Spotify Podcasts vs. Audio Features: Statistical Testing

Lara Karacasu

2023-04-12

Data Preparation

```
# Read in CSV
data1 <- read.csv(file = "C:/Users/larak/Downloads/refined_metadata_v2.3.csv", header = T, stringsAsFac
data1 <- as_tibble(data1)
```

```
# Convert audio feature columns to double type
data1 <- data1 %>%
  mutate_at(c(13:100), as.character) %>%
  mutate_at(c(13:100), as.double)
```

```
# Print head of data
head(data1)
```

```
## # A tibble: 6 x 100
##       X show_n~1 episo~2 episo~3 rss_l~4 apple~5 apple~6 apple~7 apple~8 show_~9
##   <int> <fct>    <fct>   <fct>   <fct>   <fct>     <dbl>   <int> <fct>   <fct>
## 1     0 Kop On!~ "[\"ep~ ['2-26~ https:~ https:~     4.5      22 Sports  show_0~
## 2     1 2 Massa~ "['cre~ ['5-14~ https:~ https:~     5        25 Health~ show_1~
## 3     2 2. Bund~ "['2. ~ ['12-0~ https:~ https:~     4.9      17 Sports  show_5~
## 4     3 2Minute~ "['epi~ ['7-13~ https:~ https:~     5        18 Sports  show_4~
## 5     4 4th & G~ "[\"mu~ ['9-23~ https:~ https:~     4.7     204 Sports  show_5~
## 6     5 76ers P~ "['rew~ ['10-0~ https:~ https:~     4.6     108 Sports  show_2~
## # ... with 90 more variables: episode_filename_prefix <fct>, duration <fct>,
## #   F0semitoneFrom27.5Hz_sma3nz_amean <dbl>,
## #   F0semitoneFrom27.5Hz_sma3nz_stddevNorm <dbl>,
## #   F0semitoneFrom27.5Hz_sma3nz_percentile20.0 <dbl>,
## #   F0semitoneFrom27.5Hz_sma3nz_percentile50.0 <dbl>,
## #   F0semitoneFrom27.5Hz_sma3nz_percentile80.0 <dbl>,
## #   F0semitoneFrom27.5Hz_sma3nz_pctlrange0.2 <dbl>, ...
```

Statistical Testing

We are focusing on six audio features: pitch, pitch modulation, volume, volume modulation, and speech rate. In our dataset, these are modeled by the following, respectively: F0semitoneFrom27.5Hz_sma3nz_amean, jitterLocal_sma3nz_amean, loudness_sma3_amean, loudness_sma3_stddevNorm, and VoicedSegmentsPerSec. We also have two independent variables: average rating and average engagement. In our dataset, these are modeled by the following, respectively: apple_stars and apple_ratings.

We will subset our data to obtain two buckets: high and low success podcasts. Our podcasts with both star ratings and engagement ratings above the median of the dataset are considered high success, and podcasts with both metrics below the median of the dataset are considered low success.

```
# split the data into two groups based on median of ratings and engagement
top_stars <- quantile(data1$apple_stars, 0.9, na.rm = TRUE)
top_ratings <- quantile(data1$apple_ratings, 0.9, na.rm = TRUE)
print(top_stars)
```

```
## 90%
##   5
```

```
print(top_ratings)
```

```
##   90%
## 831.8
```

```
low_stars <- quantile(data1$apple_stars, 0.1, na.rm = TRUE)
low_ratings <- quantile(data1$apple_ratings, 0.1, na.rm = TRUE)
print(low_stars)
```

```
## 10%
## 4.3
```

```
print(low_ratings)
```

```
## 10%
##   4
```

```
median_stars = median(data1$apple_stars, na.rm = TRUE)
median_ratings = median(data1$apple_ratings, na.rm = TRUE)

success <- as.factor(case_when(
    data1$apple_stars < median_stars & data1$apple_ratings < median_ratings ~ "low",
    data1$apple_stars > median_stars & data1$apple_ratings > median_ratings ~ "high",
    TRUE ~ "medium"
  ))

#success <- as.factor(case_when(
#    data1$apple_stars <= low_stars & data1$apple_ratings <= low_ratings ~ "low",
#    data1$apple_stars >= top_stars & data1$apple_ratings >= top_ratings ~ "high",
#    TRUE ~ "medium"
#  ))

data1 <- data1 %>%
  mutate(data1, success) %>%
  relocate(success, .after = apple_ratings)
print(data1)
```

```
## # A tibble: 423 x 101
```

```
##        X show_~1 episo~2 episo~3 rss_l~4 apple~5 apple~6 apple~7 success apple~8
##    <int> <fct>   <fct>   <fct>   <fct>   <fct>     <dbl>   <int> <fct>   <fct>
##  1     0 Kop On~ "[\"ep~ ['2-26~ https:~ https:~     4.5      22 low     Sports
##  2     1 2 Mass~ "['cre~ ['5-14~ https:~ https:~     5        25 medium  Health~
##  3     2 2. Bun~ "['2. ~ ['12-0~ https:~ https:~     4.9      17 medium  Sports
##  4     3 2Minut~ "['epi~ ['7-13~ https:~ https:~     5        18 medium  Sports
##  5     4 4th & ~ "[\"mu~ ['9-23~ https:~ https:~     4.7     204 medium  Sports
##  6     5 76ers ~ "['rew~ ['10-0~ https:~ https:~     4.6     108 medium  Sports
##  7     6 81 All~ "['ser~ ['9-29~ https:~ https:~     4.9      22 medium  Sports
##  8     7 90 Day~ "['rai~ ['9-13~ https:~ https:~     4.5     146 medium  TV & F~
##  9     8 90s Ba~ "['a p~ ['4-25~ https:~ https:~     4.9     602 high    Comedy
## 10     9 A Brea~ "['whe~ ['11-2~ https:~ https:~     4.2      36 low     Societ~
## # ... with 413 more rows, 91 more variables: show_filename_prefix <fct>,
## #   episode_filename_prefix <fct>, duration <fct>,
## #   F0semitoneFrom27.5Hz_sma3nz_amean <dbl>,
## #   F0semitoneFrom27.5Hz_sma3nz_stddevNorm <dbl>,
## #   F0semitoneFrom27.5Hz_sma3nz_percentile20.0 <dbl>,
## #   F0semitoneFrom27.5Hz_sma3nz_percentile50.0 <dbl>,
## #   F0semitoneFrom27.5Hz_sma3nz_percentile80.0 <dbl>, ...
```

```
data2 <- data1 %>%
  filter(success == "low" | success == "high")
print(count(data2))
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   190
```

Hypothesis 1

Hypothesis 1: Podcasts with higher ratings and engagement will differ significantly from Podcasts with lower ratings and engagement across numerous acoustic features.

```
# conduct t-tests for each acoustic feature comparing high and low success groups
ttest_F0semitoneFrom27.5Hz <- t.test(data2$F0semitoneFrom27.5Hz_sma3nz_amean ~ data2$success, alternati
ttest_jitterLocal <- t.test(data2$jitterLocal_sma3nz_amean ~ data2$success, alternative = "two.sided")
ttest_loudness_amean <- t.test(data2$loudness_sma3_amean ~ data2$success, alternative = "two.sided")
ttest_loudness_stddevNorm <- t.test(data2$loudness_sma3_stddevNorm ~ data2$success, alternative = "two.
ttest_VoicedSegmentsPerSec <- t.test(data2$VoicedSegmentsPerSec ~ data2$success, alternative = "two.sid

# perform Benjamini-Hochberg correction
pvalues <- c(ttest_F0semitoneFrom27.5Hz$p.value, ttest_jitterLocal$p.value, ttest_loudness_amean$p.valu
adjusted_pvalues <- p.adjust(pvalues, method = "BH")

# print the p-values and adjusted p-values for each t-test
print(paste("T-test for F0semitoneFrom27.5Hz: p-value =", ttest_F0semitoneFrom27.5Hz$p.value, ", adjust
```

```
## [1] "T-test for F0semitoneFrom27.5Hz: p-value = 0.106841703126624 , adjusted p-value = 0.28951858067
```

```
print(paste("T-test for jitterLocal: p-value =", ttest_jitterLocal$p.value, ", adjusted p-value =", adj
```

```
## [1] "T-test for jitterLocal: p-value = 0.201163791763453 , adjusted p-value = 0.335272986272422"
```

```
print(paste("T-test for loudness_amean: p-value =", ttest_loudness_amean$p.value, ", adjusted p-value =
```

```
## [1] "T-test for loudness_amean: p-value = 0.115807432268201 , adjusted p-value = 0.289518580670504"
```

```
print(paste("T-test for loudness_stddevNorm: p-value =", ttest_loudness_stddevNorm$p.value, ", adjusted
```

```
## [1] "T-test for loudness_stddevNorm: p-value = 0.748082088925546 , adjusted p-value = 0.748082088925
```

```
print(paste("T-test for VoicedSegmentsPerSec: p-value =", ttest_VoicedSegmentsPerSec$p.value, ", adjust
```

```
## [1] "T-test for VoicedSegmentsPerSec: p-value = 0.357366547504361 , adjusted p-value = 0.446708184380
```

Hypothesis 2

Hypothesis 2: Podcasts with higher ratings and engagement will have significantly: higher pitch, more pitch modulation, higher average volume, more volume modulation, and higher speech rate.

To test this hypothesis, we need to compare the median values of each acoustic feature for podcasts with high versus low ratings and engagement. Podcasts which have higher star ratings AND a higher number of raters on Apple Podcasts are considered high success podcasts, while podcasts with lower star ratings and a lower number of raters on Apple Podcasts are considered low success podcasts.

We subset the data into a high and low success groups. We then conducts t-tests for each acoustic feature, comparing the high success group to the low success group. The resulting p-values are stored in a vector, and the Benjamini-Hochberg correction is applied using the p.adjust function. Finally, the p-values with the Benjamini-Hochberg correction are printed for each feature.

```
# conduct t-tests for each acoustic feature comparing high and low success groups
ttest_F0semitoneFrom27.5Hz <- t.test(data2$F0semitoneFrom27.5Hz_sma3nz_amean ~ data2$success, alternati
ttest_jitterLocal <- t.test(data2$jitterLocal_sma3nz_amean ~ data2$success, alternative = "greater")
ttest_loudness_amean <- t.test(data2$loudness_sma3_amean ~ data2$success, alternative = "greater")
ttest_loudness_stddevNorm <- t.test(data2$loudness_sma3_stddevNorm ~ data2$success, alternative = "grea
ttest_VoicedSegmentsPerSec <- t.test(data2$VoicedSegmentsPerSec ~ data2$success, alternative = "greater

# perform Benjamini-Hochberg correction
pvalues <- c(ttest_F0semitoneFrom27.5Hz$p.value, ttest_jitterLocal$p.value, ttest_loudness_amean$p.valu
adjusted_pvalues <- p.adjust(pvalues, method = "BH")

# print the p-values and adjusted p-values for each t-test
print(paste("T-test for F0semitoneFrom27.5Hz: p-value =", ttest_F0semitoneFrom27.5Hz$p.value, ", adjust
```

```
## [1] "T-test for F0semitoneFrom27.5Hz: p-value = 0.053420851563312 , adjusted p-value = 0.144759290335
```

```
print(paste("T-test for jitterLocal: p-value =", ttest_jitterLocal$p.value, ", adjusted p-value =", adj
```

```
## [1] "T-test for jitterLocal: p-value = 0.899418104118273 , adjusted p-value = 0.899418104118273"
```

```
print(paste("T-test for loudness_amean: p-value =", ttest_loudness_amean$p.value, ", adjusted p-value =
```

```
## [1] "T-test for loudness_amean: p-value = 0.0579037161341007 , adjusted p-value = 0.144759290335252"
```

```
print(paste("T-test for loudness_stddevNorm: p-value =", ttest_loudness_stddevNorm$p.value, ", adjusted
```

```
## [1] "T-test for loudness_stddevNorm: p-value = 0.625958955537227 , adjusted p-value = 0.782448694421
```

```
print(paste("T-test for VoicedSegmentsPerSec: p-value =", ttest_VoicedSegmentsPerSec$p.value, ", adjust
```

```
## [1] "T-test for VoicedSegmentsPerSec: p-value = 0.17868327375218 , adjusted p-value = 0.297805456253
```

Additional Testing

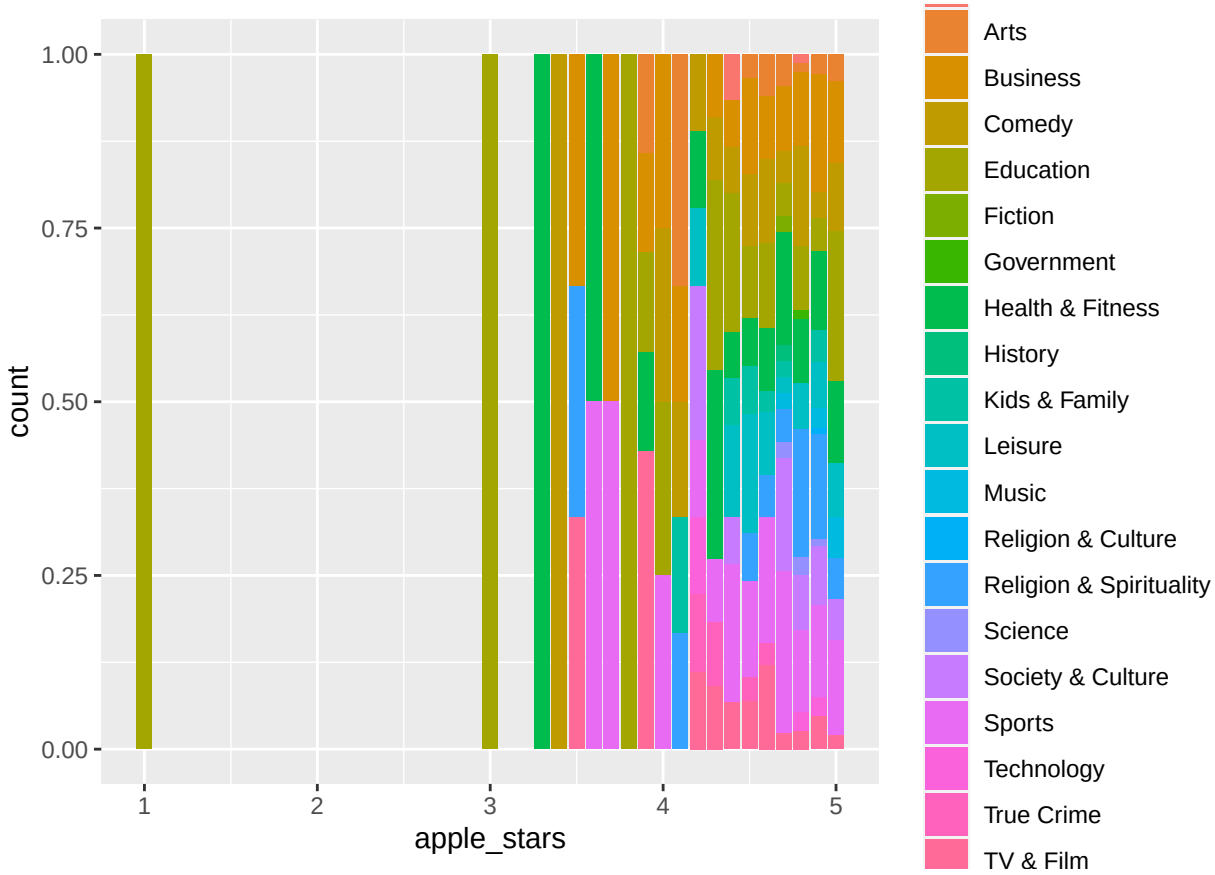
Grouping by Number of Ratings Alone

Grouping by Star Ratings Alone

Grouping by Show Genre

Visualizations

```
gg1 <- ggplot(data = data1) +
  geom_bar(mapping = aes(x = apple_stars, fill = apple_category), position = "fill")
gg1
```


1.00
0.75
0.50
0.25
0.00
count
1
2
3
4
5
apple_stars
Arts
Business
Comedy
Education
Fiction
Government
Health & Fitness
History
Kids & Family
Leisure
Music
Religion & Culture
Religion & Spirituality
Science
Society & Culture
Sports
Technology
True Crime
TV & Film