**INSY 5378**

**PROJECT 3:**

**MOOD BASED MUSIC PLAYLIST**

**Team 02**:

Dhanashri Ostwal (dhanashrivilas.ostwal@mavs.uta.edu) – 1001277328

Priyanka Sekhar (priyanka.sekhar@mavs.uta.edu) - 1001215254

Saranya Ravichandran (saranya.ravichandran@mavs.uta.edu) -1001160582

**BUSINESS / RESEARCH QUESTION ADDRESSED**

The idea of our project is two folds:

**Analyzing by building a Model:**

Building a model by training and testing a set number of songs along with their lyrics in order to come up with the classification of songs into two categories (“happy” and “sad”). This classification is for three cities (New York, Chicago and Dallas)

**Extending the application for the benefit of the user:**

The use of this classification for visualizing the confidence level of all the songs with respect to the cities and displaying the weather condition of the selected city for the particular date. In the end, based on this analysis we are recommending a list of songs to the user depending on whichever confidence level (“happy” or “sad”) of the sentiment is higher compared to the other

**IMPLICATIONS OF THE PROJECT**

From this project, we got a clear idea about the kind of songs (“happy” or “sad”) preferred by the people in each city, depending on the weather condition in the area. Also, depending on this model, we were able to recommend the user, a list of songs which was formulated by identifying the weather condition.

**EVALUATION OF MODELS**

The premise of our project is based on the “Classification model”. We used two types of classifiers: Naïve Bayes Classifier and SGD (Stochastic Gradient Descent) Classifier. We chose the Classification model for our project since our whole idea was based on predicting the mood (class label) of the songs as “happy” and “sad”.

We used the Naïve Bayes Classifier as it is computationally efficient and they are pretty efficient to train in the batch-learning mode. Also, the predictive performance of this classifier is good in context of text based categorization. For our project, we first used the Naïve Bayes Classifier to classify all our songs in train dataset into “happy” and “sad” categories by analyzing the lyrics of these songs. After which, we created a model based on our final training dataset by using the SGD classifier.

SGD classifier is a simple and efficient approach to discriminative learning of linear classifiers like Support Vector Machines (SVM) and Logistic Regression. SGD has been successfully applied to large scale and sparse machine learning problems encountered in text classification. SGD is very helpful when it comes to ease of implementation.

Next, we passed our testing dataset which we obtained by scraping three radio stations through our trained model. This classified the songs in testing dataset into “happy” and “sad” categories.

**DATA PREPROCESSING**

For fulfilling the purpose of our project, we collected the data for songs in the following two ways:

1. For training our model, we collected the train data from the “Last.fm” dataset which is a subset of Million Song Dataset.

<http://labrosa.ee.columbia.edu/millionsong/lastfm#work>.

We got the Artist and Song title information from the dataset. We had to scrape the corresponding lyrics for each song and their artist by passing them as arguments to these links: <http://www.songlyrics.com/> and <http://www.lyricsmode.com/>.

At the end we had a complete dataset by combining an existing dataset with the data

we collected.

1. For testing purposes, we collected the test data by scraping the songs played by the radio stations of three cities from 2016/11/14 to 2016/11/20 (7 days):

Dallas ---- (<http://jackontheweb.cbslocal.com/playlist/2016/11/20/)>

New York ----(<http://923amp.cbslocal.com/playlist/)> and

Chicago ----(<http://b96.cbslocal.com/playlist/2016/11/17/)>.

From these websites we got information such as the Date, Artist, Song Name, Time at which the song was played. Once we got this information, next we had to scrape the lyrics corresponding to the songs as we did for train data.

**Note:** We scraped the test data for 18 cities played during the 7 days. Due to time constraints, we were not able to include that in the front end. The code snippet showing the scraping will be attached in the below section.

**The link that we have used:** <http://www.cbsradio.com/market>

We are scraping the list of cities and corresponding links of the radio station and saving it and using those for further scraping the lyrics.

**Weather data collection:**

For obtaining the weather condition pertaining to a particular date and city, we made use of the API on the website <http://api.wunderground.com/> .We collected the historical weather data needed by passing the date, state and the city as the arguments.

A json file will be generated. From the json file we fetched only the weather condition required for our project.

**DESCRIPTIVE STATISTICS ON THE DATASET**

1. Training Data set:

We collected close to 700 songs along with their lyrics having artist, lyrics and the classification of the songs as “happy” or “sad” as the fields.

Note: Classification here was done by using Naïve Bayes Classifier.

1. Testing Dataset:

Approximately 115 songs were collected for each of the three cities for the 7 days taken into consideration. This dataset had time, artist, song name, lyrics as the fields.

Note: Collection of these tuples was done by web scraping.

**LESSONS LEARNT**

We learnt a lot of things from this project. Ranging from new techniques to various layers involved in building a complex model like ours. Most of which are explained below:

1. Data Collection isn’t easy

When we were doing some initial research for our project and looking for music dataset, we came across a number of such data sets, but none of them met our needs. One of these data sets was “The Million Song Dataset”, which is most widely used for any research involving music/songs. But, the data set didn’t have all the information we needed for fulfilling the purpose of our project. Which ultimately led us to creating our own data set by web scraping.

1. Scraping is fun!

We realized that scraping the web would render the content we need in a structured way as we will be pulling the data straight from the HTML page. Doing this helped us extract and manipulate the data with ease. Once we built the scraping model, we were able to extract data from all the pages of the radio stations.

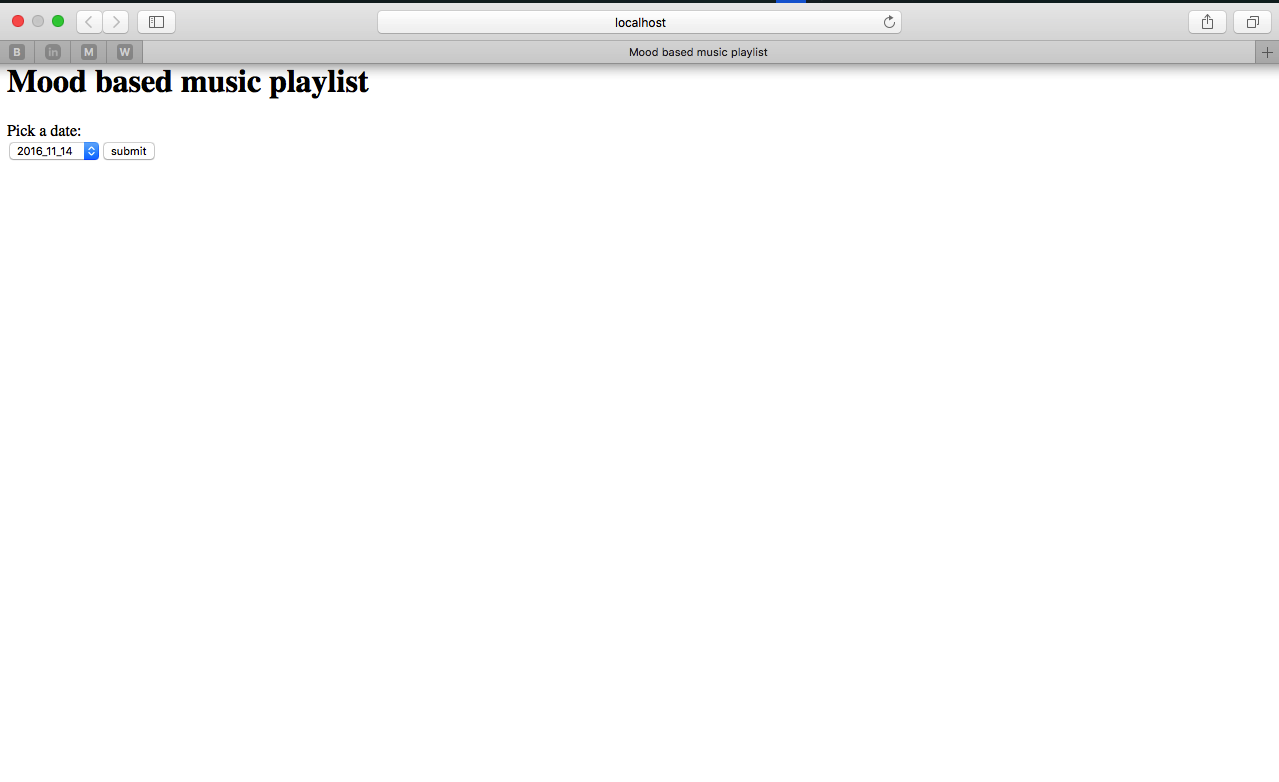
Another advantage of web scraping is that it is universally available. There’s no need to wait for a site to open up an API or even contact any organization. Once we find the data and we are able to figure out some basic access patterns, we are good to go.

1. There’s more to SVM than meets the eye

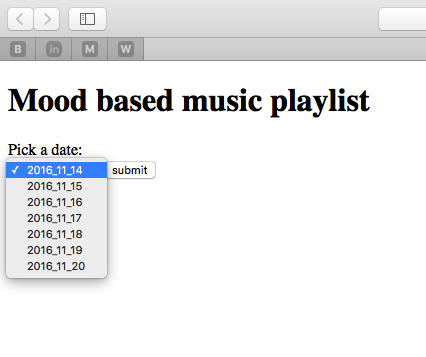
Implementing SVM classifier is generally a complex process. So, we thought of using SGD (Stochastic Gradient Descent), a variant of SVM for generating our model. SGD is an incremental anytime algorithm, which means that it can be applied to data streams or data sets that are too large. After we implemented the SGD, by analyzing the results we came to a conclusion that the efficiency of the classifier is very high. Also, it was interesting to implement as we had lots of opportunities for tuning the code according to our needs.

**SCREENSHOTS**

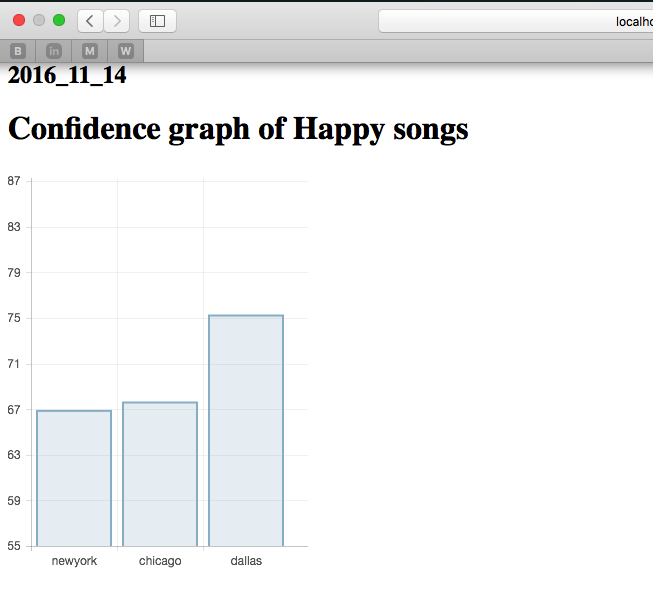
Landing page of our web application “Mood based music playlist”



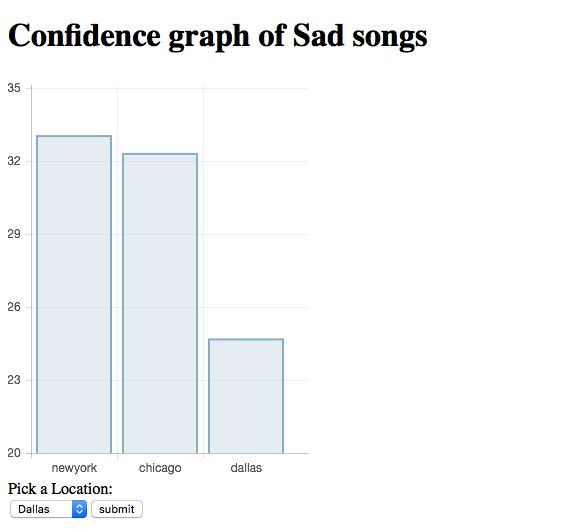
Date Picker option to select the date we want to generate the graphs for:



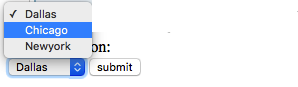
Confidence graph of the Happy Songs for each city for the chosen date (11/14/2016):



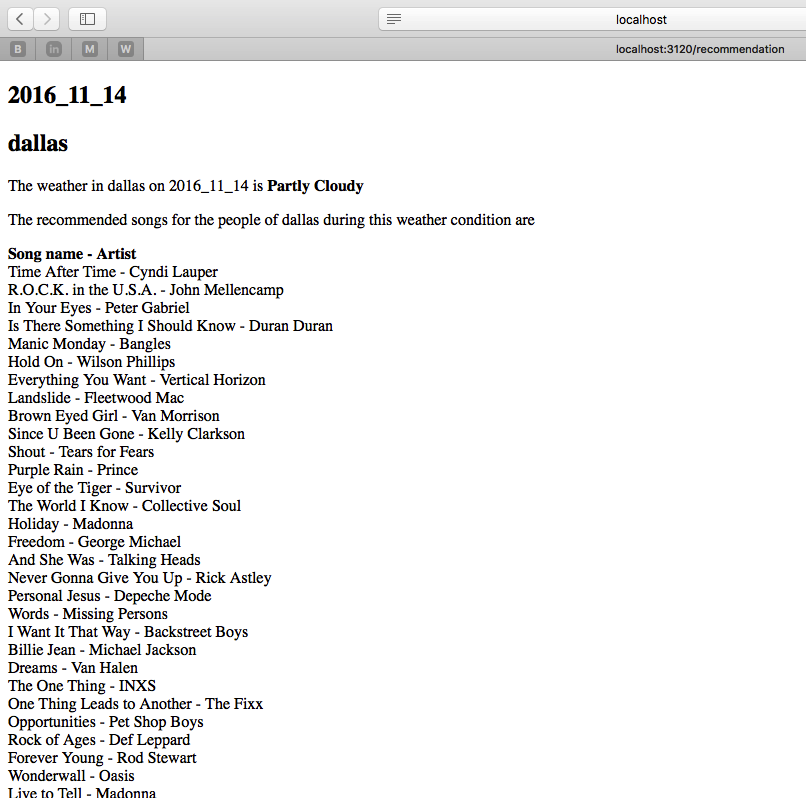
Confidence graph of Sad songs for each city for the chosen date (11/14/2016):



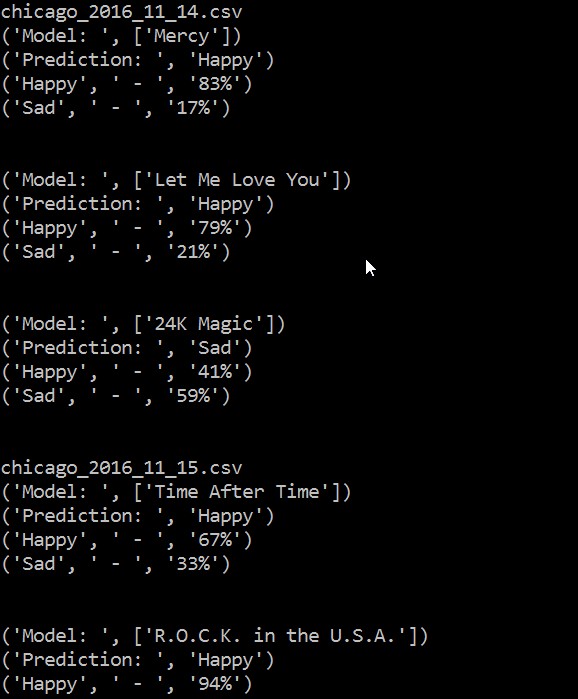
Location Picker for allowing the user to pick a desired location to get a list of song recommendations:



List of Songs Recommendations for the chosen date (11/14/2016) and the city (Dallas) according to the weather condition:



Model –



This is the snapshot of the SGD model. It gives us the prediction and the percentage of the classes happy and sad.

**CODE SNIPPETS**

1. Web Scraping:



This snippet is used to scrape the data for 18 different cities for 7 days. For each city we are fetching the corresponding radio station link. Since we had to scrape differently based on the city, we have 2 lists of cities for comparison.

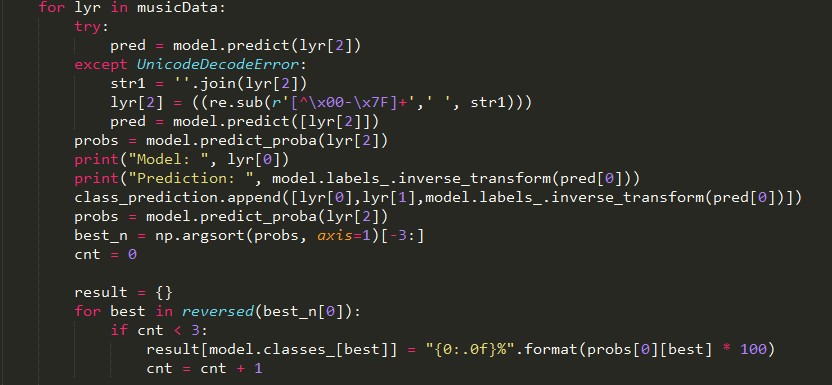


The above snippet shows how we scrape the song name, artist and the time the song was played, for each date from the links we got from the previous snippet. We stored the scraped data into different CSV files.

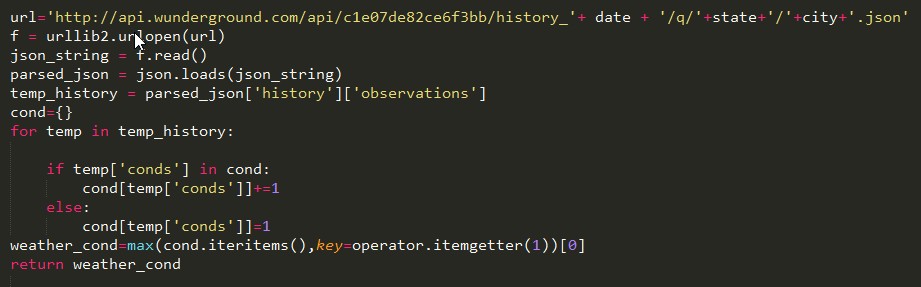


The above code explains how we scraped the lyrics for the corresponding songs. We passed the song title and artist as arguments to the link.

1. Classification Model:

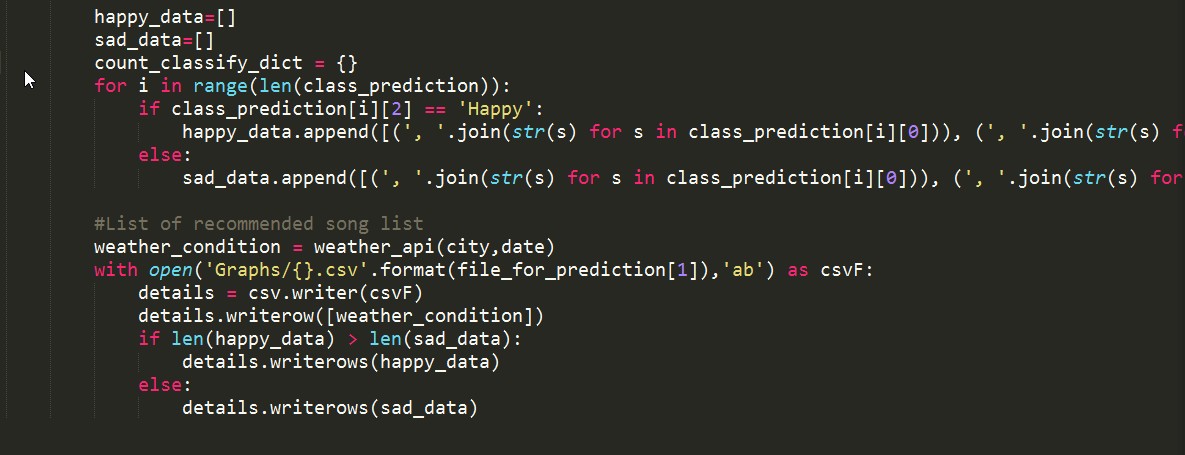


The code snippet is used to classify the song lyrics into ‘happy’ and ‘sad’ songs. This prediction is done using Naïve Bayes Classifier.



The above code is to fetch the weather information using the API for the corresponding city and date. We pass the date, city and state as arguments to the link with historical data.

1. Recommendation of songs:

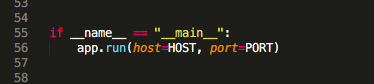


The code snippet above explains how we generate a recommendation of songs. Initially we separate the happy song classification and sad songs classification into 2 different lists for each city on a particular date. Later we compare the length of the lists to find the highest confidence level for each city and write it to a different CSV file. This list of songs is given as a recommendation when the user selects the city and date.

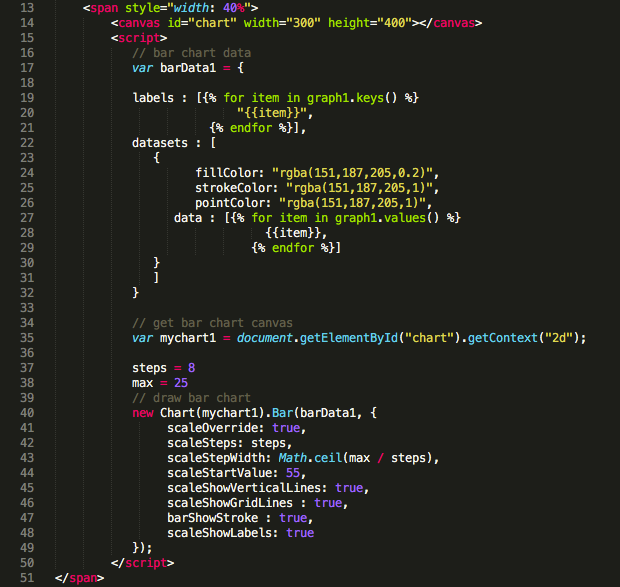
1. Front End (Python):



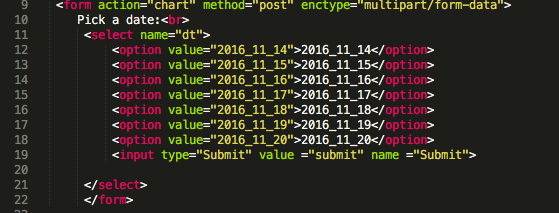
This part of the code allows the python script to be executed as Flask. So that when we run the python file, we will be redirected to <http://localhost:3120/>. And our landing HTML page which is rendered first is “first.html” located in our “Templates” folder.



By including this in our Flask script, we are indicating the program to look for the “main” function in the python script and execute it first by running on the specified HOST and PORT.



The above code snippet is for creating a canvas in the html page and generating bar graphs by specifying labels, colors and steps.



HTML code snippet for creating a drop down menu for the 7 dates taken into consideration (11/14/2016 to 11/20/2014).

**CONCLUSION:**

Almost everyone likes to listen to music, one or the other kind. With this project, we were able to understand how the weather in a particular city impacted the selection of songs by the listeners. To further get a better insight, we analyzed this trend spread across 7 different days and were able to visualize and conclude how much the selection of songs varied in these cities (Dallas being the Happiest and New York listening to “less happy” songs during the same period). Also, providing the user with a recommendation list of the songs depending on the city proved to be a better extension of our application, by making it more user friendly.