Billboard Analysis

#Team_Maroon_4
#CMSC_12200

Team **Maroon 4**

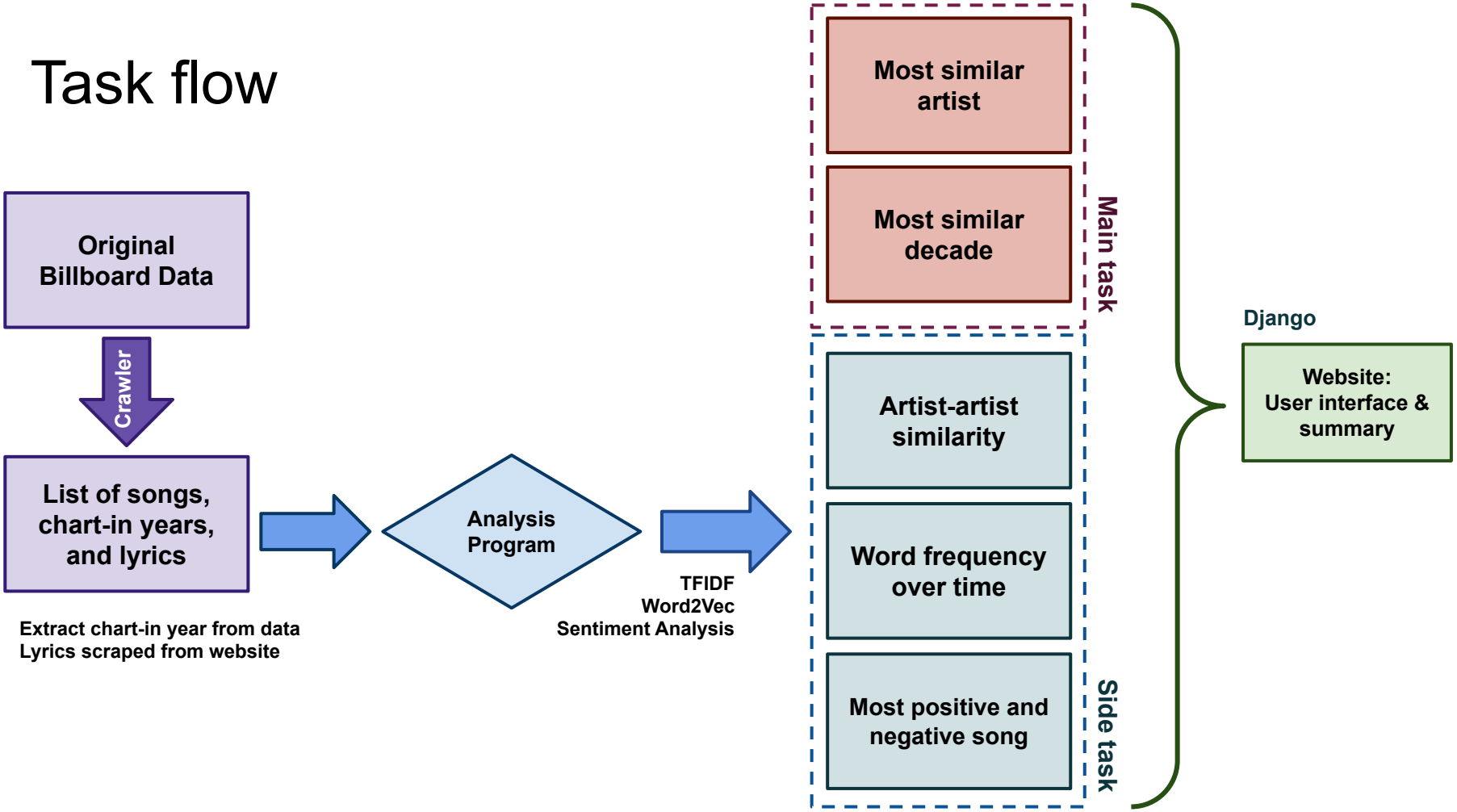
```
for name, year, major in Team:

    print("{:s}: {:s} / {:s}".format(name, year, major))

>>>
```

- **Will Hwang:** Sophomore / Statistics
- **Yongju Kim:** Junior / Economics
- **Whan Lee:** Sophomore / Economics
- **H.I. Park:** Junior / Statistics

Objective

Quantitative comparison of successful Billboard Top 100 songs and a randomly constructed string (supposed to have some context)

Task flow


Original Billboard Data
Crawler
List of songs, chart-in years, and lyrics
Extract chart-in year from data
Lyrics scraped from website
Analysis Program
TFIDF
Word2Vec
Sentiment Analysis
Most similar artist
Most similar decade
Main task
Artist-artist similarity
Word frequency over time
Most positive and negative song
Side task
Django
Website: User interface & summary

Lyrics Crawler

- Cleans original Billboard data
- Filters out “One Hit Wonders“ and keep “established” artists
  (by excluding “redundancy,” or multi-time chart-in by a same song)
- Web crawling + lyrics extraction & cleaning + adds first chart-in years of songs
  (“year” in the context of our data might be different from the release year)
- Store results TXT file

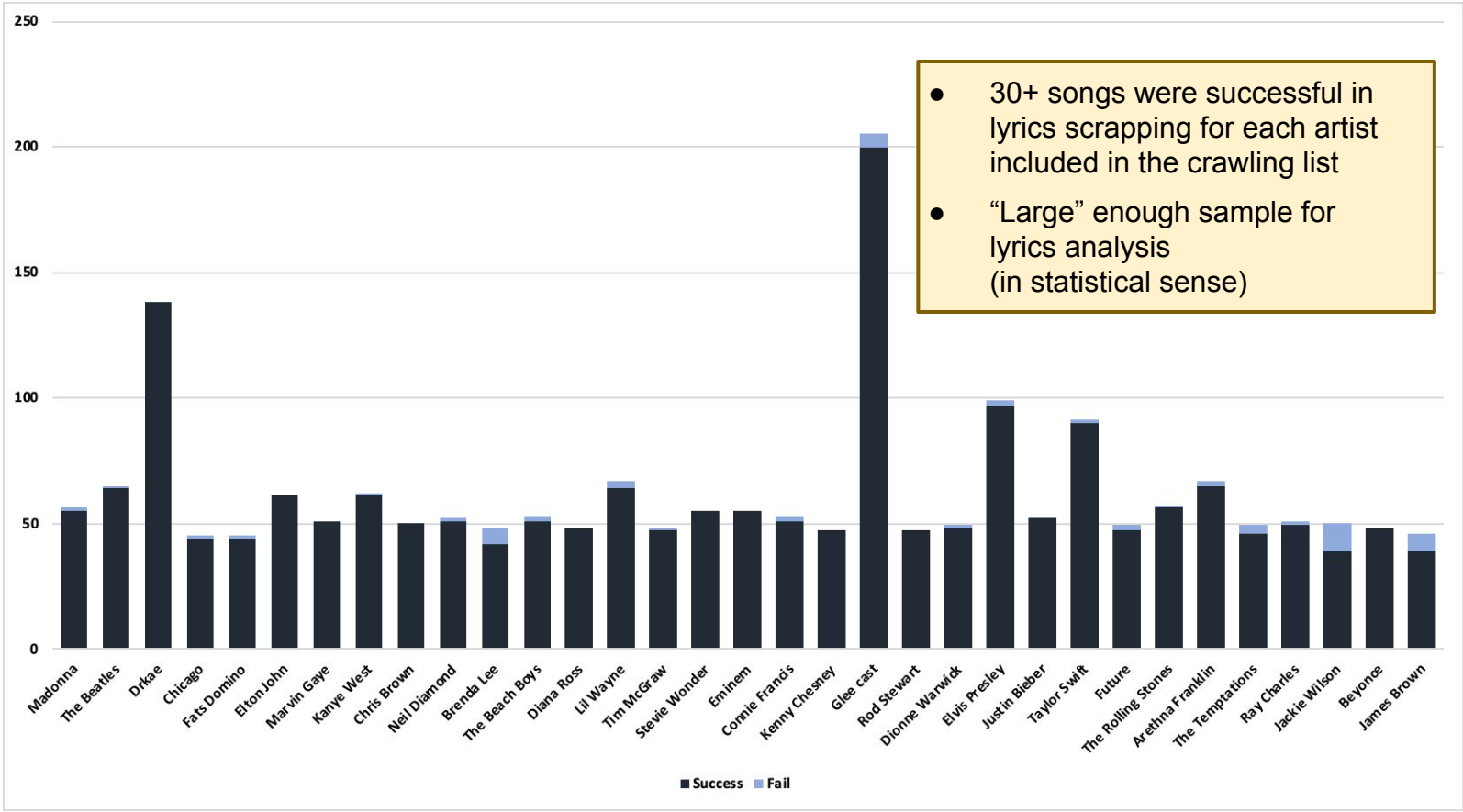
Websites used:

- songlyrics.com
- lyricsondemand.com
- lyrics.az
- genius.com

Crawling criteria:

- An artist should have least 45 different chart-in songs in order to be included in the crawling list
- If failed scraping lyrics in one site, the crawler tries another site (4 websites in total)
- If lyrics cannot be scrapped from all of 4 websites, sort the song as “failed in crawling”

Overall crawling performance:
2002 songs successfully crawled | 57 songs failed | success rate 97.23%



Main Task: Computing Similarity

- Stopwords: “I”, “The”, “Me”, “Have”, etc. - Did not remove them!

TFIDF

- Reflection of how important a word is to a document
- Same technique as in PA3 of CMSC 12100

Word2Vec

- Represents each word by a 100-dimensional vector through Neural Networks
- Looks at 5 words before/ 5 words after the word to reconstruct linguistic contexts of words
- Preserves relationships among words
- “men” + “women” - “queen” = “king”

Sentiment Analysis

- Measures how positive and/or negative the lyrics are
- Uses NLTK package

Weights

- Multiplied TFIDF distance by 10^3 and sentiment distance by 10^-1 to roughly equalize variance

```
[In [10]: w.find_var_equal_constant()
To equalize variance, multiply tfidf by  1215.5854371924484
To equalize variance, multiply sentiment by  0.10277386121529077
```

- Weight TFIDF, Word2Vec, and Sentiment scores by a ratio
- Weight how similar the lyric is to a single song/ how similar the lyric is to the artist as a whole

```
The best weights is :  (1, 2, 2, 4, 1)
```

Side Task I: Most Positive/Negative Song Given ...

- Artist: Given artist name, what is his/her/their most positive/negative song?
  - Example
    - Input: Drake
    - Output:
      - Positive: Drake - Best I ever had - 2009
      - Negative: Drake - 0 to 100 - 2014
- Year: Given year, what was the most positive/negative song?
  - Example
    - Input: 2010
    - Output:
      - Positive: Justin Bieber - Somebody to love - 2010
      - Negative: Lil Wayne - What’s wrong with them - 2010

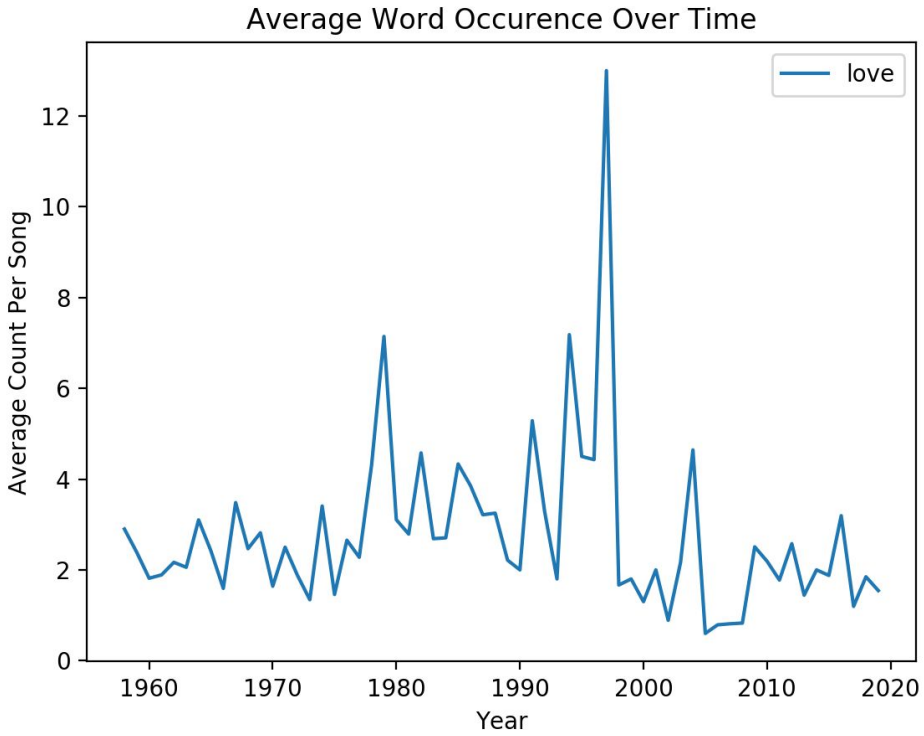
Negative

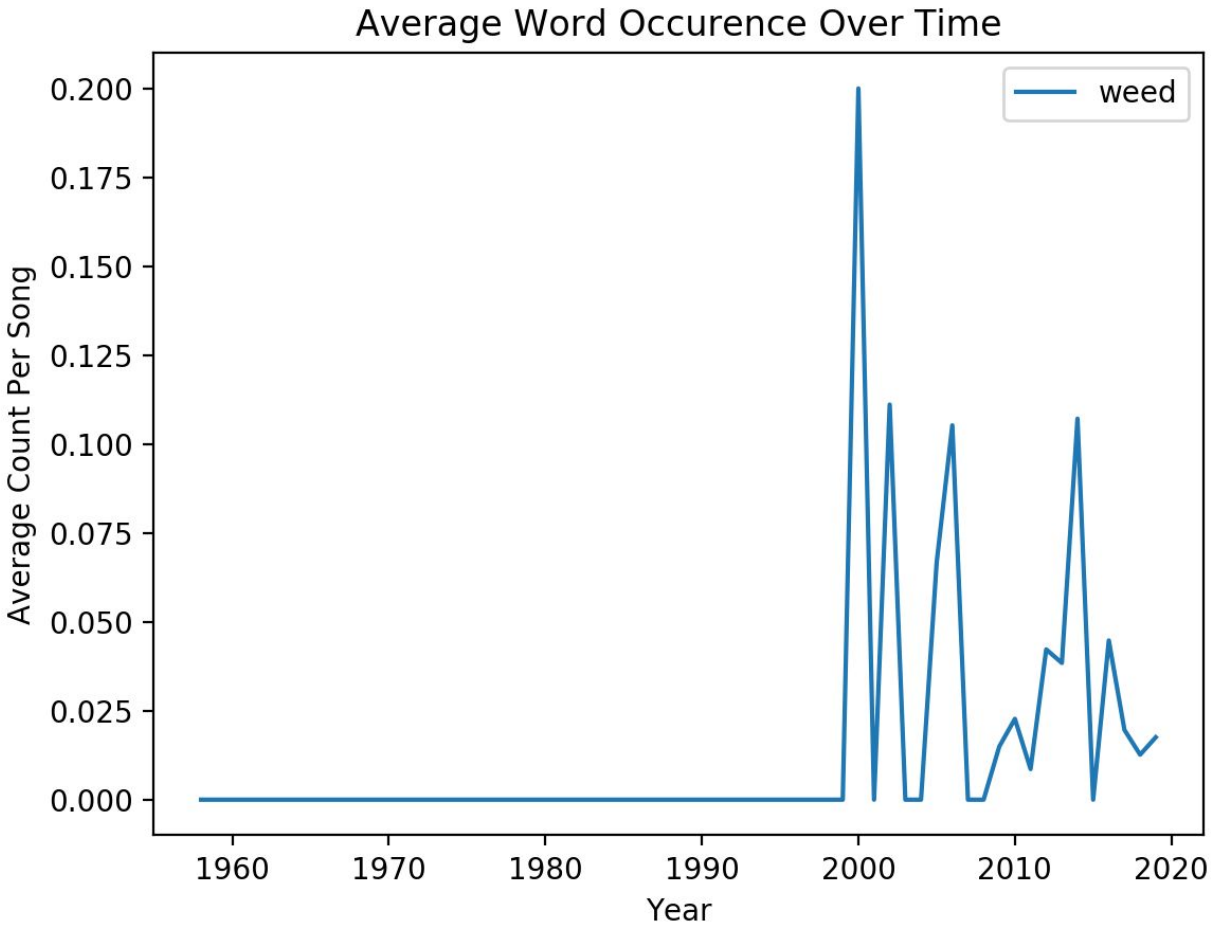
0 to 100

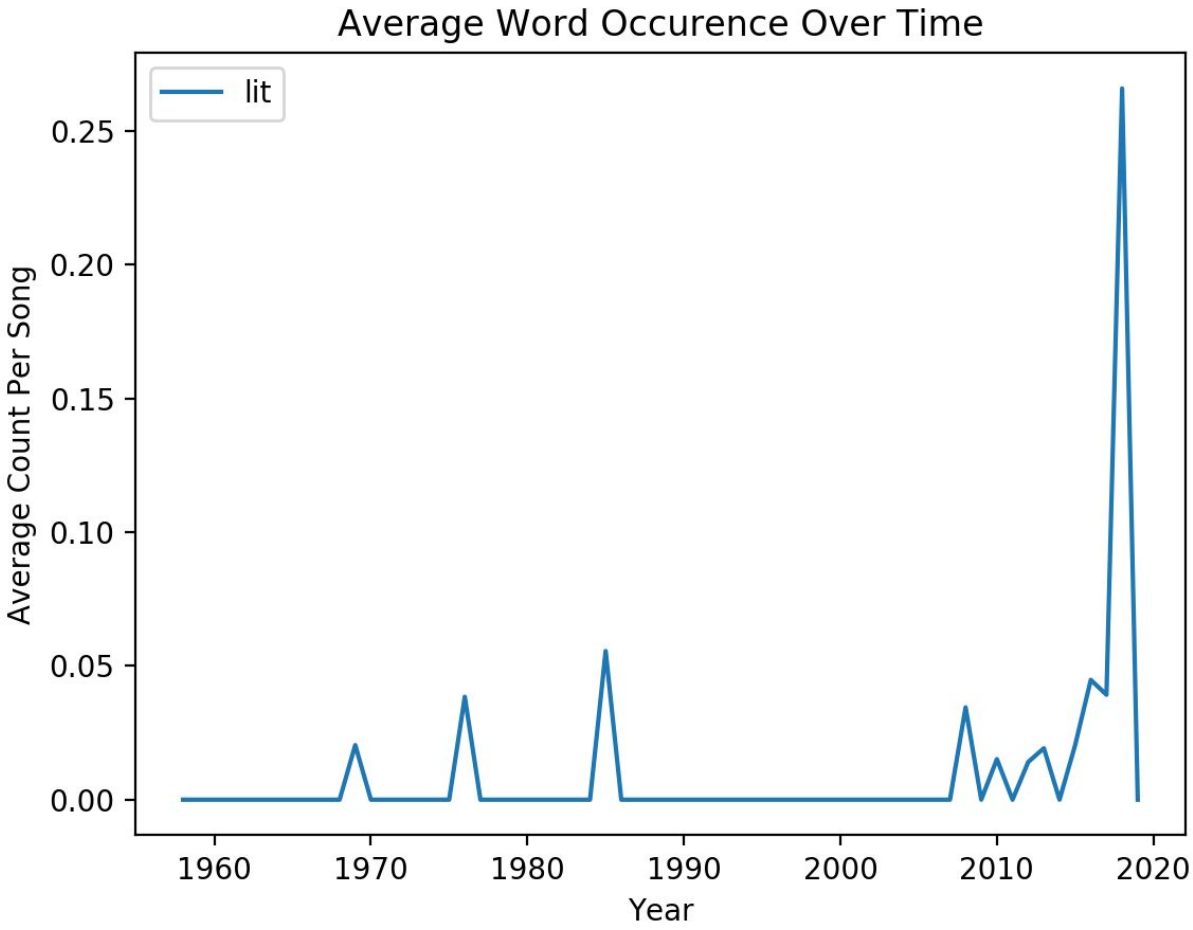
Positive

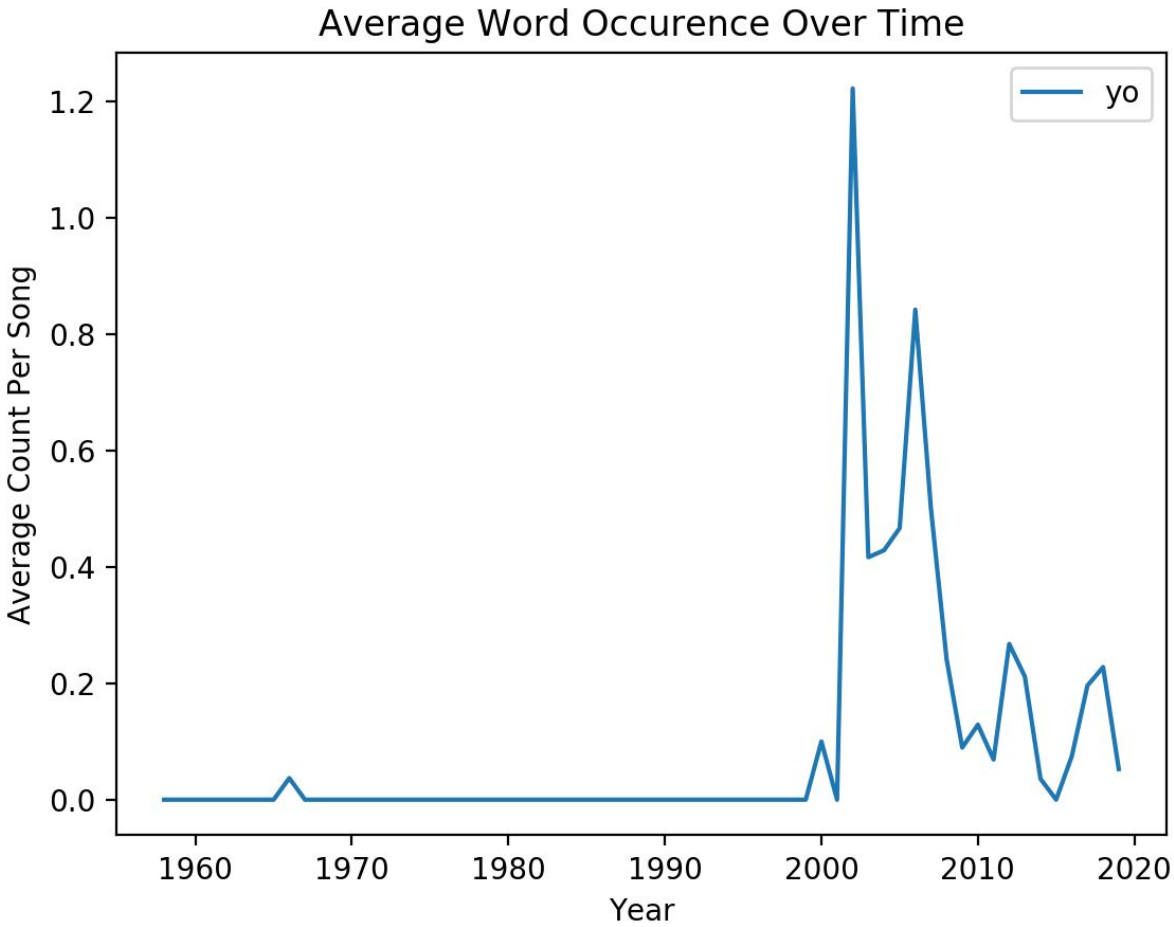
Best I Ever Had

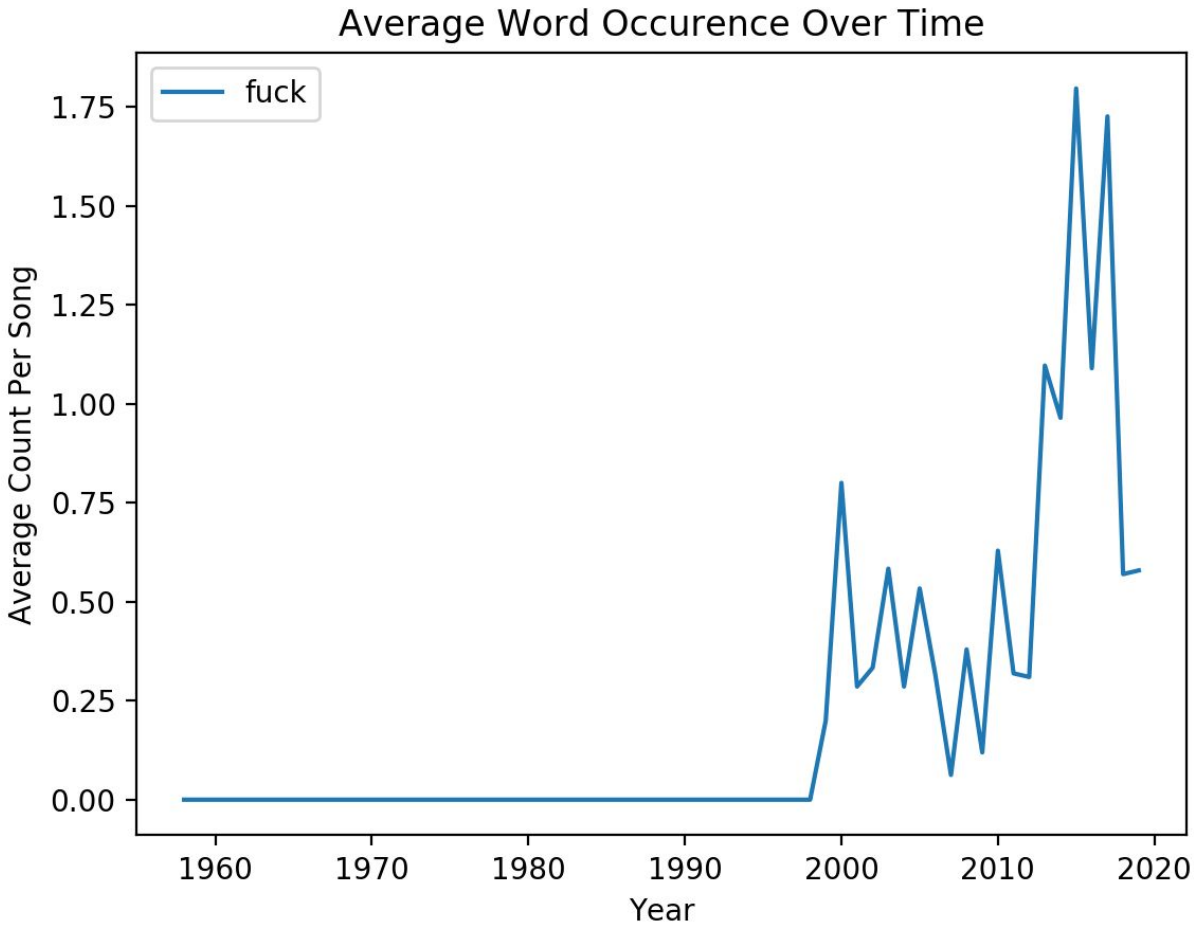
Side Task II: Visualization - Word Frequency




Average Word Occurence Over Time
weed
Average Count Per Song
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
1960
1970
1980
1990
2000
2010
2020
Year


Average Word Occurence Over Time
lit
Average Count Per Song
0.25
0.20
0.15
0.10
0.05
0.00
1960
1970
1980
1990
2000
2010
2020
Year


Average Word Occurence Over Time
yo
Average Count Per Song
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1960
1970
1980
1990
2000
2010
2020
Year


Average Word Occurence Over Time
fuck
Average Count Per Song
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
1960
1970
1980
1990
2000
2010
2020
Year

Side Task III: Most Similar Artists

- Given artist name in the data, who is his/her/their most similar artist?
  - Example
    - Input: Drake
    - Output: Kanye West

Implications

- Multi-level comparison of linguistic contexts
  - Normalized Word Frequency (TF-IDF)
  - Word relationship (Word Vectorization)
  - Sentiment Comparison (NLTK)

- Possible Applications
  - Plagiarism Detection
  - Writer/Speaker Identification: Customer Reviews/Survey Analysis
  - Cross-sectional analysis of famous authors/artists/students