Scraping and Analysing Song Lyrics and Metadata

2020-02-03 Data Wrangling (XB_0014)

F.D.	Brunet de Rochebrune	fbe690
R.K.	Dijkstra	rda630
J.E.	Van der Lei	jli530

Music is everywhere in today's society. The success of a song seems like a hit or miss. To investigate these matters further, we are interested in how different songs, artists and genres in music differ from each other and what their characteristics are. In order to do this analysis, we need a platform capable of supplying us with information regarding songs; Genius is a platform filled with lyrics from popular songs fit for our purpose. That led us to compose the following research question: *What are the characteristics of the genres, most popular songs and artists on genius.com? And how do they differ?* To answer this question we answered several sub questions regarding the uniqueness, explicitness, song duration, bpm, sentiment and word use within songs. Lastly, we also addressed the use of the Genius platform by artists and viewers.

Data sources

Our main source of data is Genius' own API.[1] This could supply us with the following information about individual songs: release date, pageviews, Pyongs[2], trending (hot), verified status[3], chart position, name and artist. The API returns data in JSON format and requires a `song_id`. This ID is an internal ID generated by Genius themselves for every song on the platform. There is no way of obtaining it unless we use their search API to look for a specific song's metadata. We had to find a different way to approach this, as looking up every song manually would take too much time.

The initial idea was to look at the most popular songs on Genius, however, the platform does not offer an API for that. We came to the conclusion that the best way to go about this is to scrape the website, as it showed the current most popular songs. Upon further inspection we found out that the website actually uses an undocumented API in the background to fetch data on the front page.[4] This API gave information about the current most popular songs – exactly what we were looking for. It also had the benefit of supplying a song ID with every element. This last part allowed us to combine this data with the previously mentioned, documented, Genius API. The undocumented API had some limits, but we were able to circumvent them by making multiple smaller requests. This allowed us to successfully scrape the top 150 songs per

[1] `https://docs.genius.com/`
[2] Pyongs are Genius' own like/recommend system. A user can place a song on their profile by clicking the Pyong button next to it.
[3] Verified artists are artists who have a profile on Genius themselves. This allows them to write annotations on their own song's lyrics with a 'verified' tag. These annotations go above the normal crowdsourced annotations made by Genius' users.
[4] `https://genius.com/api/songs/chart?time_period=all_time&chart_genre=all&page=1&per_page=10`

genre. We decided to limit this amount to said value to keep the database of reasonable size, but also not too big, as lyrics processing would take exponentially longer on larger sets.

We did not find any easy, or rather, API way of obtaining the actual lyrics, so we decided to resort to scraping the website. This was a reasonable easy task as we now had all the song IDs in a database. Only the URL needed to be changed with this ID and the correct HTML element needed to be selected. The result was stored and merged with the final dataframe.

Genius' API was good for some song information and most importantly the lyrics, but it lacked critical metadata that we would like to analyse, such as song duration, BPM and if it is an explicit song or not. This last point could theoretically be found by analysing the lyrics for profane words, however, some songs might be explicit in the meaning of the words, and not for using profanity per se. This was the reason we decided to use an external data provider.

There are a lot of public music services who offer streaming and, more importantly, an API. The first attempt was using Spotify's, however, it quickly showed difficulties and limits for our use. We stumbled upon the Deezer API, a less popular streaming service mostly used in France. Even though the service is based in France, it still offered the same – or sometimes even more – information than Spotify's offerings. We also discovered that the API can be used without any authentication nor rate limiting.[5] The challenge now was combining our current dataset with Deezer's new information.

We cannot just lookup the song ID we have from Genius' database in Deezer's API, as they both use their own ID system. We decided to use Deezer's search API to search each element of the current dataset, and return the first element found – this was almost always the correct result.

A problem we ran into was that some songs in Genius' lists were Russian. This is a problem for when we want to search in Deezer's API, as all the songs were transliterated in their database. Searching Russian/Cyrillic text will give zero results. We used the package `cyrtranslit` to transliterate all Russian songs and `urllib` to correctly make the request work with their API. From the Deezer API we were thus able to obtain information regarding BPM, duration and explicitness of songs.

All data was saved into a single CSV file, for able to import as DataFrame. The last data access was on 13 January 2020.

Data wrangling methods

From the data sources listed previously, we were able to obtain general information and lyrics from songs. The Genius and Deezer API both returned JSON formats, which could then be loaded into a pandas dataframe. In order to obtain lyrics data we had to use a web scraper, with the help of the `song_id` as obtained from the API we were able to make a request to the lyrics page. On the specific lyrics page we were able to extract the lyrics with the help of the `beautifulsoup4` HTML parser, these lyrics are now in "raw"-string format – including newline characters. In order to make this data fit for use we applied multiple NLP data wrangling methods. Lyrics data contains a lot of punctuation marks, brackets and parentheses; which are problems that can easily be tackled with the help of regular expressions.

[5] `https://api.deezer.com/search?q=eminem+rap+god`

We furthermore tokenized each of the lyrics, in order to correctly separate the words from each other. The tokenizer we used is called: TokTok Tokenizer, a tokenizer capable of identifying and tokenizing slang language (which came in useful as many lyrics contained slang and ad libs). Furthermore, we stemmed (with the help of the `nltk` package) each of the tokenized words, in order to convert it to its base form and to remove distinction between singular/plural forms of the words. To gain more insight in word-use in songs, we had to remove stopwords from the lyrics, as they do not provide us with any real additional information. Moreover, we also applied sentiment analysis with the help of the `vaderSentiment` package, a package also capable of conducting sentiment analysis on text subject to slang and abbreviations. We also implemented a check on whether a specific word was profane with the help of the `better-profanity` package.

One last step to conduct our analysis, was to construct a song-term-frequency matrix[6] and a song-bigram[7]-frequency matrix. These frequency matrices were constructed with the help of the `Counter` class of the `collections` module in python, which could then be converted into a pandas dataframe. The result of this wrangling is an extensive dataset, fit for analysis. In order to visualise important matters we used the matplotlib, seaborn and pandas package. *In our presentation we have highlighted several important figures which support our conclusions.*

Conclusion

Our analysis gave us a good understanding of the most popular song lyrics, genres, artists and use of the Genius platform. The following findings are among our most significant discoveries.

The genres Rap and Country are the most different from each other in terms of views, uniqueness and explicitness, with Rap getting the most views, Country using the most unique words and Rap having the most explicit songs and words. As for artist usage of the Genius platform, the genres Rap and R&B have the most active artists (in terms of verified artists), whereas Rock and Country have the least amount of verified artists percentage wise.

Our sentiment analysis shows that Rap has an overall more negative sentiment, compared to the other genres. Moreover, Rap lyrics differ from the other genres in terms of word use; Rap uses more explicit words, whereas the other genres had approximately the same word use. In the whole lyrics corpus, the word “like” is the most used word, the bi-word “yeah yeah” is the most occurring bi-word. The three most viewed artists on Genius turn out to be Drake, Kendrick Lamar and Eminem, with Eminem having the highest pyong/view ratio which suggests he is the most appreciated artist on the platform.

One limitation of our research was the absence of data of songs from the 19th century. Due to the origin and relatively young age of the Genius platform, not many songs from the 19th century were available, disallowing us to make a valid time series of metadata of songs. Furthermore, not many country songs were available on Genius (as it was originally a Rap platform). The final limitation of our research was the time constraint, if more time would be available, we could have scraped more songs and made a more comprehensive comparison between genres. Further research could address these issues.

[6] A matrix with rows representing the songs in the corpus, the columns representing all the unique stemmed words in the song corpus. The content of the matrix denotes how many times a word occurs in a particular song. The matrix is very sparse.
[7] The same matrix as the song-term frequency matrix, but now the columns represent the different bi-grams in a song (stopwords stripped).