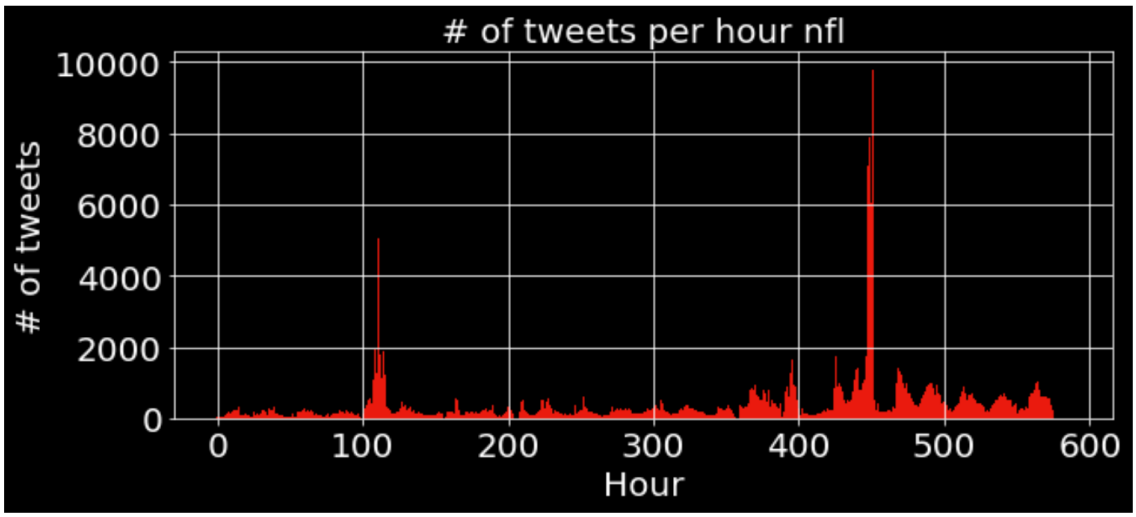
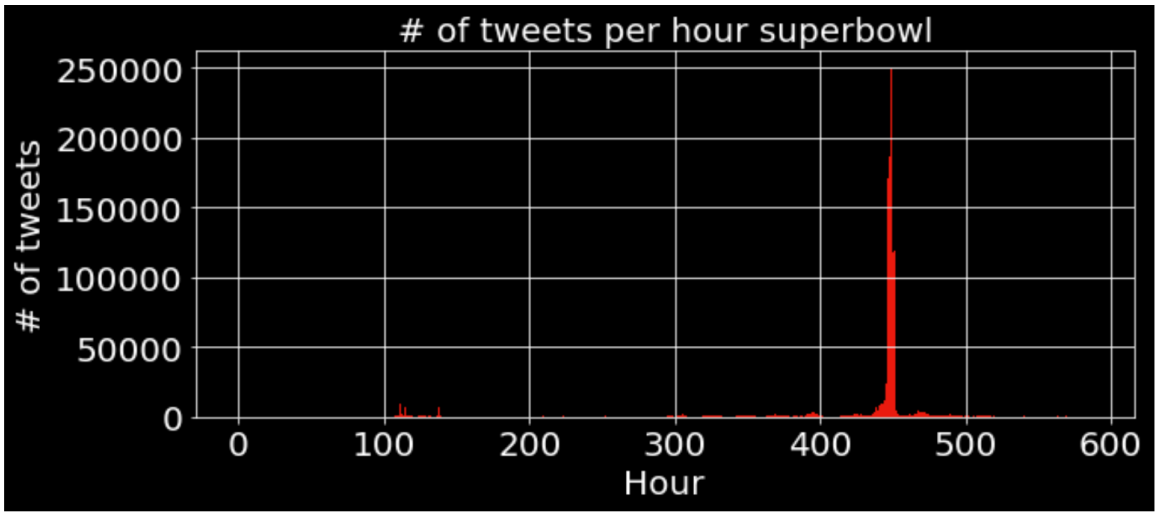
Project 3 report
Liyuan Zhao
005692591

Question 27: Report the following statistics for each hashtag. Average number of tweets per hour. Average number of followers of users posting the tweets per tweet (to make it simple, we average over the number of tweets; if a user posted twice, we count the user and the user's followers twice as well). Average number of retweets per tweet.

```
Filename: tweets_#gohawks.txt
Average number of tweets per hour: 292.09326424870466
Average number of followers of users posting the tweets per tweet:  2217.9237355281984
Average number of retweets per tweet:  2.0132093991319877
Tweetcounts:  169122
First time of Tweet:  2015-01-14 00:04:41-08:00
Last  time of Tweet:  2015-02-07 02:17:49-08:00
---------------------------------------------------------------------------------------------
-------------------------
Filename: tweets_#gopatriots.txt
Average number of tweets per hour: 40.888695652173915
Average number of followers of users posting the tweets per tweet:  1427.2526051635405
Average number of retweets per tweet:  1.4081919101697078
Tweetcounts:  23511
First time of Tweet:  2015-01-14 01:50:11-08:00
Last  time of Tweet:  2015-02-06 23:54:35-08:00
---------------------------------------------------------------------------------------------
-------------------------
Filename: tweets_#nfl.txt
Average number of tweets per hour: 396.97103918228277
Average number of followers of users posting the tweets per tweet:  4662.37544523693
Average number of retweets per tweet:  1.5344602655543254
Tweetcounts:  233022
First time of Tweet:  2015-01-14 00:00:04-08:00
Last  time of Tweet:  2015-02-07 10:55:36-08:00
---------------------------------------------------------------------------------------------
-------------------------
Filename: tweets_#patriots.txt
Average number of tweets per hour: 750.6320272572402
Average number of followers of users posting the tweets per tweet:  3280.4635616550277
Average number of retweets per tweet:  1.7852871288476946
Tweetcounts:  440621
First time of Tweet:  2015-01-14 00:07:18-08:00
Last  time of Tweet:  2015-02-07 10:55:00-08:00
---------------------------------------------------------------------------------------------
-------------------------
Filename: tweets_#sb49.txt
Average number of tweets per hour: 1275.5557461406518
Average number of followers of users posting the tweets per tweet:  10374.160292019487
Average number of retweets per tweet:  2.52713444111402
Tweetcounts:  743649
First time of Tweet:  2015-01-14 04:31:15-08:00
Last  time of Tweet:  2015-02-07 10:55:36-08:00
---------------------------------------------------------------------------------------------
-------------------------
Filename: tweets_#superbowl.txt
Average number of tweets per hour: 2067.824531516184
Average number of followers of users posting the tweets per tweet:  8814.96799424623
Average number of retweets per tweet:  2.3911895819207736
Tweetcounts:  1213813
First time of Tweet:  2015-01-14 00:13:07-08:00
Last  time of Tweet:  2015-02-07 10:00:08-08:00
---------------------------------------------------------------------------------------------
```

Question 28: Plot “number of tweets in hour” over time for #SuperBowl and #NFL (a bar plot with 1-hour bins).





Question 29: Describe your task.

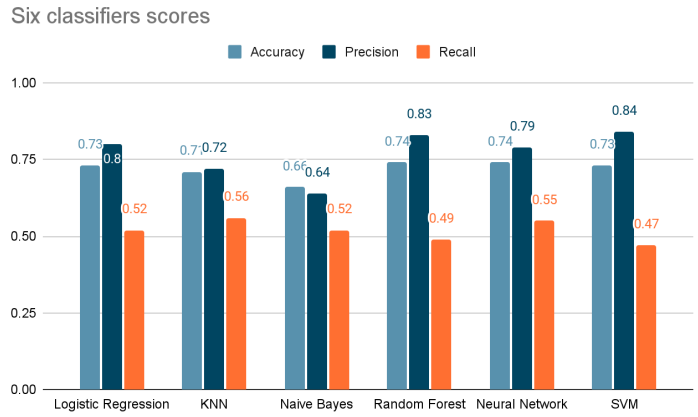
For this task, we are predicting the **fan base**, either of Seattle Seahawks or New England Patriot, from the given tweets dataset. The attribute of a tweet sentiment reveals a lot of information about the author of the tweet. For example, in Super Bowl 2015 the authors of positive sentiment tweets would originate from their state in this case authors of tweets from Seattle Washington would have positive tweets about their home team Seahawks and negative tweets about the opposing team, in this case the New England Patriots.

In the first half of the task, We are using tweets_#superbowl.txt to predict the location of the author of each tweet, in this case either Washington or Massachusetts. The tweet file contains one tweet in each line and tweets are sorted with respect to their posting time. Since each tweet is a JSON string, I load through Python as a dictionary, and then use Pandas to convert it as a dataframe file for further analysis. In order to track the location and encode the information, we are using the encoding function to return different numbers (1, -1, 0) for different locations text input, and append results to the list of titles. In the second half of the task, we split the data into train and test datasets and then implemented 6 different binary classifiers with Logistic Regression(L1 and L2 penalty), KNN, Naive Bayes, Random Forest Classifier, Neural network classifier and SVM and applied gridsearch for finding the best parameters, and report out the Precision/Recall Scores, ROC curves and confusion matrices.

Figure 1 shows the consolidated accuracy, precision and recall scores for all classifiers.

Classifier	Scores		
	Accuracy	Precision	Recall
Logistic Regression	0.73	0.80	0.52
KNN	0.71	0.72	0.56
Naive Bayes	0.66	0.64	0.52
Random Forest	0.74	0.83	0.49
Neural Network	0.74	0.79	0.55
SVM	0.73	0.84	0.47

Figure1. Compare scores for all classifiers

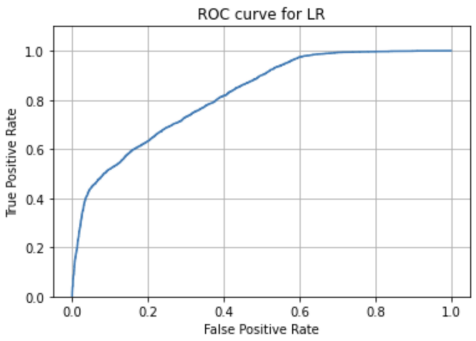


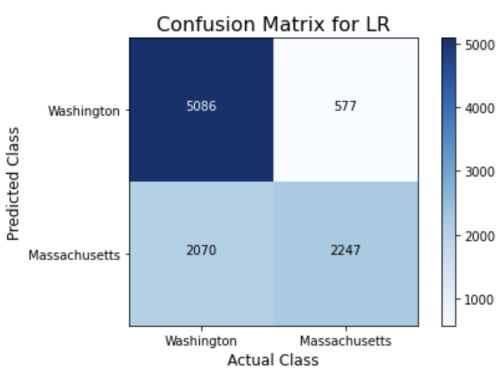
Result:

We can infer from the above table and plot that the Neural Network and Random Forest have the best accuracy. SVM has the best precision followed by Random Forest. We can use the relevant classifiers in the appropriate scenarios. We are reporting the ROC curves for all of the 6 classifiers mentioned below, the confusion matrix and the accuracy, precision and recall curves.

	precision	recall	f1-score	support
0	0.71	0.90	0.79	5663
1	0.80	0.52	0.63	4317
accuracy			0.73	9980
macro avg	0.75	0.71	0.71	9980
weighted avg	0.75	0.73	0.72	9980

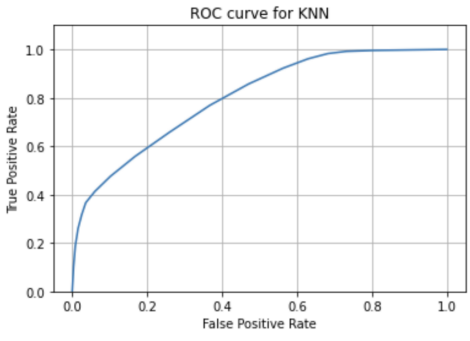
```
[[5086  577]
 [2070 2247]]
```

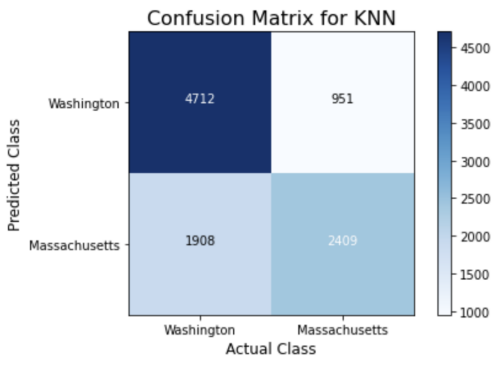




	precision	recall	f1-score	support
0	0.71	0.83	0.77	5663
1	0.72	0.56	0.63	4317
accuracy			0.71	9980
macro avg	0.71	0.70	0.70	9980
weighted avg	0.71	0.71	0.71	9980

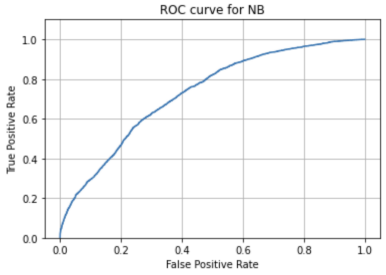
```
[[4712  951]
 [1908 2409]]
```

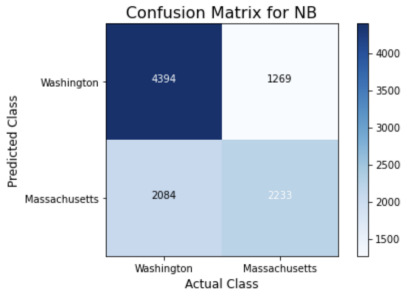




	precision	recall	f1-score	support
0	0.68	0.78	0.72	5663
1	0.64	0.52	0.57	4317
accuracy			0.66	9980
macro avg	0.66	0.65	0.65	9980
weighted avg	0.66	0.66	0.66	9980

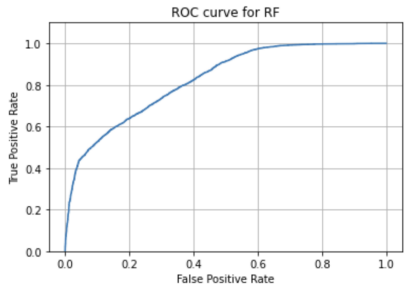
```
[[4394 1269]
 [2084 2233]]
```

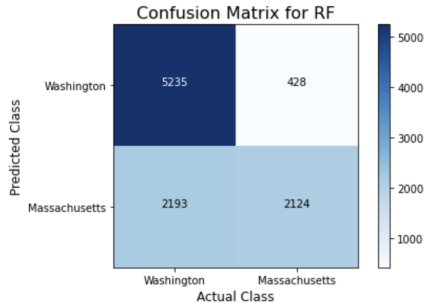




	precision	recall	f1-score	support
0	0.70	0.92	0.80	5663
1	0.83	0.49	0.62	4317
accuracy			0.74	9980
macro avg	0.77	0.71	0.71	9980
weighted avg	0.76	0.74	0.72	9980

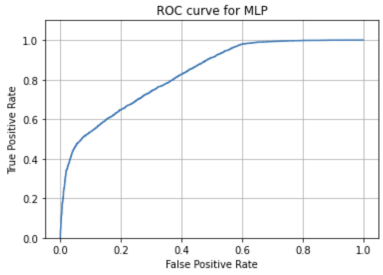
```
[[5235  428]
 [2193 2124]]
```

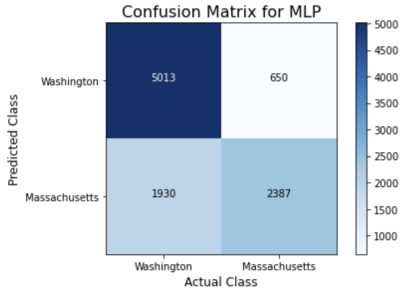




	precision	recall	f1-score	support
0	0.72	0.89	0.80	5663
1	0.79	0.55	0.65	4317
accuracy			0.74	9980
macro avg	0.75	0.72	0.72	9980
weighted avg	0.75	0.74	0.73	9980

```
[[5013  650]
 [1930 2387]]
C:\Users\ericz\AppData\Roaming\Python\Python39\site-packages\sklearn
ged yet.
  warnings.warn(
```





	precision	recall	f1-score	support
0	0.70	0.93	0.80	5663
1	0.84	0.47	0.61	4317
accuracy			0.73	9980
macro avg	0.77	0.70	0.70	9980
weighted avg	0.76	0.73	0.72	9980

```
[[5284  379]
 [2275 2042]]
```