Sunday night was a big night for Taylor Swift – not only was she nominated for multiple [Billboard Music Awards](https://www.billboard.com/articles/news/bbma/8456842/billboard-music-awards-2018-winners-list-bbmas); she took home Top Female Artist and Top Selling Album. So I thought it was a good time for some [more Taylor Swift-themed statistical analysis](http://www.deeplytrivial.com/2018/05/statistics-sunday-taylor-swift-vs-lorde.html).

When I started this blog back in 2011, my goal was to write deep thoughts on trivial topics – specifically, to overthink and overanalyze pop culture and related topics that appear fluffy until you really dig into them. Recently, I’ve been blogging more about statistics, research, R, and data science, and I’ve loved getting to teach and share.

But sometimes, you just want to overthink and overanalyze pop culture.

So in a similar vein to the [text analysis](http://www.deeplytrivial.com/2018/05/statistics-sunday-taylor-swift-vs-lorde.html) I’ve been [demonstrating on my blog](https://www.deeplytrivial.com/2018/05/statistics-sunday-welcome-to-sentiment.html), I decided to answer a question I’m sure we all have – as Taylor Swift moved from country sweetheart to mega pop star, how have the words she uses in her songs changed?

I’ve used the geniusR package on a couple posts, and I’ll be using it again today to answer this question. I’ll be pulling in some additional code, some based on code from the [*Text Mining with R: A Tidy Approach*](https://www.amazon.com/Text-Mining-R-Tidy-Approach-ebook/dp/B071K9RT9Z/ref=sr_1_fkmr0_1?ie=UTF8&qid=1526611640&sr=8-1-fkmr0&keywords=text+analysis+in+R+julia+silge) book I recently devoured, some written to try to tackle this problem I’ve created for myself to solve. I’ve shared all my code and tried to credit those who helped me write it where I can.

First, we want to pull in the names of Taylor Swift’s 6 studio albums. I found these and their release dates on Wikipedia. While there are only 6 and I could easily copy and paste them to create my data frame, I wanted to pull that data directly from Wikipedia, to write code that could be used on a larger set in the future.

library(rvest)

## Loading required package: xml2

TSdisc <- 'https://en.wikipedia.org/wiki/Taylor\_Swift\_discography'  
  
disc <- TSdisc %>%  
 read\_html() %>%  
 html\_nodes(xpath = '//\*[@id="mw-content-text"]/div/table[2]') %>%  
 html\_table(fill = TRUE)

Since html() is deprecated, I replaced it with read\_html(), and I got errors if I didn’t add fill = TRUE. The result is a list of 1, with an 8 by 14 data frame within that single list object. I can pull that out as a separate data frame.

TS\_albums <- disc[[1]]

The data frame requires a little cleaning. First up, there are 8 rows, but only 6 albums. Because the Wikipedia table had a double header, the second header was read in as a row of data, so I want to delete that, because I only care about the first two columns anyway. The last row contains a footnote that was included with the table. So I removed those two rows, first and last, and dropped the columns I don’t need. Second, the information I want with release date was in a table cell along with record label and formats (e.g., CD, vinyl). I don’t need those for my purposes, so I’ll only pull out the information I want and drop the rest. Finally, I converted year from character to numeric – this becomes important later on.

library(tidyverse)

TS\_albums<-TS\_albums[2:7,1:2]  
  
TS\_albums <- TS\_albums %>%  
 separate(`Album details`, c("Released","Month","Day","Year"),  
 extra='drop') %>%  
 select(c("Title","Year"))  
  
TS\_albums$Year<-as.numeric(TS\_albums$Year)

I asked geniusR to download lyrics for all 6 albums. (Note: this code may take a couple minutes to run.) It nests all of the individual album data, including lyrics, into a single column, so I just need to unnest that to create a long file, with album title and release year applied to each unnested line.

library(geniusR)  
  
TS\_lyrics <- TS\_albums %>%  
 mutate(tracks = map2("Taylor Swift", Title, genius\_album))

## Joining, by = c("track\_title", "track\_n", "track\_url")  
## Joining, by = c("track\_title", "track\_n", "track\_url")  
## Joining, by = c("track\_title", "track\_n", "track\_url")  
## Joining, by = c("track\_title", "track\_n", "track\_url")  
## Joining, by = c("track\_title", "track\_n", "track\_url")  
## Joining, by = c("track\_title", "track\_n", "track\_url")

TS\_lyrics <- TS\_lyrics %>%  
 unnest(tracks)

Now we’ll tokenize our lyrics data frame, and start doing our word analysis.

library(tidytext)  
  
tidy\_TS <- TS\_lyrics %>%  
 unnest\_tokens(word, lyric) %>%  
 anti\_join(stop\_words)

## Joining, by = "word"

tidy\_TS %>%  
 count(word, sort = TRUE)

## # A tibble: 2,024 x 2  
## word n  
##   
## 1 time 198  
## 2 love 180  
## 3 baby 118  
## 4 ooh 104  
## 5 stay 89  
## 6 night 85  
## 7 wanna 84  
## 8 yeah 83  
## 9 shake 80  
## 10 ey 72  
## # ... with 2,014 more rows

There are a little over 2,000 unique words across TS’s 6 albums. But how have they changed over time? To examine this, I’ll create a dataset that counts word by year (or album, really). Then I’ll use a binomial regression model to look at changes over time, one model per word. In their book, Julia Silge and David Robinson demonstrated how to use binomial regression to examine word use on the authors’ Twitter accounts over time, including an adjustment to the p-values to correct for multiple comparisons. So I based on my code off that.

words\_by\_year <- tidy\_TS %>%  
 count(Year, word) %>%  
 group\_by(Year) %>%  
 mutate(time\_total = sum(n)) %>%  
 group\_by(word) %>%  
 mutate(word\_total = sum(n)) %>%  
 ungroup() %>%  
 rename(count = n) %>%  
 filter(word\_total > 50)  
  
nested\_words <- words\_by\_year %>%  
 nest(-word)  
  
word\_models <- nested\_words %>%  
 mutate(models = map(data, ~glm(cbind(count, time\_total) ~ Year, .,  
 family = "binomial")))

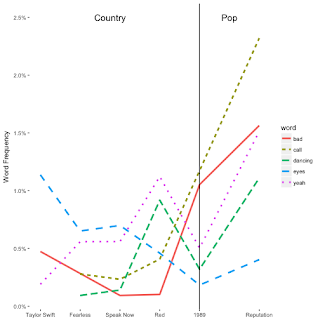
This nests our regression results in a data frame called word\_models. While I could unnest and keep all, I don’t care about every value the GLM gives me. What I care about is the slope for Year, so the filter selects only that slope and the associated p-value. I can then filter to select the significant/marginally significant slopes for plotting (p < 0.1).

library(broom)  
  
slopes <- word\_models %>%  
 unnest(map(models, tidy)) %>%  
 filter(term == "Year") %>%  
 mutate(adjusted.p.value = p.adjust(p.value))  
  
top\_slopes <- slopes%>%  
 filter(adjusted.p.value < 0.1) %>%  
 select(-statistic, -p.value)

This gives me five words that show changes in usage over time: bad, call, dancing, eyes, and yeah. We can plot those five words to see how they’ve changed in usage over her 6 albums. And because I still have my TS\_albums data frame, I can use that information to label the axis of my plot (which is why I needed year to be numeric). I also added a [vertical line](http://ggplot2.tidyverse.org/reference/geom_abline.html) and [annotations](https://www.r-graph-gallery.com/233-add-annotations-on-ggplot2-chart/) to note where [TS believes she shifted from country to pop](http://tasteofcountry.com/taylor-swift-on-going-country-to-pop/).

library(scales)

words\_by\_year %>%  
 inner\_join(top\_slopes, by = "word") %>%  
 ggplot(aes(Year, count/time\_total, color = word, lty = word)) +  
 geom\_line(size = 1.3) +  
 labs(x = NULL, y = "Word Frequency") +  
 scale\_x\_continuous(breaks=TS\_albums$Year,  
 labels=TS\_albums$Title) +  
 scale\_y\_continuous(labels=scales::percent) +  
 geom\_vline(xintercept = 2014) +  
 theme(panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.background = element\_blank()) +  
 annotate("text", x = c(2009.5,2015.5), y = c(0.025,0.025),  
 label = c("Country", "Pop") , size=5)

[](https://i2.wp.com/4.bp.blogspot.com/-9NIC3WQKzmY/Wv9ogsWH2QI/AAAAAAAALlU/aZ82D4Ifgi4knWDmiiaDjofcu5c209FKQCPcBGAYYCw/s1600/unnamed-chunk-8-1.png?ssl=1)

The biggest change appears to be in the word “call,” which she didn’t use at all in her self-titled album, and used at low rates until “1989” and, especially, “Reputation.” I can ask for a few examples of “call” in her song lyrics, with [grep](https://www.rdocumentation.org/packages/base/versions/3.5.0/topics/grep).

library(expss)

callsubset <- TS\_lyrics[grep("call", TS\_lyrics$lyric),]  
callsubset <- callsubset %>%  
 select(Title, Year, track\_title, lyric)  
set.seed(2012)  
callsubset<-callsubset[sample(nrow(callsubset), 3), ]  
callsubset<-callsubset[order(callsubset$Year),]  
as.etable(callsubset, rownames\_as\_row\_labels = FALSE)

| **Title** | **Year** | **track\_title** | **lyric** |
| --- | --- | --- | --- |
| Speak Now | 2010 | Back to December (Acoustic) | When your birthday passed, and I didn’t call |
| Red | 2012 | All Too Well | And you call me up again just to break me like a promise |
| Reputation | 2017 | Call It What You Want | Call it what you want, call it what you want, call it |

On the other hand, she doesn’t sing about “eyes” as much now that she’s moved from country to pop.

eyessubset <- TS\_lyrics[grep("eyes", TS\_lyrics$lyric),]  
eyessubset <- eyessubset %>%  
 select(Title, Year, track\_title, lyric)  
set.seed(415)  
eyessubset<-eyessubset[sample(nrow(eyessubset), 3), ]  
eyessubset<-eyessubset[order(eyessubset$Year),]  
as.etable(eyessubset, rownames\_as\_row\_labels = FALSE)

| **Title** | **Year** | **track\_title** | **lyric** |
| --- | --- | --- | --- |
| Taylor Swift | 2006 | A Perfectly Good Heart | And realized by the distance in your eyes that I would be the one to fall |
| Speak Now | 2010 | Better Than Revenge | I’m just another thing for you to roll your eyes at, honey |
| Red | 2012 | State of Grace | Just twin fire signs, four blue eyes |

Bet you’ll never listen to Taylor Swift the same way again.

A few notes: I opted to examine any slopes with p < 0.10, which is greater than conventional levels of significance; if you look at the adjusted p-value column, though, you'll see that 4 of the 5 are < 0.05 and one is only slightly greater than 0.05. But I made the somewhat arbitrary choice to include only words used more than 50 times across her 6 albums, so I could get different results by changing that filtering value when I create the words\_by\_time data frame. Feel free to play around and see what you get by using different values!