Stage 2 report

Introduction

The best example to illustrate the commercialization of the arts is today's music industry. Being a multi-billion dollar industry, artists and labels have to work on finding a balance between making music that appeals to their fans and fulfills their artistic expression, while also makes money. In the digital landscape, the number of plays or streams a song gets is indicative of it's financial success. In other words, the more popular a song becomes, the more money it brings in. If this really is the case, then being able to idenfity how popular a song can become before it is even released would be of great interest to labels and record companies. So, this naturally raises the question that we will be focusing on in this report:

Given data on attributes of a song, can we predict how popular it will become?

Dataset

To answer this question, our group will be working with data on different audio-based features of a song to predict it's popularity when released on a music streaming platform. The dataset we will be using is from the Tidy Tuesday dataset collection [https://github.com/rfordatascience/tidytuesday/blob/master/data/2021/2021-09-14/readme.md]. We have cleaned and combined the `audio_features.csv` and `billboard.csv` datasets provided to obtain the dataset we will use for the rest of our analysis. The raw data files have 32 variables, of which we will be using 21 relevant variables, removing the variables that don't affect our dataset, such as URLs and id's for the songs. We also narrowed the billboard data by selecting only the rows which a song peaks on the chart, and ignoring other weeks, then removing the week specific variables. The full transformation can be found in the `Data_Transformation_script_Billboard.R` script. The remaining data represents different features of a song such as its key, tempo, loudness, danceability, etc. as well as metrics reflecting its popularity on billboard and spotify charts respectively.

Objective

Our goal with this project/report is to create a regression model to predict how "popular" a song may be based on it's audio-based features. This means we will be looking at each song independent of cultural trends, the associated artist's popularity, and other external influences. We will be looking into ideas such as what features popular songs have in common, if any specific feature is highly linked to a song performing well or being well recieved, and whether we can identify any subgroups with their own unique trends.

Motivation

We as a group agreed to working with this dataset and on this question because we are all avid music lovers. With varied interest and tastes amongst all of us, we thought it would be interesting to see how an emperical analysis of what is or isn't "popular music" would compare to our own personal preferences, and how much each of us may agree or disagree with the findings.

Exploratory Data Analysis

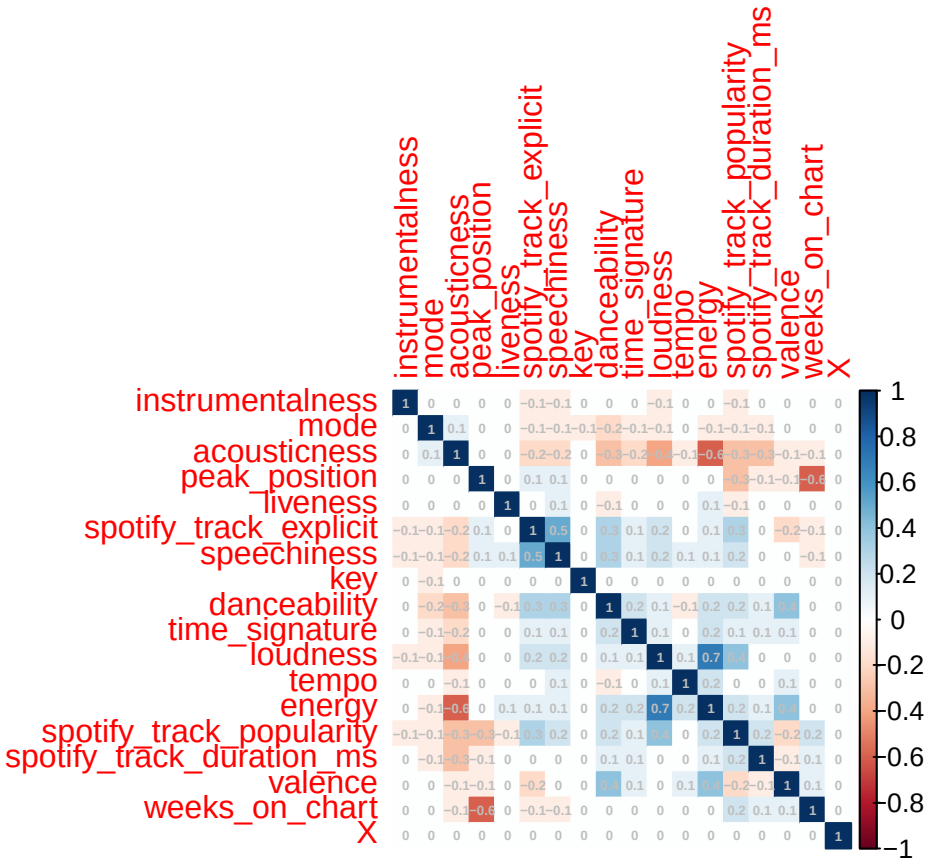
Quick briefing of selection of response variable

We first cleaned our dataset by removing obviously non-useful variables like names. (However, while we recognize that different genres may have differing levels of popularity, for simplicity we will leave out this variable as the "genre" of certain songs may be ambiguous and there are too many categories to keep track of.)

```
bd_clean <- select(songs, -performer, -song, -spotify_genre,
    -spotify_track_album, -date)
```

Next we make a correlation plot to have a broad overview of what variables may be connected to our response as well as with each other:

```
options(repr.plot.width = 12, repr.plot.height = 12)
cor_matrix <- round(cor(bd_clean), 1)
corrplot(cor_matrix, method = "color", addCoef.col = "grey",
    order = "AOE", number.cex = 0.4)
```



We had briefly explained in our stage 2 report that we originally chose `peak_performance` as our response variable, but with its noticeably low correlations with most of the predictors, we switched our response to `spotify_track_popularity`

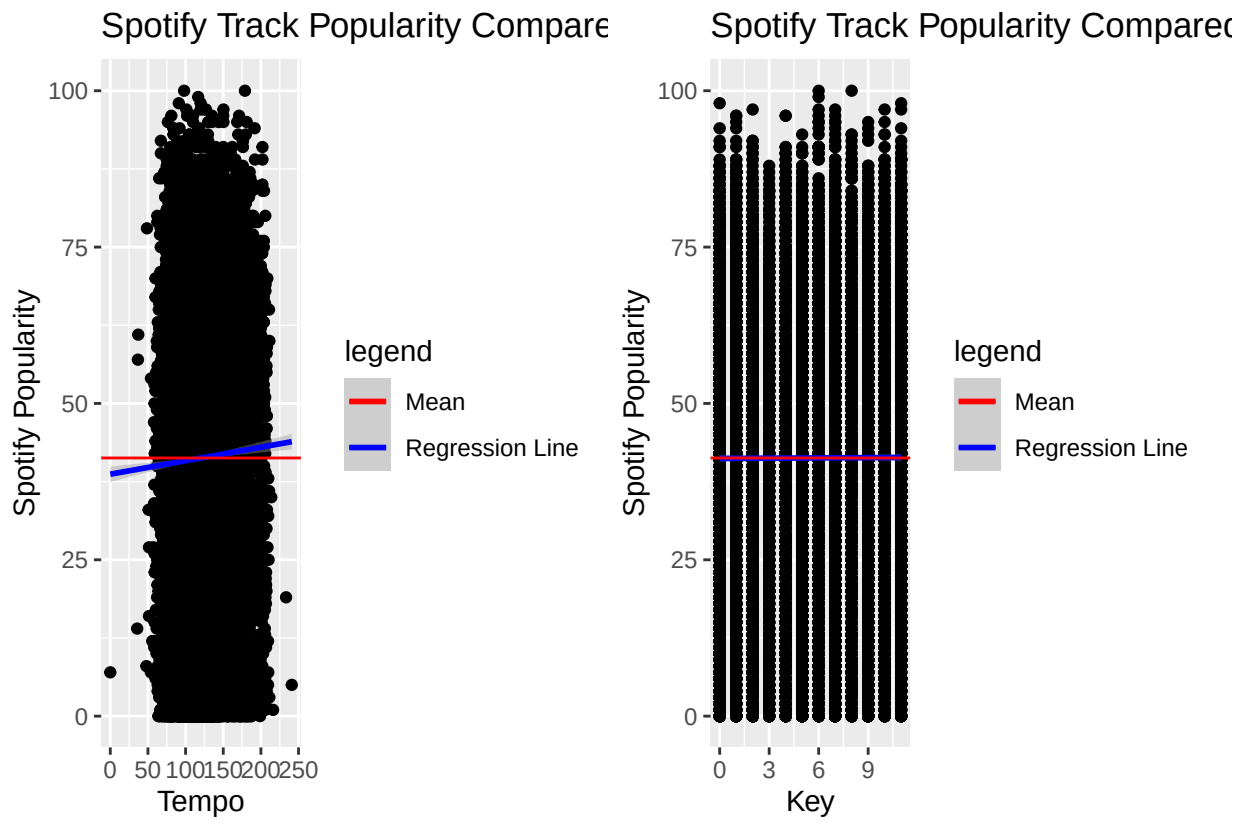
Exploration of Spotify Track Popularity on Billboard as Response Variable

`spotify_track_popularity` is correllated with almost all parameters in our data set. This means we are much more likely to be able to actually predict it's popularity based on the values of the other parameters, which will preform better than `peak_performance` did.

Here we see the two least correlated variables with respect to `spotify_track_popularity`:

```
options(repr.plot.width = 10, repr.plot.height = 4)
mean_popularity <- mean(songs$spotify_track_popularity)
p1 <- ggplot(data = songs, aes(tempo, spotify_track_popularity)) +
    geom_point() + geom_smooth(method = "lm", aes(color = "Regression Line")) +
    geom_hline(aes(yintercept = mean_popularity, color = "Mean")) +
    scale_color_manual(name = "legend", values = c("red", "blue")) +
    labs(x = "Tempo", y = "Spotify Popularity", title = "Spotify Track Popularity Compared to Tempo")

p2 <- ggplot(data = songs, aes(key, spotify_track_popularity)) +
    geom_point() + geom_smooth(method = "lm", aes(color = "Regression Line")) +
    geom_hline(aes(yintercept = mean_popularity, color = "Mean")) +
    scale_color_manual(name = "legend", values = c("red", "blue")) +
    labs(x = "Key", y = "Spotify Popularity", title = "Spotify Track Popularity Compared to Key")
p1 + p2
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



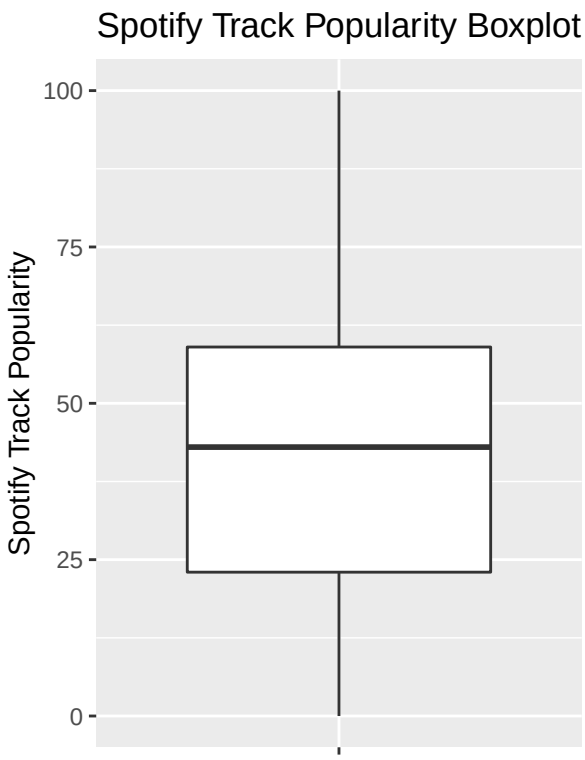
These two variables have been explained in greater detail in the earlier report, but are still quite relevant

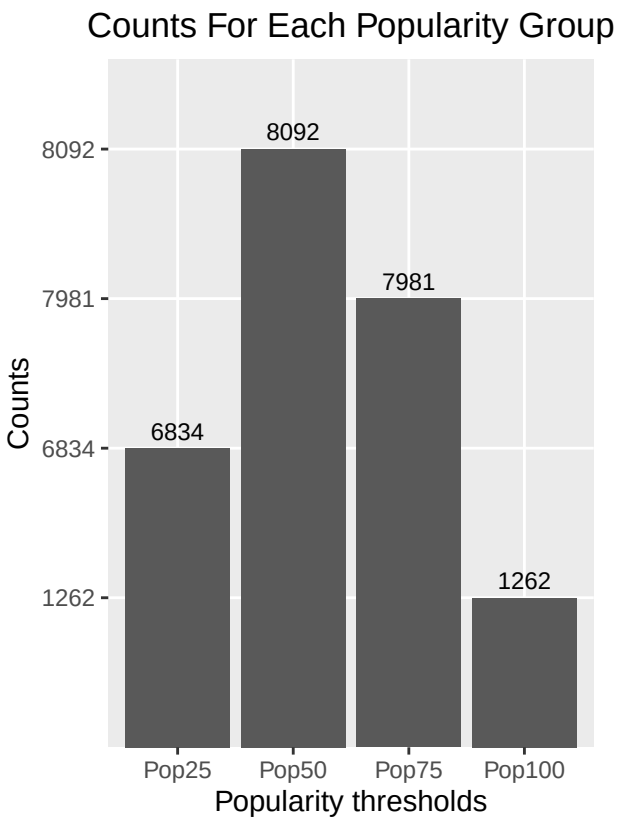
to mention as in our data analysis in training and testing models, these two were filtered out due to 0 correlation with the response `spotify_track_popularity` as well as to cut down on the number of our predictors to make model selection that much easier.

As we've seen with all these variables in the correlation matrix we will likely need many in tandem in order to create a model to predict the popularity of the songs, but we do have correlation in some form across many variables, so `spotify_track_popularity` should be a better response variable than `peak_performance`, so we will use this as our response variable.

Further Exploration Of Data

```
options(repr.plot.width = 10, repr.plot.height = 4)
popularity_boxplot = songs %>%
    ggplot(aes(x = "", y = spotify_track_popularity)) + geom_boxplot() +
    ggtitle("Spotify Track Popularity Boxplot") + labs(x = "",
    y = "Spotify Track Popularity") + theme(plot.title = element_text(hjust = 0.5))



pop25 = songs %>%
    filter(spotify_track_popularity <= 25)
pop50 = songs %>%
    filter(spotify_track_popularity > 25 & spotify_track_popularity <=
        50)
pop75 = songs %>%
    filter(spotify_track_popularity > 50 & spotify_track_popularity <=
        75)
pop100 = songs %>%
    filter(spotify_track_popularity > 75)


pop25_df = cbind("Pop25", nrow(pop25))
pop50_df = cbind("Pop50", nrow(pop50))
pop75_df = cbind("Pop75", nrow(pop75))
pop100_df = cbind("Pop100", nrow(pop100))

pop_all_df = as.data.frame(rbind(pop25_df, pop50_df, pop75_df,
    pop100_df))
colnames(pop_all_df) = c("Pops", "n")
ordered_pops = c("Pop25", "Pop50", "Pop75", "Pop100")

pop_n_plot = pop_all_df %>%
    ggplot(aes(x = Pops, y = n)) + geom_col() + scale_x_discrete(limits = ordered_pops) +
    labs(x = "Popularity thresholds", y = "Counts") + ggtitle("Counts For Each Popularity Group") +
    theme(plot.title = element_text(hjust = 0.5)) + geom_text(aes(label = n),
    vjust = -0.5, size = 3, position = position_dodge(0.9))


popularity_boxplot + pop_n_plot
```






```
summary(songs$spotify_track_popularity)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   23.00   43.00   41.28   59.00  100.00
```

Spotify popularity ranges from 0 to 100. The boxplot alongside the five number summary helps us understand that the median popularity is 43. This suggests that a majority of the songs are ranked lower in popularity, hence the higher popularity ranks are coveted.

To better understand the spread of the data, four groups have been constructed, `Pop25`, which contains data for all values of `spotify_track_popularity` $<$ 25; `Pop50` for 25 $\leq$ `spotify_track_popularity` $<$ 50; `Pop75` for 50 $\leq$ `spotify_track_popularity` $<$ 75; and finally, `Pop100` for `spotify_track_popularity` $>$ 75. It is evident that a large portion of the songs get ranked between 25 $\leq$ `spotify_track_popularity` $<$ 75, while only 1262 out of the 24169 entries are ranked equal to or above 75.