FinalProjectLyrics

Saulai Vue, Jake Christenson, Kaleb Laramore

2025-04-11

# Part 1:

Your first task is to create a list of top songs, dating back to 1958 (when Billboard introduced it’s Hot 100 yearly chart). You may want to start with just the yearly top song, but your work should be general enough to account for multiple songs per year. You may narrow your search to a particular genre if you like. You may use any website that provides this information, though you may try to find one that makes part 2 as simple as possible.

* Initially, we scraped from 1958-2024 for all Top 20 Billboard Songs from wikipedia. This function generated our list of songs and artist in our data frame.
* With this, we wrote a function to be used to webscrape for artist and song titles from wikipedia.

### Extra File Generating Artist and Song Names

Attached are extended files from other markdown files that helped in webscraping

* From Github file code (<https://raw.githubusercontent.com/vsaulai/STAT-345-FinalProject-Lyrics/refs/heads/main/webscraping/JakeChristiansenCode.Rmd>)
* Note: Some markdown files were created by different group members and worked at different times while working on the group project. Files may not be integrated seemlessly together at the moment.

# Part 2:

For the top songs in part 1, gather some basic information: artist, title, year, genre (if appropriate), length, and other variables you think might be informative (sales figures, etc.). While webscraping we wanted genres to be incorporated with our data frame. However, finding a site that can freely give us the genre type of a song was challenging so we abandoned this idea. Next

* We initially wanted a genre incorporated in our data set but it was beyond our ability. While attempting to scrape for genres, it did not work and had problems in the CSS selector or website host.
* Therefore, we just settled on: artist, song title, year, and rank of song. Eventually, we added other information like: common phrases repeated, number of times the song title is repeated in lyrics, and word count.
* Next, I imported the ‘udpipe’ language model and wrote a function to count the word types for each song lyrics and got these amounts in our data frame, and will eventually add word type counts to the data set.

# Part 3:

Find a lyric hosting service (such as www.azlyrics.com or www.songlyrics.com, though these aren’t the only options) that provides full lyrics to songs. Ideally, the URLs for these songs follow a reproducible pattern. Write a function that can automatically capture these song lyrics for your top songs from part 1, and then gather the lyrics. Do your best to keep this function general, but you may need to write code for specific instances.

* Next, we chose the site host “songlyrics.com” by using the artist and song name we generated previously, we wrote a function to generate a url from the site host. With this, we wrote two main functions: one for generating the URL and one for webscraping.

### URL Generate Function

This generated our URL from our Main Data Frame. Here is an example of the URL we generated.

## https://www.songlyrics.com/domenico-modugno/volare-nel-blu-dipinto-di-blu-lyrics/

## Webscraping(cont.)

For the Second Function we took the URL we generated from the previous function and did the webscraping.

Here is an example of what the scraped lyrics turned out to be.

### Song: “You and I” (1983) by Eddie Rabbit and Crystal Gayle

## Just you and I  
## Sharing our love together  
## And I know in time  
## We'll build the dreams we treasure  
## And we'll be alright  
## Just you and I  
## Just you and I   
## Sharing our love together  
## And I know in time   
## We'll build the dreams we treasure  
## And we'll be alright  
## Just you and I  
## And I remember our first embrace  
## That smile that was on your face  
## The promises that we made oohoohooh  
## And now your love is my reward  
## And I love you even more  
## Than I ever did before  
## Just you and I   
## We care and trust each other  
## With you in my life   
## There'll never be another  
## We'll be alright  
## Just you and I  
## And I remember our first embrace  
## That smile that was on your face  
## The promises that we made oohoohooh  
## And now your love is my reward  
## And I love you even more  
## Than I ever did before  
## We made it you and I

### Extra File from Webscraping and Formatting

The two functions were used in these seperate markdown files to help generate our final data set. Download Code is located below and from github. - From file code (<https://raw.githubusercontent.com/vsaulai/STAT-345-FinalProject-Lyrics/refs/heads/main/webscraping/SaulaiVueCode.Rmd>) AND (<https://raw.githubusercontent.com/vsaulai/STAT-345-FinalProject-Lyrics/refs/heads/main/webscraping/missingSVCode.Rmd>)

# Data Formatting:

* To format the lyric strings to get our main data set, we removed all punctuation(except apostrophe), unnecessary spaces, words within brackets, and section titles like [Verse], [Chorus], [Artist Duet Part], and other random words that may have unintentionally appeared during webscraping.
* Next, we implemented other functions to our lyrics to get the most common words used (phrase\_repetition), and amount of times the title (title\_repetition\_count) was repeated in. And renamed our column titles.

## Udpipe Language Model

* ‘udpipe’ package is a language model used for tagging, converting words, and defining words by their parts of speech interpreted by the model from their semantic context.
* In our case, we’re using this model to estimate the word’s POS(Parts of Speech) category, i.e. noun, verb, adjective, etc. from the context of the song lyric.

## Parts of Speech Counted by Function (our column names)

* These are UPOS(Universal Parts of Speech) tagset categories found in our data set from columns 10 - 27.
* noun, verb, adjective, adposition, auxilary verb, adverb, proper\_noun, interjection, numeral, particle, pronoun, subordinating\_conj, coordinating\_conj, determiner, punctuation, symbol, other,

## Function to Count Word Type Using Udpipe

* Wrote a function that tags words in a string, trims white spaces, annotates and counts the word type in string and enters it back into the data frame.
* This counts the tokenized tags of each word and stores it as an integer count.

## Extra File from Data Formatting

The two functions were used in these seperate markdown files to help generate our final data set.

* From file code (<https://raw.githubusercontent.com/vsaulai/STAT-345-FinalProject-Lyrics/refs/heads/main/DataFormattingSV.Rmd>)
* Note: Some markdown files were created by different group members working at different on the group project, and are not seemlessly integrated together or required a file that may not be there anymore.

## Renaming the Parts of Speech for Clarity

When running the “udpipe” language model with our function it classifies words with the universal POS tags and are indicated below ### Parts of Speech (POS) Tags:

* noun = NOUN e.g. (office, car, table, freedom, happiness)
* verb = VERB e.g. (run, jump, sleep, exist, can, was, has)
* adjective = ADJ e.g. (big, small, red, nice, American, his, their, smallest)
* adposition = ADP e.g. (in, on, at, by)
* adverb = ADV e.g. (literally, elegantly, quickly, always, here, very)
* auxiliary\_verb = AUX e.g. (might, had, may, shall, ought to, are, is)
* proper\_noun = PROPN e.g. (Sarah, Titanic, Facebook, Microsoft, Rocky Rococo)
* interjection = INTJ e.g. (Phew, Ahem, Ahoy, Huh, Eww, Hurray, Oh)
* numeral = NUM e.g. (one, two, three, 1, 2, 3)
* particle = PART e.g. (up, out, on, to, off, in) i.e. (to know, cut in, jump off)
* pronoun = PRON e.g. (us, she, he, each, we, I, Them)
* subordinating\_conj = SCONJ e.g. (after, because, even though, once, that, if, unless)
* coordinating\_conj = CCONJ e.g. (for, and, nor, but, or, yet, so)
* determiner = DET e.g. (a, an, the, my, your, this, that, some)
* punctuation = PUNCT e.g. (most punctuation, ain’t, findin’, shan’t,)
* symbol = SYM e.g. (Yin and Yang, Peace Sign)
* other = X (unknowns)

## Final Main Data Frame Created from Webscraping

* Here is an example of what the udpipe language model tagged from our lyrical text for the first two songs.

## word\_count adjective adposition adverb AUX coordinating\_conj determiner  
## 1 258 4 31 9 8 7 23  
## 2 246 27 10 29 23 6 12  
## interjection noun numeral particle pronoun proper\_noun verb  
## 1 19 46 20 1 20 50 22  
## 2 8 42 2 4 55 11 37

### Removing NA values from Songs and Artists

* Some songs failed from webscrapping, either the website host did not have any lyrics or the url caused problems.
* Lost approximately +100 potential songs by their rankings.

### Removing Unnecessary Columns

* Eventually, adding the row sums of total words estimated by udpipe model and finding the proportion of parts of speech.
* Therefore, removing all non words that showed up in the data count to get the total.
* Total Observations: 1190 Songs with Lyrics

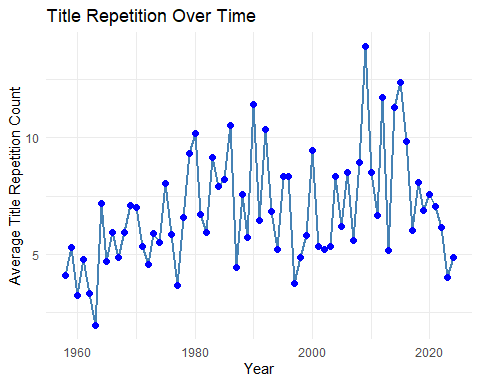
## Average Word Count List (Rounded up to Two Decimal Points)

* Took the average of each Song and grouped them together with their respective year’s average

## # A tibble: 4 × 5  
## Year avg\_title\_count avg\_word\_count avg\_adjective avg\_adposition  
## <int> <dbl> <dbl> <dbl> <dbl>  
## 1 1958 4.09 176. 9.55 16.0  
## 2 1959 5.3 180. 8.9 15.0  
## 3 1960 3.21 177. 14.7 13.6  
## 4 1961 4.76 160. 5.35 10.3

# Part 4 and 5:

Create two measures of song repetitiveness. Write a function (or two) to measure song repetitiveness, and apply it to each of the songs from part 1. Suggestions for “repetitiveness” include (but are definitely not limited to): “Do songs repeat the same phrase frequently?” and “Do songs repeat their song title frequently”



## Summary of Title Repetition Over Time

Have songs become more repetitive over time? Summarize and visualize your repetitive measures from part 4.

### Early Years (1958–1980s):

* Song titles appeared less frequently in lyrics.
* Lyrics were more story-driven, focusing on narratives rather than repetition.
* Genres like classic rock, jazz, and folk had minimal reliance on repeated hooks.

### 1990s–2010s:

* Gradual increase in title repetition.
* Pop and hip-hop music adopted catchier hooks, making song titles repeat more often.
* Radio play and streaming services started rewarding songs with memorable choruses.

### 2010s–2020s:

* Significant spike in title repetition.
* Songs now repeat titles multiple times per chorus, making them more recognizable.
* Genres like hip-hop, EDM, and commercial pop use repetition as a marketing tool.

# Part 6:

(If possible) Extend your work to more songs! Consider more questions, like “Does genre matter?”.

## Splitting into Periods by Decades into A List

Next, We split our data into decades. 1958-1969, 1970-1979, 1980-1989, 1990-1999, 2000-2009, 2010-2019, 2020-2024.

## Splitting POS types into categories

Taking the proportion of each song’s POS type to total POS word for each year

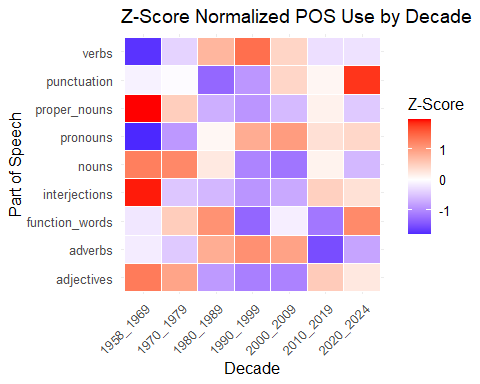
## Categories

* Proper Nouns: are significant because it’s naming a direct person, place, or thing. Which may hint towards a decade’s cultural reference.
* Verbs + Auxiliary Verbs: were my preference of POS category, Actions are significant POS word types.
* Adjectives: are descriptive POS and are significant to stand alone
* Adverbs: are describing actions and may have an argument to be in verbs but we chose to leave it separated. Also contains particles which are of the same function like adverbs i.e. (“up”, “on”, “out”, I ran “into” my professor)
* Pronouns: hint at specifically someone but not whom i.e.(Him, Her, They, It, He, She, etc.) could be referencing an Ex boyfriend or girlfriend. So, we thought this might be a significant count.
* Interjections: are emotional words like “Hurray!”, “Hey”, “Ow”, they can oftentimes be related to slang which I though significant
* Punctuation: may not seem relevant but it is sometimes indicating slang or ‘double’ words like can’t, shouldn’t, Ain’t, etc. Slang words are a pretty significant in English Slang.
* Function Word Group: is everything else: Determiners (“a”, “an”, “the”); Conjunctions both subordinating and coordinating(“for”, “and”, “but”, “whereas”, “unless”); Adposition (“in”, “with”, “by”) ‘NA.’ is other words that the language model did not comprehend.

Below is an example of Proportions of POS types

## decade nouns proper\_nouns verbs adjectives  
## 1 1958\_1969 0.1507351 0.04511018 0.2332876 0.04903672  
## 2 1970\_1979 0.1498772 0.03153423 0.2472570 0.04727228  
## 3 1980\_1989 0.1402742 0.02052303 0.2588450 0.03914901

## Heat Map for Z-Score Standard Normal Distribution



Z-score is based off of a standard normal distribution

## What does the Z-score mean from the heat map?:

### Z-score > 0 :

* The proportion of that POS in this decade is above average compared to all other decades of the same POS.
* The higher the value, the more unusually high that POS usage was in that decade.

### Z-score < 0 :

* The proportion of that POS in this decade is below average compared to all other decades of the same POS.

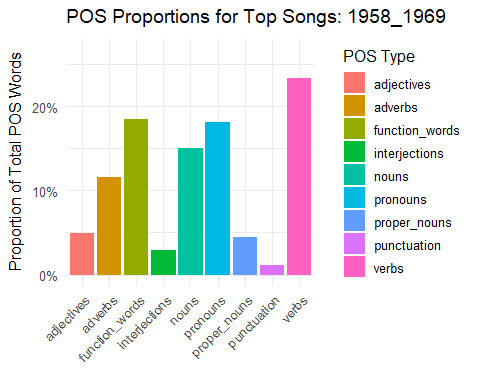
### Z-score near 0 :

* The proportion of POS usage is very close across the average of all decades.

## Key Observations

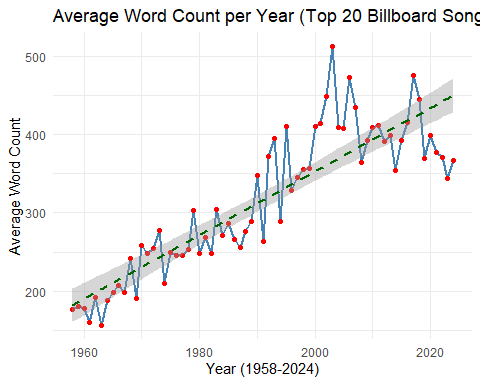
* Between the 1980’s to early 2000’s, verbs usage in the Top 20 Songs of the Year were above the global average of verb usage during all other time periods. Verbs are used to describe “actions” and may be a linguistic preference to song writers from this time period.
* Meanwhile, verb usage was significantly below the average proportion from 1958-1969. It could be because the amount of instrumental songs were popular during this decade.
* Proper Noun’s i.e. names of places, personal names, etc. were more significantly found in the top songs from 1958-1969. Colloquial references to other names were also more popular in these songs.
* Across the decade the pronoun usage starting out was low in the 60’s and 70’s, but the average proportion of pronoun usage grew with time. This isn’t a surprise because references to “He”, “She”, “it”, “them”, etc. are rather popular in romantic songs nowadays.
* Punctuations, often an indicator for slang like “Ain’t”, “’Til”, “findin’” were more popular in the top songs from 2020-2024, and the proportion was above the standard mean.
* Interjections, i.e. “Ahoy”, “Hey”, “Ugh”, “Gee, etc. were especially popular in the top songs from 1958-1969 often dominated by Elvis Presley, and Elton John.

## Animated Stacked Area Plot Code



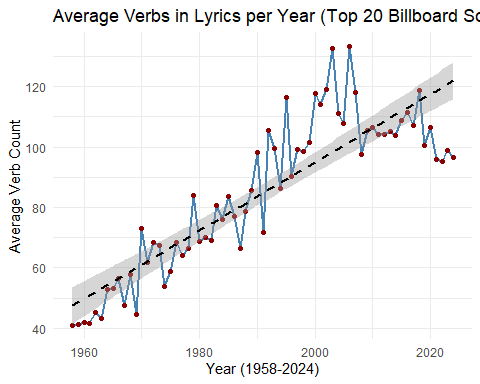
### Key Observations:

* Most of the English language structure has relatively the same proportion of parts of speech types.
* What differs are the trends that are gradually shifting the preferences of most songwriters or our semantics.
* Some likely key errors could be within the language model, either not detecting accurately the semantic meaning behind modern words or lingo used in the past or present.
* The semantic context of the lyrical sentence structure may differ.



## Average Amount of Words Count per Year(Top 20 Billboard Songs)

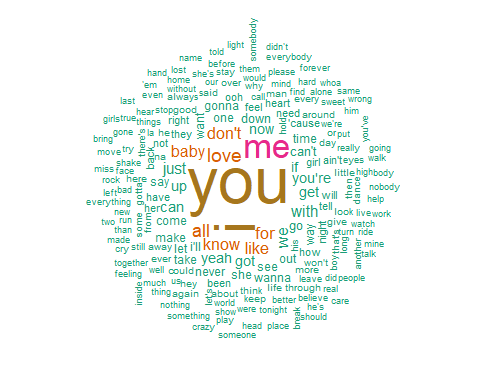
* Takes every Top Billboard Songs Per Year and measures the average word count per year
* Word counts increased among songs since 1958.



## Verb Count per Year

* As the same with word count, the amount of verb count per year has significantly increased since 1958.

# Wordcloud: Top 200 Most Frequently Used Words in Top 20 Billboard Songs (1958-2024)

(Removed most function words e.g. “a”, “an”, “it”, etc.) 

## Conclusions:

* Songs repeating their own titles in their lyrics have been on average increasing since 2010
* Words counts of songs have gotten longer over time
* Older songs tended to favor instrumentals or solo parts which could be why the amount of lyrical words were generally less from 1958-1979.
* Verb usage in the 1980’s and 90’s were proportionately in more top songs and above the standard mean across all other decades.