Project 5
Application - Twitter data

Donna Branchevsky
UID: 404473772

Pavan Holur
UID: 204403134

Megan Williams
UID: 104478182

Tadi Ravi Teja Reddy
UID: 505227246

University of California, Los Angeles

Winter 2019

0 Introduction

This project involves the prediction of popularity of a tweet or set of tweets in Twitter, given attributes and the tendencies observed in training tweets. The training set of tweets is large and rich full of features, suitable to be exploited by prediction algorithms. The dataset provided contains 6 files namely: (the total size is 12 GB of text data)

1. "tweets_#gohawks.txt"
2. "tweets_#gopatriots.txt"
3. "tweets_#nfl.txt"
4. "tweets_#patriots.txt"
5. "tweets_#sb49.txt"
6. "tweets_#superbowl.txt"

Preliminary tests show that in fact file 2 is the smallest file and file 6 the largest. The text files are compiled bsed on the tag associated with the tweet recorded; i.e the file "tweets_#*.txt" contains only tweets with * tags in the tweet. This report intends to model the popularity of tweets using various features and models and provide analytic about their relative performance.

1 Question 1

First, we report trivial statistics for each hashtag. Note that all features cannot be extracted at once from the files. **The data set is too large and limited RAM capacity may halt the performance of the copy function in datasets** 5 **and** 6**.** Information is extracted using JSON accessors and/or Pandas tools.

1.1 Average Number of Tweets per Hour

We consider integer number of hours starting from the first full hour after the incomplete hour and ending the last full hour before the incomplete hour. This is done by implementing the following logic: (similar for *min_citation_date*)

$$max\_citation\_date\_new = \frac{max\_citation\_date}{3600} * 3600 \tag{1}$$

The results are given in the table below. Note that we average across consecutive hours along the whole timeline of provided data.

Metric/Tag	#gohawks	#gopatriots	#nfl	#patriots	#sb49	#superbowl
Avg. Num / Hr	292.4877	40.9547	397.0212	750.8939	1276.8565	2072.1174

Table 1: Average Number of Tweets per Hour

1.2 Average Number of Followers of Users per Tweet

A similar assessment is done across the number of tweets, this time considering the number of followers for the average tweets. With the same trimmed dates as in equation 1, we get table 2.

Metric/Tag	#gohawks	#gopatriots	#nfl	#patriots	#sb49	#superbowl
Avg. Foll / Tweet	2217.9237	1427.2526	4662.3754	3280.4635	10374.1602	8814.9679

Table 2: Average Followers of Users per Tweet

1.3 Average Number of Retweets per Tweet

Another assessment is done across the number of tweets, this time considering the number of retweets for the average tweets. With the same trimmed dates as in equation 1, we get table 3.

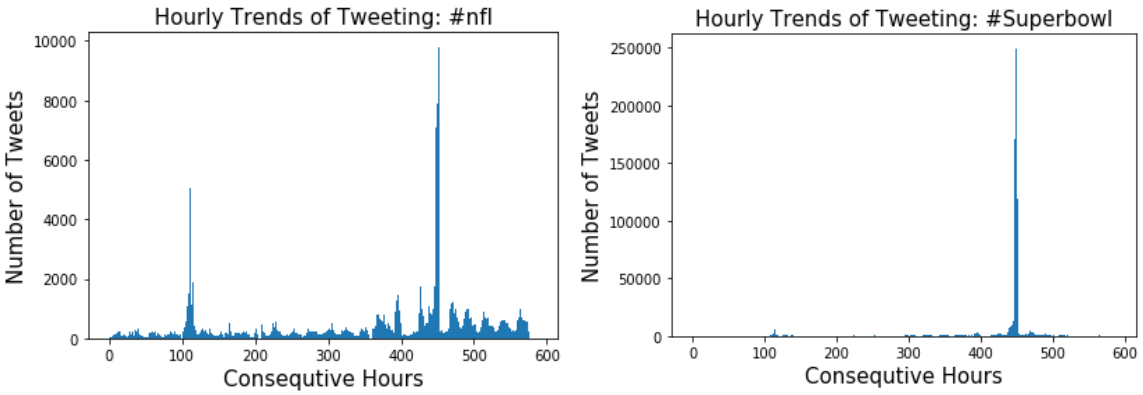
Metric/Tag	#gohawks	#gopatriots	#nfl	#patriots	#sb49	#superbowl
Avg. Retweets / Tweet	2.0132	1.4082	1.5344	1.7853	2.5271	2.3912

Table 3: Average Retweets per Tweet

2 Question 2

The number of tweets are now plotted on a histogram with bins for every integer count of hour from the earliest feasible begin citation date and latest feasible citation date. The histograms would represent the approximate times when a certain tag's popularity "heats" up.

Logically, we see the #superbowl tag erupt during the Superbowl and the #nfl bursts could be attributed to 2 major events, one of which is the Superbowl. **It is notable that the disparity of the tweets is so high that in some sense equation** 1 **is not necessarily useful.**



(a) #nfl

(b) #superbowl

Figure 1: Number of Tweets per hour through dataset

3 Question 3

Next, we look at Linear Regression as applied to each tag separately. Our simple regressor will not require any tuning hyperparameters. Building the feature extractor, and passing the target and feature set to the regression model, we are obliged to report the Mean Squared Error (MSE), and R-measure (R2).

It will also be useful to observe the significance of each feature in our regressor, so we will also extract the t-test and p-value. The tags are listed independently; inter-tag analysis is obscure because each tag is unique in many ways. The MSE, R-squared measure and t,p value table is provided below for each tag. Any value of order $< 10^{-6}$ is set to 0. The features used for the naive approach are listed below. (aggregated when not mentioned)

1. Number of tweets (N)
2. Number of retweets (NR)
3. Sum of followers (SNF)
4. Max of followers (MF)
5. Time of Day (in hours) (TD)

3.1 #gohawks

The low R-squared value indicates poor modelling. This makes sense as we are sweeping across all data points (through spikes) to predict a linear trend, which is not physically observed in Question 2. But within the confines of this model, **the features "number of tweets" and "number of retweets" are the top features with the former being the dominant one**. This results were obtained by the lower p-values as seen in the tables below.

	MSE	**R2**
Value	761167.1337	0.4764

Table 4: MSE, R2 heuristics for #gohawks

Feature	**t-val**	**p-val**
number of tweets	7.8164	0.0
number of retweets	-3.1305	0.0018
sum of followers	-2.4251	0.0156
max of followers	0.4117	0.6807
time of day	0.2914	0.7708

Table 5: T-val, P-val comparison of Features for #gohawks

3.2 #gopatriots

The slightly higher R-squared value indicates better modelling. Within the confines of this model, **the features "number of tweets" and "number of retweets" are the top features with the later being the dominant one**. This results were obtained by the lower p-values as seen in the tables below. In both the models seen till now, the feature "max of followers" seems to be performing the worst in contributing to the regression.

	MSE	**R2**
Value	27683.7421	0.6293

Table 6: MSE, R2 heuristics for #gopatriots

Feature	t-val	p-val
number of tweets	1.071	0.2846
number of retweets	2.5853	0.01
sum of followers	-0.5242	0.6003
max of followers	-0.0889	0.9292
time of day	-0.1964	0.8444

Table 7: T-val, P-val comparison of Features for #gopatriots

3.3 #nfl

The R2 value shows the model is still poor at capturing the trends fully. Within the confines of this model however, **the feature "number of tweets" is the top feature with the remaining features except time of day not much worse**. This results were obtained by the lower p-values as seen in the tables below.

	MSE	R2
Value	270842.2338	0.5706

Table 8: MSE, R2 heuristics for #nfl

Feature	t-val	p-val
number of tweets	4.1899	0.0
number of retweets	-2.5854	0.01
sum of followers	4.574	0.0
max of followers	-3.524	0.0005
time of day	0.1127	0.9103

Table 9: T-val, P-val comparison of Features for #nfl

3.4 #patriots

The slightly higher R-squared value indicates better modelling. Within the confines of this model, **the feature "number of tweets" and "max of followers" are the top features with the former being the dominant one**. This results were obtained by the lower p-values as seen in the tables below. **The max followers could be useful in this model as logically, followers of a Patriots fan are mostly Patriot fans.**

	MSE	R2
Value	5198554.8773	0.6684

Table 10: MSE, R2 heuristics for #patriots

Feature	t-val	p-val
number of tweets	12.915	0.0
number of retweets	-1.1764	0.2399
sum of followers	-0.4163	0.6773
max of followers	1.3356	0.1822
time of day	-0.4296	0.6676

Table 11: T-val, P-val comparison of Features for #patriots

3.5 #sb49

The higher R-squared value indicates better modelling but the MSE is very high. This implies that the error was created by a few large deviations but for the most part, the linear regressor captured the trends well. This makes sense as the tag is very specific to the game played on Superbowl day. Within the confines of this model, **the features "number of tweets" and "max of followers" are the top features with the former being the dominant one**. This results were obtained by the lower p-values as seen in the tables below. All features are pretty good except time of day.

	MSE	R2
Value	16244201.2156	0.8046

Table 12: MSE, R2 heuristics for #sb49

Feature	t-val	p-val
number of tweets	12.997	0.0
number of retweets	-2.035	0.0423
sum of followers	0.7754	0.4384
max of followers	2.1606	0.0311
time of day	-0.6531	0.5139

Table 13: T-val, P-val comparison of Features for #sb49

3.6 #superbowl

The higher R-squared value indicates better modelling but the MSE is very high. This implies that the error was created by a few large deviations but for the most part, the linear regressor captured the trends well. This matches our graphs in Question 2. Within the confines of this model, **all features are significant with "time of day" being the most irrelevant**. This results were obtained by the lower p-values as seen in the tables below.

	MSE	R2
Value	52663033.1874	0.7998

Table 14: MSE, R2 heuristics for #superbowl

Feature	t-val	p-val
number of tweets	28.4874	0.0
number of retweets	-5.5339	0.0
sum of followers	-6.2537	0.0
max of followers	4.8794	0.0
time of day	-0.4662	0.6412

Table 15: T-val, P-val comparison of Features for #superbowl

4 Question 4

Now we consider additional features that we find in the twitter data, and those which we believe can improve the linear regression of our each hashtag. In specific, we considered the following 5 additional features in our analysis: (for this section the window is 3600 seconds)

1. Number of Impressions of tweets (NI)
2. The total ranking score (RS)
3. Total friend count of users posting the tag (UCF)
4. Total Listed count of users posting the tag (LC)
5. Followers of the original author of the tweet (FOA)

Now we defend our intuition about these features playing a role in improving Linear Regression. More **impressions** could correlate to more popularity and more tweets following it. The **sum of the ranking scores** could represent the quality of tweets of all users in an hour preceding the target. **Total friend count** could correlate to the influence a certain person has on potential tweets by his friends in the future. **The listed count** could examine the rate at which a potential tweeter sees a tweet and thinks about tweeting and **the followers of the original tweet** might influence the number of users who "react" to the current tweet to post more in the immediate future.

We concatenate these features to the existing 5 features already extracted and in the assumption of orthogonality of the considered features, the results of the Linear Regressor on the hash tags are independently discussed.

4.1 #gohawks

This result is improved from the baseline. We can see that a higher R2 and lower MSE imply better fitting to the data. Additionally the feature analysis indicates that the **aggregated listed count is an excellent parameter to predict a tweet in the succeeding hour, while sum of followers and followers of original author are good features as well**. Number of retweets is not a bad feature either.

	MSE	**R2**
Value	541059.4294	0.6278

Table 16: Improved MSE, R2 heuristics for #gohawks

Feature	**t-val**	**p-val**
number of tweets	-1.5252	0.1278
number of retweets	-2.5217	0.012
sum of followers	-4.6768	0.0
max of followers	1.3849	0.1666
time of day	0.4139	0.6791
number of impressions	1.9011	0.0578
ranking score	1.5223	0.1285
friend count of user	0.6604	0.5092
listed count	10.2646	0.0
foll. of orig. author	-4.8636	0.0

Table 17: Improved T-val, P-val comparison of Features for #gohawks

4.2 #gopatriots

This result is improved from the baseline. We can see that a higher R2 and lower MSE imply better fitting to the data. Additionally the feature analysis indicates that the **aggregated listed count is an excellent parameter to predict a tweet in the succeeding hour, while sum of followers and followers of original author are good features as well**. This is interestingly similar to #gohawks. These features might be specifically good for fan bases.

	MSE	**R2**
Value	24841.7768	0.6674

Table 18: Improved MSE, R2 heuristics for #gopatriots

Feature	t-val	p-val
number of tweets	-2.361	0.0186
number of retweets	-0.4949	0.6209
sum of followers	-2.7361	0.0064
max of followers	0.5865	0.5578
time of day	0.236	0.8135
number of impressions	2.4084	0.0163
ranking score	2.6492	0.0083
friend count of user	0.1136	0.9096
listed count	4.6933	0.0
foll. of orig. author	-3.8809	0.0001

Table 19: Improved T-val, P-val comparison of Features for #gopatriots

4.3 #nfl

This result is improved from the baseline. We can see that a higher R2 and lower MSE imply better fitting to the data. Additionally the feature analysis indicates that the **number of tweets, ranking score and listed count are excellent features**.

	MSE	R2
Value	254800.5276	0.5961

Table 20: Improved MSE, R2 heuristics for #nfl

Feature	t-val	p-val
number of tweets	4.1598	0.0
number of retweets	-3.3227	0.0009
sum of followers	1.8576	0.0637
max of followers	-1.7152	0.0868
time of day	-0.4511	0.6521
number of impressions	-1.3539	0.1763
ranking score	-3.7096	0.0002
friend count of user	-2.2985	0.0219
listed count	5.4615	0.0
foll. of orig. author	-2.4199	0.0158

Table 21: Improved T-val, P-val comparison of Features for #nfl

4.4 #patriots

This result is improved from the baseline. We can see that a higher R2 and lower MSE imply better fitting to the data. Additionally the feature analysis indicates that the **number of tweets, sum of followers, ranking score, listed count and followers of original author the strongest features**. Among these, number of tweets is the strongest feature.

	MSE	**R2**
Value	4174354.1958	0.7337

Table 22: Improved MSE, R2 heuristics for #patriots

Feature	**t-val**	**p-val**
number of tweets	8.2656	0.0
number of retweets	1.8505	0.0647
sum of followers	4.5124	0.0
max of followers	-3.1899	0.0015
time of day	-0.4198	0.6748
number of impressions	-1.9247	0.0548
ranking score	-7.2117	0.0
friend count of user	-1.019	0.3087
listed count	5.8649	0.0
foll. of orig. author	-5.3544	0.0

Table 23: Improved T-val, P-val comparison of Features for #patriots

4.5 #sb49

This result is improved from the baseline. We can see that a higher R2 and lower MSE imply better fitting to the data. Additionally the feature analysis indicates that the **number of tweets, sum of followers, number of impressions, ranking score and friend count of user are excellent features**. Among these, friend count of user is the strongest feature.

	MSE	**R2**
Value	13507555.0099	0.8375

Table 24: Improved MSE, R2 heuristics for #sb49

Feature	t-val	p-val
number of tweets	6.8611	0.0
number of retweets	1.7696	0.0773
sum of followers	4.6139	0.0
max of followers	0.119	0.9053
time of day	-1.1719	0.2417
number of impressions	-4.1389	0.0
ranking score	-6.8594	0.0
friend count of user	7.752	0.0
listed count	-0.7751	0.4386
foll. of orig. author	-1.9062	0.0571

Table 25: Improved T-val, P-val comparison of Features for #sb49

4.6 #superbowl

This result is improved from the baseline. We can see that a higher R2 and lower MSE imply better fitting to the data. Additionally the feature analysis indicates that the **listed count and number of impressions are the best features with sum of followers a close third**.

	MSE	R2
Value	40567005.5249	0.8458

Table 26: Improved MSE, R2 heuristics for #superbowl

Feature	t-val	p-val
number of tweets	1.5723	0.1164
number of retweets	-2.6246	0.0089
sum of followers	3.7518	0.0002
max of followers	1.7523	0.0803
time of day	0.2582	0.7964
number of impressions	-4.3967	0.0
ranking score	-1.5161	0.13
friend count of user	2.6552	0.0081
listed count	5.6289	0.0
foll. of orig. author	-3.3665	0.0008

Table 27: Improved T-val, P-val comparison of Features for #superbowl

At the end of the analysis across tags, we believe that the following 5 features could help in modelling the Linear Regressor across tags best: (aggregated across a fixed window)

1. Listed Count (LC)
2. Number of Tweets (N)
3. Followers of Original Author (FOA)
4. Ranking Score (RS)
5. Number of retweets (NR)
6. Sum of Followers (SNF) (*highly correlated to FOA if the tweet is original.* Use the better one when necessary, but NOT both)

5 Question 5

To verify that our analyses are legitimate, we cross check by plotting the target variable w.r.t to the top features for each tag. This is measured by the lowest p-values described in the tables in Question 4. We expect a well-correlated plot; i.e there must exist a linear trend that describes the relationship between the feature and target. **Note that the linear regressor in general cannot well predict the target at the peak of the superbowl tweeting period as the majority of tweets it is being trained on have a near** 0 **target.** The highly unbalanced tweet rate observed in part 1 cannot be solved with naive Linear Regression as the outliers are not modelled well.

5.1 #gohawks

We find the plots for this tag reasonable. The diffuse points observed are poorly modelled but the p-value is still low because the high variance points are sparse. Filtering these, we can see a linear relationship. **Note: We don't consider BOTH number of followers and number of followers of original author; ONLY one because they are highly correlated**

5.2 #gopatriots

We find the plots for this tag reasonable. The diffuse points observed are poorly modelled but the p-value is still low because the high variance points are sparse. Filtering these, we can see a linear relationship.

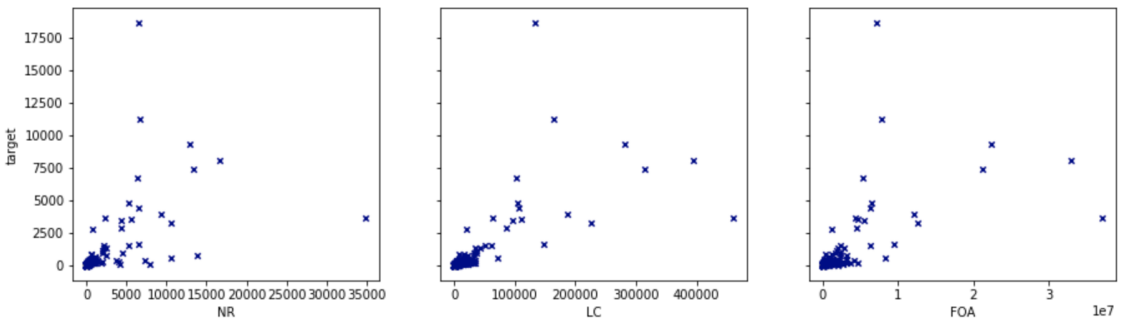


Figure 2: Analysis of target vs. top 3 features for #gohawks

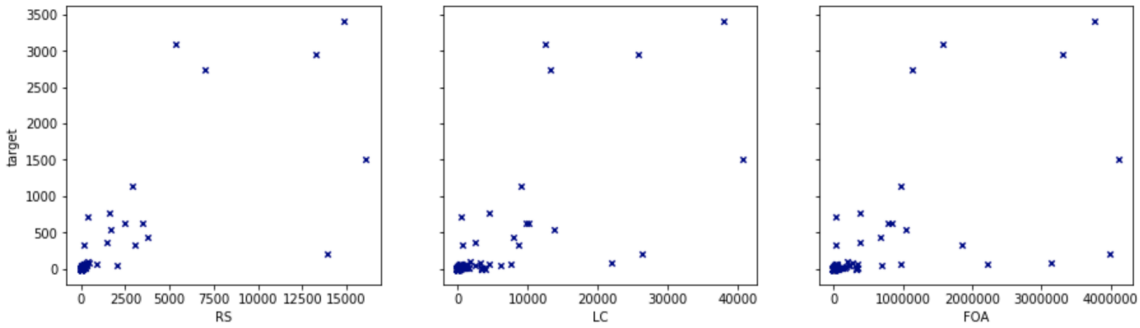


Figure 3: Analysis of target vs. top 3 features for #gopatriots

5.3 #nfl

We find the plots for this tag reasonable. The diffuse points observed are poorly modelled but the p-value is still low because the high variance points are sparse. Filtering these, we can see a linear relationship.

5.4 #patriots

We find the plots for this tag reasonable. The diffuse points observed are poorly modelled but the p-value is still low because the high variance points are sparse. Filtering these, we can see a linear relationship. Wechoose 3 of those features having the tied lowest p-score.

5.5 #sb49

We find the plots for this tag reasonable. The diffuse points observed are poorly modelled but the p-value is still low because the high variance points are sparse. Filtering these, we can see a linear relationship. We choose 3 of those features having the tied lowest p-score (using the ones with higher t-val).

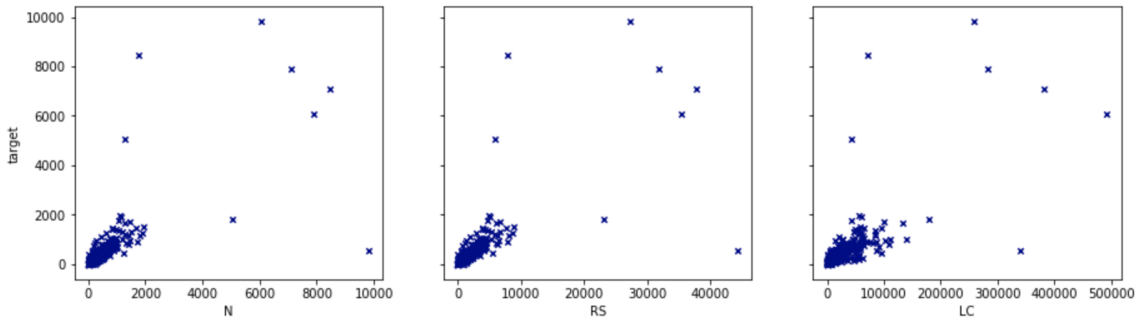


Figure 4: Analysis of target vs. top 3 features for #nfl

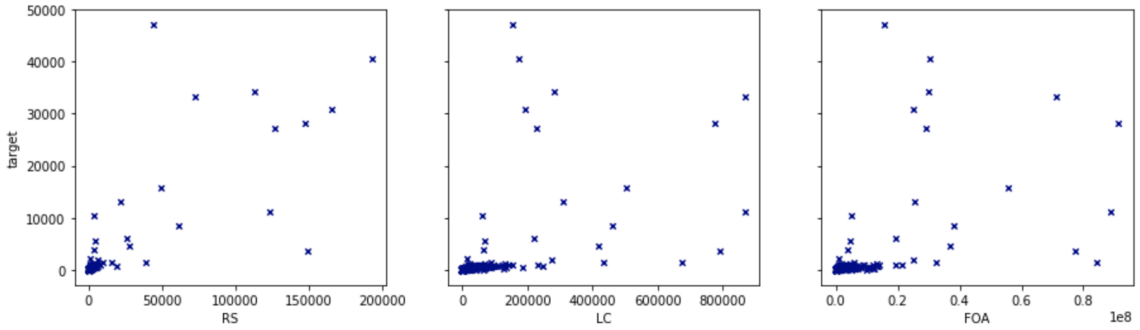


Figure 5: Analysis of target vs. top 3 features for #patriots

5.6 #superbowl

We find the plots for this tag reasonable. The diffuse points observed are poorly modelled but the p-value is still low because the high variance points are sparse. Filtering these, we can see a linear relationship. We choose 3 of those features having the tied lowest p-score (using the ones with higher t-val).

It appears that despite the high p-values being computed on a sparse twitter data set, **the main event of the superbowl is not well modelled**. The high tweet spikes around the day of the superbowl are not well captured by a single linear regressor and the error function does not optimize to the large errors because there are so few. **As a result we see the graphs in this section model well for the majority of points which have a low target value but fail for the sparse high target samples.**

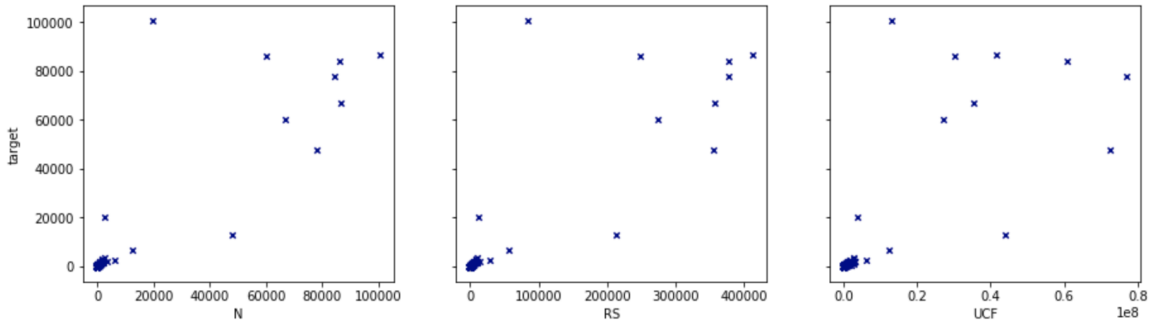


Figure 6: Analysis of target vs. top 3 features for #sb49

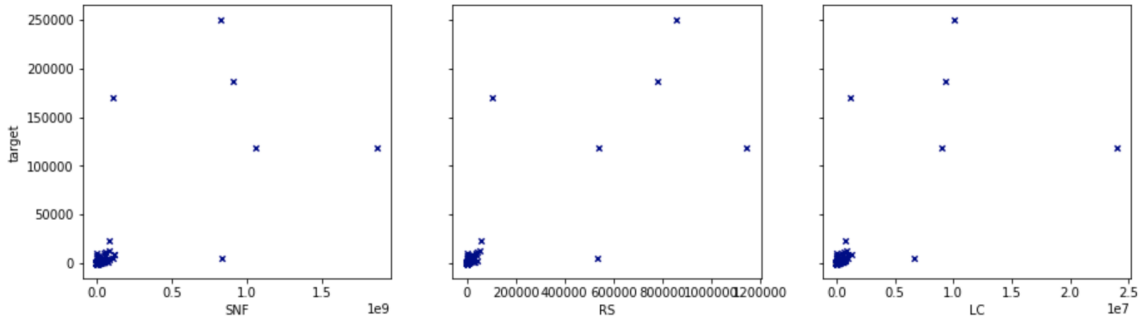


Figure 7: Analysis of target vs. top 3 features for #superbowl

6 Question 6

In this question we create different regression models for 3 different periods of time based on the Super Bowl's date and time. First, when the hashtags are still dormant, second, (during the game) when they are more active and third, after they pass their high-activity time. The timelines are listed below:

1. Before Feb. 1, 8:00 a.m.: 1-hour window
2. Between Feb. 1, 8:00 a.m. and 8:00 p.m.: 5-minute window
3. After Feb. 1, 8:00 p.m.: 1-hour window

For each hashtag, we train 3 regression models, one for each of these time periods. We compute the MSE and R-squared score for each interval. We also perform 5 fold cross validation for each window and present the train and test errors.

6.1 #gohawks

Metric	Window-1	Window-2	Window-3
MSE score	548396.80	75963.80	2086.16
R-squared score	0.47	0.46	0.81
Train MSE	514121.63	75357.88	1588.95
Test MSE	1016216.25	81500.29	11320.34

Table 28: Scores for #gohawks

6.2 #gopatriots

Metric	Window-1	Window-2	Window-3
MSE score	1820.96	14683.36	19.57
R-squared score	0.56	0.43	0.89
Train MSE	1749.46	14203.60	13.17
Test MSE	2809.54	22800.02	511.33

Table 29: Scores for #gopatriots

6.3 #nfl

Metric	Window-1	Window-2	Window-3
MSE score	65249.76	19847.22	17181.40
R-squared score	0.52	0.83	0.80
Train MSE	64496.99	18929.77	16863.88
Test MSE	74474.00	31931.34	20299.81

Table 30: Scores for #nfl

6.4 #patriots

Metric	Window-1	Window-2	Window-3
MSE score	344635.92	695440.51	9371.60
R-squared score	0.56	0.70	0.90
Train MSE	322635.03	686897.98	8881.89
Test MSE	582373.77	795906.30	23363.09

Table 31: Scores for #patriots

6.5 #sb49

Metric	Window-1	Window-2	Window-3
MSE score	7723.98	1306598.01	52993.31
R-squared score	0.87	0.86	0.86
Train MSE	7293.52	1284066.95	48047.53
Test MSE	12512.03	1551450.44	111100.36

Table 32: Scores for #sb49

6.6 #superbowl

Metric	Window-1	Window-2	Window-3
MSE score	513194.71	5069445.56	110019.39
R-squared score	0.44	0.92	0.85
Train MSE	498795.68	4923005.70	103902.41
Test MSE	722781.70	6915762.69	181996.22

Table 33: Scores for #superbowl

From the scores we observe that #gopatriots gave the least MSE score and #sb9 gave the best R squared values.

7 Question 7

Next, we aggregate all the data from each hashtag and train 3 different regression models. Aggregation is done by concatenation of all the tag files and the windowing now includes files from all the different tags.

Metric	Window-1	Window-2	Window-3
MSE score	4304812.64	15991650.78	408727.91
R-squared score	0.43	0.86	0.87
Train MSE	4150762.04	15679060.72	356374.77
Test MSE	6282385.48	24143718.91	1218286.85

Table 34: Scores for aggregated data

We observe that the MSE scores from aggregated data is not as good when compared to models trained on individual hashtag. For all 6 hashtags the MSE scores across all window periods is better than the MSE scores from aggregated data. This could

denote that each tag exhibits an individualized trend, one that cannot be congealed into one prediction model.

8 Question 8

For this section, we use a Random Forest Regressor and Gradient Boosting Regressor on aggregated data for the following set of parameters:

1. max_depth: [10, 20, 40, 60, 80, 100, 200, None]
2. max_features: ['auto', 'sqrt'],
3. min_samples_leaf: [1, 2, 4],
4. min_samples_split: [2, 5, 10],
5. n_estimators: [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]

The top 3 combinations for the Random Forest Regressor and Gradient Boosting Regressor are listed below (we performed 5 fold cross validation for MSE values):

max_depth	**max_features**	**min_samples_leaf**	**min_samples_split**	**n_estimators**	**Train MSE**	**Test MSE**
10	sqrt	1	2	200	37951483.7	210908391.3
10	auto	1	2	200	39827153.72	213825745.7
40	sqrt	1	2	400	39423689.04	219415215.1

Table 35: Scores for Random forest regressor

max_depth	**max_features**	**min_samples_leaf**	**min_samples_split**	**n_estimators**	**Train MSE**	**Test MSE**
None	sqrt	1	10	1400	9.88512E-08	221626894.13
200	sqrt	1	10	1200	9.86427E-08	222720506.4
10	sqrt	1	10	2000	9.96006E-08	228524300.5

Table 36: Scores for Gradient Boosting regressor

These scores do not look good due to the fact that the feature set has not been scaled and a non-linear model's performance is highly correlated to the scaling of the feature set. As some of our features have very large values, the MSE is astronomical (this is then resolved in the NN model, analyzed later).

9 Question 9

The best estimator found in the regression for section 8 is determined to be the Random Forest Regressor with a max depth of 10, a square root maximizer, min leaf

samples 1, split min samples 2 and the number of estimators 200. The test RMSE in shown in table 35. For comparison the gradient boosting regressor results are in table 36. This model essentially captures the regressors that describe the non-linear trends in tweeting habits. However as we have not scaled our data for this section, a simple OLS linear model would still perform better in this scenario and indeed it does. The values of a simple OLS model are given here:

1. MSE: 1.02×10^8
2. R-Squared: 0.871

The OLS model does not require scaling and hence performs better than the previous two models.

10 Question 10

Next, we perform the same grid search on Gradient Boosting regressor and Random forest regressor in windows described in Question 6. The best parameters for each window is as listed below:

10.1 Gradient Boosting regressor

max_depth	max_features	min_samples_leaf	min_samples_split	n_estimators	Train MSE	Test MSE
40	sqrt	4	10	1600	0.415047445	5713098.29
80	sqrt	4	5	800	6.731883505	5747838.47
10	sqrt	4	10	1800	0.414038165	5753185.66

Table 37: Scores for Gradient Boosting regressor for data from Feb. 1, 8:00 a.m.: 1-hour window.

max_depth	max_features	min_samples_leaf	min_samples_split	n_estimators	Train MSE	Test MSE
40	sqrt	1	2	1800	9.91134E-08	18504875.14
20	sqrt	1	2	2000	9.87848E-08	18711913.67
None	sqrt	1	5	1000	9.91374E-08	18820446.31

Table 38: Scores for Gradient Boosting regressor for data between Feb. 1, 8:00 a.m. and 8:00 p.m.: 5-minute window.

max_depth	max_features	min_samples_leaf	min_samples_split	n_estimators	Train MSE	Test MSE
100	sqrt	2	5	2000	9.99674E-08	362082.61
20	sqrt	2	10	2000	9.99429E-08	362715.36
100	sqrt	2	5	1600	9.96205E-08	363449.94

Table 39: Scores for Gradient Boosting regressor for data after Feb. 1, 8:00 p.m.: 1-hour window.

Comparing the MSE values obtained for Gradient Boosting regressor in question 10 with question 8. We observe that the test RMSE values are better when we split them into aforementioned windows. The best parameters obtained from question 8 don't agree with the ones obtained for any of the windows.

10.2 Random forest regressor

max_depth	max_features	min_samples_leaf	min_samples_split	n_estimators	Train MSE	Test MSE
20	sqrt	4	5	200	2637117.43	4125318.96
100	sqrt	4	10	200	2688558.48	4169119.68
60	sqrt	4	10	400	2687471.46	4176426.38

Table 40: Scores for Random forest regressor for data from Feb. 1, 8:00 a.m.: 1-hour window.

max_depth	max_features	min_samples_leaf	min_samples_split	n_estimators	Train MSE	Test MSE
10	sqrt	2	5	200	7138916.39	18160227.33
80	sqrt	2	5	200	7086817.87	18189702.63
100	sqrt	2	5	200	7133558.72	18285723.86

Table 41: Scores for Random forest regressor for between Feb. 1, 8:00 a.m. and 8:00 p.m.: 5-minute window.

max_depth	max_features	min_samples_leaf	min_samples_split	n_estimators	Train MSE	Test MSE
100	sqrt	2	5	200	118223.00	315782.33
200	sqrt	2	5	1000	119947.09	319174.73
None	sqrt	2	5	1200	119265.83	319865.29

Table 42: Scores for Random forest regressor for data after Feb. 1, 8:00 p.m.: 1-hour window.

Similarly, we also compared the MSE values for 3 different windows of Random forest regressor to aggregated on the whole dataset. We observed that dividing the whole dataset into three windows improved the performance. Also, the best parameters obtained in question 6 don't agree with the one's obtained in question 10.

11 Question 11

In this section, we use the MLPRegressor provided by sklearn to regress our aggregated data. To toggle the layer size and number of layers we considered the following options. This means for our grid search, each layer size for tested with all the possible number of layers listed below, yielding a total of 132 combinations.

1. Layer Size: $10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 500$

2. Number of Layers: $2, 4, 5, 6, 8, 10, 12, 14, 16, 18, 20$

See the table below for a summary of architectures we tried and their MSE results. We see that layer size 500 with 14 layers was the best architecture.

Layer Size	**Number of Layers**	**mean_test_score q11**
500	16	1.1E+11
500	18	1.4E+11
500	14	8.2E+10
500	20	2E+12
500	12	2.4E+11

Table 43: MSE versus number of layers and layer size, question 11.

12 Question 12

Now we take our best performing architecture from question 11, and re-implement the MLPRegressor. This time, we use StandardScaler, which makes our data feature set mean 0 and standard deviation 1, before feeding it into the MLPRegressor. We do see a performance increase by roughly 3 orders of magnitude. See the table below:

Layer Size	**Number of Layers**	**mean_test_score q12**
500	16	4.7E+08
500	18	5E+08
500	14	5.5E+08
500	20	5.5E+08
500	12	6.2E+08

Table 44: MSE versus number of layers and layer size, question 12.

Here, all MSE values are of the same magnitude, with layer size 500 and 16 layers being optimal. Thus scaling features for non-linear models generally proves to be helpful in regression models.

13 Question 13

We see that using the window lengths described in question 6, we were able to see a similar performance increase using the standard scalar. For all our windows, the best architecture has layer size 500 and 20 layers.

Layer Size	Number of Layers	mean_test_score q13_1
500	16	4954980
500	18	4945896
500	14	4962415
500	20	4907180
500	12	5167190

Table 45: MSE versus number of layers and layer size, question 13. Before Feb. 1, 8:00 a.m.: 1-hour window.

Layer Size	Number of Layers	mean_test_score q13_2
500	16	2.8E+08
500	18	2.8E+08
500	14	2.9E+08
500	20	2.8E+08
500	12	2.9E+08

Table 46: MSE versus number of layers and layer size, question 13. Between Feb. 1, 8:00 a.m. and 8:00 p.m.: 5-minute window.

Layer Size	Number of Layers	mean_test_score q13_3
500	16	4407618
500	18	3830289
500	14	4493959
500	20	3230713
500	12	4565239

Table 47: MSE versus number of layers and layer size, question 13. After Feb. 1, 8:00 p.m.: 1-hour window.

14 Question 14

The rigidity in a linear model is emphasized in the observed predictions below. Note that while for the second period (during the game), the predictions are closer, the values suffer in period 1 and period 3. Meanwhile the MSE values are lower for periods 1 and 3 as the target variable itself is quite low in comparison to period 2.

Actual values	Predictions
79.00	551.52
94.00	601.67
101.00	580.04
122.00	665.35
120.00	573.10

Table 48: Predictions for sample0 period1

Actual values	Predictions
3834.00	4304.73
2258.00	4509.67
1455.00	2962.63
1235.00	1942.59
1123.00	1956.05

Table 49: Predictions for sample0 period2

Actual values	Predictions
48.00	393.50
94.00	352.80
45.00	382.84
77.00	369.21
87.00	367.96

Table 50: Predictions for sample0 period3

Actual values	Predictions
180.00	613.24
202.00	566.80
294.00	614.48
555.00	726.85
846.00	882.50

Table 51: Predictions for sample1 period1

Actual values	Predictions
995.00	1692.14
870.00	1767.45
960.00	1582.29
861.00	2011.24
903.00	1595.18

Table 52: Predictions for sample1 period2

Actual values	Predictions
87.00	346.10
43.00	390.89
27.00	339.82
44.00	337.82
46.00	362.84

Table 53: Predictions for sample1 period3

Actual values	Predictions
141.00	748.46
102.00	600.28
144.00	525.51
104.00	560.27
61.00	616.00

Table 54: Predictions for sample2 period1

Actual values	Predictions
19.00	858.47
25.00	872.71
27.00	879.22
29.00	913.81
28.00	874.08

Table 55: Predictions for sample2 period2

Actual values	Predictions
90.00	390.78
40.00	405.03
58.00	376.59
87.00	346.10
43.00	390.89

Table 56: Predictions for sample2 period3

Name	Test MSE
sample0_period1.txt	242201.80
sample0_period2.txt	1751840.87
sample0_period3.txt	92962.60
sample1_period1.txt	90869.38
sample1_period2.txt	696164.56
sample1_period3.txt	94547.67
sample2_period1.txt	255808.28
sample2_period2.txt	729667.51
sample2_period3.txt	102675.57

Table 57: MSE values for test files

15 Question 15

In this section we attempt to use the textual content of the tweets to predict the location of the user. To generate our training data, we consider all tweets using #superbowl who's author resides in either Massachusetts or Washington. To label our data, we extract the location from `json_object['tweet']['user']['location']`. We consider a user to be in Massachusetts if their location contains the substring 'MA' or 'Massachusetts'. In addition, we consider a user to be in Washington if their location contains 'WA', or if their location contains 'Washington' and not 'DC' or 'D.C'.

Next we take all of the tweets from the selected users, and convert these tweets to the TF-IDF representation from Project 1. We begin by removing stopwords and performing lemmatization on the tweets. Then we use sklearn's `CountVectorizer` to get the word frequencies, and finally calculate the term frequency-inverse document frequencies. We use 3 different types of classifiers, namely a logistic classifier, a SVM, and a naive Bayes classifier.

15.1 Logistic Regression

Before performing classification on our data, we fist reduce the dimensionality with PCA. We swept the number of latent dimensions, r, from 1 to 250 in increments of 50, and trained the logistic classifier on the reduced-dimensional data. We use 10-fold cross validation to evaluate the models. The average validation accuracy is shown in the graph below. Based on this graph, we chose to use $r = 200$.

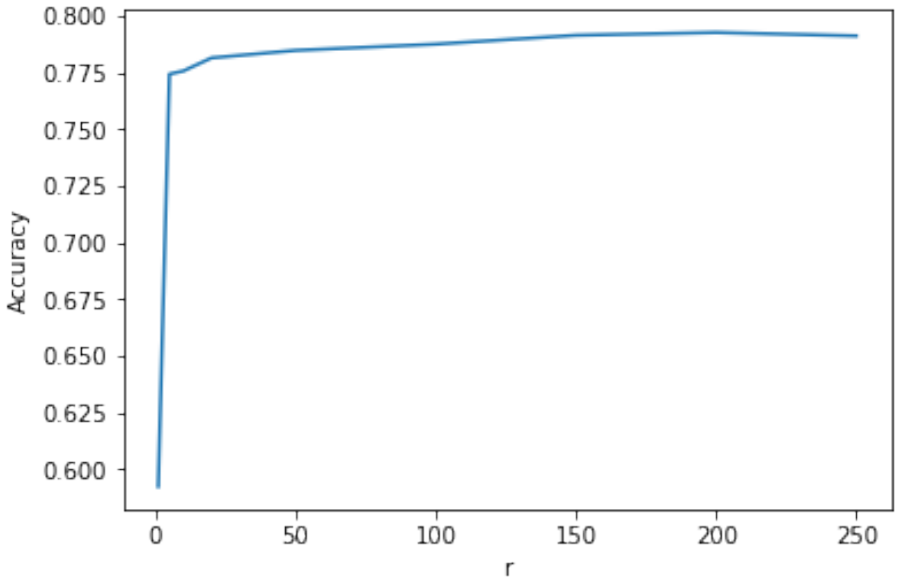


Figure 8: Average accuracy of the logistic classifier vs number of latent dimensions r

We also optimized our model over the regularization parameter C; it is swept from 10^{-3} to 10^{4}. The validation accuracy over C is shown below.

We found that the best values of C is 100. In the table at the end of this section, we present the average accuracy, precision, and recall obtained from 10-fold cross-validation for this best model. We also split the data into train and validation sets to calculate the confusion matrix and ROC curve, shown below.

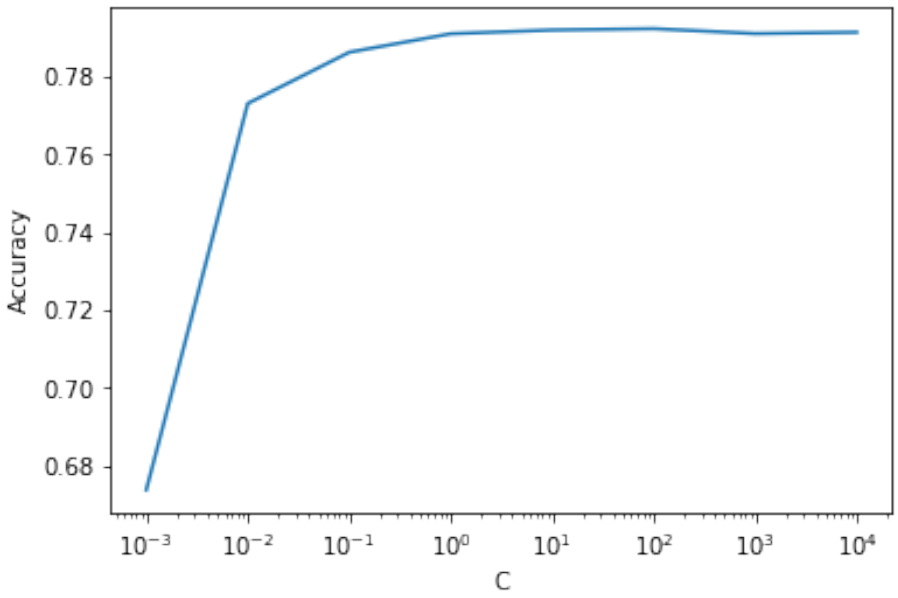


Figure 9: Average accuracy of the logistic classifier for various values of the regularization parameter C

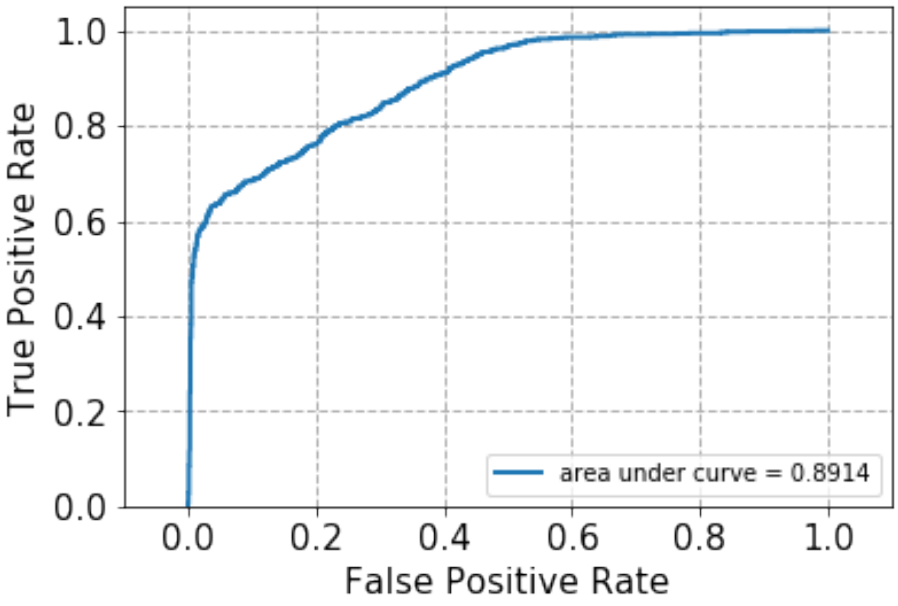
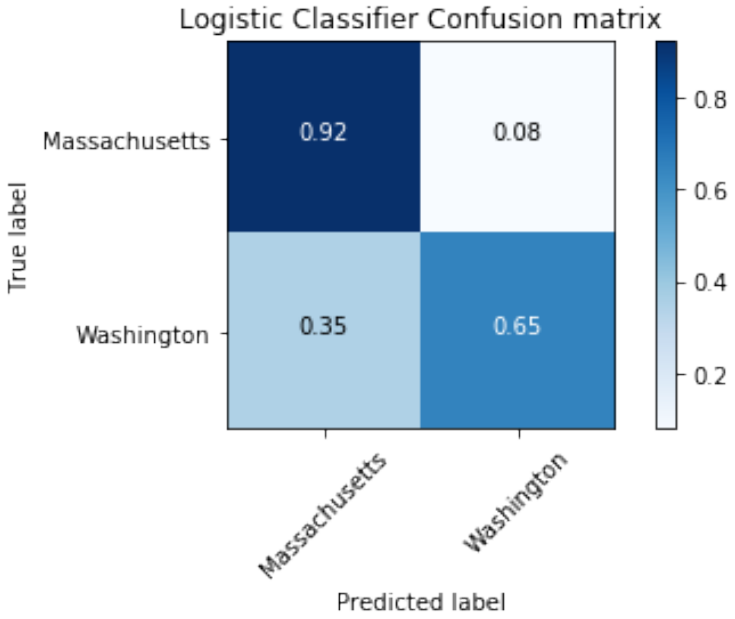


Figure 10: ROC curve for Logistic Classifier



15.2 SVM

In addition to training a logistic classifier, we also tried using a SVM. We again decided to use $r = 200$ for our latent dimension. We used a linear kernel, and used 10-fold cross-validation to optimize the regularization strength C over the range 10^{-3} to 10^3. The average validation accuracy is shown below for each values of C.

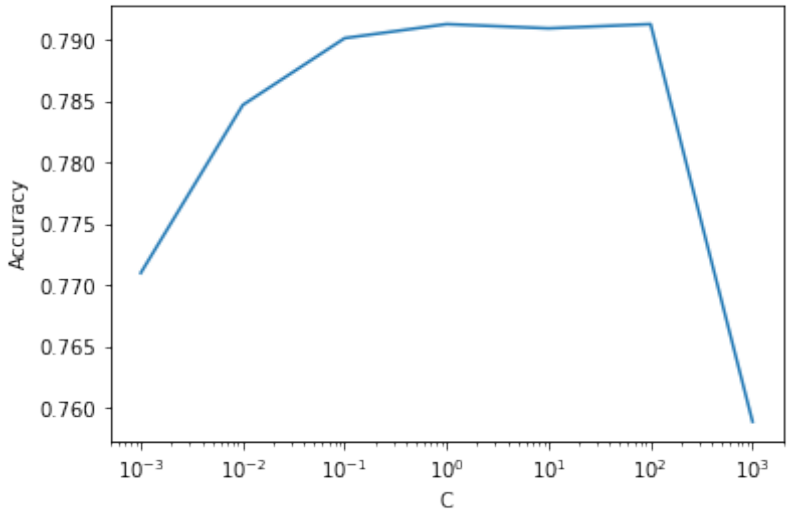


Figure 11: Average validation accuracy vs C for the SVM

The best model we found had $C = 1$. The ROC curve and confusion matrix for this model are shown below. The average recall, precision, and accuracy are shown in Table 58. We found that the SVM model performed very similarly to the logistic classifier.

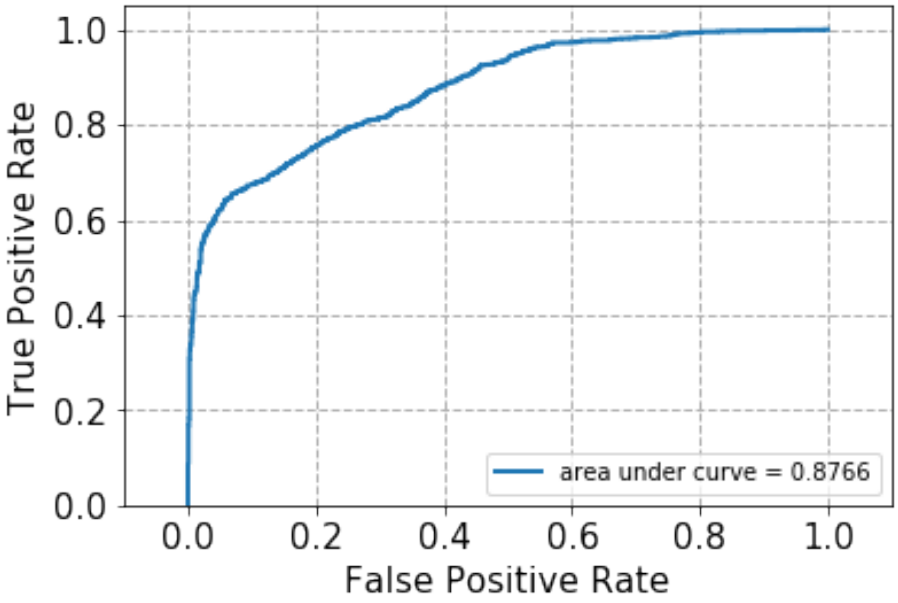
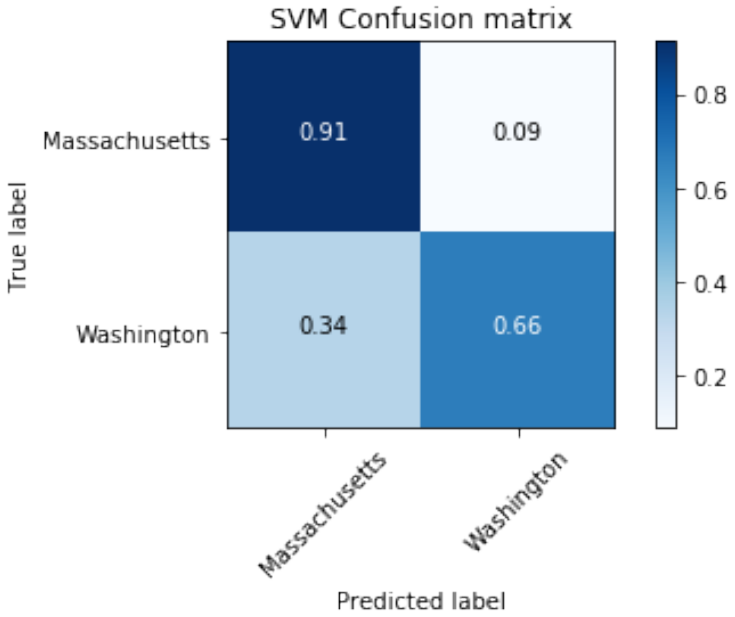


Figure 12: ROC curve for SVM



15.3 Naive Bayes

Finally, we use a Naive Bayes Classifier as our model. We used 10-fold cross-validation to calculate the average accuracy, precision, and recall, shown in the table below. We also calculate the confusion matrix and ROC curve.

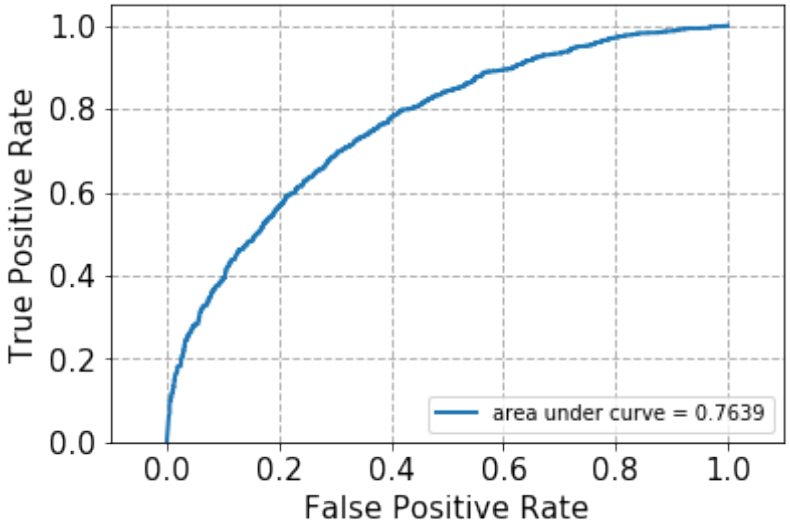
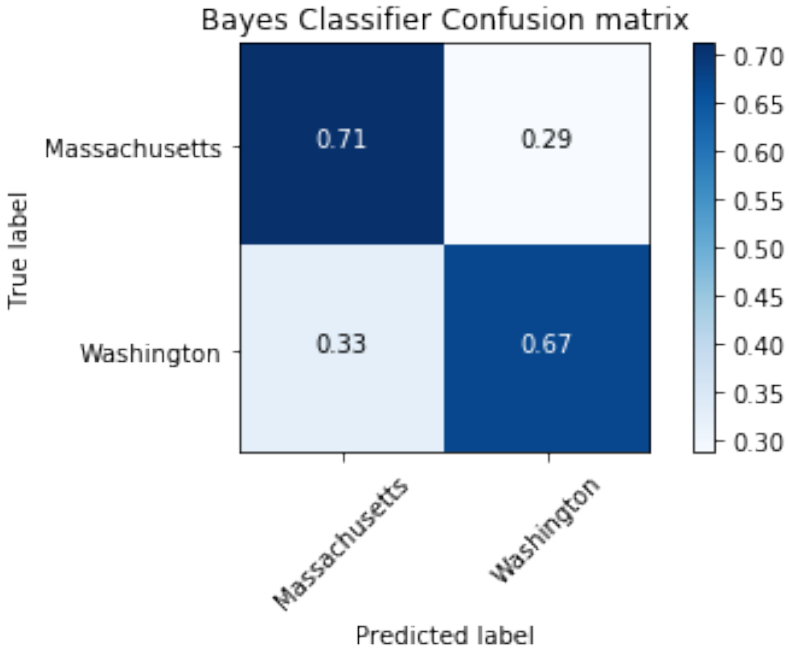


Figure 13: ROC curve for Bayes classifier



Overall we found that the logistic classifier and SVM have very similar results, while

the Bayes classifier performed quite poorly in comparison. The results are summarized in the table below.

	Accuracy	Precision	Recall
Logistic	0.792758	0.881613	0.654658
SVM	0.791651	0.890462	0.643611
Naive Bayes	0.683462	0.685574	0.662831

Table 58: Average accuracy, precision, and recall obtained from 10-fold cross-validation on 3 different models

Question 16

For the final part of our project we use the given data to infer whether or not a significant game event occurred during specific time periods. In addition, if a significant game event occurred, we wish to generate a new tweet characterizing the event, with the ultimate goal of creating a model that can live tweet a game.

We define significant events as touchdowns, interceptions, and field goals. Whenever such an event occurs, there is generally a spike in the number of tweets as users begin posting about the event. These spikes can be observed in the figure below. Several of the spikes appear to correspond to critical game moments, such as an interception that occurred at minute 22 and a touchdown at minute 42. Thus we decided to use the number of tweets as a feature to predict game events. In addition, we use another feature that is a binary value corresponding to whether or not the number of tweets at the current time period is a local maximum. This allows us to identify the peaks in the number of tweets.

In practice we find that there is often a delay of a few minutes between an event and a tweet being posted. For this reason, we consider intervals of 3 minutes and we wish to predict whether or not a major event occurred in the previous 3 minutes. The game can be broken down into 73 3-minute intervals. Of these intervals, 11 contained significant events, including 7 touchdowns, 1 fields goal, and 3 interceptions. We assign a time period a label of 1 if an event occurred during that time period, and a label of 0 otherwise.

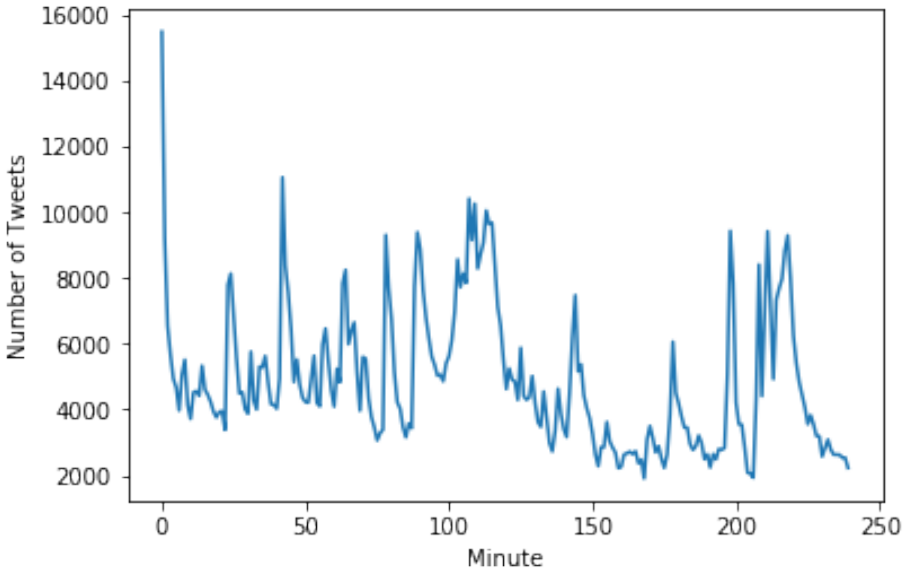


Figure 14: Total number of tweets during each 3-minute interval

While a spike in tweets can be a good indicator of game events, relying on the number of tweets alone can lead to many false positives. This is because other events, such as the halftime show and viral commercials can cause spikes as well. Looking at the graph above, there is a major spike that occurred around to 100 minute mark, during the halftime show. Therefore, we need to use other features to predict game events. One such feature that we can use is fan sentiment. We begin by identifying users that are fans of each team. We define Patriots fans as those who have used the hashtag #gopatriots at least once, and Seahawks fans as those who have used #gohawks. We found 427 users who used both hashtags, and we ignore these users. Next, we identified roughly 20 positive words such as 'good', 'yes', 'win', and 20 negative words such as 'bad', 'lose', 'angry', which are found in sentimental tweets. For each tweet we count the number of positive words and negative words in the tweet. A tweet can then be classified as positive, negative, or neutral depending on if it has more, less, or the same number of positive and negative words. During each time interval, we calculate the average sentiment for fans of both teams. This is plotted in the graph below. Significant game events show a large discrepancy in sentiment between teams, whereas during other events such the halftime show teams tend to express similar sentiments. Therefore, we also use the difference in sentiment between the two teams as a feature to predict events.

The model that we choose to predict events is a Random Forest Classifier. For each time period we use the number of tweets and fan sentiment to predict if an event occurred. To evaluate our model, we use leave-one-out cross-validation. The accuracy,

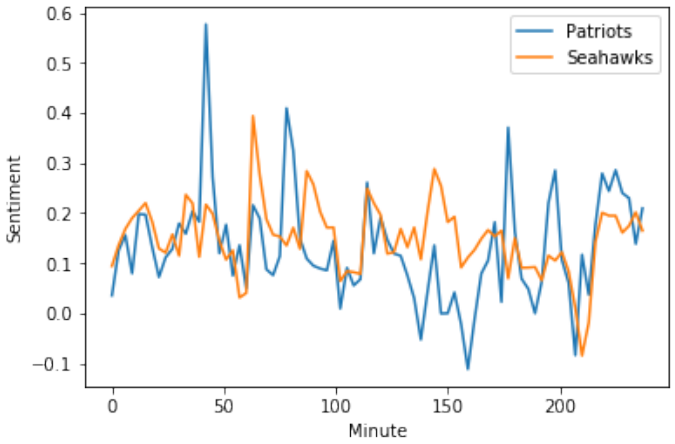


Figure 15: Average fan sentiment during each 3-minute interval

precision, and recall are shown in the table below. We also present the confusion matrix. Ultimately, we were able to correctly identify 9 out of the 11 game events. The two events that we failed to predict were both interceptions. One of the interceptions was overshadowed by a touchdown, which made it difficult to predict.

	Accuracy	Precision	Recall
Random Forest Classifier	0.93	0.75	0.82

Table 59: Performance of the random forest classifier in predicting events

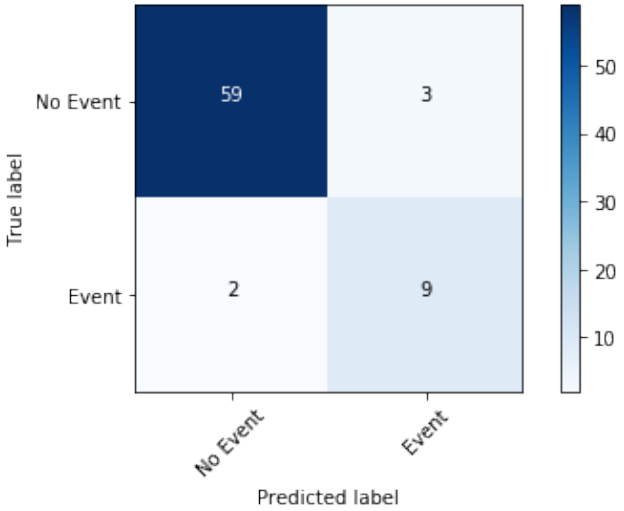


Figure 16: Confusion Matrix for event detection with the random forest classifier

Finally, we take our predicted events and generate tweets about them. To accomplish this, we use tweets from an event to train a recurrent neural network (RNN). The RNN that we use for this task is a basic model designed for text generation in the style of Shakespeare [https://www.tensorflow.org/tutorials/sequences/text_generation]. It consists of a single gated recurrent unit (GRU) layer with a sigmoid activation followed by a dense layer.

At first, we tried training the RNN on all of the tweets from an event time period. We then use the RNN to generate several new tweets. Hopefully, as the majority of users are tweeting about the event, the RNN will learn to generate tweets relevant to the event. Ultimately, we found that we were fairly successful in generating tweets that accurately characterize an event. Several example tweets that we generated are shown in the figures below. Some of these tweets are able to correctly identify whether a touchdown or interception has occurred. Moreover, they can sometimes recognize which players were involved in significant plays, and even report the current score. For example, in one tweet below, the tweet acknowledged a touchdown pass from Tom Brady to Brandon LaFell. In another, the score was correctly reported to be 14-14.

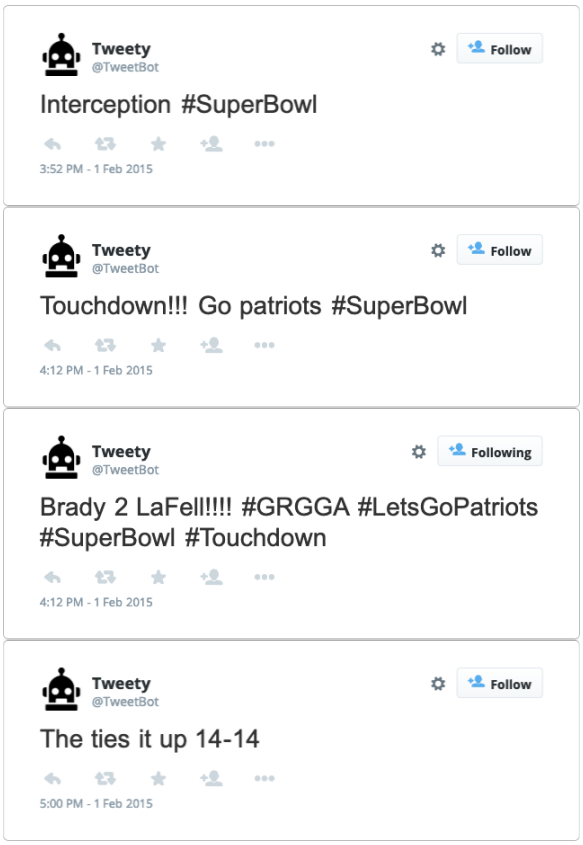


Figure 17: Tweets generated from our model during predicted events

However, our method of generating tweets is not perfect. Several tweets were generated which make no sense or are completely unrelated to the game. Some of these tweets are shown below.

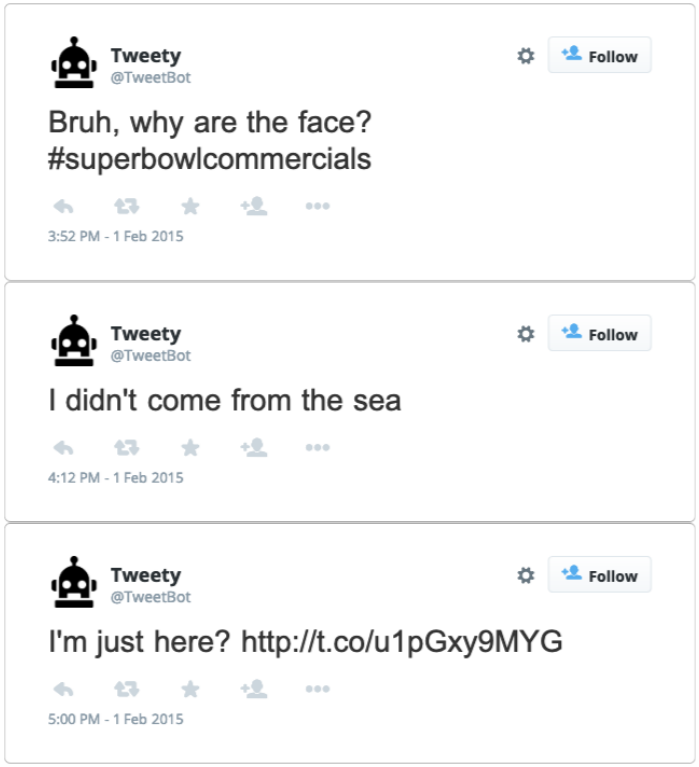


Figure 18: Questionable tweets generated from our model

To generate tweets that are more relevant to the current event, we can be selective about the tweets that we feed the RNN. Fans of a team are more likely to tweet accurate information about an event when the event benefits their team. Therefore, we look at the sentiment score for each team, and use that to predict which team scored the touchdown or made the interception. Then we train the RNN on the tweets only from fans of the team that has a positive reaction to the event.

When using the sentiment score to predict which team scored or made an interception, we were able to correctly characterize 89% of our predicted events. The only event that we were not able to characterize was a field goal made by the Seahawks in the 2nd half. This is likely due to the large amount of negative sentiments expressed by Seahawks fans at being forced to kick a field goal instead of scoring a touchdown. This particular issue makes characterizing field goals difficult.

We found that training the RNN on tweets only from fans of the team that benefited

from the event generally increases the relevance of the generated tweet to the event. In the table below, selected tweets our model generated are provided along with a description of the original event.

Actual Event	Generated Tweet
Seahawk Jeremy Lane makes an interception, is injured	Interception!!! #GoHawks YES LANE YES!!! I hope he is okay! #GoHawks #SB49
Patriots score the first touchdown	TOUCHDOWN!!!!!! #GoPatriots Touchdown! #Patriots strike first! #SB49 #NFLCanada #GoPatriots
Marshawn Lynch scores touchdown for Seahawks	TOUCHDOWN SEAHAWKS! Lynch !! #SuperBowl #superbowlw9 #GoHawks #SB49
Rob Gronkowski scores the 2nd touchdown for the Patriots.	Touchdown 2 #patriots You got Gronk'd #SuperBowlXLIX Go Patriots!!! #SuperBowlXLIX #GoPats
Seahawks score a touchdown with 6 seconds left in the half	TOUCHDOWN HAWKS! #GoHawks #SuperBowlXLIX Touchdown with 6 Seconds left in the half #SB49
Seahawks score a field goal.	Back to the game #GoPatriots #GoPatriots
Seahawks score a touchdown, bringing the score to 14-24.	@Seahawks amazing touchdown!!!!!!! #GoHawks #BeastMode #SEAvsNE #SuperBowl #SB49 10 point lead #GoHawks
Patriots score a touchdown, 21-24.	TOUCHDOWN PATRIOTS! #EPPNDOLA! We're back in this! #SB49 SanerBowl — NE #Patriots 21-24 SEA #SeahawksStill #SuperBowlXLIX
Patriots score touchdown with 2 min left, bringing the final score to 28-24.	TOOOOUCHDOWWWWNNNN!!!!!! #GoPatriots #SuperBowl 2 minutes leftave a big #SuperBowl for Brady. 28 24 for Patriots

Table 60: Select tweets generated for predicted events

16 Conclusion

Tweet data during the Superbowl is a great resource to predict many target variables, including the number of tweets in the following hour (the majority of focus in this project) as explored in questions 1 through 15. Apart from the number of tweets, this data is compatible with many regression estimates including popularity of a tweet, and team support. In fact the textual content of a tweet can even be used to suggest new tweets, as a mixture of models that predict events AND standard tweet architecture. This is explored in question 16. Data such as this promises to be valuable in understanding fan behaviour and a myriad of other tangible and intangible descriptions of how people view sports and how success of a team is perceived.