Spotify Podcasts vs. Audio Features: Statistical Testing

Lara Karacasu & Quinn Booth

2023-04-23

Data Preparation

```r
# 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)

# Replace empty string factor
levels(data1$apple_category)[1] <- "Not specified"

# 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>, ...
```

Grouping by Number of Ratings (for t-tests)

```r
#top_ratings <- quantile(data1$apple_ratings, 0.5, na.rm = TRUE)
#low_ratings <- quantile(data1$apple_ratings, 0.5, na.rm = TRUE)
```

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

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

data3 <- data1 %>%
  mutate(data1, ratings_success) %>%
  relocate(ratings_success, .after = apple_ratings)

data3 <- data3 %>%
  filter(ratings_success == "low" | ratings_success == "high") %>%
  drop_na()
print(count(data3))
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   398
```

Grouping by Star Ratings (for t-tests)

```r
#top_stars <- quantile(data1$apple_stars, 0.5, na.rm = TRUE)
#low_stars <- quantile(data1$apple_stars, 0.5, na.rm = TRUE)

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

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

data4 <- data1 %>%
  mutate(data1, stars_success) %>%
  relocate(stars_success, .after = apple_ratings)

data4 <- data4 %>%
  filter(stars_success == "low" | stars_success == "high") %>%
  drop_na()
print(count(data4))
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   326
```

Statistical Testing

We are focusing on five audio features: fundamental frequency, jitter, shimmer, HNR, and loudness. In our dataset, these are modeled by the following, respectively: F0semitoneFrom27.5Hz_sma3nz_amean, jitterLocal_sma3nz_amean, shimmerLocaldB_sma3nz_amean, HNRdBACF_sma3nz_amean, and loudness_sma3_amean 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.

Hypothesis 1

Podcasts with higher star ratings will differ significantly from podcasts with lower star ratings across numerous acoustic features.

Hypothesis 2

Podcasts with more raters will differ significantly from podcasts with fewer raters across numerous acoustic features.

Testing against number of ratings

```
# conduct t-tests for each acoustic feature comparing high and low success groups
ttest_fundamental_frequency <- t.test(data3$F0semitoneFrom27.5Hz_sma3nz_amean ~ data3$ratings_success, a
ttest_jitter <- t.test(data3$jitterLocal_sma3nz_amean ~ data3$ratings_success, alternative = "two.sided
ttest_shimmer <- t.test(data3$shimmerLocaldB_sma3nz_amean ~ data3$ratings_success, alternative = "two.s
ttest_HNR <- t.test(data3$HNRdBACF_sma3nz_amean ~ data3$ratings_success, alternative = "two.sided")
ttest_loudness <- t.test(data3$loudness_sma3_amean ~ data3$ratings_success, alternative = "two.sided")

# perform Benjamini-Hochberg correction
pvalues <- c(ttest_fundamental_frequency$p.value, ttest_jitter$p.value, ttest_shimmer$p.value, ttest_HN
adjusted_pvalues <- p.adjust(pvalues, method = "BH")

# print the p-values and adjusted p-values for each t-test
print(paste("T-test for fundamental frequency: p-value =", ttest_fundamental_frequency$p.value, ", adju
```

```
## [1] "T-test for fundamental frequency: p-value = 0.0193629494413261 , adjusted p-value = 0.02420368
```

```
print(paste("T-test for jitter: p-value =", ttest_jitter$p.value, ", adjusted p-value =", adjusted_pval
```

```
## [1] "T-test for jitter: p-value = 0.000661550780491062 , adjusted p-value = 0.00330775390245531"
```

```
print(paste("T-test for shimmer: p-value =", ttest_shimmer$p.value, ", adjusted p-value =", adjusted_pv
```

```
## [1] "T-test for shimmer: p-value = 0.0069267692193665 , adjusted p-value = 0.0115446153656108"
```

```
print(paste("T-test for HNR: p-value =", ttest_HNR$p.value, ", adjusted p-value =", adjusted_pvalues[4]
```

```
## [1] "T-test for HNR: p-value = 0.00202372956382691 , adjusted p-value = 0.00505932390956728"
```

```
print(paste("T-test for loudness: p-value =", ttest_loudness$p.value, ", adjusted p-value =", adjusted_
```

```
## [1] "T-test for loudness: p-value = 0.404544560030309 , adjusted p-value = 0.404544560030309"
```

Testing against stars

```
# conduct t-tests for each acoustic feature comparing high and low success groups
ttest_fundamental_frequency <- t.test(data4$F0semitoneFrom27.5Hz_sma3nz_amean ~ data4$stars_success, al
ttest_jitter <- t.test(data4$jitterLocal_sma3nz_amean ~ data4$stars_success, alternative = "two.sided")
ttest_shimmer <- t.test(data4$shimmerLocaldB_sma3nz_amean ~ data4$stars_success, alternative = "two.sid
ttest_HNR <- t.test(data4$HNRdBACF_sma3nz_amean ~ data4$stars_success, alternative = "two.sided")
ttest_loudness <- t.test(data4$loudness_sma3_amean ~ data4$stars_success, alternative = "two.sided")

# perform Benjamini-Hochberg correction
pvalues <- c(ttest_fundamental_frequency$p.value, ttest_jitter$p.value, ttest_shimmer$p.value, ttest_HN
adjusted_pvalues <- p.adjust(pvalues, method = "BH")

# print the p-values and adjusted p-values for each t-test
print(paste("T-test for fundamental frequency: p-value =", ttest_fundamental_frequency$p.value, ", adju
```

```
## [1] "T-test for fundamental frequency: p-value = 0.63097153372713 , adjusted p-value = 0.78871441715
```

```
print(paste("T-test for jitter: p-value =", ttest_jitter$p.value, ", adjusted p-value =", adjusted_pval
```

```
## [1] "T-test for jitter: p-value = 0.189409463009983 , adjusted p-value = 0.473523657524959"
```

```
print(paste("T-test for shimmer: p-value =", ttest_shimmer$p.value, ", adjusted p-value =", adjusted_pv
```

```
## [1] "T-test for shimmer: p-value = 0.29174154963596 , adjusted p-value = 0.486235916059934"
```

```
print(paste("T-test for HNR: p-value =", ttest_HNR$p.value, ", adjusted p-value =", adjusted_pvalues[4]
```

```
## [1] "T-test for HNR: p-value = 0.947461634283994 , adjusted p-value = 0.947461634283994"
```

```
print(paste("T-test for loudness: p-value =", ttest_loudness$p.value, ", adjusted p-value =", adjusted_
```

```
## [1] "T-test for loudness: p-value = 0.0846579934505076 , adjusted p-value = 0.423289967252538"
```

Hypothesis 3

Podcasts with higher ratings and engagement will have significantly: higher fundamental frequency, lower jitter, lower shimmer, lower HNR, greater loudness.

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.

Checking for negative relationship (high attribute = low rating/stars)

Testing against number of ratings

```
# conduct t-tests for each acoustic feature comparing high and low success groups
ttest_fundamental_frequency <- t.test(data3$F0semitoneFrom27.5Hz_sma3nz_amean ~ data3$ratings_success, a
ttest_jitter <- t.test(data3$jitterLocal_sma3nz_amean ~ data3$ratings_success, alternative = "less")
ttest_shimmer <- t.test(data3$shimmerLocaldB_sma3nz_amean ~ data3$ratings_success, alternative = "less")
ttest_HNR <- t.test(data3$HNRdBACF_sma3nz_amean ~ data3$ratings_success, alternative = "less")
ttest_loudness <- t.test(data3$loudness_sma3_amean ~ data3$ratings_success, alternative = "less")

# perform Benjamini-Hochberg correction
pvalues <- c(ttest_fundamental_frequency$p.value, ttest_jitter$p.value, ttest_shimmer$p.value, ttest_HNR
adjusted_pvalues <- p.adjust(pvalues, method = "BH")

# print the p-values and adjusted p-values for each t-test
print(paste("T-test for fundamental frequency: p-value =", ttest_fundamental_frequency$p.value, ", adju
```

```
## [1] "T-test for fundamental frequency: p-value = 0.990318525279337 , adjusted p-value = 0.99898813521
```

```
print(paste("T-test for jitter: p-value =", ttest_jitter$p.value, ", adjusted p-value =", adjusted_pvalu
```

```
## [1] "T-test for jitter: p-value = 0.000330775390245531 , adjusted p-value = 0.00165387695122766"
```

```
print(paste("T-test for shimmer: p-value =", ttest_shimmer$p.value, ", adjusted p-value =", adjusted_pva
```

```
## [1] "T-test for shimmer: p-value = 0.00346338460968325 , adjusted p-value = 0.00865846152420813"
```

```
print(paste("T-test for HNR: p-value =", ttest_HNR$p.value, ", adjusted p-value =", adjusted_pvalues[4]
```

```
## [1] "T-test for HNR: p-value = 0.998988135218086 , adjusted p-value = 0.998988135218086"
```

```
print(paste("T-test for loudness: p-value =", ttest_loudness$p.value, ", adjusted p-value =", adjusted_
```

```
## [1] "T-test for loudness: p-value = 0.797727719984846 , adjusted p-value = 0.998988135218086"
```

```
print(levels(data3$ratings_success))
```

```
## [1] "high"   "low"    "medium"
```

Testing against stars

```
# conduct t-tests for each acoustic feature comparing high and low success groups
ttest_fundamental_frequency <- t.test(data4$F0semitoneFrom27.5Hz_sma3nz_amean ~ data4$stars_success, al
ttest_jitter <- t.test(data4$jitterLocal_sma3nz_amean ~ data4$stars_success, alternative = "less")
ttest_shimmer <- t.test(data4$shimmerLocaldB_sma3nz_amean ~ data4$stars_success, alternative = "less")
ttest_HNR <- t.test(data4$HNRdBACF_sma3nz_amean ~ data4$stars_success, alternative = "less")
ttest_loudness <- t.test(data4$loudness_sma3_amean ~ data4$stars_success, alternative = "less")

# perform Benjamini-Hochberg correction
pvalues <- c(ttest_fundamental_frequency$p.value, ttest_jitter$p.value, ttest_shimmer$p.value, ttest_HN
adjusted_pvalues <- p.adjust(pvalues, method = "BH")

# print the p-values and adjusted p-values for each t-test
print(paste("T-test for fundamental frequency: p-value =", ttest_fundamental_frequency$p.value, ", adju
```

```
## [1] "T-test for fundamental frequency: p-value = 0.684514233136435 , adjusted p-value = 0.95767100032
```

```
print(paste("T-test for jitter: p-value =", ttest_jitter$p.value, ", adjusted p-value =", adjusted_pval
```

```
## [1] "T-test for jitter: p-value = 0.905295268495008 , adjusted p-value = 0.957671003274746"
```

```
print(paste("T-test for shimmer: p-value =", ttest_shimmer$p.value, ", adjusted p-value =", adjusted_pv
```

```
## [1] "T-test for shimmer: p-value = 0.85412922518202 , adjusted p-value = 0.957671003274746"
```

```
print(paste("T-test for HNR: p-value =", ttest_HNR$p.value, ", adjusted p-value =", adjusted_pvalues[4]
```

```
## [1] "T-test for HNR: p-value = 0.473730817141997 , adjusted p-value = 0.957671003274746"
```

```
print(paste("T-test for loudness: p-value =", ttest_loudness$p.value, ", adjusted p-value =", adjusted_
```

```
## [1] "T-test for loudness: p-value = 0.957671003274746 , adjusted p-value = 0.957671003274746"
```

Checking for positive relationship (high attribute = high rating/stars)

Testing against number of ratings

```r
# conduct t-tests for each acoustic feature comparing high and low success groups
ttest_fundamental_frequency <- t.test(data3$F0semitoneFrom27.5Hz_sma3nz_amean ~ data3$ratings_success, a
ttest_jitter <- t.test(data3$jitterLocal_sma3nz_amean ~ data3$ratings_success, alternative = "greater")
ttest_shimmer <- t.test(data3$shimmerLocaldB_sma3nz_amean ~ data3$ratings_success, alternative = "greate
ttest_HNR <- t.test(data3$HNRdBACF_sma3nz_amean ~ data3$ratings_success, alternative = "greater")
ttest_loudness <- t.test(data3$loudness_sma3_amean ~ data3$ratings_success, alternative = "greater")

# perform Benjamini-Hochberg correction
pvalues <- c(ttest_fundamental_frequency$p.value, ttest_jitter$p.value, ttest_shimmer$p.value, ttest_HNI
adjusted_pvalues <- p.adjust(pvalues, method = "BH")

# print the p-values and adjusted p-values for each t-test
print(paste("T-test for fundamental frequency: p-value =", ttest_fundamental_frequency$p.value, ", adju
```

```
## [1] "T-test for fundamental frequency: p-value = 0.00968147472066305 , adjusted p-value = 0.02420368
```

```r
print(paste("T-test for jitter: p-value =", ttest_jitter$p.value, ", adjusted p-value =", adjusted_pvalu
```

```
## [1] "T-test for jitter: p-value = 0.999669224609754 , adjusted p-value = 0.999669224609754"
```

```r
print(paste("T-test for shimmer: p-value =", ttest_shimmer$p.value, ", adjusted p-value =", adjusted_pva
```

```
## [1] "T-test for shimmer: p-value = 0.996536615390317 , adjusted p-value = 0.999669224609754"
```

```r
print(paste("T-test for HNR: p-value =", ttest_HNR$p.value, ", adjusted p-value =", adjusted_pvalues[4])
```

```
## [1] "T-test for HNR: p-value = 0.00101186478191346 , adjusted p-value = 0.00505932390956728"
```

```r
print(paste("T-test for loudness: p-value =", ttest_loudness$p.value, ", adjusted p-value =", adjusted_
```

```
## [1] "T-test for loudness: p-value = 0.202272280015154 , adjusted p-value = 0.337120466691924"
```

```r
print(levels(data3$ratings_success))
```

```
## [1] "high"   "low"    "medium"
```

Testing against stars

```r
# conduct t-tests for each acoustic feature comparing high and low success groups
ttest_fundamental_frequency <- t.test(data4$F0semitoneFrom27.5Hz_sma3nz_amean ~ data4$stars_success, al
ttest_jitter <- t.test(data4$jitterLocal_sma3nz_amean ~ data4$stars_success, alternative = "greater")
ttest_shimmer <- t.test(data4$shimmerLocaldB_sma3nz_amean ~ data4$stars_success, alternative = "greater
ttest_HNR <- t.test(data4$HNRdBACF_sma3nz_amean ~ data4$stars_success, alternative = "greater")
ttest_loudness <- t.test(data4$loudness_sma3_amean ~ data4$stars_success, alternative = "greater")

# perform Benjamini-Hochberg correction
```

```r
pvalues <- c(ttest_fundamental_frequency$p.value, ttest_jitter$p.value, ttest_shimmer$p.value, ttest_HNR
adjusted_pvalues <- p.adjust(pvalues, method = "BH")

# print the p-values and adjusted p-values for each t-test
print(paste("T-test for fundamental frequency: p-value =", ttest_fundamental_frequency$p.value, ", adju
```

```
## [1] "T-test for fundamental frequency: p-value = 0.315485766863565 , adjusted p-value = 0.39435720857
```

```r
print(paste("T-test for jitter: p-value =", ttest_jitter$p.value, ", adjusted p-value =", adjusted_pvalu
```

```
## [1] "T-test for jitter: p-value = 0.0947047315049917 , adjusted p-value = 0.236761828762479"
```

```r
print(paste("T-test for shimmer: p-value =", ttest_shimmer$p.value, ", adjusted p-value =", adjusted_pva
```

```
## [1] "T-test for shimmer: p-value = 0.14587077481798 , adjusted p-value = 0.243117958029967"
```

```r
print(paste("T-test for HNR: p-value =", ttest_HNR$p.value, ", adjusted p-value =", adjusted_pvalues[4]]
```

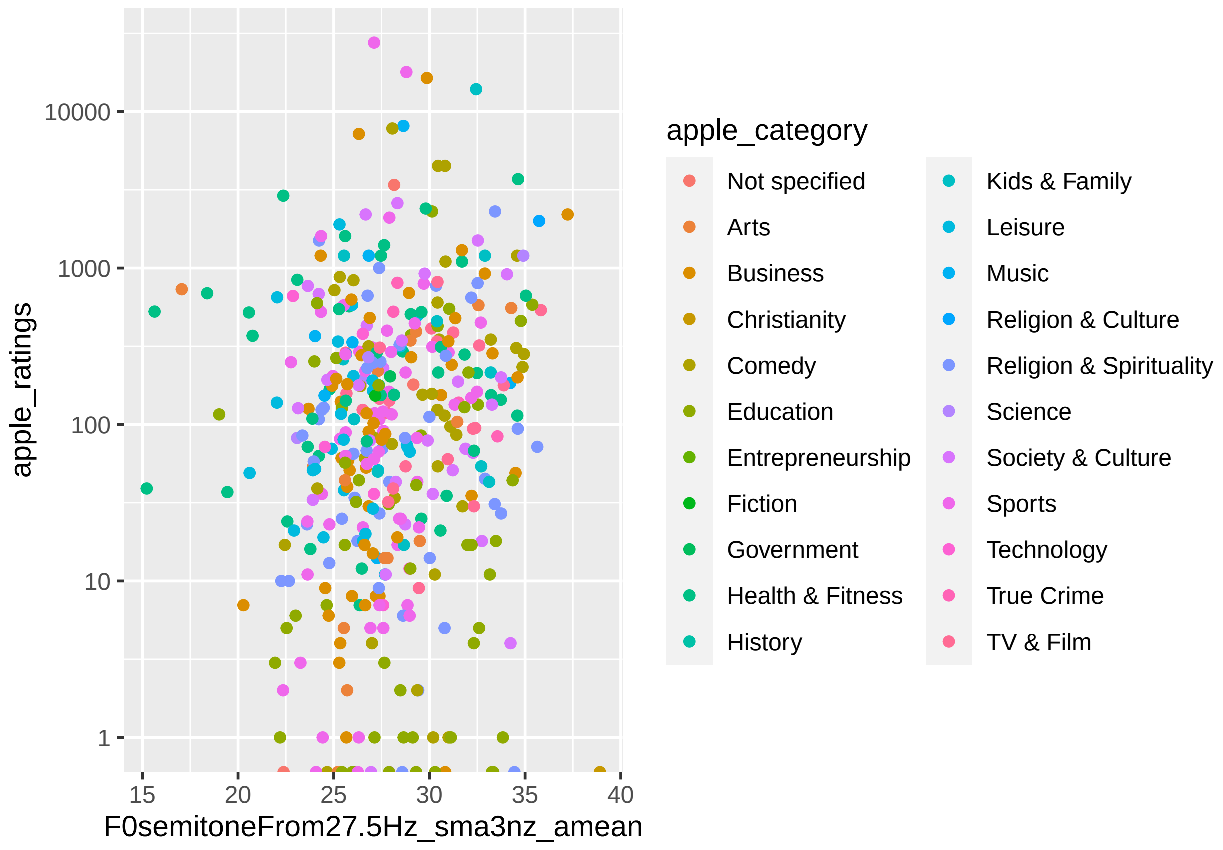
```
## [1] "T-test for HNR: p-value = 0.526269182858003 , adjusted p-value = 0.526269182858003"
```

```r
print(paste("T-test for loudness: p-value =", ttest_loudness$p.value, ", adjusted p-value =", adjusted_p
```

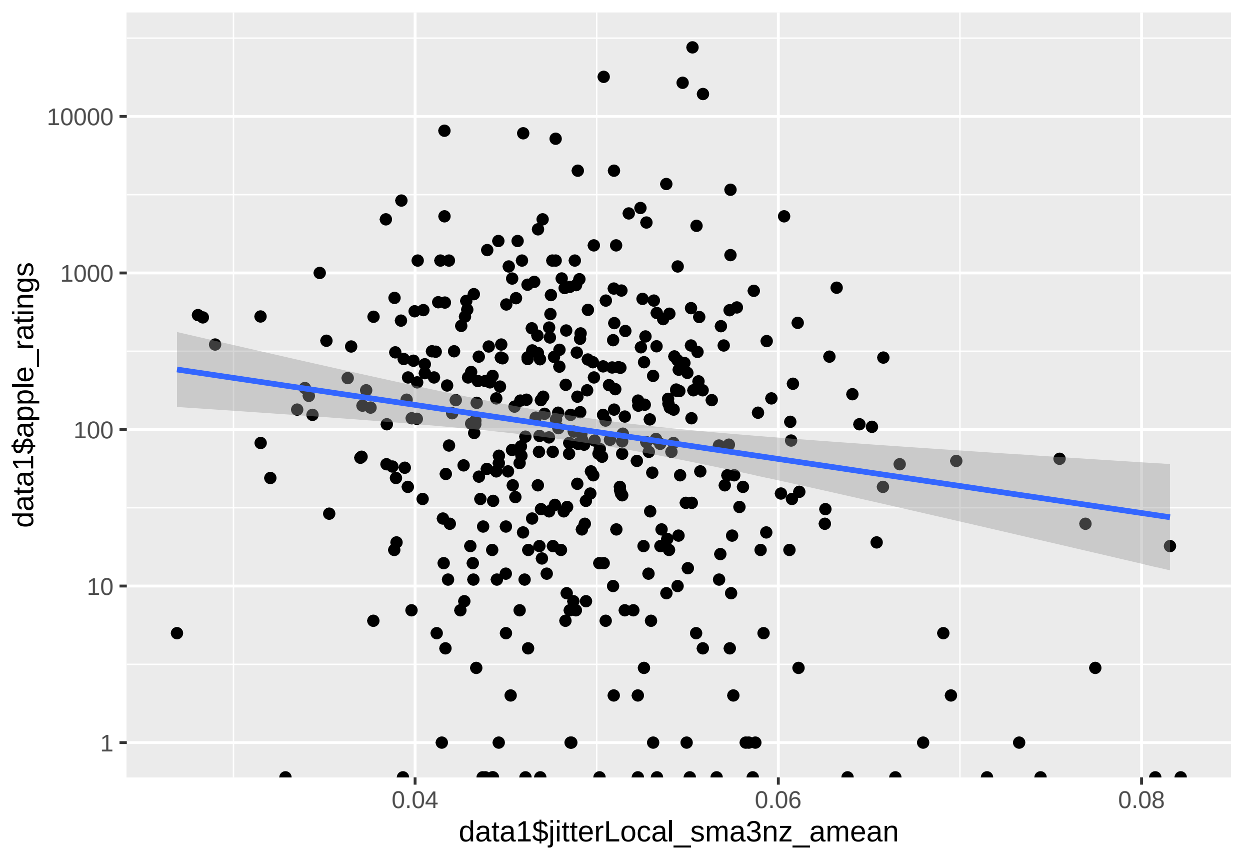
```
## [1] "T-test for loudness: p-value = 0.0423289967252538 , adjusted p-value = 0.211644983626269"
```

Visualizations of the Results

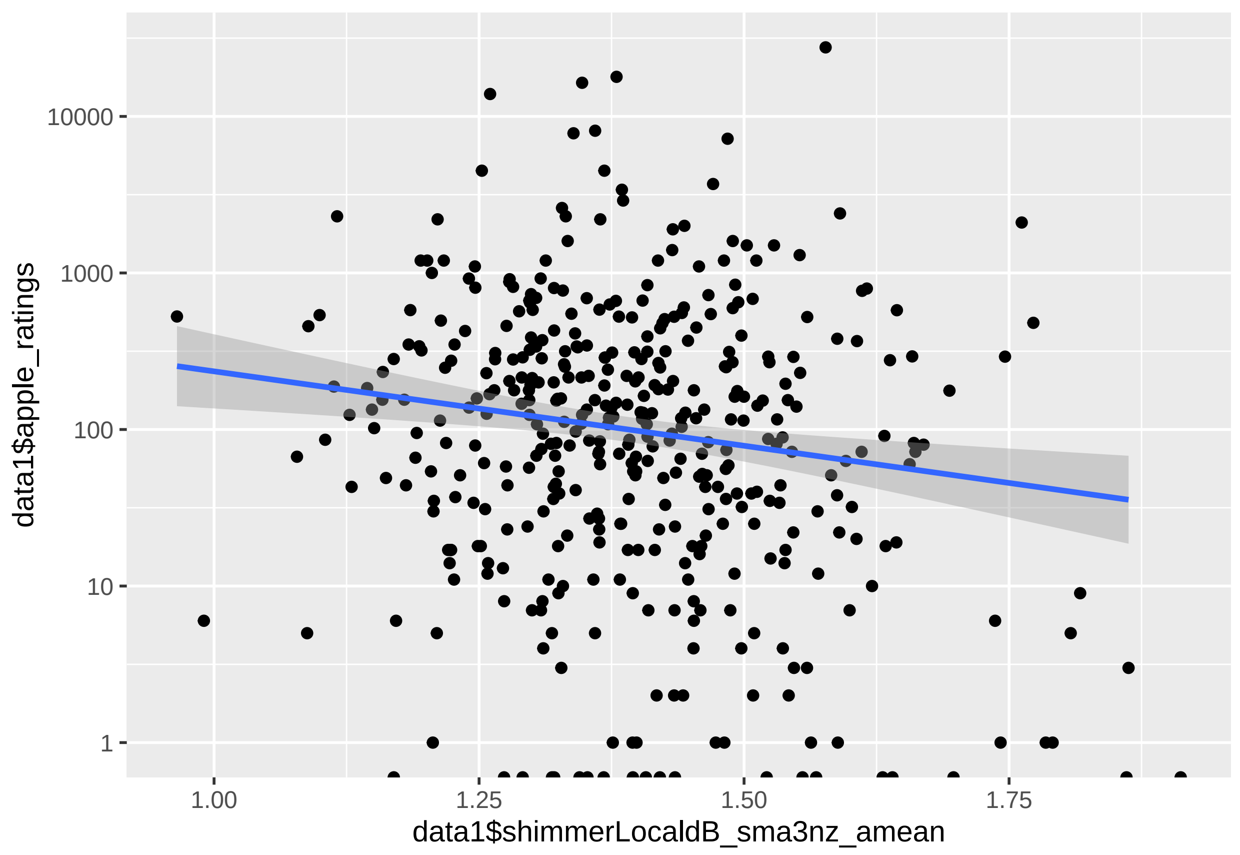
```r
# Fundamental Frequency by Category, by Number of Raters
gg3 <- ggplot(data = data1) +
  geom_point(mapping = aes(x = F0semitoneFrom27.5Hz_sma3nz_amean, y = apple_ratings, color = apple_categ
  scale_y_continuous(trans='log10')
gg3
```



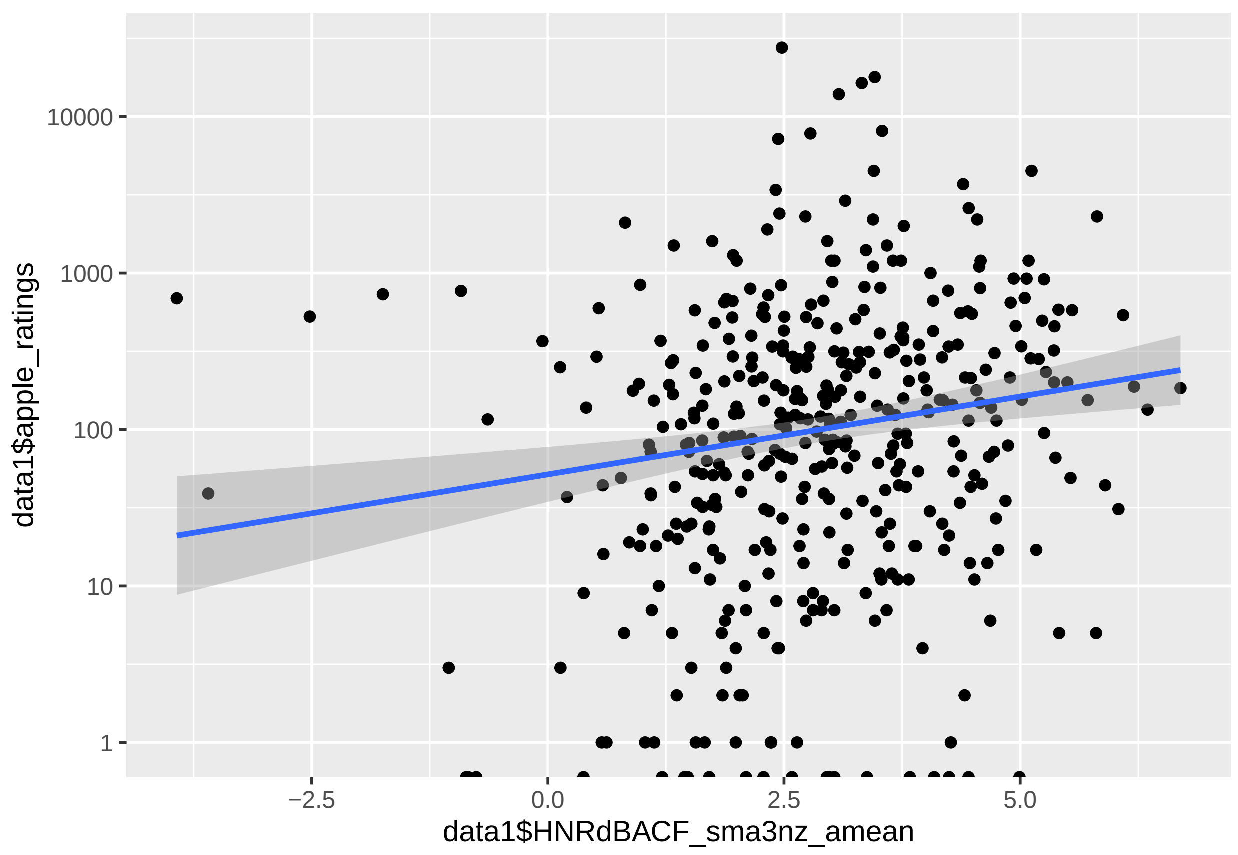
```r
# Jitter by Category, by Number of Raters

#gg3 <- ggplot(data = data1) +
#  geom_point(mapping = aes(x = jitterLocal_sma3nz_amean, y = apple_ratings, color = apple_category)) +
#  scale_y_continuous(trans='log10') +
#  geom_smooth(method = 'lm')
#gg3

gg3 <- ggplot(data=data1,aes(x=data1$jitterLocal_sma3nz_amean, y=data1$apple_ratings)) +
  geom_point() +
  geom_smooth(method="lm") +
  scale_y_continuous(trans='log10')
gg3
```



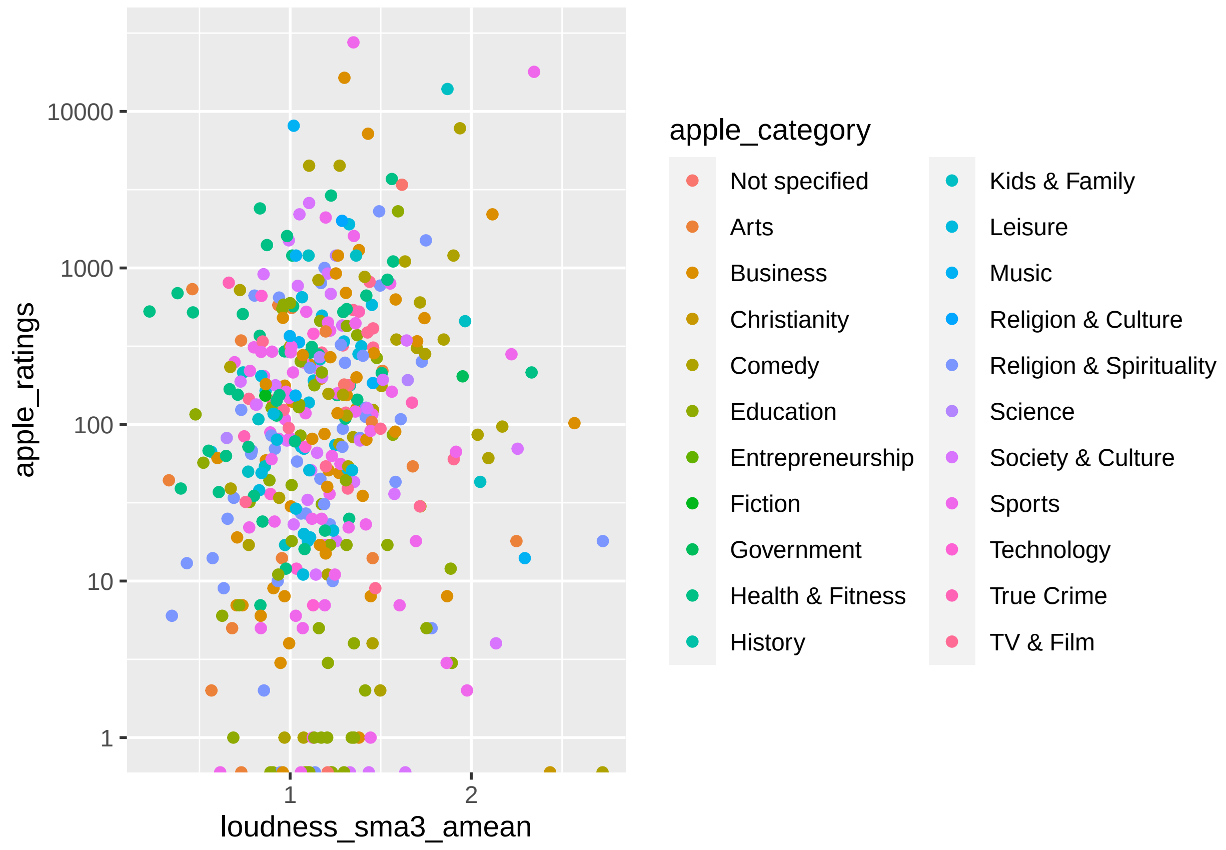
```
# Shimmer by Category, by Number of Raters

#gg3 <- ggplot(data = data1) +
#  geom_point(mapping = aes(x = shimmerLocaldB_sma3nz_amean, y = apple_ratings, color = apple_category)
#  scale_y_continuous(trans='log10')
#gg3

gg3 <- ggplot(data=data1,aes(x=data1$shimmerLocaldB_sma3nz_amean, y=data1$apple_ratings)) +
  geom_point() +
  geom_smooth(method="lm") +
  scale_y_continuous(trans='log10')
gg3
```



```
# HNR by Category, by Number of Raters

#gg3 <- ggplot(data = data1) +
#  geom_point(mapping = aes(x = HNRdBACF_sma3nz_amean, y = apple_ratings, color = apple_category)) +
#  scale_y_continuous(trans='log10')
#gg3

gg3 <- ggplot(data=data1,aes(x=data1$HNRdBACF_sma3nz_amean, y=data1$apple_ratings)) +
  geom_point() +
  geom_smooth(method="lm") +
  scale_y_continuous(trans='log10')
gg3
```

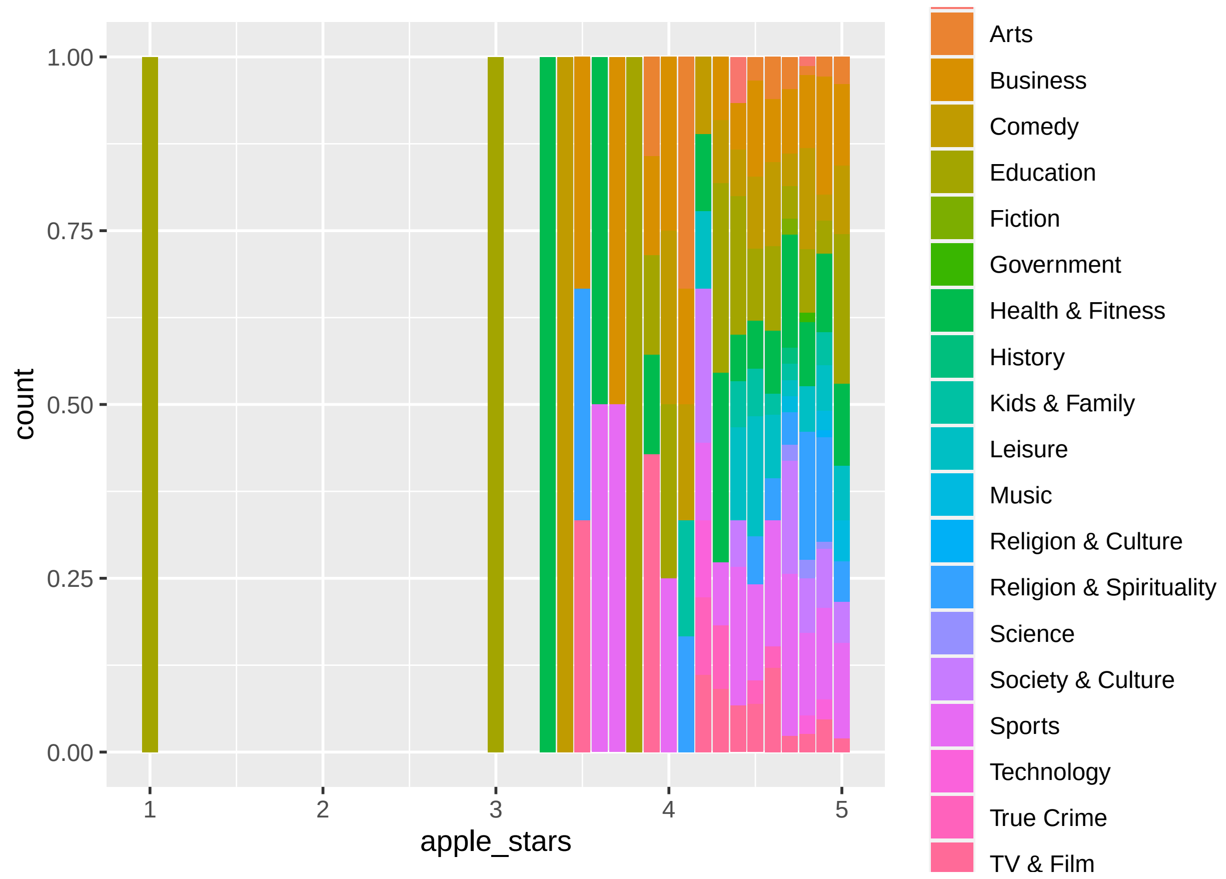


```
# Loudness by Category, by Number of Raters
gg3 <- ggplot(data = data1) +
  geom_point(mapping = aes(x = loudness_sma3_amean, y = apple_ratings, color = apple_category)) +
  scale_y_continuous(trans='log10')
gg3
```

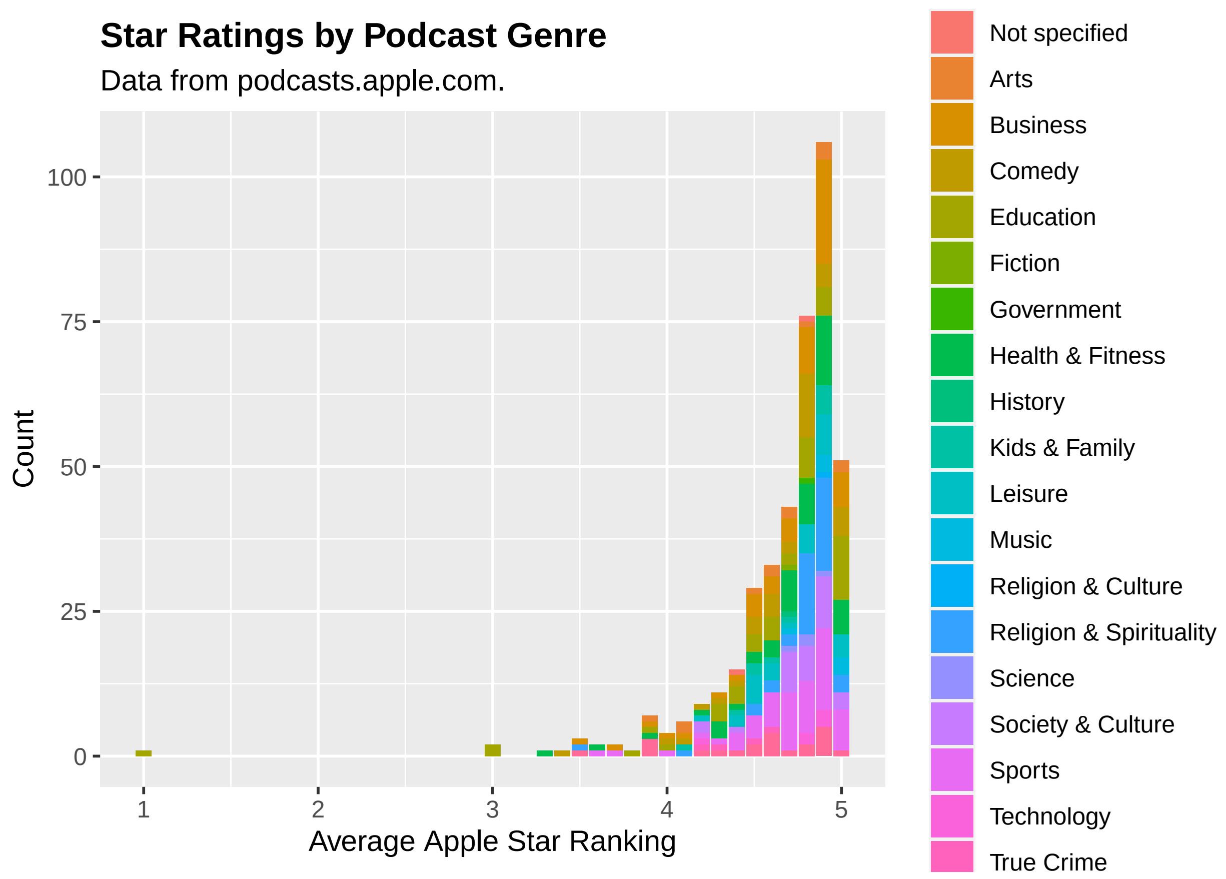


Misc. Visualizing the Dataset

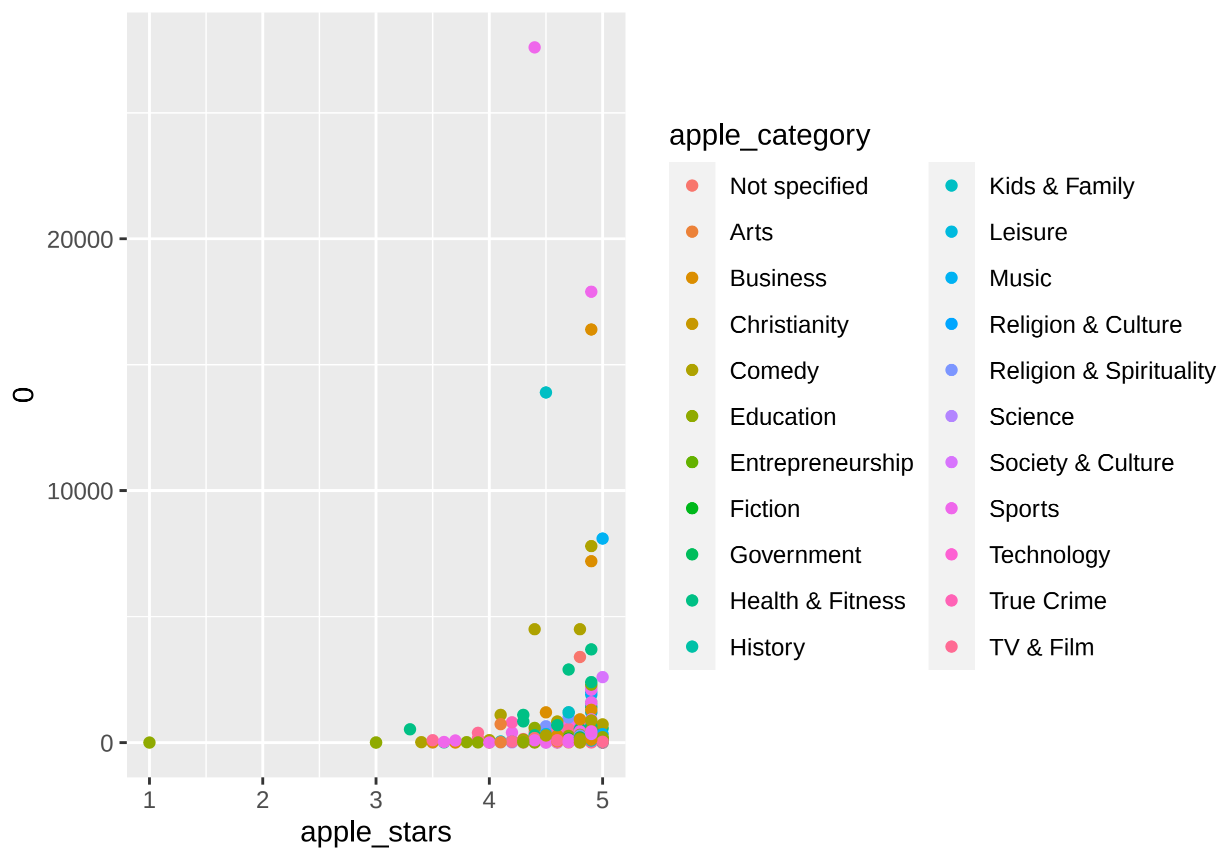
```
# Apple Stars by Category, By Proportion
gg1 <- ggplot(data = data1) +
  geom_bar(mapping = aes(x = apple_stars, fill = apple_category), position = "fill")
gg1
```



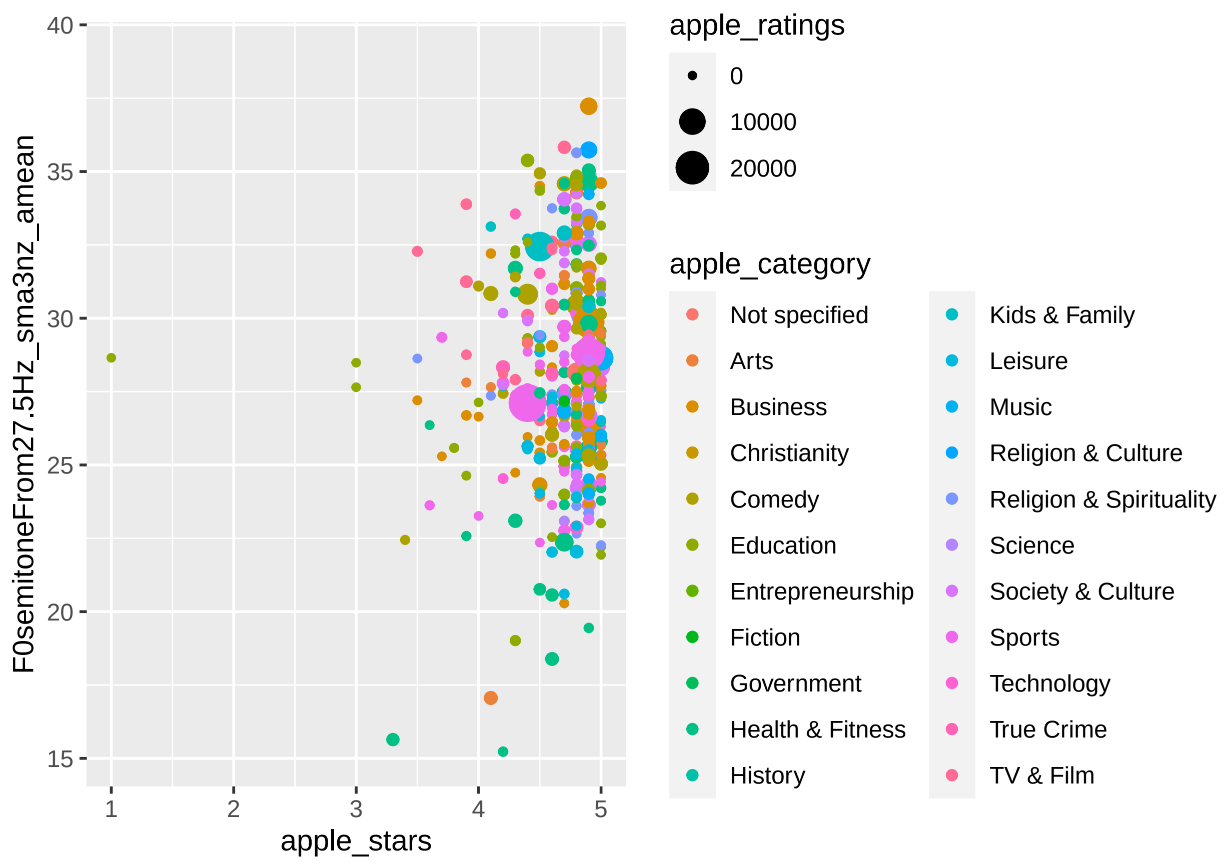
```
# Apple Stars by Category, by Count
gg2 <- ggplot(data = data1) +
  geom_bar(mapping = aes(x = apple_stars, fill = apple_category)) +
  labs(title = "Star Ratings by Podcast Genre",
       subtitle = "Data from podcasts.apple.com.",
       x = "Average Apple Star Ranking",
       y = "Count") +
  theme(plot.title=element_text(face = "bold")) +
   scale_fill_discrete(name = "Genre")
gg2
```



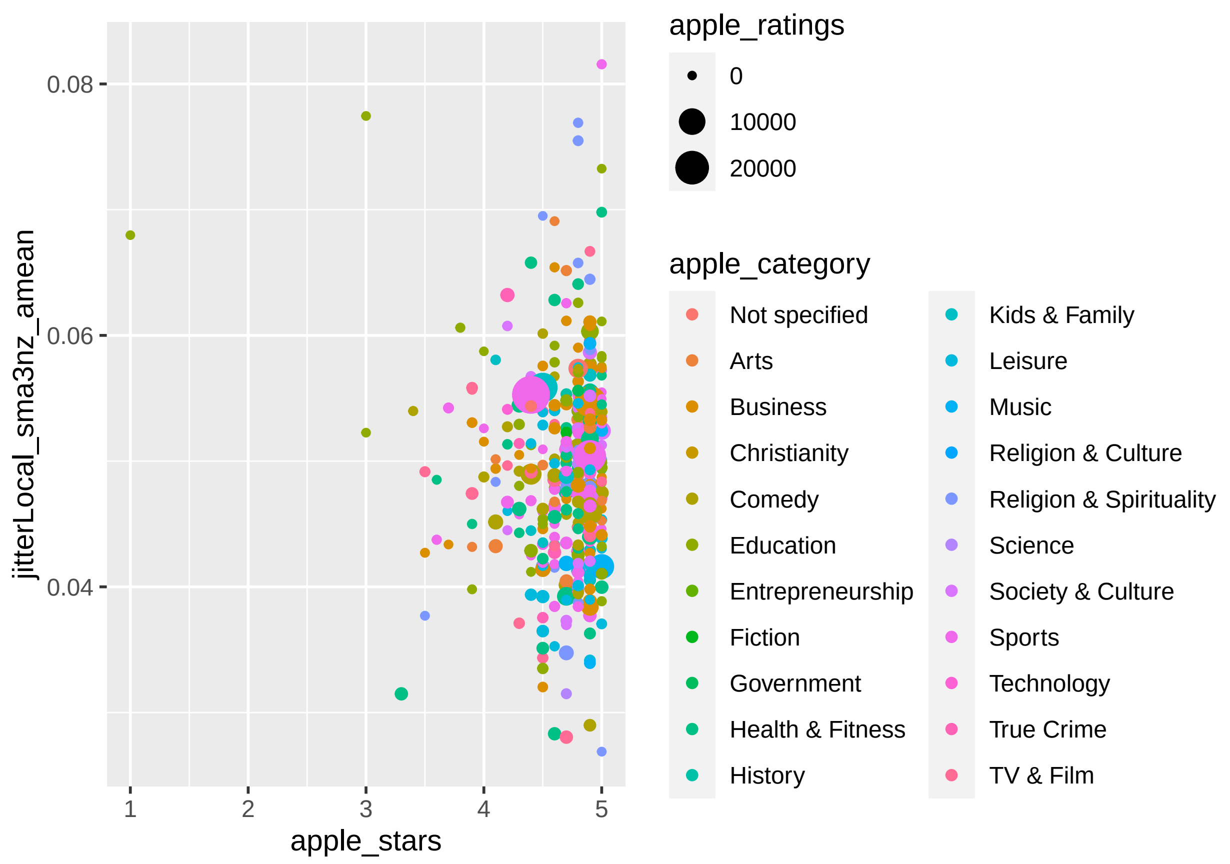
```r
# Apple Stars by Category, by Number of Raters
gg3 <- ggplot(data = data1) +
  geom_point(mapping = aes(x = apple_stars, y = apple_ratings, color = apple_category)) +
  scale_y_continuous(c(0, 30000, 1000))
gg3
```



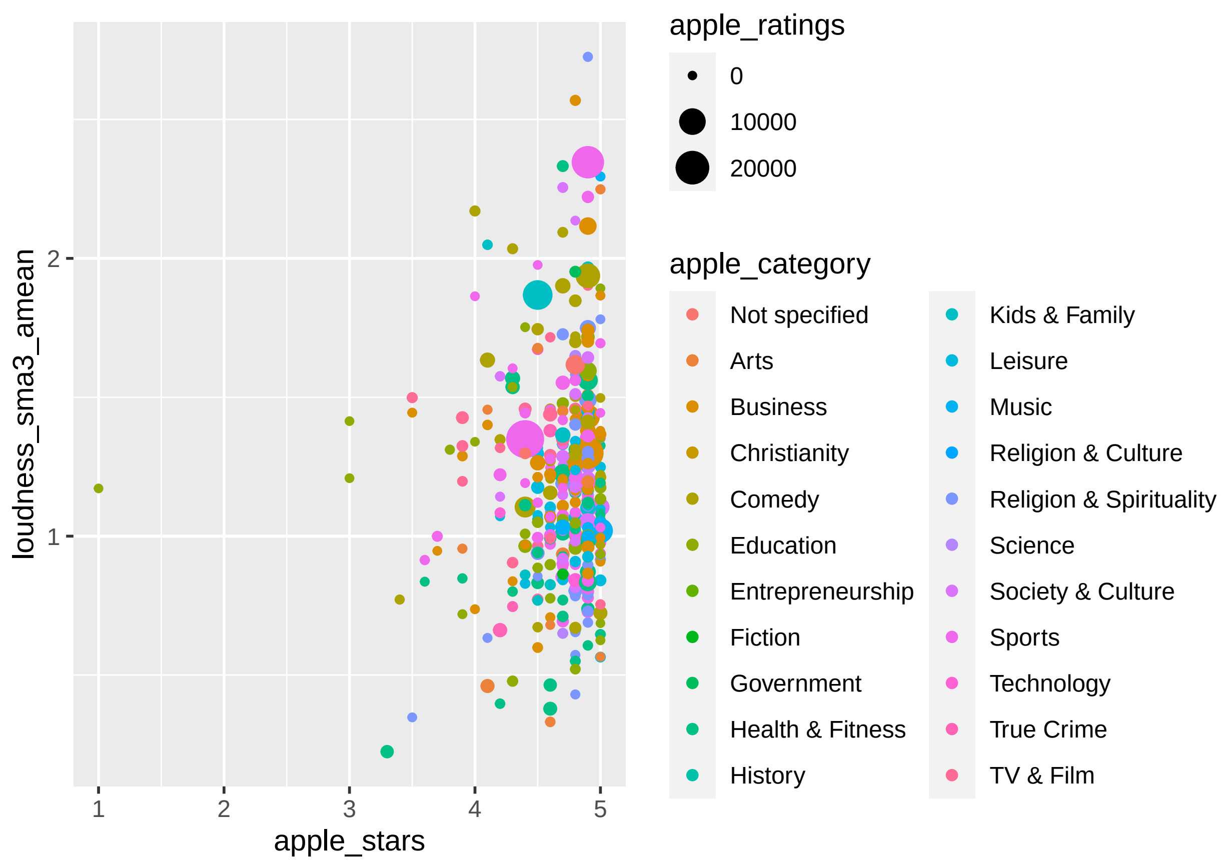
```
# Bubble Plot of Stars vs. Raters vs. Category
gg_b1 <- ggplot(data = data1, aes(x = apple_stars, y = F0semitoneFrom27.5Hz_sma3nz_amean, size = apple_
  geom_point()
gg_b1
```



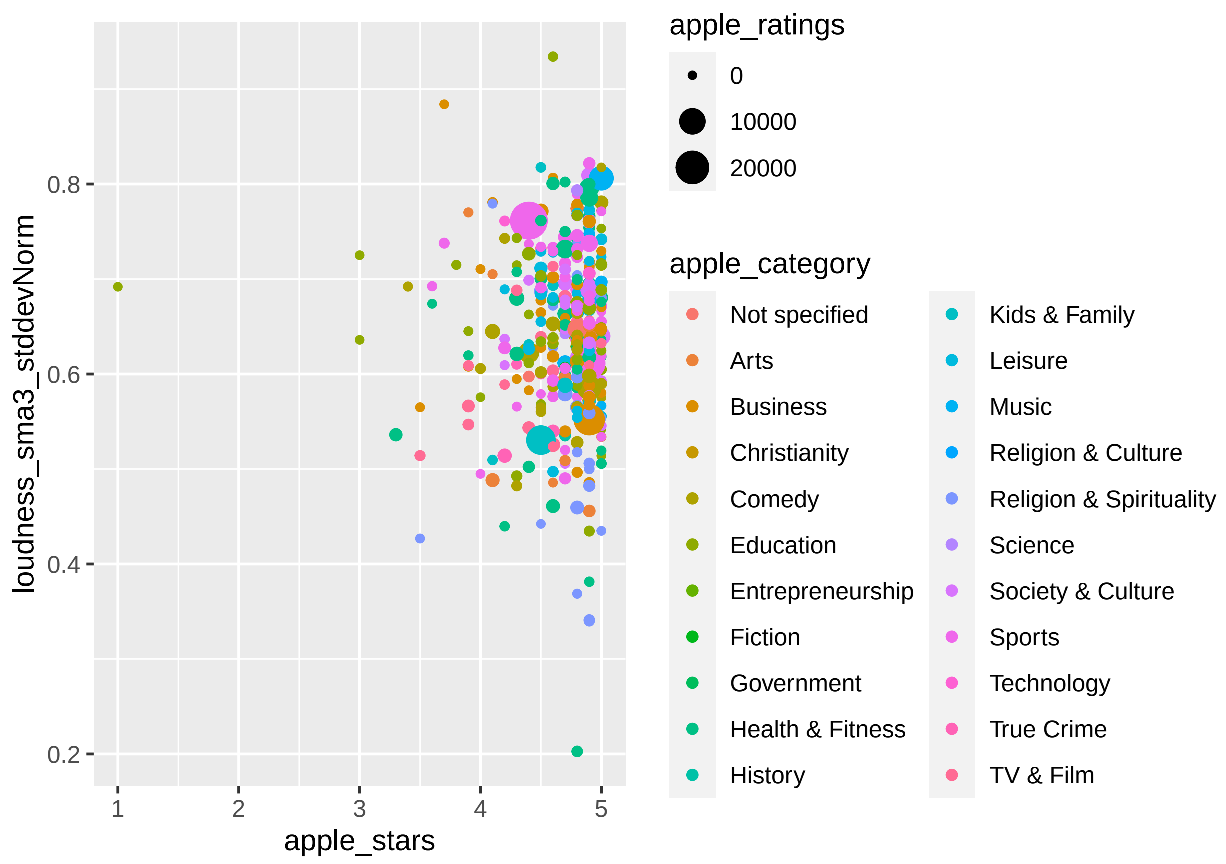
```r
# Bubble Plot of Stars vs. Raters vs. Category
gg_b2 <- ggplot(data = data1, aes(x = apple_stars, y = jitterLocal_sma3nz_amean, size = apple_ratings, c
  geom_point()
gg_b2
```



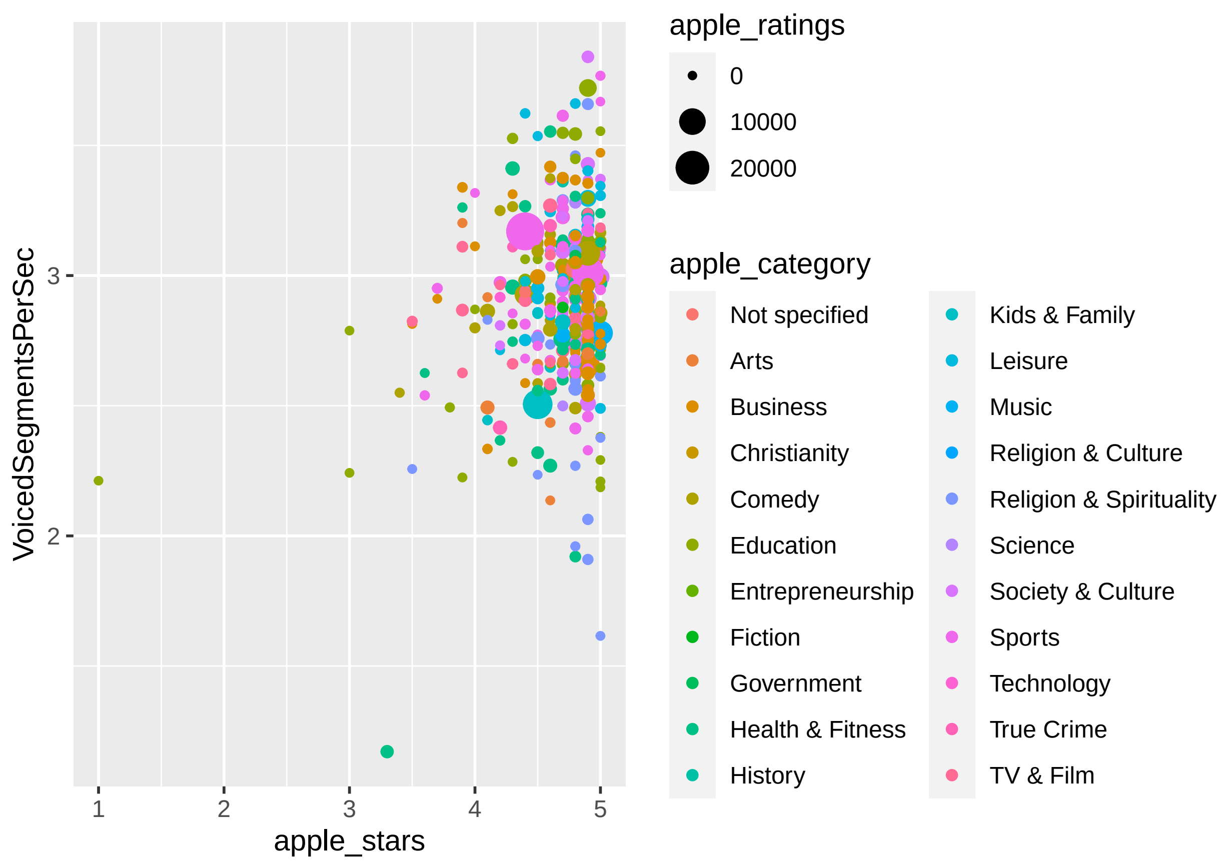
```
# Bubble Plot of Stars vs. Raters vs. Category
gg_b3 <- ggplot(data = data1, aes(x = apple_stars, y = loudness_sma3_amean, size = apple_ratings, color
  geom_point()
gg_b3
```



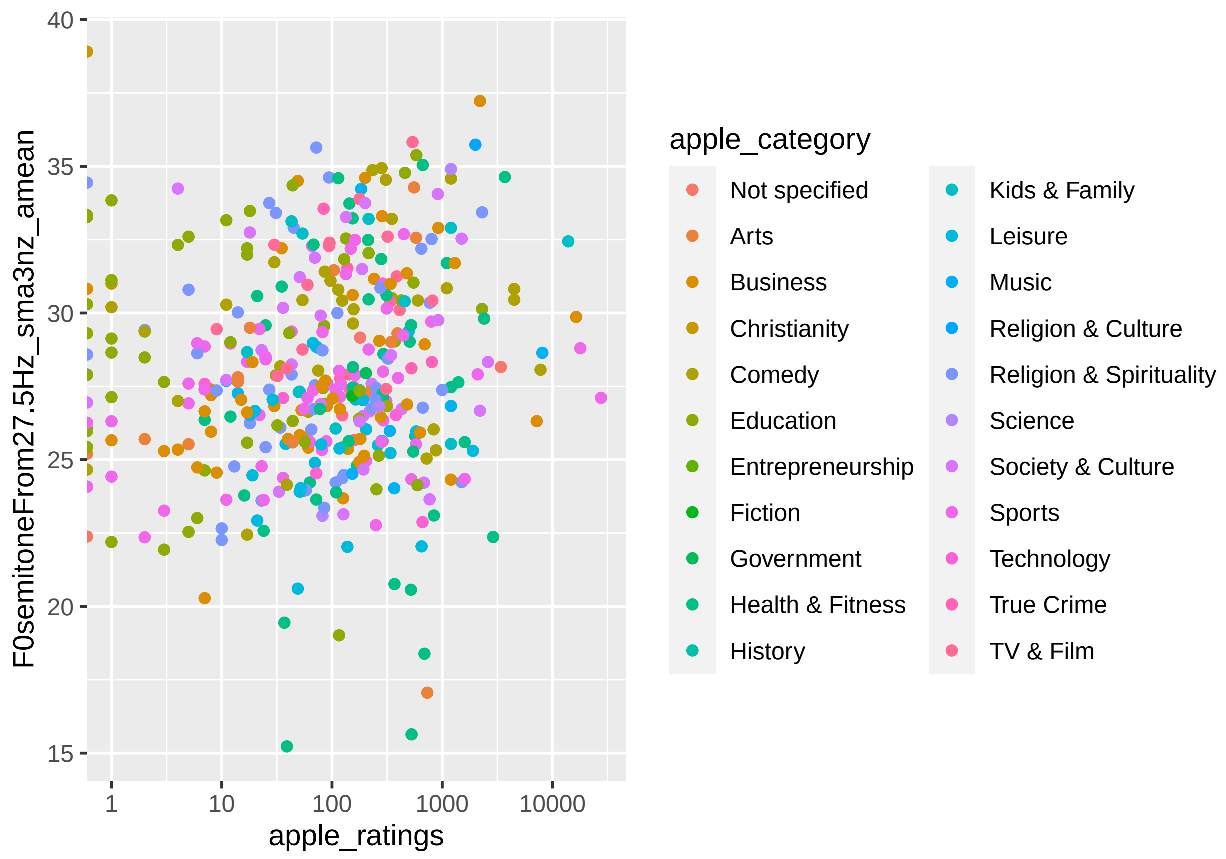
```
# Bubble Plot of Stars vs. Raters vs. Category
gg_b4 <- ggplot(data = data1, aes(x = apple_stars, y = loudness_sma3_stddevNorm, size = apple_ratings, 
  geom_point()
gg_b4
```



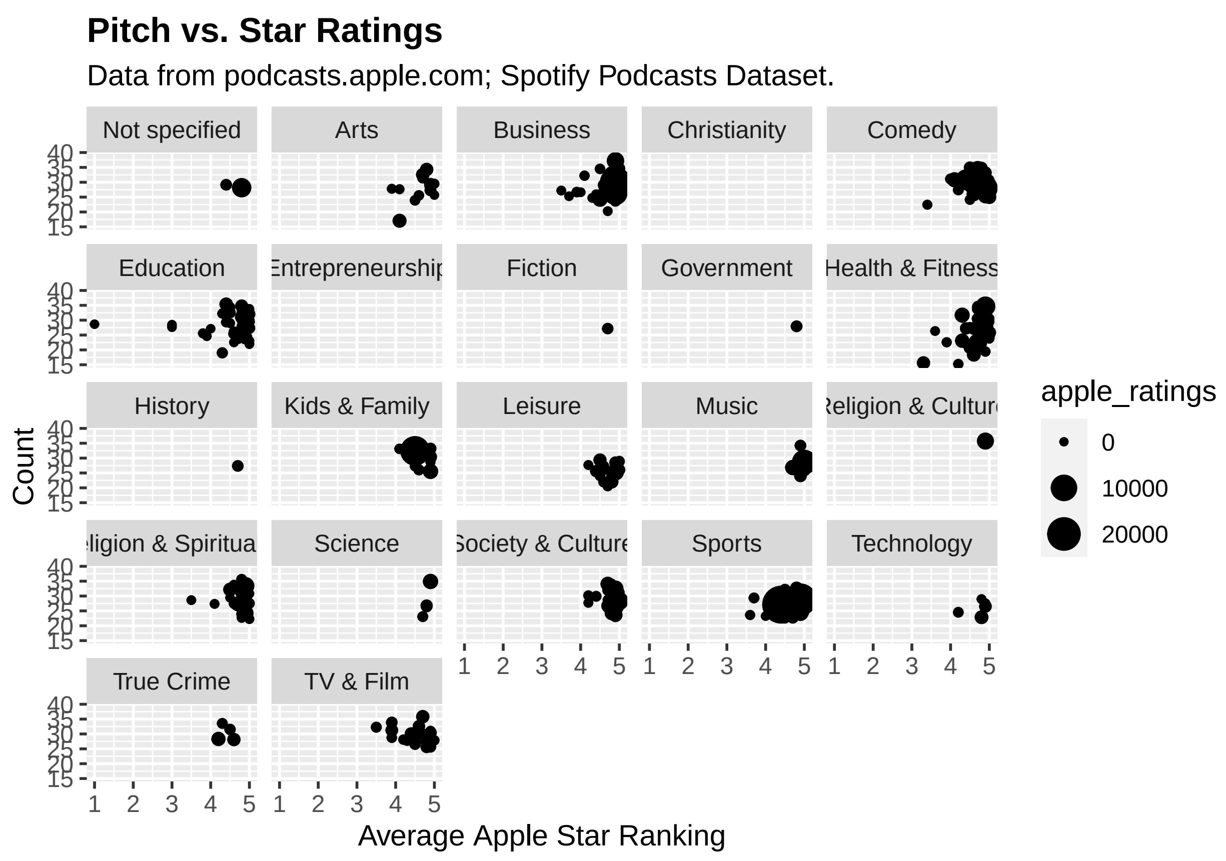
```
# Bubble Plot of Stars vs. Raters vs. Category
gg_b5 <- ggplot(data = data1, aes(x = apple_stars, y = VoicedSegmentsPerSec, size = apple_ratings, colo
  geom_point()
gg_b5
```



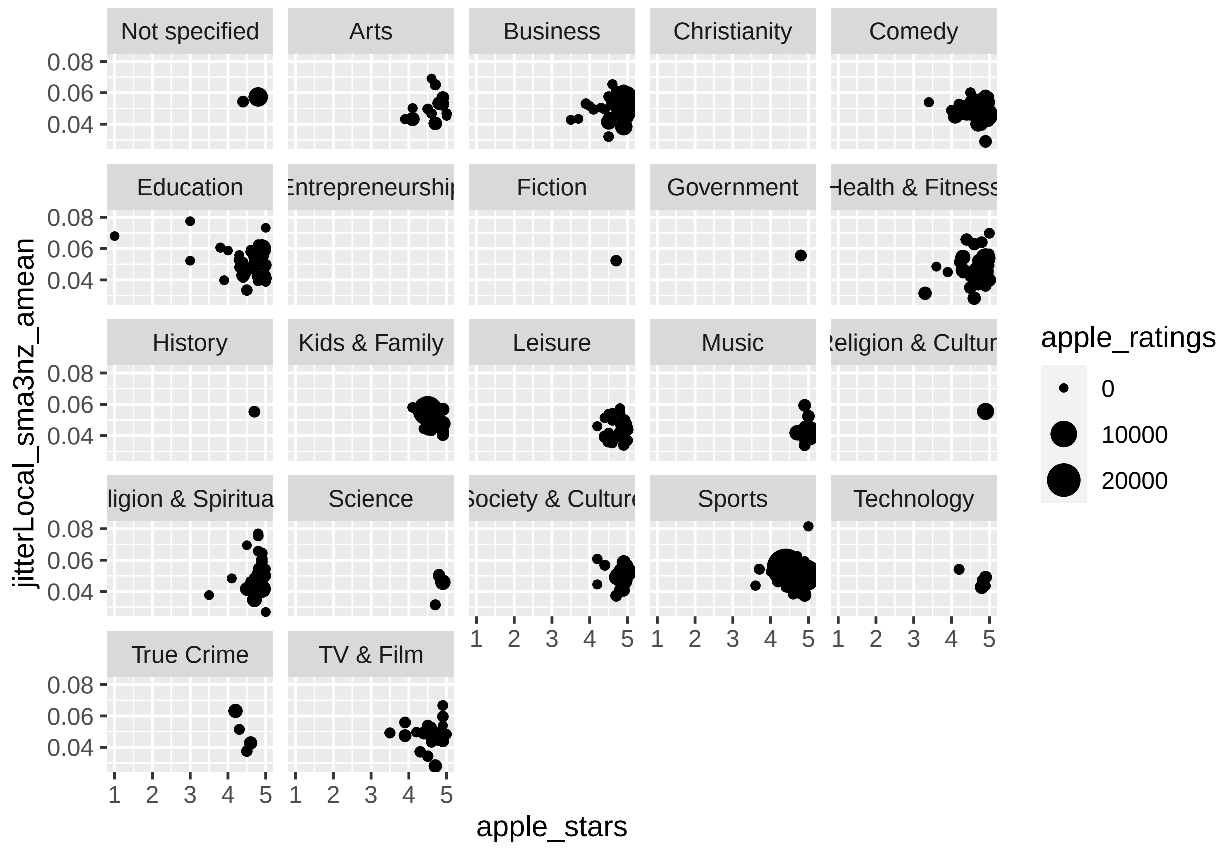
```
# Bubble Plot of Raters vs. Stars vs. Category
gg_r1 <- ggplot(data = data1, aes(x = apple_ratings, y = F0semitoneFrom27.5Hz_sma3nz_amean, color = appl
  geom_point() +
    scale_x_continuous(trans='log10')
gg_r1
```



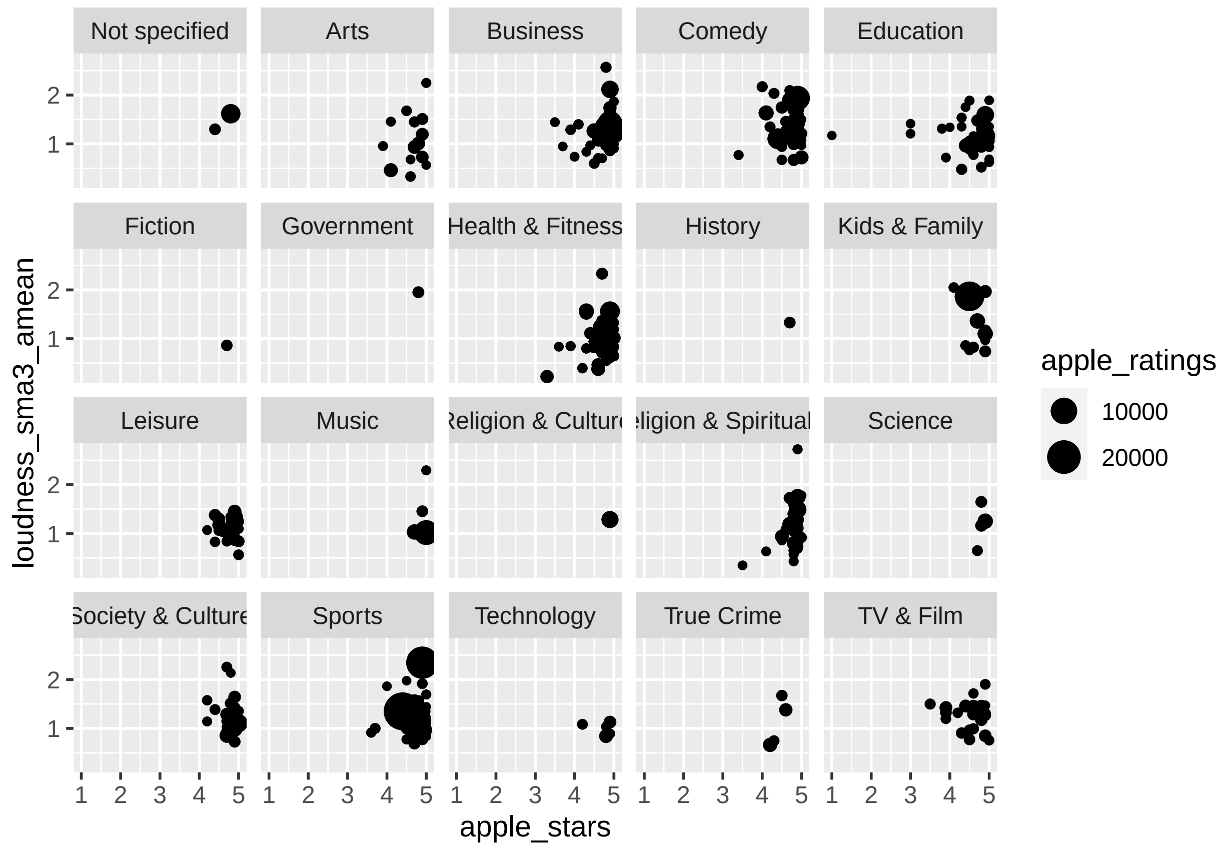
```
# Facet Wrap of Stars vs. Raters vs. Category for features

gg5 <- ggplot(data = data1) +
  geom_point(mapping = aes(x = apple_stars, y = F0semitoneFrom27.5Hz_sma3nz_amean, size = apple_ratings
  facet_wrap(~ apple_category) +
  labs(title = "Pitch vs. Star Ratings",
       subtitle = "Data from podcasts.apple.com; Spotify Podcasts Dataset.",
       x = "Average Apple Star Ranking",
       y = "Count") +
  theme(plot.title=element_text(face = "bold")) +
   scale_fill_discrete(name = "Number of Raters")
gg5
```



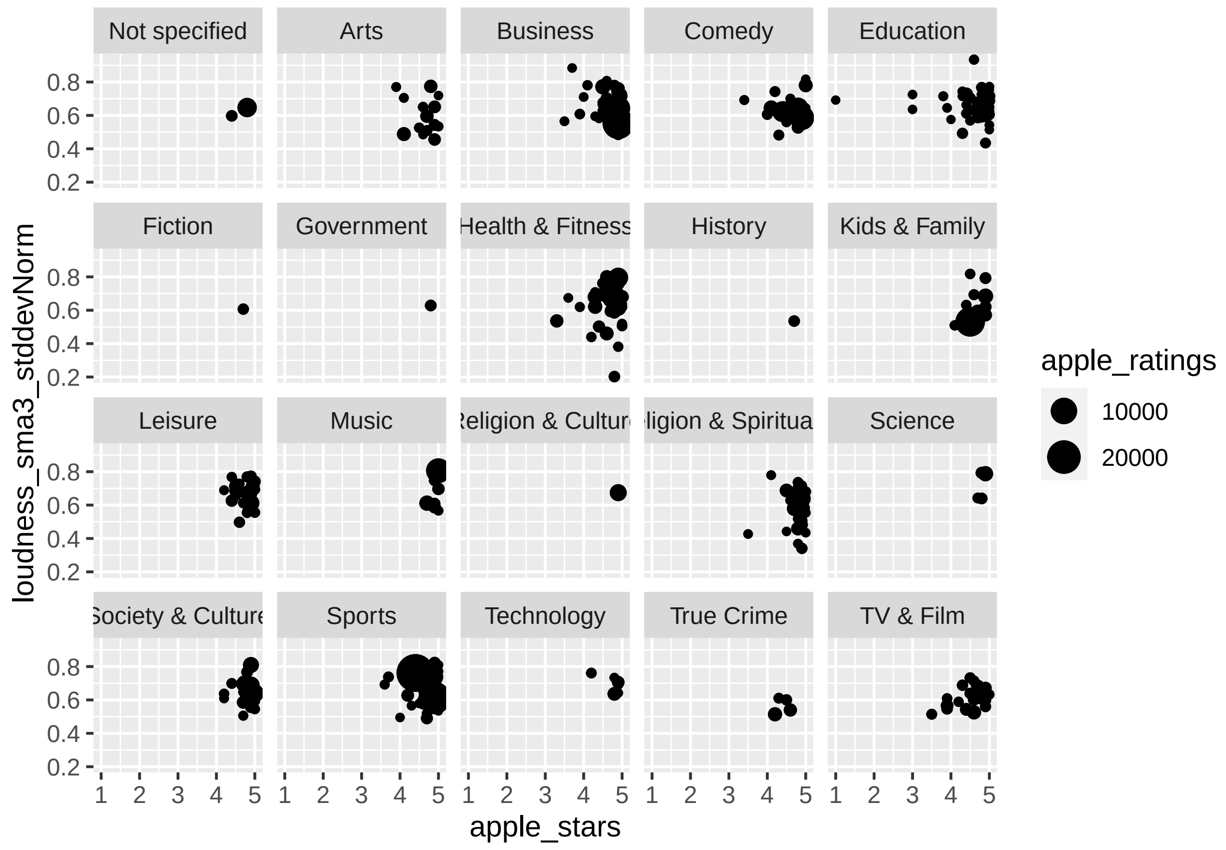
```r
# Facet Wrap of Stars vs. Raters vs. Category for features

gg5 <- ggplot(data = data1) +
  geom_point(mapping = aes(x = apple_stars, y = jitterLocal_sma3nz_amean, size = apple_ratings)) +
  facet_wrap(~ apple_category)
gg5
```



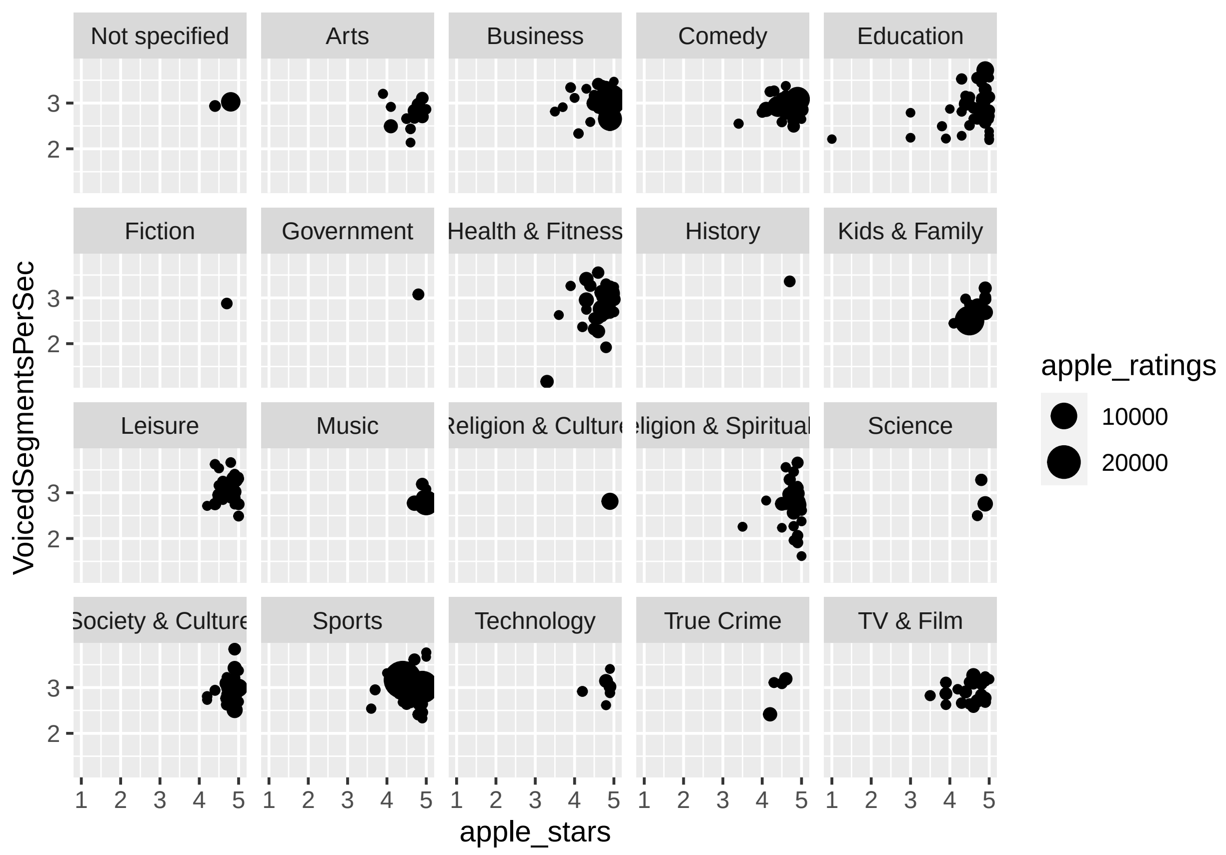
```
# Facet Wrap of Stars vs. Raters vs. Category for features

gg5 <- ggplot(data = na.omit(data1)) +
  geom_point(mapping = aes(x = apple_stars, y = loudness_sma3_amean, size = apple_ratings)) +
  facet_wrap(~ apple_category)
gg5
```



```
# Facet Wrap of Stars vs. Raters vs. Category for features

gg5 <- ggplot(data = na.omit(data1)) +
  geom_point(mapping = aes(x = apple_stars, y = loudness_sma3_stddevNorm, size = apple_ratings)) +
  facet_wrap(~ apple_category)
gg5
```



```r
# Facet Wrap of Stars vs. Raters vs. Category for features

gg5 <- ggplot(data = na.omit(data1)) +
  geom_point(mapping = aes(x = apple_stars, y = VoicedSegmentsPerSec, size = apple_ratings)) +
  facet_wrap(~ apple_category)
gg5
```


Not specified
Arts
Business
Comedy
Education
Fiction
Government
Health & Fitness
History
Kids & Family
Leisure
Music
Religion & Culture
Religion & Spiritual
Science
Society & Culture
Sports
Technology
True Crime
TV & Film
VoicedSegmentsPerSec
apple_stars
apple_ratings
10000
20000