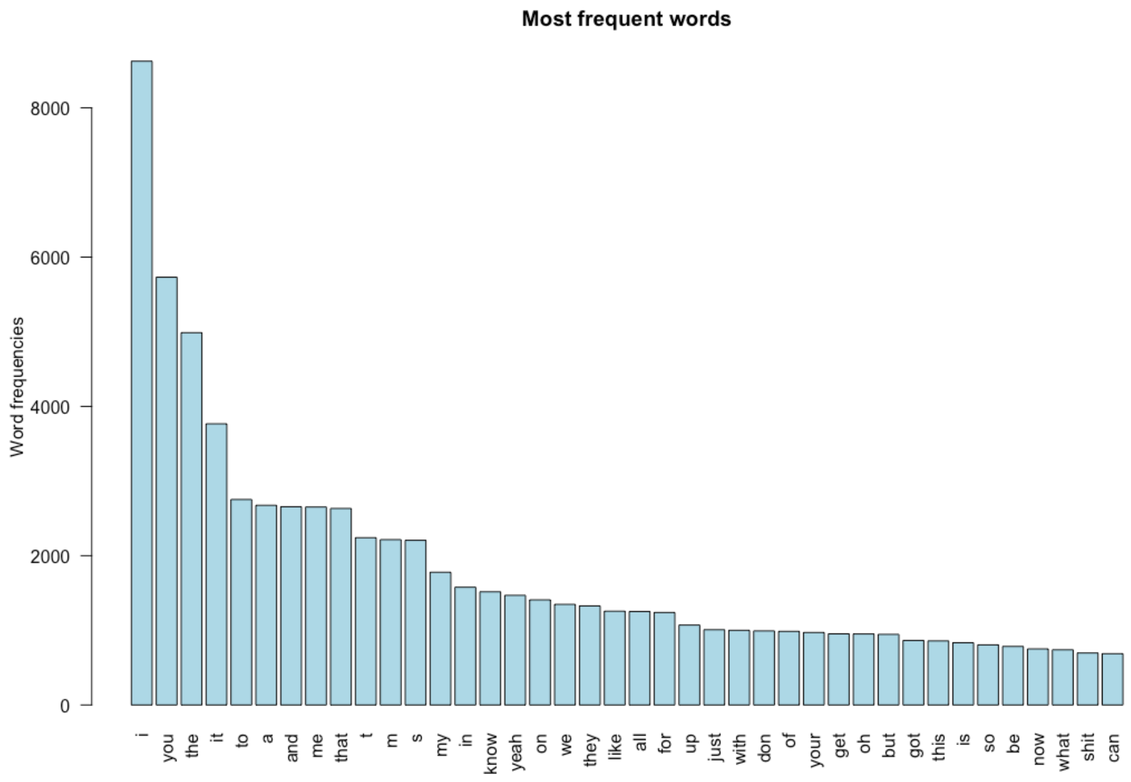
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having energy below 0.02 , means which are of lowest intensity on user. And we found out that lowest energy songs contain "you", "I", "que".





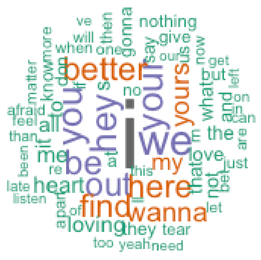
Explicit TRUE

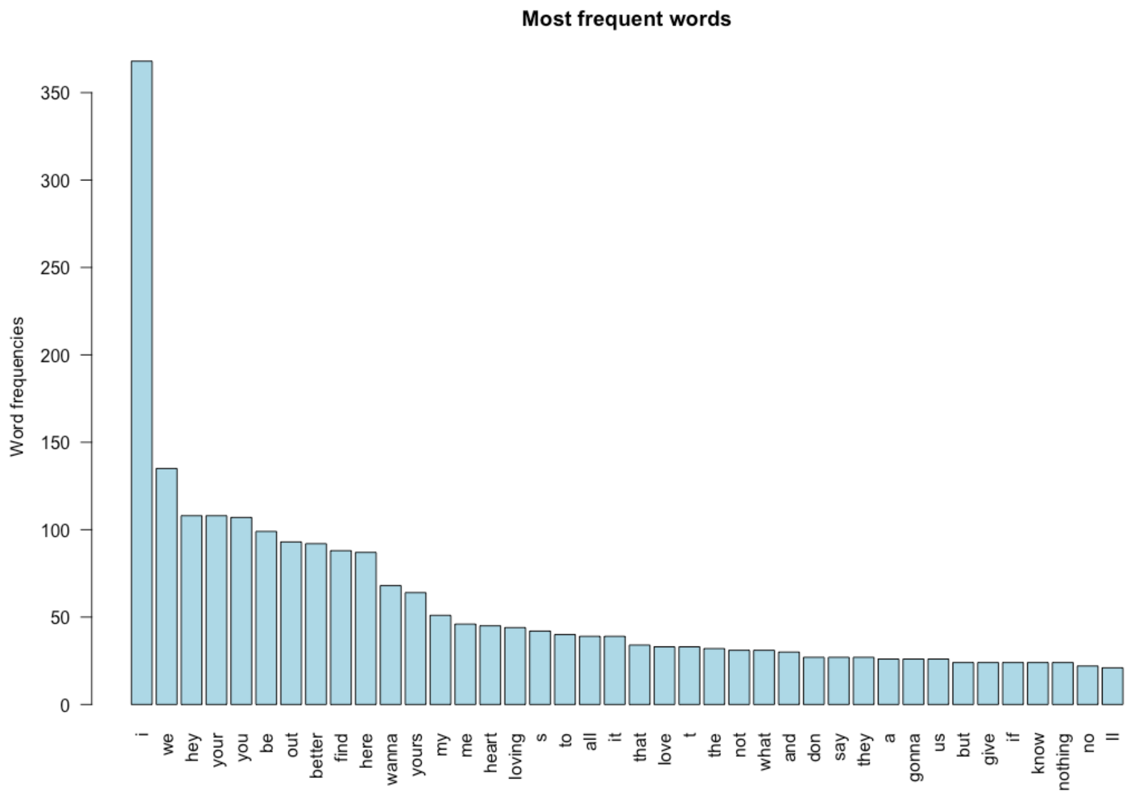
you
i
the
it
that
and
me
my
to
a
know
yeah
m
we
on
they
in
s
for
all
like
t
don



Liveness 0.05

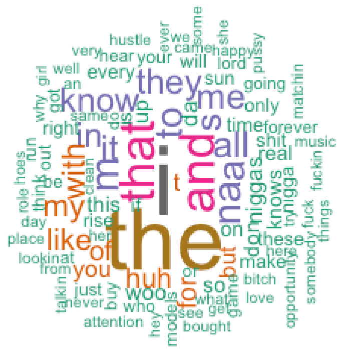
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having liveness below 0.05 , means where no audience or live nature is involved, basically recorded in a studio with no component of human crowd or noise. And we found out that lowest live nature songs contain “i”, “we”, “hey”.


i
we
your
hey
you
be
out
better
yours
here
find
wanna
my
nothing
gonna
give
say
then
will
ve
when
one
more
know
matter
afraid
feel
than
been
late
listen
don
to
all
it
me
heart
loving
apart
off
too
yeah
need
they
tear
let
not
that
love
just
the
and
what
but
now
us
our
get
left
on
in
can
are
no
this
at
re
s

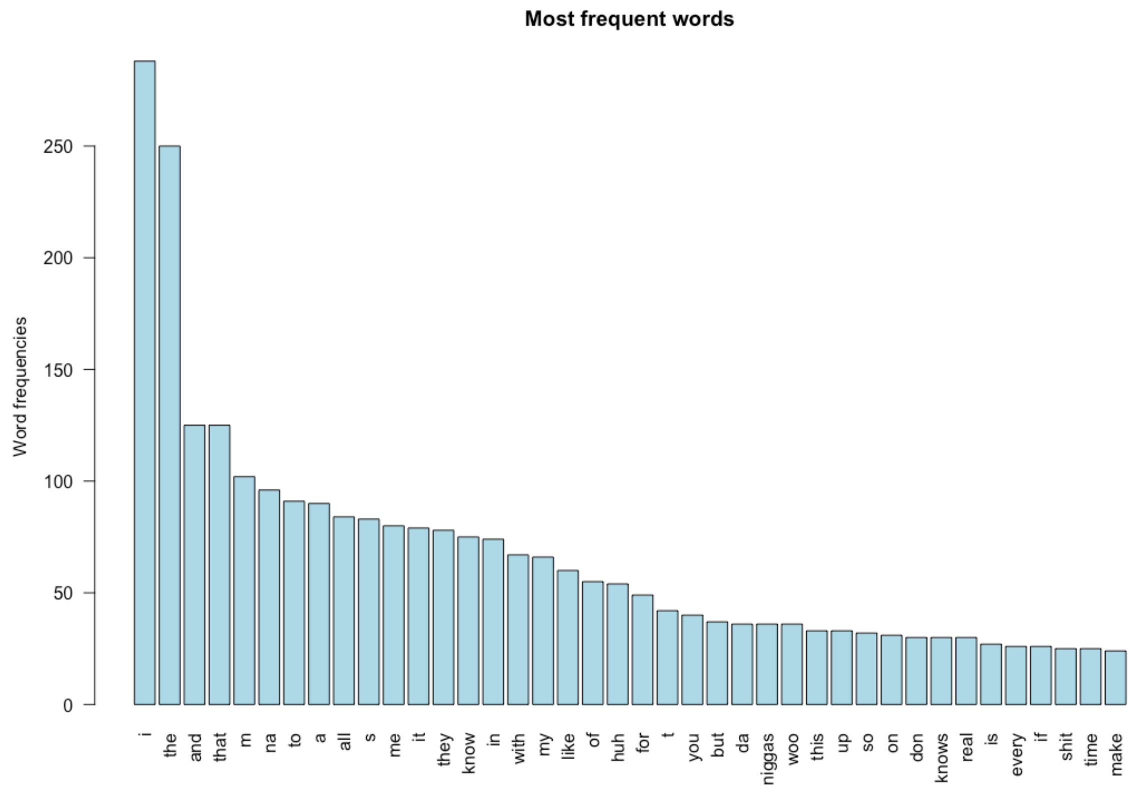


Liveness 0.75

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having liveness above 0.75 , means songs which were performed and recorded live with major component of human crowd or noise. And we found out that above average live nature songs contain "and", "the", "I".

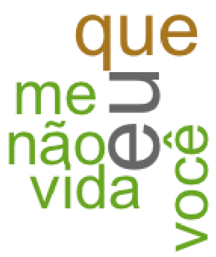


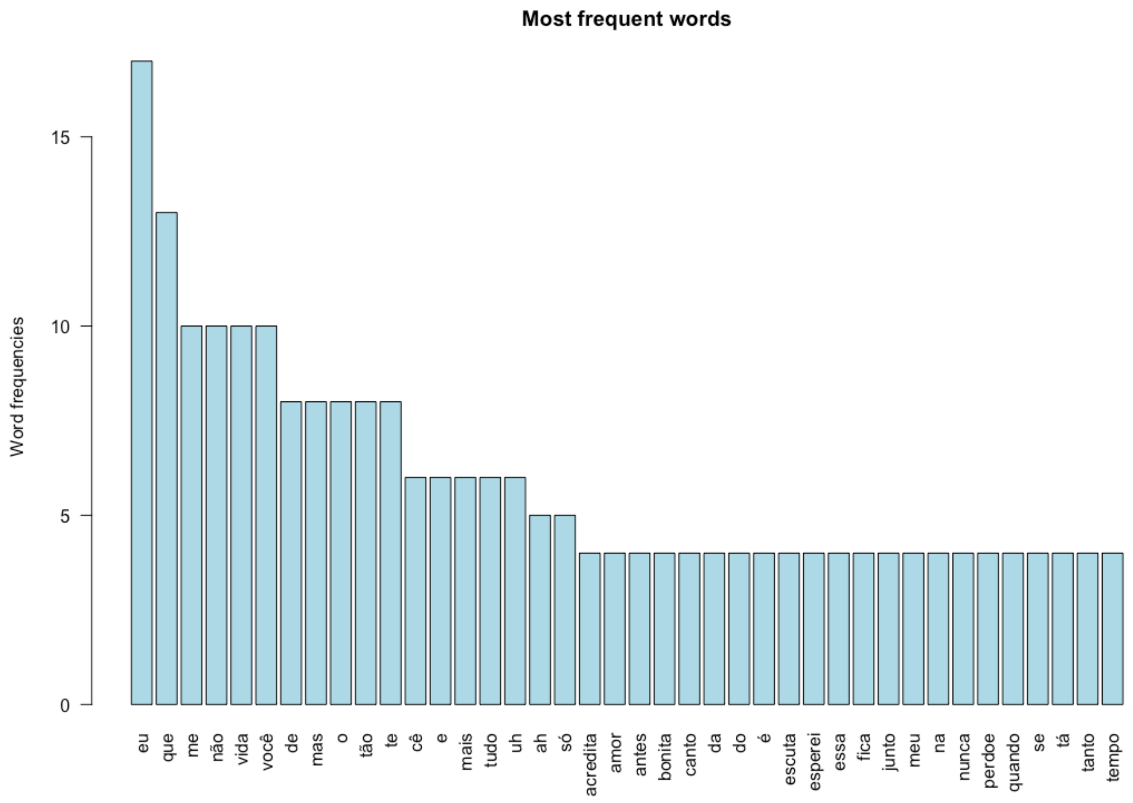
the
i
and
that
to
know
they
me
all
naa
s
m
it
in
with
my
like
of
you
huh
for
but
t
up
da
niggas
on
don
knows
nigga
these
make
real
shit
time
only
going
sun
will
lord
your
hear
every
this
rise
woo
so
who
just
right
got
out
be
run
think
hoes
role
day
place
lookinat
from
talkin
never
attention
buy
hey
models
see
bought
get
what
game
bitch
love
here
opportunity
somebody
things
fuck
fuckin
music
forever
matchin
try
pussy
she
happy
came
some
we
ever
hustle
very
well
an
girl
why
same
do
is
clean
her



Loudness 20.0

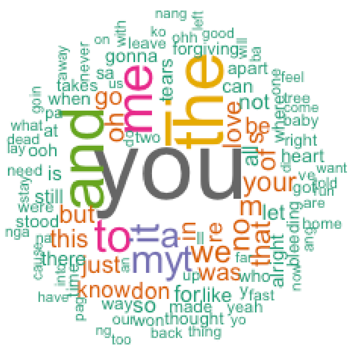
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having loudness above 20.0 , means where loudness is high , basically with high decibel(dB). For eg. Heavy metal songs, rock songs etc.. And we found out that loud nature songs contain “eu”, “que”, “me”.


que
me
eu
não
vida
você

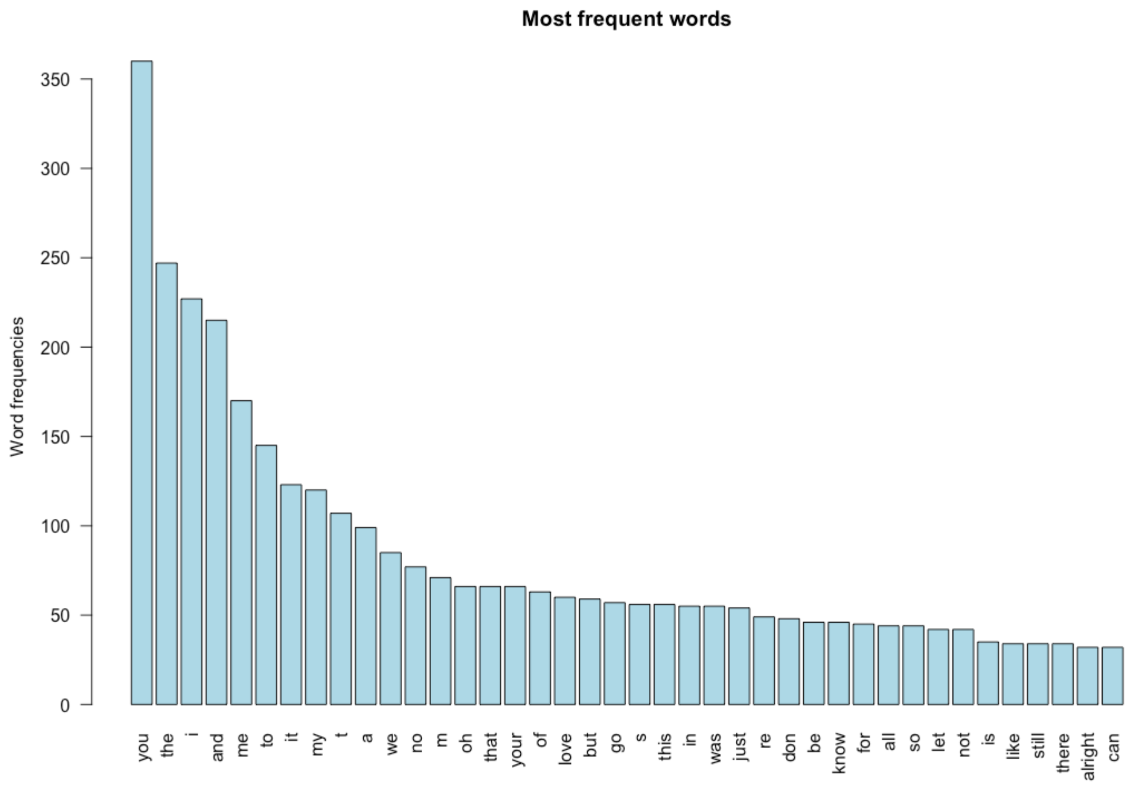


Speechiness 0.03

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having speechiness below 0.03 , means where there is very low or no human or other creatures voice involved. And we found out that speechless nature songs contain "I", "you", "the".

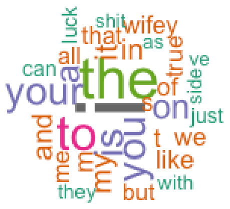


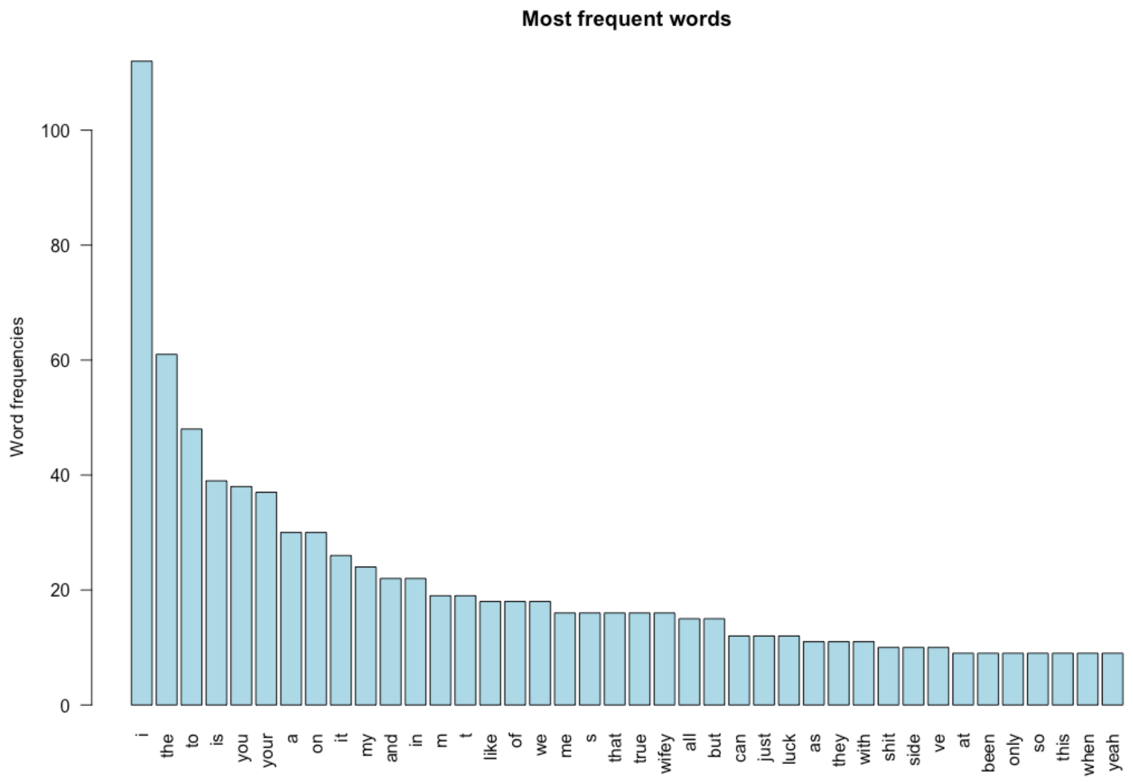
you
the
and
me
to
it
a
my
we
your
that
no
of
love
this
but
just
know
don
was
for
like
be
all
not
go
oh
in
re
m
so
there
stood
still
is
ooh
at
what
dead
lay
need
stay
were
nga
cause
have
when
takes
away
never
sa
gonna
on
with
leave
nang
ko
ohh
left
good
forgiving
will
ba
tears
apart
can
feel
one
where
free
come
baby
right
heart
want
ve
got
told
run
let
are
home
bleeding
alright
now
who
fast
far
up
two
way
won
thought
made
yeah
yo
back
thing
too
ng
out



Speechiness 0.7

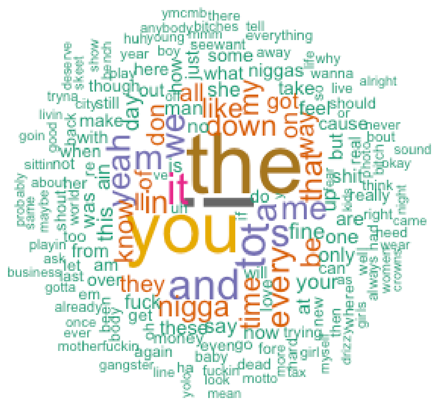
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having speechiness above 0.7 , meaning for most part of the song human voice or speech is their. For eg. A Podcast. And we found out that songs with high speechiness contain "the", "i", "to".


the
i
to
your
you
is
my
and
on
a
it
in
that
wifey
of
true
we
like
but
with
they
me
m
t
s
just
side
ve
as
all
can
luck
shit

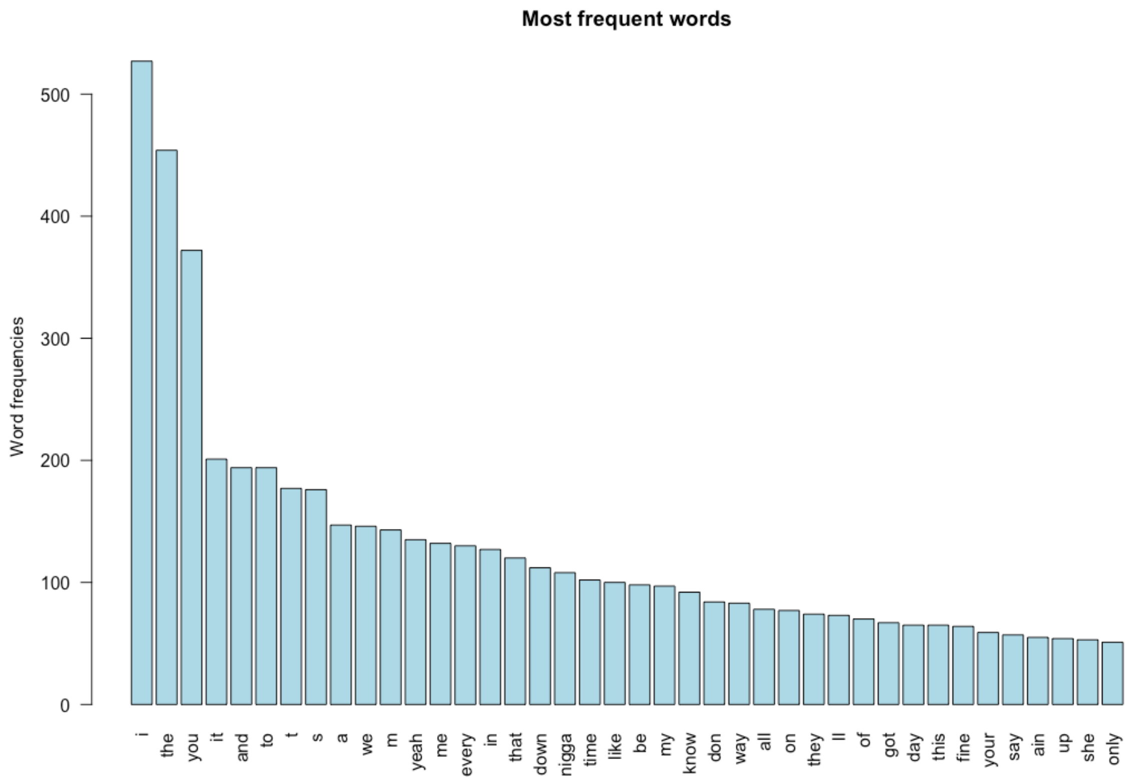


Tempo 185.0

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having tempo 185.0 basically the song which are very fast. For eg. Rap songs. And we found out that high tempo nature songs contain "I", "you", "the".

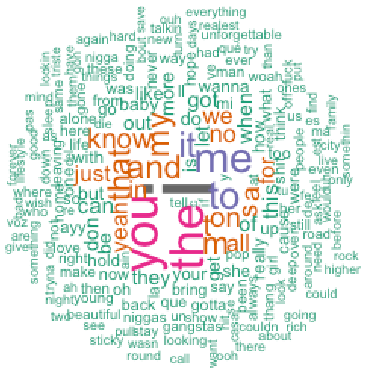


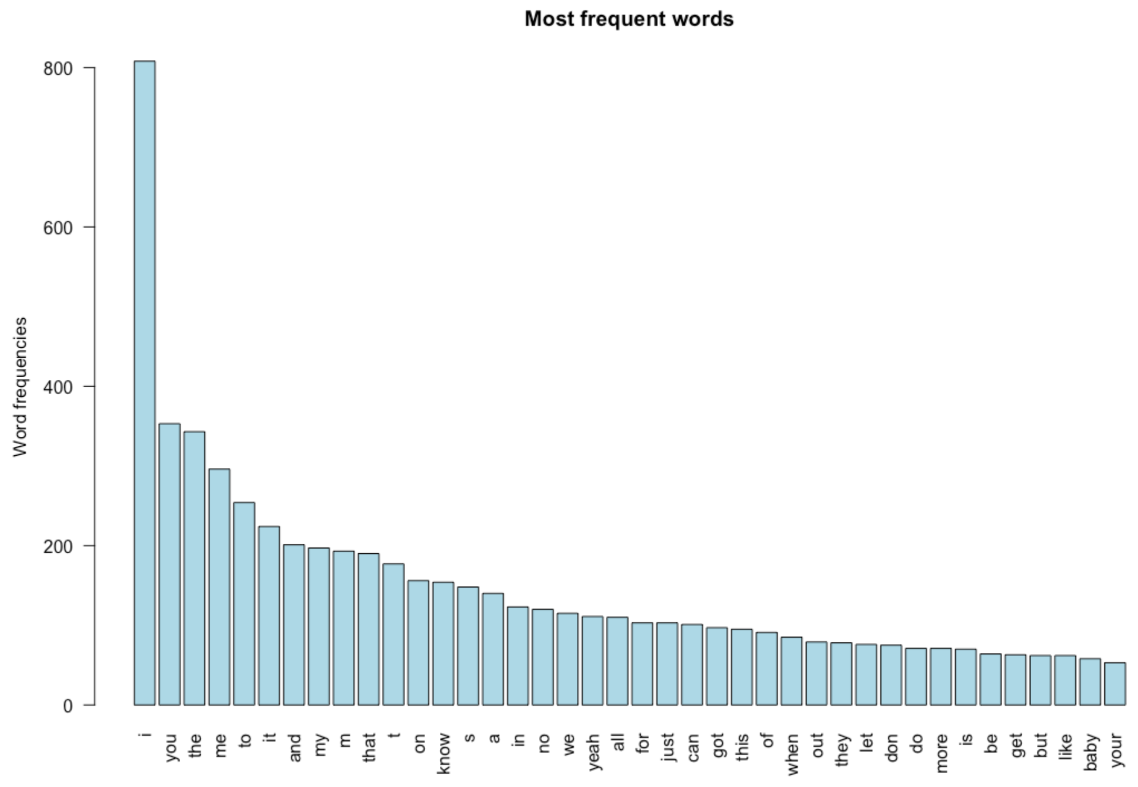
the
you
and
it
yeah
we
m
lin
know
tot
a
me
s
every
time
that
way
be
nigga
down
like
my
don
all
got
on
up
they



Valence 0.1

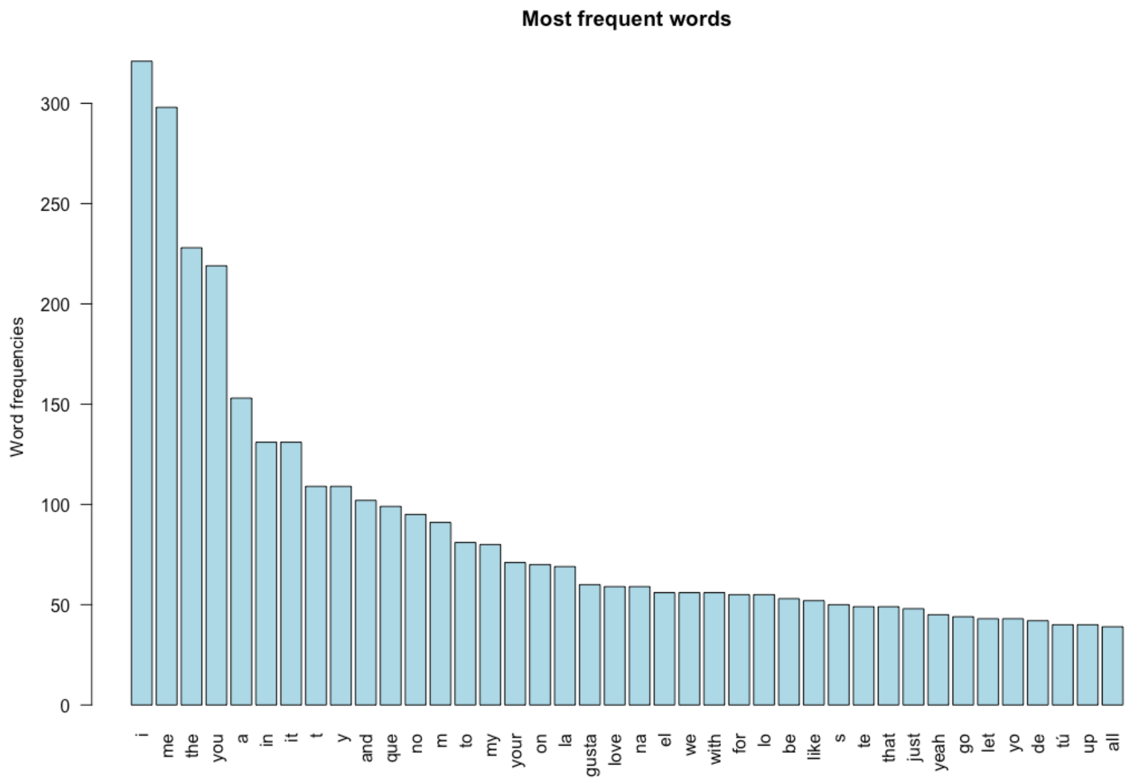
Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having valence 0.1 basically the songs which are very negative in nature. They can be fearful, jealously, agony songs. And we found out that negative nature songs contain "you", "the", "I".





Valence 0.85

Here, we made a words frequency bar chart(barplot()) and a word plot adjacent to bar chart, of all the tracks having valence 0.85 basically the songs which brings positive energy. That make us feel good, joyous.


Most frequent words
Word frequencies
0
50
100
150
200
250
300
i
me
the
you
a
in
it
t
y
and
que
no
m
to
my
your
on
la
gusta
love
na
el
we
with
for
lo
be
like
s
te
that
just
yeah
go
let
yo
de
tú
up
all

SHINY APP :

Preprocessing of data:

1) We got the lyrics of songs as **list**, which we converted into **dataframe**.
2) Then we **sorted** them and then **merged** them.
3) We also got some **verses** (like I love you), but we needed only words and not verses, so we seperated them out.
4) Then frequency count was arranged in **descending** order from the data.
5) Then we set a **threshold** value, to only get words whose appearances were more than **20** times.
6) Then we made a **histogram** of top **40** words.
7) Then we plotted the heatmap of correlation between threshold musical attributes and song lyrics(happy songs, sad songs) which gave us a score about at the what **proportion** they exists in a category(eg. If in a song there are 200 happy words and total there are 2000 words, then proportion will be 200/2000 = .1).

Inside shiny

In the shiny app we have tried to make following plots:

1)In the first part we have made plots regarding personal Spotify data where we have added bar plots regarding following information which when clicked on the side panel shows:

(i)Most listened artist (more than 3 hours) (ii)Playback activity by date and time of day

(iii)playback activity by day type (iv)playback activity by time of the day and weekday-line chart

v playback activity by time of the day and weekday (vi) playback activity by time of the day

(vii)playback activity per week and hour.

2)In the second part we have tried to made a **barplot** of word frequency and **wordcloud** of the words used in the song of all artists. Here we have chosen some characters or say properties of the song and for each one we have mad above mention plot. Following are the properties taken into considerations:

i **Acousticness** (ii) **Danceability** (iii) **Duration** (iv) **Energy** (v) **explicitness** (vi) **instrumentalness**

vii **liveness** (viii) **loudness** (ix)**speechiness** (x) **valence** (xi) **tempo**

3) In the third part we have tried to made a **barplot** of word frequency and **wordcloud** of the words used in the song of upcoming artists. In side panel first we will select country and then we will get a list of **upcoming artist** from that **country**, after then **music attributes** will come, after selecting a musical attribute we will get option of **higher and lower**. After chossing all the options we will get a desired **wordplot** with a frequency wordplot (**barplot**) of top 40 words. Then there is also a heatmap with it, which shows correlation between lyrics and mood. For eg. Happy words, sad words etc..

WHAT WE DID WITH THE DATA (conclusion)

1. We can now do analysis of Spotify account, by getting user's (who is willing to share his/her Spotify information) account playback activity, and then further do analysis on who is user's favourite singer, which is user's favourite song, how much time that user spends listening Spotify tracks, at what time interval etc., both in data and graphical way.
2. We can get the information about different words no. of appearance in a particular track lyrics or lyrics of a collection of track.
3. We also know which word in the lyrics of track convey which emotion, or generally we made a dataset of all the words classified as per emotions.
4. We also created a heatmap between different attributes of music and lyrics of different emotions, which gave us a matric score of each attribute. From which we can tell which attributes have which implication on emotions.

REFERENCES https://towardsdatascience.com/explore-your-activity-on-spotify-with-r-and-spotifyr-how-to-analyze-and-visualize-your-stream-dee41cb63526