Code Sections

I have given the ordering and outline of each notebook created for my project. I have specified which csv files are created at which points. For API and VADER lexicon uses, I included the websites I used for my code.

Part 1 Removing duplicates

Input: charts.csv

Output: 2015\_2020\_noDuplicates.csv, 2010\_2020\_noDuplicates.csv

Filtered the original dataset down to 2015-2020 and 2010-2020 and remove duplicates by dropping the same song and artist entry.

Part 2.1 Getting lyrics

Input: 2010\_2020\_noDuplicates.csv

Output: 2010\_2020\_noDuplicates\_lyrics.csv

Use the Genius API and run through all the 5,000 ish songs manually, to determine if they lyrics are correct and therefore added to the dataset. If incorrect, NaN is added.

*\*Note: I saved the same dataset twice, once using “2010\_2020\_noDuplicates\_lyrics” and again using “2010\_2020\_noDuplicates\_lyrics1” which is used in the remainder of the project\**

*Code reference:* [*https://pypi.org/project/lyricsgenius/*](https://pypi.org/project/lyricsgenius/)

Part 2.2 Missing Lyrics

Input: 2010\_2020\_noDuplicates\_lyrics1.csv

Used to print out all of the empty song with no lyrics so that they could be manually found. The notebook is basically empty now as I refreshed and re-read the dataset after entering new lyrics to ensure there is no remaining empty songs with no lyrics.

Part 2.3 Sentiment Analysis

Input: 2010\_2020\_noDuplciates\_lyrics1.csv

Output: 2010\_2020\_noDuplciates\_lyrics1.csv, 2010\_2020\_noDuplicates\_lyrics1\_sentiment.csv

Went through and removed all instances of /n or /u208 for anything unneeded. Then resaved the same dataset. I then went through each entry in the dataset, found the polarity score, split this up into each section and determined the overall positive or negative label.

*Code reference:* [*https://towardsdatascience.com/sentimental-analysis-using-vader-a3415fef7664*](https://towardsdatascience.com/sentimental-analysis-using-vader-a3415fef7664)

Part 3.1 Spotify

Input: 2010\_2020\_noDuplicates\_lyrics1\_sentiment.csv

Output: 2010\_2020\_lyrics1\_sentiment\_spotify.csv

Set up the Spotify API and did some testing. Went through all songs to get the popularity score, explicitness, and id. Saved this as the new dataset. Found all the missing values and attempted to fill them in

*Code reference:* [*https://towardsdatascience.com/extracting-song-data-from-the-spotify-api-using-python-b1e79388d50*](https://towardsdatascience.com/extracting-song-data-from-the-spotify-api-using-python-b1e79388d50)

Part 3.2 Missing Spotify

Input: 2010\_2020\_lyrics1\_sentiment\_spotify.csv

Output: 2010\_2020\_lyrics1\_sentiment\_spotify\_release.csv

Went through and attempted to find the release dates. Went through missing release dates.

*Code reference:* [*https://towardsdatascience.com/extracting-song-data-from-the-spotify-api-using-python-b1e79388d50*](https://towardsdatascience.com/extracting-song-data-from-the-spotify-api-using-python-b1e79388d50)

Part 3.3 Adding Date back

Input: 2010\_2020\_lyrics1\_sentiment\_spotify.csv, 2010\_2020\_lyrics1\_sentiment\_spotify\_release.csv

Output: 2010\_2020\_preTweet.csv

Accidently removed the date variable in the dataset so merged with a previous dataset to get them back. Also cleaned and rearranged the variables in my dataset.

Part 3.4 Checks

Input: 2010\_2020\_preTweet.csv

Checked for missing values, ect.

Part 3.5 Pre Tweets Graphs

Input: 2010\_2020\_preTweet.csv

Made a couple graphs.

Part 4.1 TWEETS nowplaying

Input: 2010\_2020\_preTweet.csv

Output: 2015\_2020\_nowplaying.csv

Testing the full archive search. Changed the artist’s name so any connectors are taken out and only the first artist is taken. Also calculate the dates two weeks from the release date and put all dates in the correct format. Collect up to 100 songs for each song, including #nowplaying hashtag, this was done by manually saying yes or no to each or it crashed. Saved the tweets and volume to a new dataset. There are missing values.

*\*Note: The newly saved “2015\_2020\_nowplaying” file is actually 2010-2020 data but was misnamed\**

*Code reference:* [*https://stackoverflow.com/questions/69849242/error-while-trying-to-extract-tweets-from-twitter*](https://stackoverflow.com/questions/69849242/error-while-trying-to-extract-tweets-from-twitter)

Part 4.2 Filling in nowplaying

Input: 2015\_2020\_nowplaying.csv

Output: 2015\_2020\_nowplaying.csv

Going through the songs without nowplaying tweets and attempted to re-find them, using variations of the song name or excluding brackets for example. Testing a bit with the sentiment analysis of tweets.

*Code reference:* [*https://stackoverflow.com/questions/69849242/error-while-trying-to-extract-tweets-from-twitter*](https://stackoverflow.com/questions/69849242/error-while-trying-to-extract-tweets-from-twitter)

Part 4.3 TWEETS pre-release

Input: 2015\_2020\_nowplaying.csv, extraData.csv

Output: 2015\_2020\_nowplaying.csv, extraData.csv

Gathering all the pre-release tweets from the songs. The dataset extraData was created just for convenience. Gathered the date 2 weeks pre-release for all songs and gathered the tweets. Had to gather the tweets by manually checking in batches of around 100-400 each time otherwise the API continued to time out and fail.

*Code reference:* [*https://stackoverflow.com/questions/69849242/error-while-trying-to-extract-tweets-from-twitter*](https://stackoverflow.com/questions/69849242/error-while-trying-to-extract-tweets-from-twitter)

Part 4.4 Sentiment Analysis

Input: 2015\_2020\_nowplaying.csv

Output: 2015\_2020\_nowplaying.csv

Changing the format of tweets so the sentiment can be analysed. This includes removing all variations of the song and artist name in terms of upper and lower cases, different connectors, inclusion and exclusion of brackets, nowplaying, ect. Then got the compound sentiment score of the tweets.

*Code reference:* [*https://towardsdatascience.com/sentimental-analysis-using-vader-a3415fef7664*](https://towardsdatascience.com/sentimental-analysis-using-vader-a3415fef7664)

Part 5.1 Training Classifiers

Input: 2015\_2020\_nowplaying.csv

Output: df\_BASIC.csv, df\_top10.csv, df\_spot75.csv

Creating datasets for Top 10, 20, 5 and Spotify > 75, 70, and 80 and training classic three classifiers without cross validation. Looking at the impact of the data being different balances and having different splitting of testing and training.

Part 5.2 Training Classifiers 2.0

Input: 2015\_2020\_nowplaying.csv

Training all classifiers for Top 10, 20, 5 and Spotify > 75, 80 and 70. This gathers results but does not use cross validation, so the results gathered are not the ones used and spoken about in my final report

Part 5.3 Final Classifier Results

Input: 2015\_2020\_nowplaying.csv

This is the final document with all of the classification models being trained, implemented cross validation, and producing the best results which have been used within the report

Part 5.4 Comparing Hit Songs

Input: df\_top10.csv, df\_spot75.csv, 2015\_2020\_nowplaying.csv

Creating graphs and doing analysis of the hit songs in each year for both Top 10 and Spotify>75. These graphs are included in my final report.

Part 6.1 Looking at Individual Song

Input: charts.csv

A follow through the entire project of one song, looking at its journey. Needed for the discussion results of my report

Part 6.2 Anomaly Analysis

Data analysis for the Anomaly data. Used in final report

Demonstration

Input: 2015\_2020\_nowplaying.csv

The notebook uses in the demonstration for my presentation of the final project.