Regression

Code

Spotify API Statistics

Linear Regression is a simple but powerful algorithm. We look at two different sets of values. One a target set (x) and the other(s) the predictor (y) sets. We want to see the relationship between those two values. This relationship is defined by two more parameters, ‘w’, the slope of the line that tells us the amount that y changes in x and b, the intercept.

I. Simple Linear Regression

Data Set

The data set used was found on Kaggle (https://www.kaggle.com/datasets/mrmorj/dataset-of-songs-in-spotify?datasetId=1018569) (Link for pdf viewers: https://www.kaggle.com/datasets/mrmorj/dataset-of-songs-in-spotify?datasetId=1018569 (https://www.kaggle.com/datasets/mrmorj/dataset-of-songs-in-spotify?datasetId=1018569)) This data set takes a large sample of songs from Spotify, a music streaming service, and lists an abundance of attributes associated with that song. All these attributes are assigned and evaluated by Spotify, and their webapp documentation goes deeper into how these various attributes are all calculated. They provide their definitions of these values on their webapp documentation page found here. (https://developer.spotify.com/documentation/web-api/reference/#/operations/get-audio-features) (Link for pdf viewers: https://developer.spotify.com/documentation/web-api/reference/#/operations/get-audio-features (https://developer.spotify.com/documentation/web-api/reference/#/operations/get-audio-features))

The two attributes I want to focus on today are energy and valence. Spotify defines energy as a floating point number from [0,1] and measures intensity and activity. They derive this by observing a song’s loudness, timbre, onset rate and general entropy. All words I do not understand. Values closer to 1 will signify a higher energy value than values closer to 0. The other attribute under the microscope in today’s linear regression is valence, another floating point number from [0,1] that measures musical positivity on an emotional spectrum. High valence tracks will have values closer to 1 and lower valence tracks will fall closer to 0.

For today’s experiment, we will take valence as our target and energy as our predictor.

Dividing the Data and Data Exploration

Let’s read in our data and look for any NA values before spliting into train/test.

Hide

```
full <- read.csv("genres_v2.csv", header = TRUE)
str(full)
sum(is.na(full$energy))
sum(is.na(full$valence))
```

Just to get a feel for all the data Spotify collects, I included all the attributes in the CSV file and listed them out. There’s some pretty neat things lying under the hood!

However, we're just looking at how a song's valence impacts its energy. So we can get rid of some extraneous variables.

Hide

```
df = subset(read.csv("genres_v2.csv"), select = -c(key, loudness, mode, speechiness, acoustic
ness, instrumentalness, liveness, tempo, type, id, uri, track_href, analysis_url, duration_m
s, time_signature, danceability, song_name, title, Unnamed..0))
```

Additionally, in the genre_2.csv file, songs are divided by genre. For now what we'll do is factor that column as it's a qualitative set.

Hide

```
df$genre <- as.factor(df$genre)
```

Here, we'll divide the data into train and test with an 80/20 split and feed it into lm1.

Hide

```
set.seed(1234)
i <- sample(1:nrow(df), .80*nrow(df), replace = FALSE)
train <- df[i, ]
test <- df[-i, ]
head(train)
head(test)
```

Data Exploration

Let's take a quick peek at our data with head, tail and summary.

Hide

```
head(df)
tail(df)
summary(df$valence)
summary(df$energy)
```

Plots

Now that our data is all organized, lets try to make a few plots to get a visual understanding. Let's take a closer look at energy.

Hide

```
hist(df$energy, col = "slategray", main = "Energy Values of Songs on Spotify", xlab = "Energ
y")
```

Clearly our data set has a high frequency of high energy songs. Let's take a look at a density plot.

Hide

```
d <- density(df$energy)
plot(d, main = "Density Plot of Energy", xlab = "Energy")
polygon(d, col = "wheat", border = "slategrey")
```

Let's do the same for valence.

Hide

```
hist(df$valence, col = "slategray", main = "Valence Values of Songs on Spotify", xlab = "Vale
nce")

d <- density(df$valence)
plot(d, main = "Density Plot of Valence", xlab = "Valence")
polygon(d, col = "wheat", border = "slategrey")
```

Now let's take a look at both energy and valence in a plot together. Since both of our x and y values are quantitative, I believe a scatter plot to be the best choice of plot.

Hide

```
plot(df$valence, df$energy, pch = '+', cex = 0.50, col = "green", xlab = 'Valence', ylab = 'E
nergy')
```

Linear Regression Model lm1

Let's take a look at our lm1 linear regression model.

Hide

```
lm1 <- lm(valence~energy, data = train)
summary(lm1)
```

Let's break down each section of our summary. * Residuals: Residuals are the difference between the observed response values and predicted response values. Across five sections of the model output we see a pretty symmetrical increase from our minimum to the 3Q. Once we go from 3Q to our maximum we see a huge spike however. So somewhere between the third quarter and max range of the model, the model begins to predict values that deviate heavily from the actual. This could likely be down to how skewed our data was towards high energy, low valence track ratings.

- Coefficients: Our standard error for both the intercept (b) and for our slope (w) are both below 0.005. We don't have a lot of variation in both our intercept and w values. We can interpret our intercept as the valence rating estimate of an average song in our data, which appears to be approximately 0.373. Our energy stat tells us that energy has a slightly negative effect on valence. As the valence of a song increase, the energy gets marginally smaller. Our standard error tells us that our intercept estimate seems to be pretty solid, as our standard error is somewhat marginal to it in comparison. However the standard error in relation to our energy variable is troubling as it's pretty significant when you compare it to the energy estimate. Our t-vales are good indicators of confidence, especially when looking at our intercept. Since these values are relatively larger than our error and in the case of the intercept, is MUCH greater than 0, we can assume there is some kind of underlying relationship here. Finally, our p-values look small, since they are both close to zero. Unfortunately our slope isn't as close to zero as we would

hope, with only a two-star significance rating. However, assuming a good p-value cutoff point is 0.05 or five percent, so we have some evidence that may suggest there is a relationship between energy and valence.

- Residual Standard Error: This metric is the average deviation each point has from the true regression line. Due to our monster data set size, we can observe a pretty low residual standard error thanks to the 33,482 degrees of freedom.
- R-Square's: These statistics provide a measure of how well the model fits the data. A number near 0 doesn't explain variance in the model well and a number closer to 1 better handles variance. As we can see both our R squared values are really close to zero. Around 0.02% of the variance between energy and valence can be explained by the model, which is absolutely horrendous. We want to shoot for a value closer to 60%.
- F-Statistic: Our F-statistic is another good measure to determine if there is a relationship between energy and valence. For our F-statistic, we want a value as far away from 1 as possible. For a data set THIS large, we can expect a value that isn't so far away from 1. Any value larger than 1 for a data set this large should be sufficient to reject the null hypothesis. Our F-statistic is 9.467, which is sufficient to reject the null hypothesis and conclude that some relationship exists between energy and valence.

Residuals

Let's plot our residuals for more data analysis.

Hide

```
plot(lm1)
```

1. Residuals v. Fitted: What we want to see here is a horizontal line with values falling equidistant from the line. It may be hard to see due to the amount of points in the data, but for the middle third of our data we have a good looking plot. We fit a flat horizontal line with a relatively even set of values below and above it. We begin to taper off at the end and get some more unruly outliers and our plot begins to look slightly parabolic, so we aren't meeting our regression assumptions super well here. We didn't see this originally, so our model is leaving out a non-linear relationship in the residuals. (Albeit slightly.)
2. Normal Q-Q: What we see here is again, the middle third or so of our data seems to fit well. Quantities from approximately [-1.5 to 1.5] fit a straight line well and marks that our residuals are normally distributed. However everywhere else on the plot we see some deviations, which may mark over-dispersed data, meaning that our data has far too much variance at the endpoints of our data.
3. Scale-Location: What we want to see here is a plot that has residuals spread equally along the predictor range. We want a horizontal line with equally spread points. Our line is fairly horizontal, and for the first half of the data we have a pretty even spread of points. As we advance on the line however, the top half of points halves and we see more of a bias of points underneath the horizontal line. So we have equal variance on our points until the last half of points.
4. Residuals v. Leverage: This essential tries to mark any influential outliers. Not all outliers are influential to a regression line. We just want to see if they conform to the regression line or not. We identify problem outliers by seeing what points fall outside of Cook's distance and alter the regression line and R^2 values too drastically. As we can see in plot 4 we have thousands of points falling outside of Cook's distance, so we have a lot of cases that are too drastic to the model. Could this be faulty data collection or an

incorrect approach to modeling?

II. Multiple Linear Regression

Here we'll add more predictors and add summaries and plots as we did before to see if we can improve upon the model.

In addition to energy, we will add danceability (another float value that evaluates a song's suitability for dancing based on a variety of musical elements like tempo, rhythm, etc.) and loudness (another float value that averages the decibels of an entire track) as predictors.

Hide

```
df2 = subset(read.csv("genres_v2.csv"), select = -c(key, mode, speechiness, acousticness, ins
trumentalness, liveness, tempo, type, id, uri, track_href, analysis_url, duration_ms, time_si
gnature, song_name, title, Unnamed..0))
df2$genre <- as.factor(df2$genre)
```

Now we'll separate the train/test data.

Hide

```
set.seed(1234)
i <- sample(1:nrow(df), .80*nrow(df), replace = FALSE)
train <- df2[i,]
test <- df2[-i, ]
head(test)
```

Here we'll build the new linear model.

Hide

```
lm2 <- lm(valence~energy + danceability + loudness, data = train)
summary(lm2)
```

And finally we'll output our residual plots.

Hide

```
plot(lm2)
```

III. Third Regression Model

Let's take a closer look at how valence is affected by genre of music.

Hide

```
lm3 <- lm(valence~genre, data = train)
summary(lm3)
plot(lm3)
```

IV. Final Findings

As we added more predictors, we get more accurate regression results. This is clearer to see in the summary section than most of the plots, due to the sheer amount of plot points. (Though there are slight differences that do point to improved results as we move from lm1 to lm3 that can be visually seen on the plots.) As we added more predictors in lm2 with danceability and loudness, we see that all coefficients (aside from the intercept) achieve a three star rating of significance, which is a good start and we also observe a better R^2 value. This is expanded when we do regression comparing valence with genre of music (seen in lm3), as this is most likely a much better predictor due to all the rules a song must have to fit into a genre and songs in a genre may have common themes within the song itself.

Final Predictions

Hide

```
pred1 <- predict(lm1, newdata = test)
mse1 <- mean((pred1 - test$energy)^2)
cor1 <- cor(pred1, test$energy)

pred2 <- predict(lm2, newdata = test)
mse21 <-mean(pred2 - test$energy)
mse22 <-mean(pred2 - test$danceability)
mse23 <-mean(pred2 - test$loudness)

cor21 <- cor(pred2, test$energy)
cor22 <- cor(pred2, test$danceability)
cor23 <- cor(pred2, test$loudness)

pred3 <- predict(lm3, newdata = test)

print(mse1)
print(cor1)

print(mse21)
print(mse22)
print(mse23)
print(cor21)
print(cor22)
print(cor23)
```

We see that with our first cor value, we have a very strong negative relationship between valence and energy. And the MSE value tells me that our predictions matched our expected values relatively well given that it's not too far from zero. We did figure out some relationship in our summary of lm1 initially, so this makes sense. Secondly, our MSE for energy and danceability were pretty solid, again being pretty close to zero. However loudness was erratic, we deviated a lot from what we expected in our test data. I imagine that loudness in decibels can really vary across songs without being directly tied to valence. Looking at our cor values we see that the strongest positive correlation falls onto danceability, with a strong positive relationship between that and valence. Interestingly enough we see the energy correlation plummet close to 0, indicating no real correlation with valence, which makes sense. Energy doesn't necessarily guarantee valence as energy could

be misconstrued for more negative emotions. It seems that loudness has an almost insignificant positive relationship with valence, which also makes sense for the same reasons that energy doesn't correlate with valence.