CAPP 30122 Winter 2020
Final Project
Team Members: Chun Hu (chunhu), Yimin Li (liym15), Tianyue Niu(tniu)

A brief overview of the final project (200 words maximum)

Using Spotify Top 50 Songs data from 2010 to 2019, we created an application to recommend the best song to user based on his/her mood and preferences.

The questions asked before we make a recommendation are:

1. How are you feeling today? Happy, neutral, or sad?
2. Would you prefer new songs? Yes, or no?
3. Do you want something that is more relaxing, neutral, or intensive?

Data Source:

- Spotify 10-year Top 50 Songs data downloaded from Kaggle (https://www.kaggle.com/leonardopena/top-spotify-songs-from-20102019-by-year)
- Lyrics scraped from genius (https://genius.com/)

What the project tried to accomplish and what it actually accomplished (200 words)

We aimed to create a song-recommendation UI that allows users to enter their mood and preferences, and get a song recommendation in return. We were able to accomplish that. The following is a break down of the steps we took to achieve that goal:

1. Downloaded Spotify Top 50 Songs by year (from 2010 to 2019) from Kaggle (data source: https://www.kaggle.com/leonardopena/top-spotify-songs-from-20102019-by-year).
2. Scraped song lyrics from genius.com using BeautifulSoup (crawler.py).
3. Conducted sentiment analysis on each song's lyrics, created a new column for sentiment score (sentiment.py).
4. Through exploratory data analysis, find the best variables to match one's mood and emotions (query.py).
5. In Django, created a view function to get the requested data and defined the URL to return the information we want to display in our home page.
6. Created a HTML page, along with a CSS formatting file, to display the data and returned the page to users to view in the browser.

The overall structure of the software (1-page maximum)

Scripts

/final_project/manage.py

Running manage.py in our root folder will start the song recommendation UI.

```
$ python manage.py runserver
```

The script file in our root folder sets a default environment variable that points to our project's setting.py file, and calls a function passing in the provided command line arguments. It is the entry point of our project.

/final_project/ui/crawler.py

The script file crawler.py scrapes the lyrics from Genius.com based on our top10s.csv file (original kaggle data). The crawler performs crawls the lyrics from the Internet, returns a dictionary and stores the lyrics collected in "lyrics_file.json", preparing for the sentimental analysis. Crawler.py would print out "Lyrics successfully found: XXX" for each song during the crawling.

To run crawler.py, run the following command inside the terminal:

```
$ python crawler.py
```

(Note: crawler.py takes a relatively long time to run.)

/final_project/ui/sentiment.py

The script file sentiment.py merges lyrics_file.json (scraped lyrics) with top10s.csv (original kaggle data), performs sentiment analysis of the lyrics for each song, calculates a sentiment score ranging from -1 to 1 (the larger the score, the more positive the lyrics), and returns a cleaned version of the data named 'top_songs.csv'.

To run sentiment.py, run the following command inside the terminal:

```
$ python sentiment.py
```

(Note: sentiment.py takes a relative long time to install and run.)

/final_project/ui/query.py

The script file query.py selects songs from top_songs.csv based on users' input and returns the title of the song recommended. If there are more than one song that match with the user's requirement, a randomly chosen song from the list would be recommended each time the user hit submit button on our UI.

/final_project/ui/views.py

The script file views.py takes a web request and returns a web response. The class HomePageView creates a form instance based on the choice fields from the request, converts form data to a dictionary, executes the query function, and returns the HTML contents of our web page.

Data Files

/final_project/ui/top10s.csv

top10s.csv is the original dataset downloaded from kaggle (link listed in previous page)

/final_project/ui/lyrics_file.json

lyrics_file.json is the output of crawler.py.

/final_project/ui/top_songs.csv

top_songs.csv is the output of sentiment.py.