Project 5: Application - Twitter data

Lumi Huang, Christian Warloe, Ke Zhao, Landi Luo

Introduction

A useful practice in social network analysis is to predict future popularity of a subject or event. Twitter, with its public discussion model, is a good platform to perform such analysis. With Twitter's topic structure in mind, the problem can be stated as: knowing current (and previous) tweet activity for a hashtag, can we predict its tweet activity in the future? More specifically, can we predict if it will become more popular and if so by how much? In this project, we will try to formulate and solve an instance of such problems.

The available Twitter data is collected by querying popular hashtags related to the 2015 Super Bowl spanning a period starting from 2 weeks before the game to a week after the game. We use data from some of the related hashtags to train a regression model and then use the model to make predictions for other hashtags. To train the model, we prepare training sets out of the data, extract features for them, and then fit a regression model on it. The regression model will try to fit a curve through observed values of features and outcomes to create a predictor for new samples. Designing and choosing good features is one of the most important steps in this process and is essential to getting a more accurate system. We use the training data to create the model and use the test data to make predictions. The test data consists of tweets containing a hashtag in a specified time window, and we use our model to predict number of tweets containing the hashtag posted within one hour immediately following the given time window.

Part 1: Popularity Prediction

---

1. A first look at the data

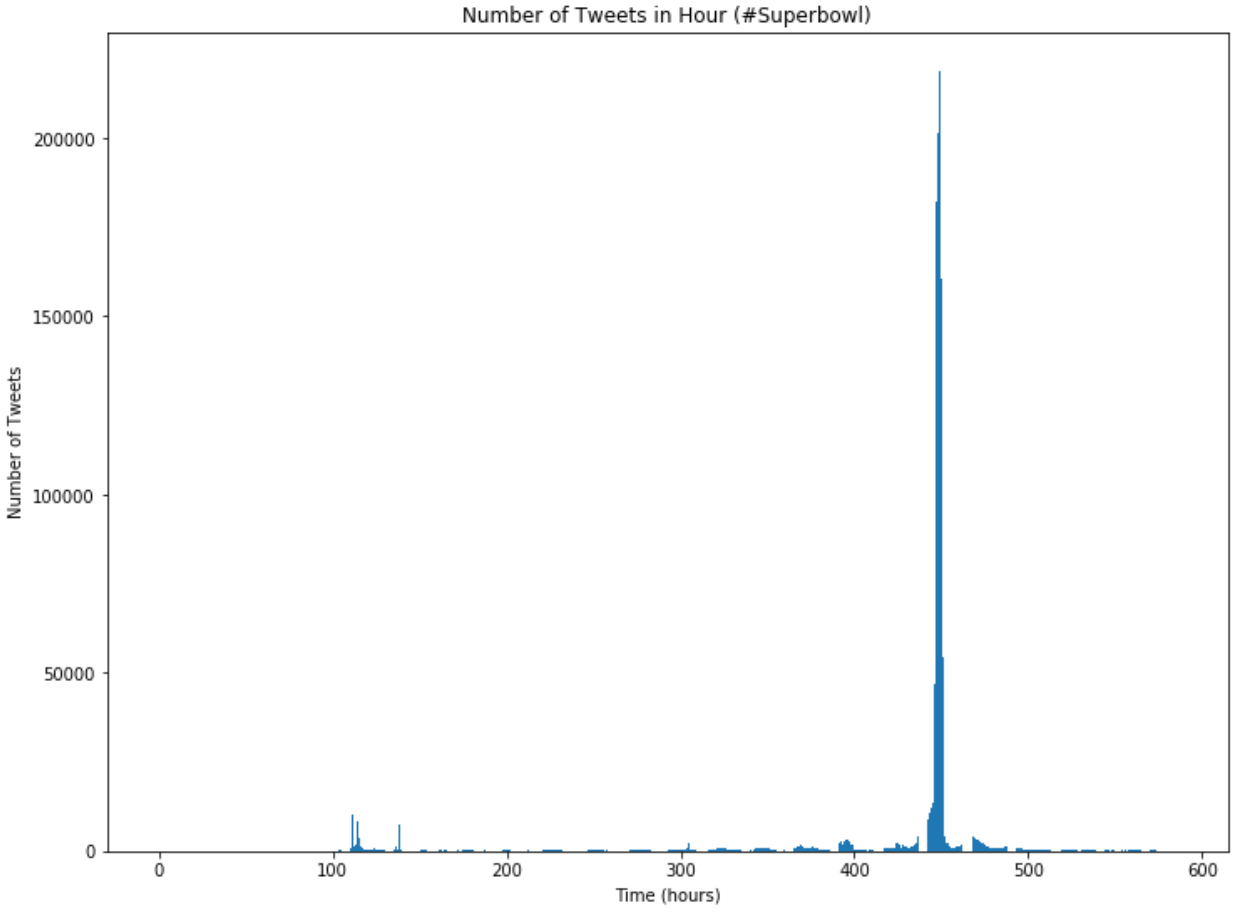
The training tweet data consist of 6 text files, each one filled with tweet data from one hashtag as indicated in the filenames.

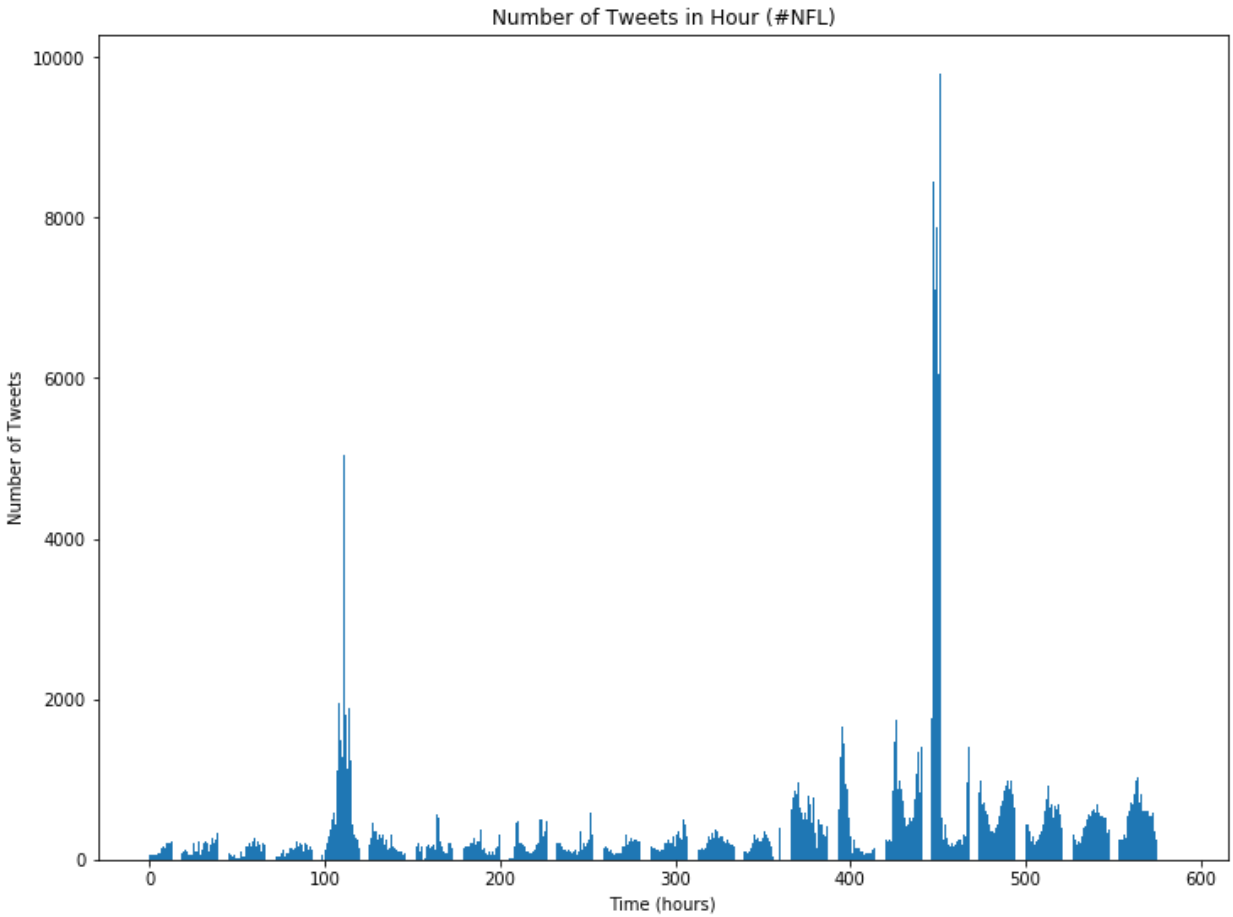
QUESTION 1: Summary Statistics

Hashtag	Average number of tweets per hour	Average number of followers of users posting the tweets per tweet	Average number of retweets per tweet

superbowl	2072.12	8814.97	2.39
gohawks	292.49	2217.92	2.01
gopatriots	40.95	1427.25	1.41
nfl	397.02	4662.38	1.53
patriots	750.89	3280.46	1.79
sb49	1276.86	10374.16	2.53

QUESTION 2: Plot "number of tweets in hour" over time for #SuperBowl and #NFL (a histogram with 1-hour bins).





The number of tweets in hour over time for both #SuperBowl and #NFL had the first peak at around 110 hours and and reached the second peak sharply at around 450 hours. These two time points may be the time that something special about SuperBowl or NFL happened.

2. Linear regression

We created time windows from the data to extract features. We used 1-hour time window (00:00 - 01:00 am, 01:00 - 02:00 am, etc.) and calculate the features in each time window, resulting in <# of hours> data points.

For each hashtag data file, we fit a linear regression model using the following 5 features to predict number of tweets in the next hour, with features extracted from tweet data in the previous hour:

- Number of tweets
- Total number of retweets
- Sum of the number of followers of the users posting the hashtag
- Maximum number of followers of the users posting the hashtag
- Time of the day (which could take 24 values that represent hours of the day with respect to a given time zone)

QUESTION 3: Assess the fit of each linear regression model using the the OLS

Hashtag	MSE	R-squared
superbowl	24579710705.824547	0.800
gohawks	79669859.83590713	0.476
gopatriots	5320955.630294414	0.627
nfl	41976324.96352415	0.570
patriots	1224056732.6249557	0.668
sb49	7742522928.1265335	0.804

#superbowl

	coef	std err	t	P>|t|	[0.025	0.975]
const	-149.5572	605.382	-0.247	0.805	-1338.565	1039.451
x1	-20.4965	43.624	-0.470	0.639	-106.177	65.184
x2	2.2766	0.080	28.537	0.000	2.120	2.433
x3	-0.2543	0.046	-5.544	0.000	-0.344	-0.164
x4	-0.0001	2.2e-05	-6.265	0.000	-0.000	-9.47e-05
x5	0.0007	0.000	4.889	0.000	0.000	0.001

The p-values for all features except for x1 (time of the day) were less than 0.05, which meant that all features except “time of the day” were significant for #superbowl.

#gohawks

	coef	std err	t	P>|t|	[0.025	0.975]
const	99.5479	72.676	1.370	0.171	-43.202	242.298
x1	1.3860	5.451	0.254	0.799	-9.321	12.093
x2	1.2823	0.165	7.767	0.000	0.958	1.607
x3	-0.1364	0.044	-3.113	0.002	-0.222	-0.050
x4	-0.0002	8.06e-05	-2.407	0.016	-0.000	-3.57e-05
x5	6.044e-05	0.000	0.402	0.687	-0.000	0.000

The p-values for x2, x3 and x4 were less than 0.05, which meant that the features “number of tweets”, “total number of retweets” and “sum of the number of followers of the users posting the hashtag” were significant for #gohawks.

#gopatriots

	coef	std err	t	P>|t|	[0.025	0.975]
const	13.2200	18.907	0.699	0.485	-23.941	50.381
x1	-0.3349	1.413	-0.237	0.813	-3.112	2.442
x2	0.3055	0.326	0.938	0.349	-0.334	0.945
x3	0.4868	0.219	2.219	0.027	0.056	0.918
x4	-0.0001	0.000	-0.421	0.674	-0.001	0.000
x5	-2.911e-05	0.000	-0.116	0.908	-0.001	0.000

Only the p-value for x3 was less than 0.05, which meant that only the feature “total number of retweets” was significant for #gopatriots.

#nfl

	coef	std err	t	P>|t|	[0.025	0.975]
const	126.4288	43.356	2.916	0.004	41.273	211.585
x1	0.2952	3.176	0.093	0.926	-5.942	6.533
x2	0.5651	0.135	4.173	0.000	0.299	0.831
x3	-0.1650	0.064	-2.578	0.010	-0.291	-0.039
x4	0.0001	2.51e-05	4.573	0.000	6.54e-05	0.000
x5	-0.0001	3.32e-05	-3.527	0.000	-0.000	-5.2e-05

The p-values for all features except for x1 were less than 0.05, which meant that all features except “time of the day” were significant for #nfl.

#patriots

	coef	std err	t	P>|t|	[0.025	0.975]
const	180.1751	183.925	0.980	0.328	-181.066	541.416
x1	-5.8597	13.765	-0.426	0.670	-32.896	21.176
x2	0.9145	0.071	12.937	0.000	0.776	1.053
x3	-0.0681	0.058	-1.178	0.239	-0.181	0.045
x4	-1.098e-05	2.63e-05	-0.417	0.677	-6.27e-05	4.07e-05
x5	0.0001	9.17e-05	1.340	0.181	-5.72e-05	0.000

Only the p-value for x2 was less than 0.05, which meant that only the feature “number of tweets” was significant for #patriots.

#sb49

	coef	std err	t	P>|t|	[0.025	0.975]
const	235.2366	365.931	0.643	0.521	-483.616	954.089
x1	-17.9341	26.914	-0.666	0.505	-70.805	34.937
x2	1.1361	0.091	12.485	0.000	0.957	1.315
x3	-0.1602	0.082	-1.953	0.051	-0.321	0.001
x4	9.695e-06	1.3e-05	0.743	0.458	-1.59e-05	3.53e-05
x5	9.417e-05	4.58e-05	2.055	0.040	4.16e-06	0.000

The p-values for x2 and x5 were less than 0.05, which meant that the features “number of tweets” and “maximum number of followers of the users posting the hashtag” were significant for #sb49.

3. Feature Analysis

QUESTION 4: Design a regression model using any features from the papers you find or other new features you may find useful for this problem. Fit your model on the data of each hashtag and report fitting MSE and significance of features.

From the linear regression results in the previous part, we observed that the significant features varied for each hashtag. We decided to remove “sum of the number of followers of the users posting the hashtag” and “time of day” features. We decided to fit regression models on each hashtag dataset using the following 8 features:

- x1 = number of tweets
- x2 = total number of retweets
- x3 = maximum number of followers of the users posting the hashtag
- x4 = total number of public lists
- x5 = maximum number of public lists
- x6 = maximum number of tweets (including retweets) issued by the users to date
- x7 = total hashtag count
- x8 = maximum ranking score

We report the fitted MSEs of our models for each hashtag and compared the results of the new models with the old models. From the table below, we observe that our new models resulted in smaller MSEs for all 6 hashtags, indicating that the new features we chose resulted in better fitted models compared to the old features.

Hashtag	**MSE for New Model**	**MSE for Old Model**

	(8 features)	(5 features)
#superbowl	16209488451.1	24579710705.8
#gohawks	60661562.1	79669859.8
#gopatriots	3957448.8	5320955.6
#nfl	30760615.2	41976324.9
#patriots	805999972.1	1224056732.6
#sb49	4887262041.2	7742522928.1

The results of the OLS regressions along with the p-values for each of the features of each hashtag are shown in the tables below:

#superbowl

	coef	std err	t	P>|t|	[0.025	0.975]
const	3802.7864	2628.994	1.446	0.149	-1360.779	8966.352
x1	-3.0734	0.469	-6.558	0.000	-3.994	-2.153
x2	-1.2712	0.076	-16.750	0.000	-1.420	-1.122
x3	-0.0004	0.000	-2.674	0.008	-0.001	-0.000
x4	0.0290	0.003	11.064	0.000	0.024	0.034
x5	-0.0640	0.019	-3.303	0.001	-0.102	-0.026
x6	-0.0032	0.001	-2.915	0.004	-0.005	-0.001
x7	2.6568	0.243	10.954	0.000	2.180	3.133
x8	-350.5528	342.571	-1.023	0.307	-1023.391	322.286

For #superbowl, all of the features were significant (p-value < 0.05) except for $x8$ (maximum ranking score).

#gohawks

	coef	std err	t	P>|t|	[0.025	0.975]
const	-354.2746	191.951	-1.846	0.065	-731.307	22.758
x1	-1.9538	0.366	-5.342	0.000	-2.672	-1.235
x2	-0.0988	0.041	-2.384	0.017	-0.180	-0.017
x3	-0.0003	0.000	-2.986	0.003	-0.001	-0.000
x4	0.0421	0.007	6.119	0.000	0.029	0.056
x5	-0.0590	0.018	-3.325	0.001	-0.094	-0.024
x6	-0.0007	0.000	-1.546	0.123	-0.002	0.000
x7	0.8637	0.204	4.233	0.000	0.463	1.264
x8	69.9009	29.347	2.382	0.018	12.258	127.544

For #gohawks, all of the features were significant (p-value < 0.05) except for `x6` (maximum number of tweets and retweets issued by the users to date)

#gopatriots

	coef	std err	t	P>|t|	[0.025	0.975]
const	-25.5221	31.203	-0.818	0.414	-86.851	35.807
x1	-6.7995	0.670	-10.153	0.000	-8.116	-5.483
x2	-1.5331	0.269	-5.707	0.000	-2.061	-1.005
x3	-3.248e-05	9.57e-05	-0.339	0.734	-0.000	0.000
x4	0.1818	0.026	7.064	0.000	0.131	0.232
x5	-0.1700	0.028	-5.966	0.000	-0.226	-0.114
x6	0.0003	0.000	1.498	0.135	-8.58e-05	0.001
x7	3.5536	0.311	11.437	0.000	2.943	4.164
x8	4.7869	5.963	0.803	0.423	-6.933	16.506

For #gopatriots, the features `x3`, `x6`, and `x8` were not significant.

#nfl

	coef	std err	t	P>|t|	[0.025	0.975]
const	-252.2840	249.122	-1.013	0.312	-741.587	237.019
x1	-2.3974	0.275	-8.706	0.000	-2.938	-1.856
x2	-0.3794	0.061	-6.203	0.000	-0.500	-0.259
x3	-1.914e-05	2.32e-05	-0.824	0.410	-6.48e-05	2.65e-05
x4	0.0173	0.002	8.437	0.000	0.013	0.021
x5	-0.0175	0.005	-3.270	0.001	-0.028	-0.007
x6	0.0002	7.67e-05	3.075	0.002	8.52e-05	0.000
x7	0.9659	0.081	11.881	0.000	0.806	1.126
x8	29.9814	33.401	0.898	0.370	-35.623	95.586

For #nfl, `x3` and `x8` were not significant among the 8 features.

#patriots

	coef	std err	t	P>|t|	[0.025	0.975]
const	260.6506	1047.800	0.249	0.804	-1797.317	2318.619
x1	3.3722	0.351	9.613	0.000	2.683	4.061
x2	-0.2981	0.076	-3.926	0.000	-0.447	-0.149
x3	-0.0001	0.000	-1.457	0.146	-0.000	5.14e-05
x4	0.0152	0.003	4.389	0.000	0.008	0.022
x5	0.0261	0.017	1.504	0.133	-0.008	0.060
x6	0.0002	0.000	0.450	0.653	-0.001	0.001
x7	-1.0409	0.150	-6.943	0.000	-1.335	-0.746
x8	-28.4826	138.658	-0.205	0.837	-300.818	243.853

For #patriots, `x3`, `x5`, `x6`, and `x8` were not significant among the 8 features.

#sb49

	coef	std err	t	P>|t|	[0.025	0.975]
const	920.7440	886.570	1.039	0.299	-820.902	2662.390
x1	2.3616	0.479	4.926	0.000	1.420	3.303
x2	0.0839	0.101	0.826	0.409	-0.115	0.283
x3	-4.38e-05	7.05e-05	-0.621	0.535	-0.000	9.48e-05
x4	-0.0032	0.002	-2.079	0.038	-0.006	-0.000
x5	0.0490	0.012	4.080	0.000	0.025	0.073
x6	0.0001	0.001	0.086	0.931	-0.003	0.003
x7	-0.7336	0.232	-3.162	0.002	-1.189	-0.278
x8	-147.6446	127.958	-1.154	0.249	-399.016	103.727

For #sb491, `x2`, `x3`, `x6`, and `x8` were not significant among the 8 features.

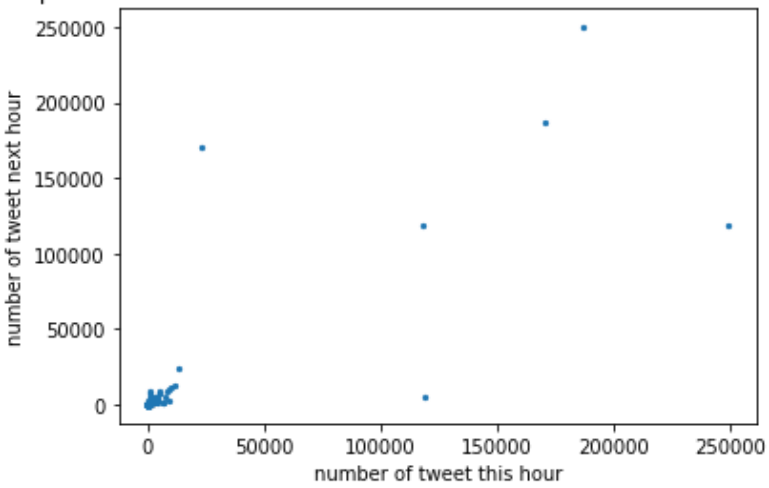
From the results above, we observed that `x8` (maximum ranking score) was not a significant feature across all 6 hashtags. The features `x1`, `x4`, and `x7` remained significant across all 6 hashtags. Thus, we conclude that "number of tweets", "total number of public lists", and "total hashtag count" were significant predictors of the number of tweets in the next hour.

QUESTION 5: For each of the top 3 features (i.e. with the smallest p-values) in your measurements, draw a scatter plot of predictant (number of tweets for next hour) versus value of that feature, using all the samples you have extracted, and analyze it.
Do the regression coefficients agree with the trends in the plots? If not, why?

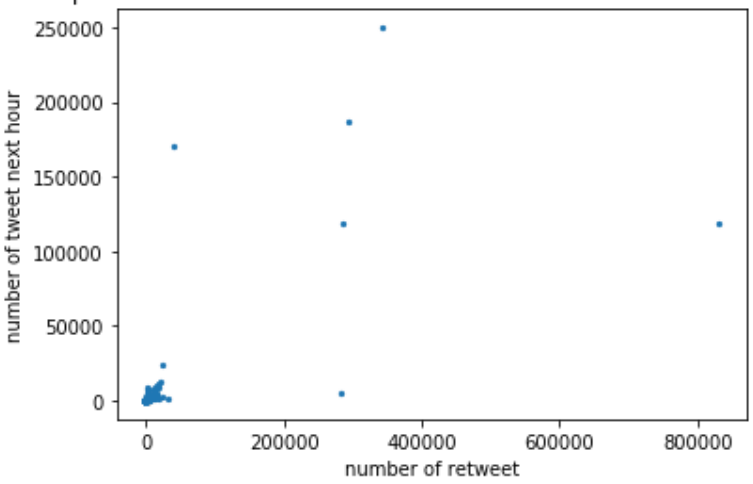
In general, the scatterplots of predictant vs. each of the top 3 features of each hashtag follow linear trends. Most of the data points are concentrated near the lower range (closer to 0) and they agree with the corresponding regression coefficients.

Scatterplots of top 3 features from #superbowl

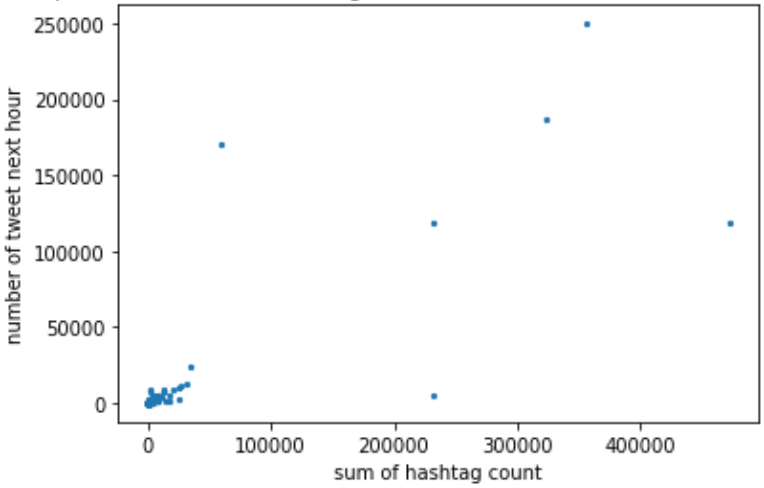
Relationship between current number of tweet and number of tweet next hour (Superbowl)



Relationship between number of retweet and number of tweet next hour (Superbowl)

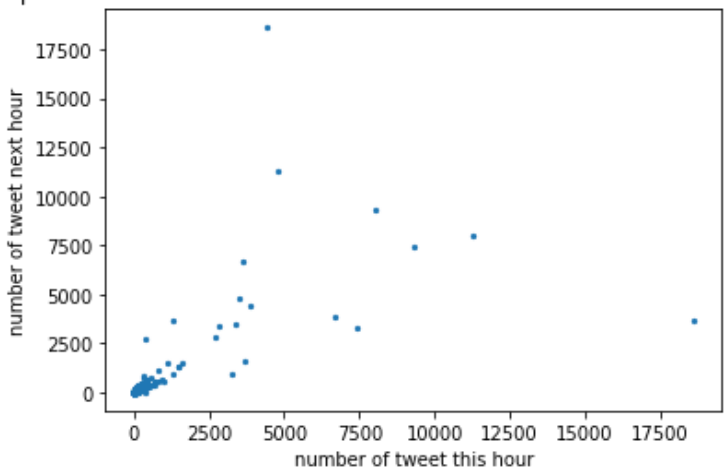


Relationship between sum of hashtag count and number of tweet next hour (Superbowl)

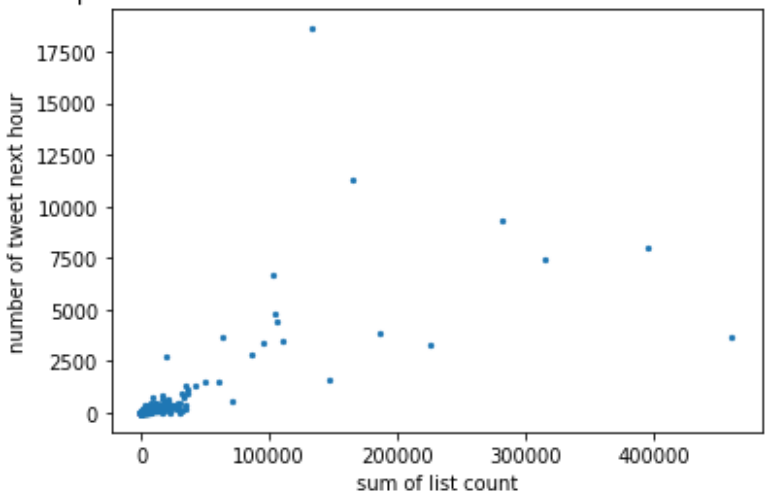


Scatterplots of top 3 features from #gohawks

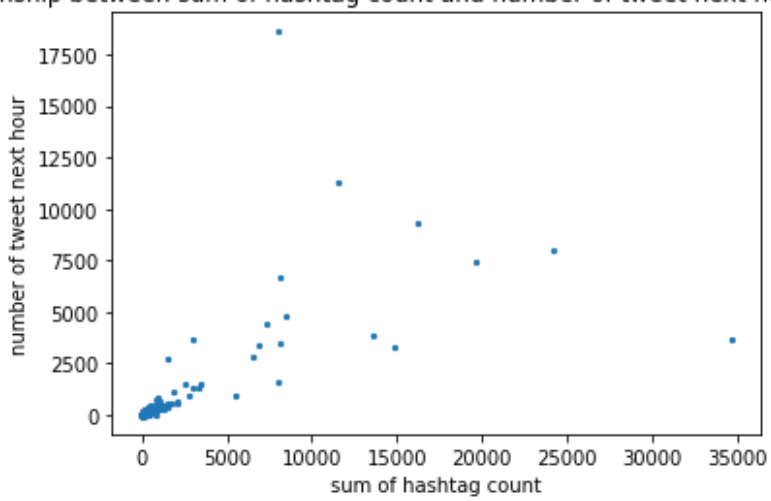
Relationship between number of tweet this hour and number of tweet next hour (#gohawks)



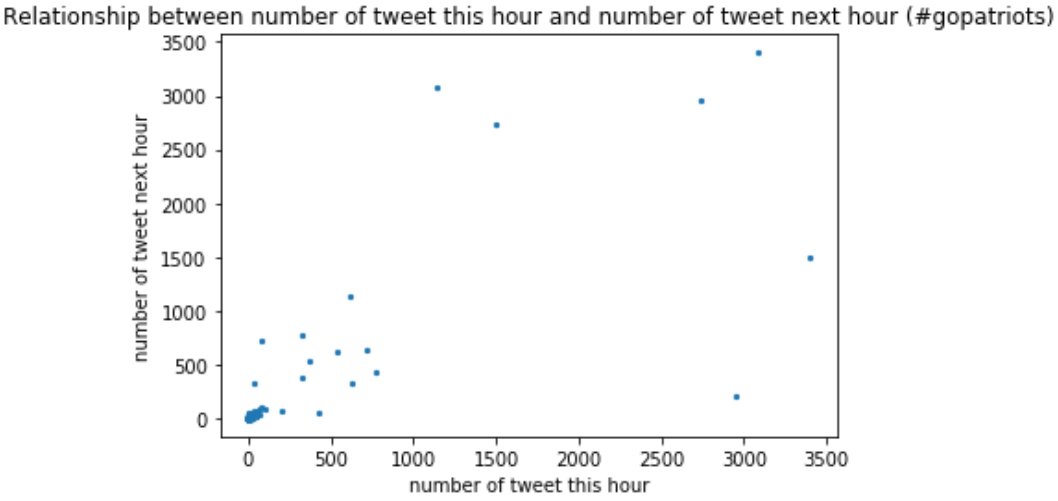
Relationship between sum of list count and number of tweet next hour (#gohawks)

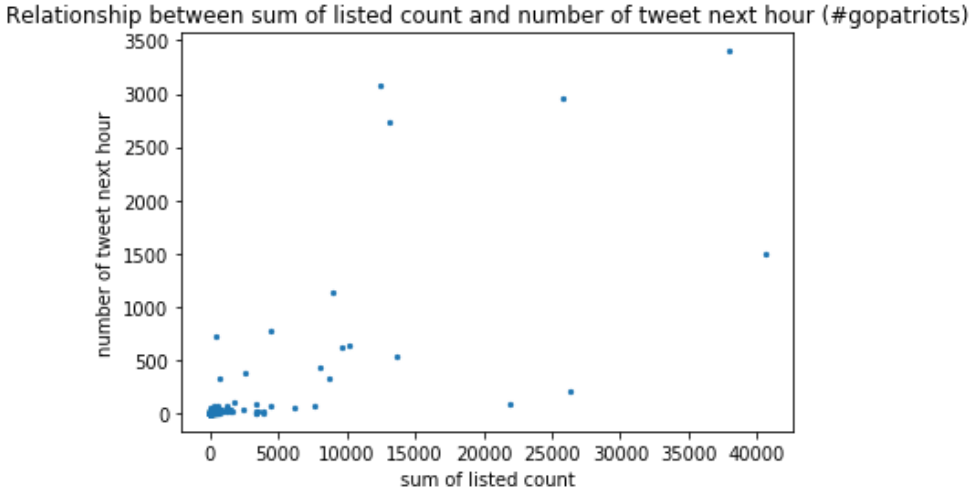


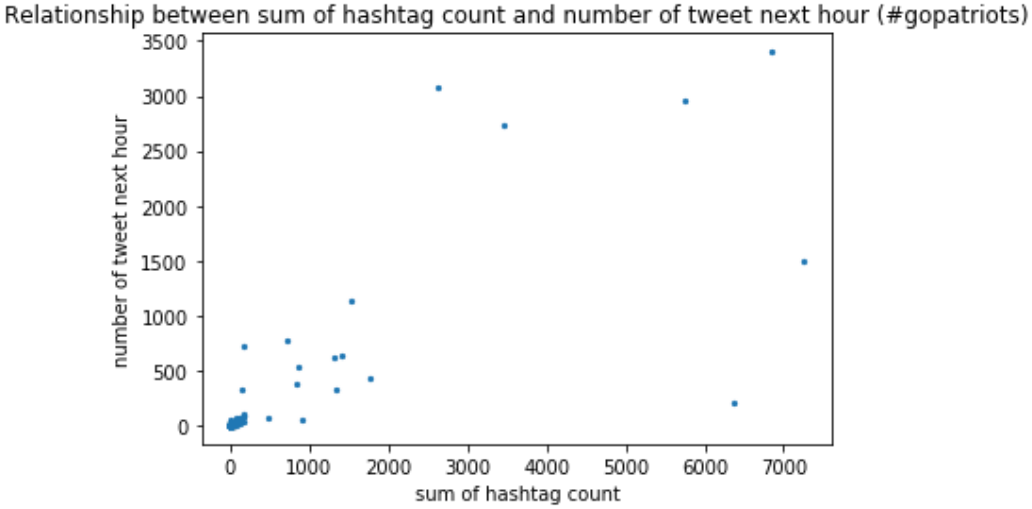
Relationship between sum of hashtag count and number of tweet next hour (#gohawks)



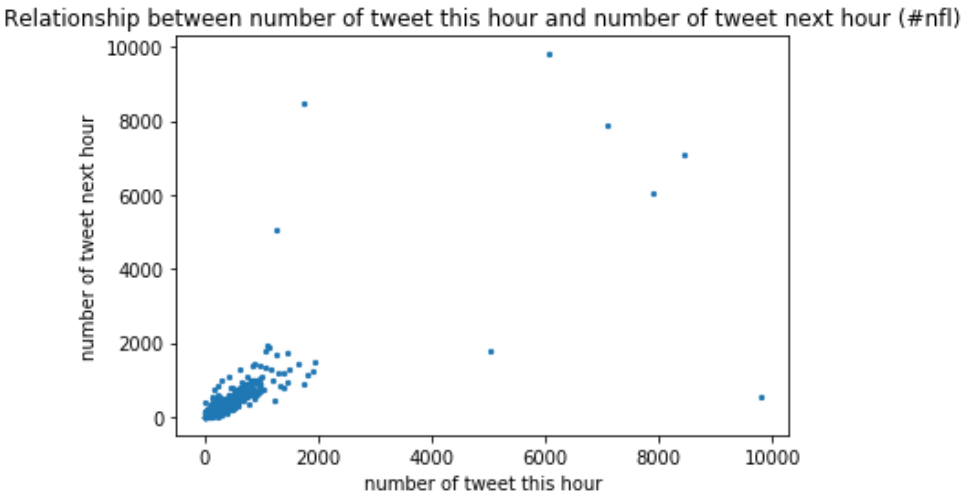
Scatterplots of top 3 features from #gopatriots

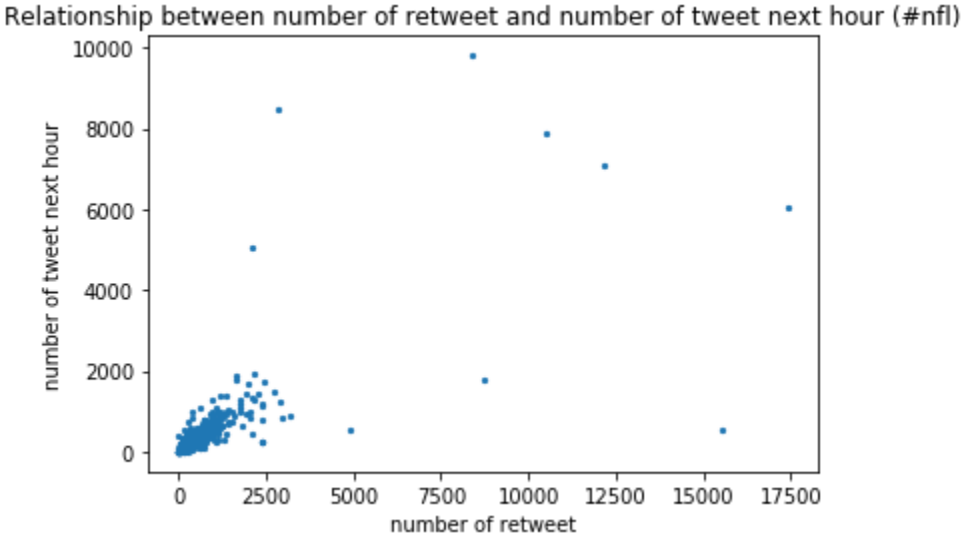


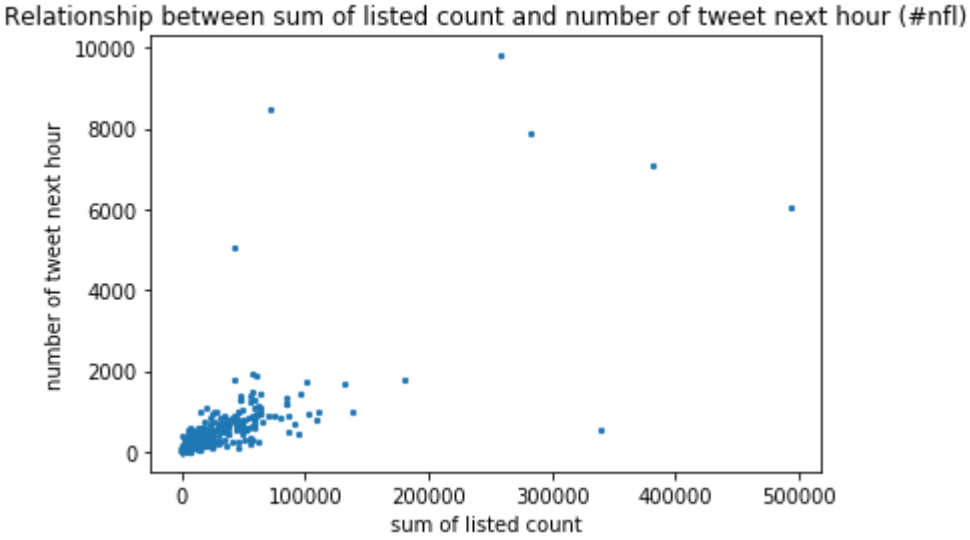




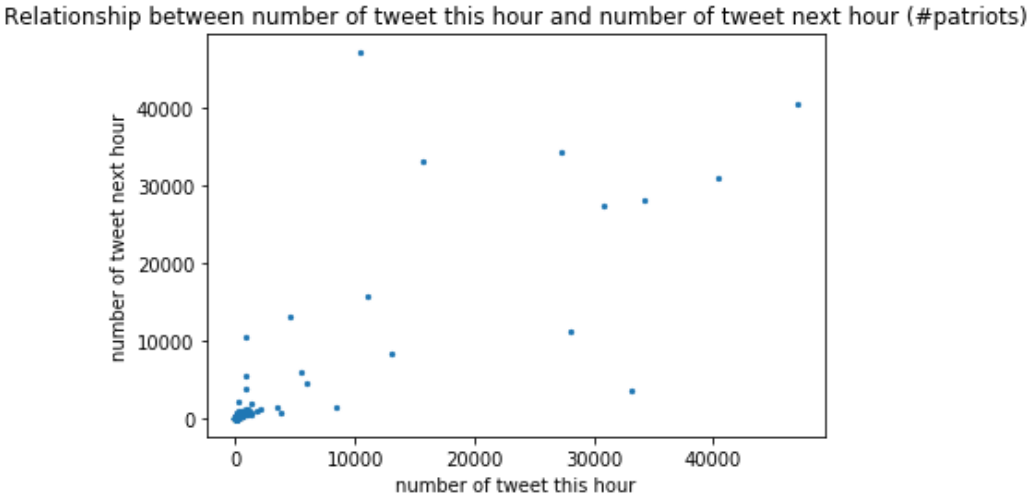
Scatterplots of top 3 features from #nfl

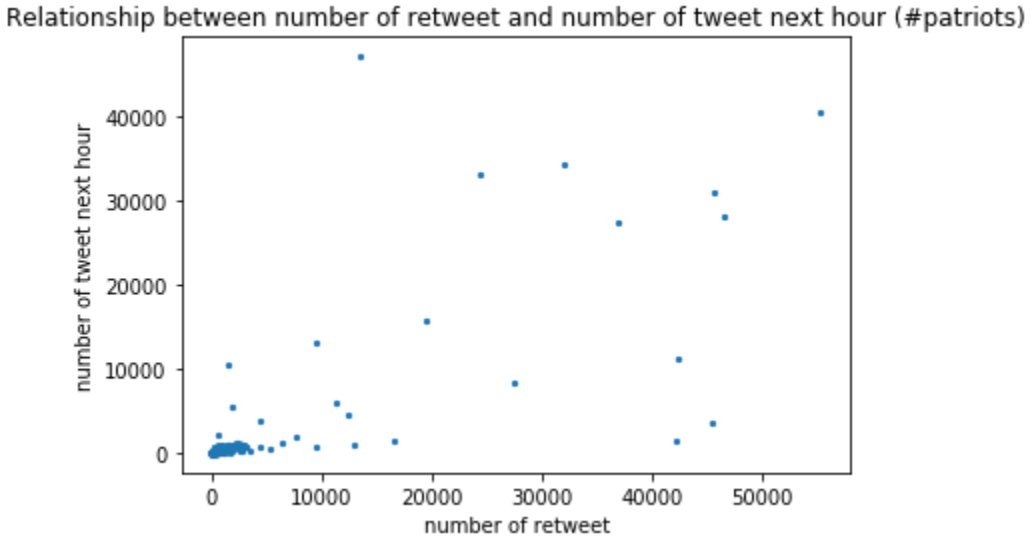


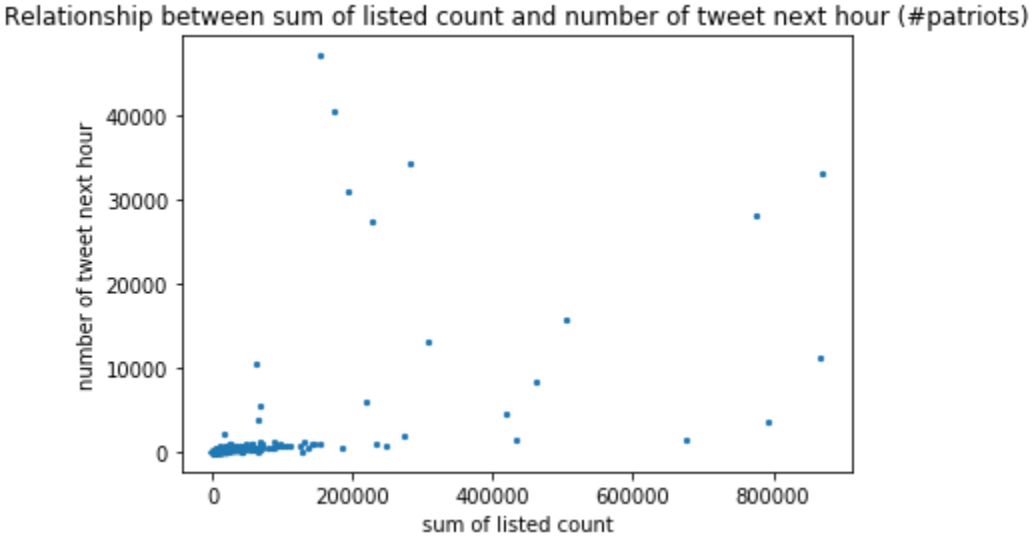




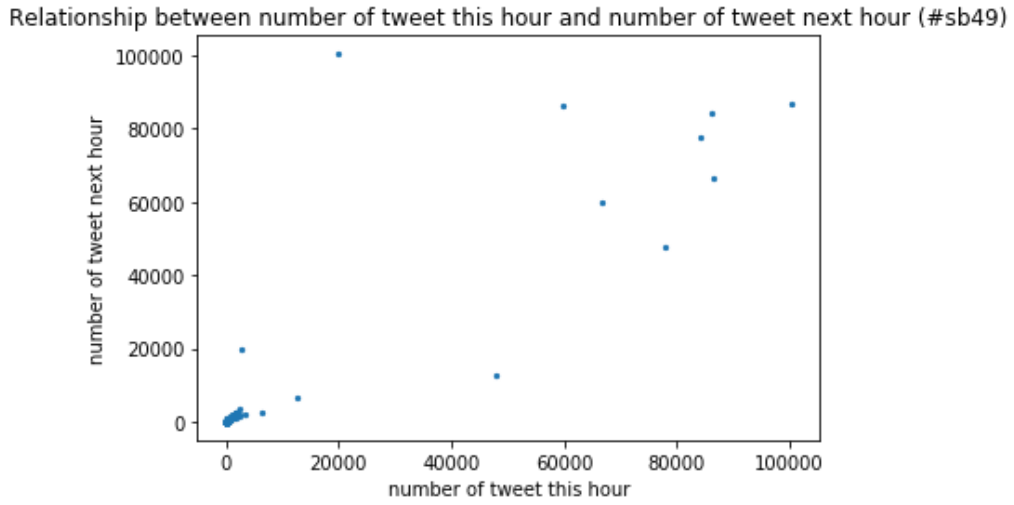
Scatterplots of top 3 features from #patriots

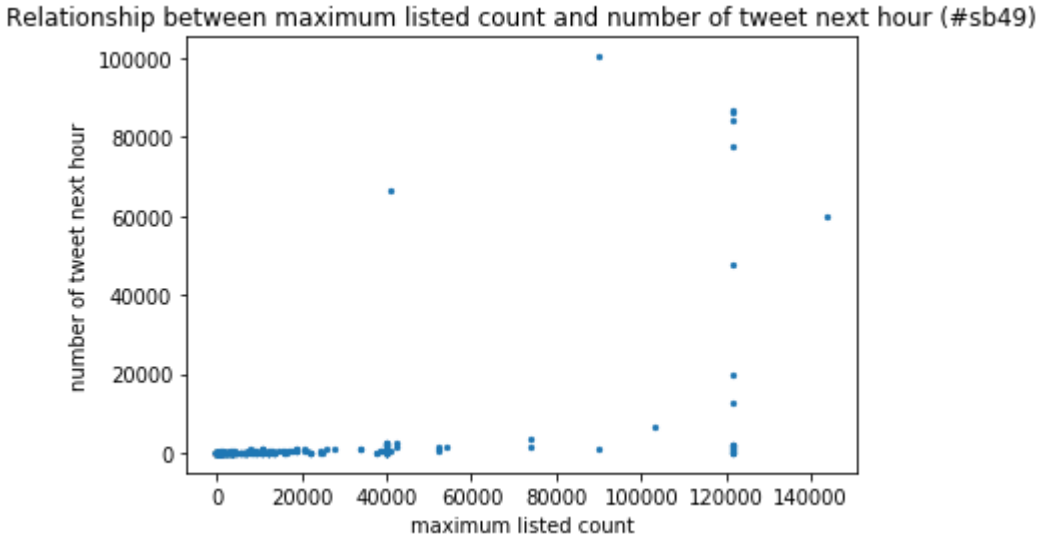


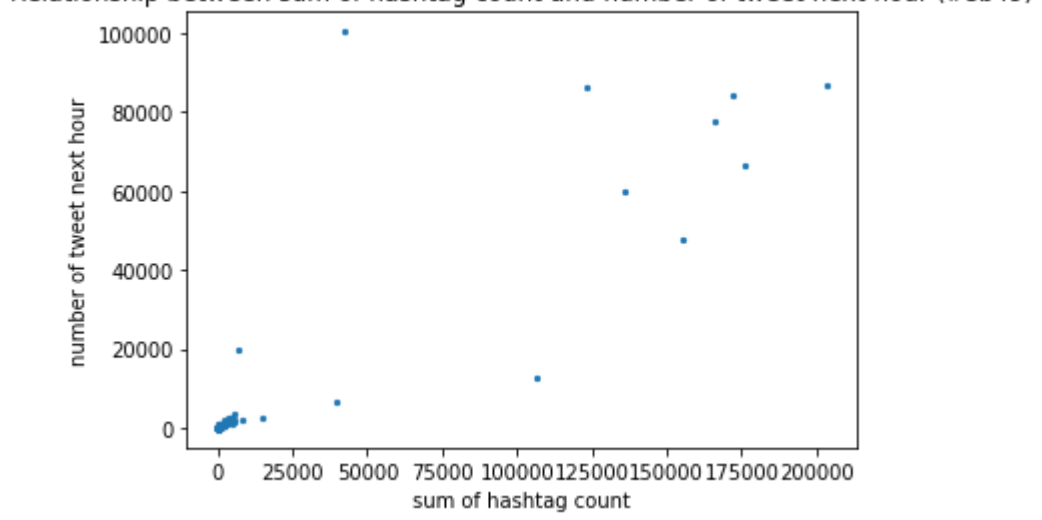




Scatterplots of top 3 features from #sb49







4. Piecewise Linear Regression

Since we know the Super Bowl's date and time, we can create different regression models for different periods of time. We define 3 time periods and their corresponding window length as follows:

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

QUESTION 6:

For each hashtag, we trained 3 regression models, one for each of the time periods listed above (the time periods are all in PST). We reported the MSE and R-squared score for each case. From the results below, we observe that in general the piecewise regression models fitted for the 1 hour window after Feb 1, 8:00 p.m. resulted in the lowest MSEs and highest R^2.

Hashtag	Before Feb 1		Feb 1		After Feb 1	
	MSE	R^2	MSE	R^2	MSE	R^2
#superbowl	40507492	0.524	2365072121	0.953	23913776	0.904
#gohawks	37556417	0.471	5329729	0.742	215683	0.833
#gopatriots	181234	0.579	568985	0.606	3477	0.842
#nfl	10491016	0.680	4841559	0.912	6707544	0.944
#patriots	38374894	0.598	128244454	0.877	2411796	0.920
#sb49	3988565	0.897	683468364	0.955	10103917	0.902

QUESTION 7:

In this part, we aggregated the data from all the hashtags. We trained 3 models, one for each of the time intervals mentioned above, to predict the number of tweets in the next hour on the aggregated data. The MSEs and R^2 for the combined models are reported below.

Before Feb 1		Feb 1		After Feb 1	
MSE	R^2	MSE	R^2	MSE	R^2

112532266	0.466	3161561547	0.942	42489741	0.705

Based on the results from the combined model for the time period before Feb 1, the MSE is higher and the R-squared is lower than those from the models trained on the individual hashtags. This indicates that the aggregated model does not perform as well compared to the individual hashtag models. For the 5-minute window on Feb 1, the MSE of the combined model is much larger than those from the individual models, but the R-squared is still pretty high. For the 1-hour window after Feb 1, the combined model results in a lower R-squared than the individual models, and the MSE is within the range of those obtained from the individual models.

5. Nonlinear Regression

In this part, we use ensemble methods and neural networks to fit the aggregated data.

5.1 Ensemble Methods

We will use `RandomForestRegressor` and `GradientBoostingRegressor` from `sklearn` as two examples of ensemble regressors on the aggregated data.

QUESTION 8:

We used grid search to find the best parameter set for the Random Forest Regressor and Gradient Boosting Regressor. The parameters we tested were max_depth (the maximum depth of the tree), max_features (the number of features to consider when looking for the best split; either the number of features or the square root of the number of features in our case), min_samples_leaf (the minimum number of samples required to be at a leaf node), min_samples_split (the minimum number of samples required to split an internal node), and n_estimators (the number of trees in the forest).

We used the following parameter set to perform the grid search:

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

Based on the 5-fold cross-validated result of the grid search, the best parameter set for the Random Forest Regressor and Gradient Boosting Regressor are shown below.

Parameter	Random Forest Regressor	Gradient Boosting Regressor
max_depth	60	80
max_features	sqrt	sqrt
min_samples_leaf	2	2
min_samples_split	2	5
n_estimators	1800	1800
MSE	2.42042e7	2.80027e7

The test errors are high, given the fact that we are training on the entire dataset, but seem better than the errors we received with the OLS models.

QUESTION 9: Compare the best estimator you found in the grid search with OLS on the entire dataset.

OLS on the entire dataset resulted in an mean squared error of 28754238351, which is significantly higher than the MSE of the best estimator found in grid search.

QUESTION 10: For each time period described in Question 6, perform the same grid search above for GradientBoostingRegressor (with corresponding time window length). Does the cross- validation test error change? Are the best parameter set you find in each period agree with those you found above?

Parameter	Before Feb 1	Feb 1	After Feb 1
max_depth	60	80	80
max_features	sqrt	sqrt	sqrt
min_samples_leaf	2	1	1
min_samples_split	5	5	10
n_estimators	1600	1800	1800
MSE	257664	160378	1297052e7

The cross validation error is lowered, which is similar to what we saw with the OLS models for aggregate vs. single time period data. We obtained similar, but slightly varied parameters for the models trained on a single time period.

5.2 Neural Network

QUESTION 11:

We regressed the aggregated data using MLPRegressor (multi-layer perceptron regressor). We tried different architectures (i.e. the structure of the network) by adjusting hidden_layer_sizes:

```
{
'hidden_layer_sizes': [(100,), (50,), (50,50), (100,25), (50, 20, 20)]
}
```

Based on the results of the 5-fold cross-validated grid search, the best hidden_layer_sizes was (50, 20, 20). The MSE from fitting the entire aggregated data using the best architecture we found was 73419261.53.

QUESTION 12:

We used StandardScaler to scale the data before feeding it to MLPRegressor, using the best architecture found in question 11 (hidden_layer_sizes = (50, 20, 20)). The resulting MSE was 20979669.71 which was lower than the MSE using the unscaled data. Thus, the performance improved after standardizing features by removing the mean and scaling to unit variance.

QUESTION 13:

Using grid search, we found the best architecture for the scaled data for each of the 3 time periods and their corresponding window length (as defined in question 6):

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

	Before Feb 1	**Feb 1**	**After Feb 1**
hidden_layer_sizes	(50,)	(50, 20, 20)	(50, 20, 20)

6. Using 6x Window to Predict

In this part, we used test data where each file in the test data contains a hashtag's tweets from a 6x-window-length time range. For example, a sample from the first time period (described in question 6) contains tweets in a 6-hour window that lies in the first time period; a sample from the second time period contains tweets in a 30 minute window that lies in the second time period.

We fit a model on the aggregate of the training data for all hashtags and predicted the number of tweets in the next hour for each test file.

QUESTION 14: Report the model you use. For each test file, provide your predictions on the number of tweets in the next time window.

We fitted the data using MLPRegressor with the best architecture found in question 11 (hidden_layer_sizes = (50, 20, 20)). We used StandardScaler to scale the data. The predictions on the number of tweets in the next time window for each sample are displayed in the table below. We can see that the predictions for the first time period (before Feb 1) and second time period (Feb 1) change drastically depending on the sample, whereas the predictions for the time period (after Feb 1) are more stable. However, in general the prediction results were quite variable on different runs, perhaps due to the fact that there were not a lot of data points to train the data on.

	Before Feb 1	**Feb 1**	**After Feb 1**
Sample 0	733	1089	119
Sample 1	1350	369	119
Sample 2	769	55	118

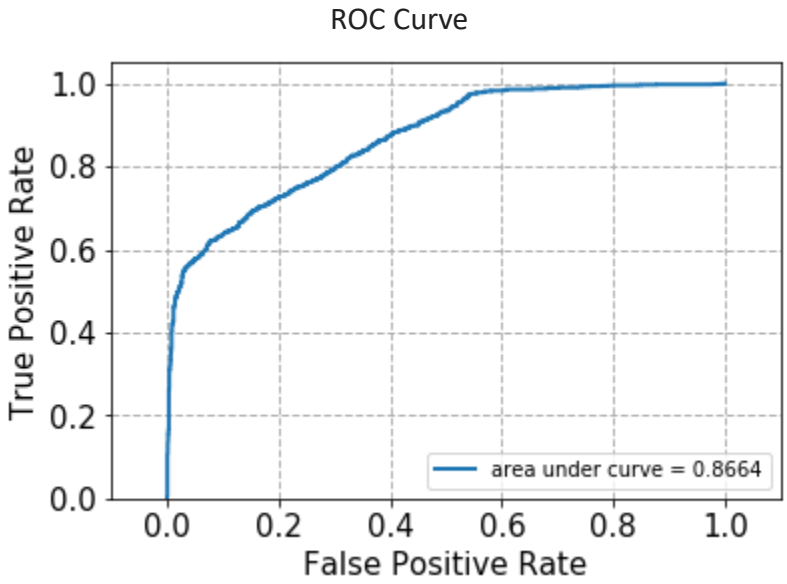
Part 2: Fan Base Prediction

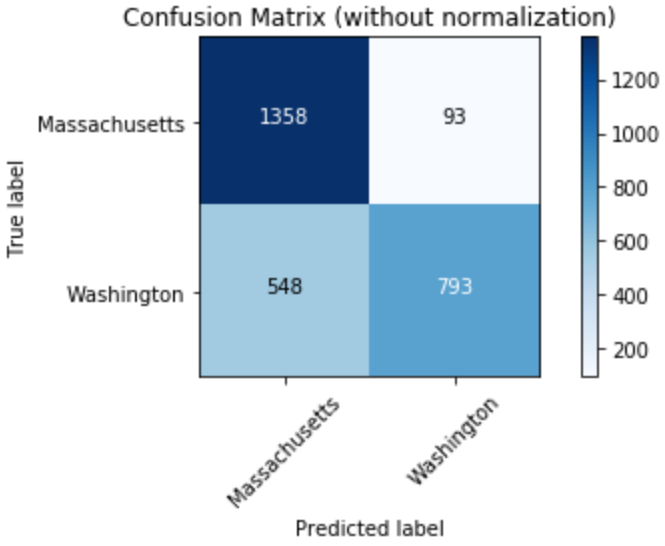
15.1 To determine which location the user was in, we looked at the location field of the tweet (`json_object['tweet']['user']['location']`). The text was converted to lowercase, and we then split up the location text into the component words. If the location contained the words were “washington” or “wa”, but not the words “dc” or “d.c”, the user was considered to be from Washington state, and if the location contained the words “massachusetts” or “ma”, the user was considered to be from Massachusetts. If the location text did not contain any of those words, then the tweet was not used for the following part. There were a total of 27911 tweets used.

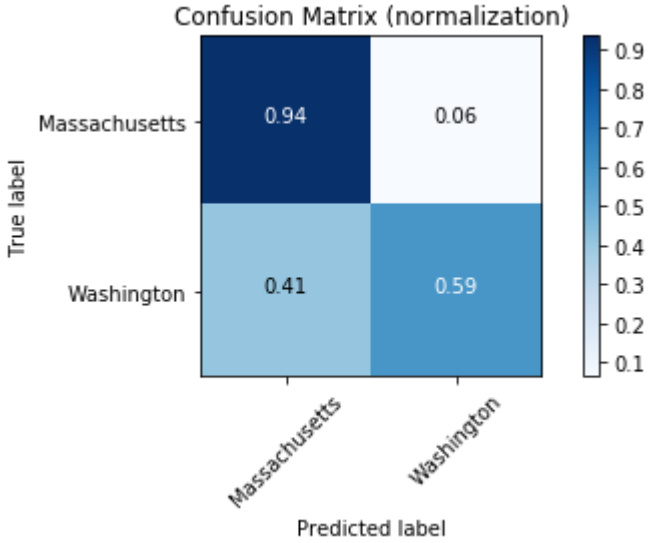
15.2 The textual context of each tweet was obtained in a similar manner to Project 1. We used a CountVectorizer with lemmatization and a minimum document frequency of 3. The count matrix was converted to a TF-IDF matrix, and then LSA reduction was performed to reduce the number of features to 100. The data set was split into 90% training data and 10% test data. We then trained 3 classifiers on the data, a linear SVM classifier, a neural network classifier, and a random forest classifier.

Linear SVM

Accuracy: 0.7704154727793696
Recall: 0.5913497390007457
Precision: 0.8950338600451467







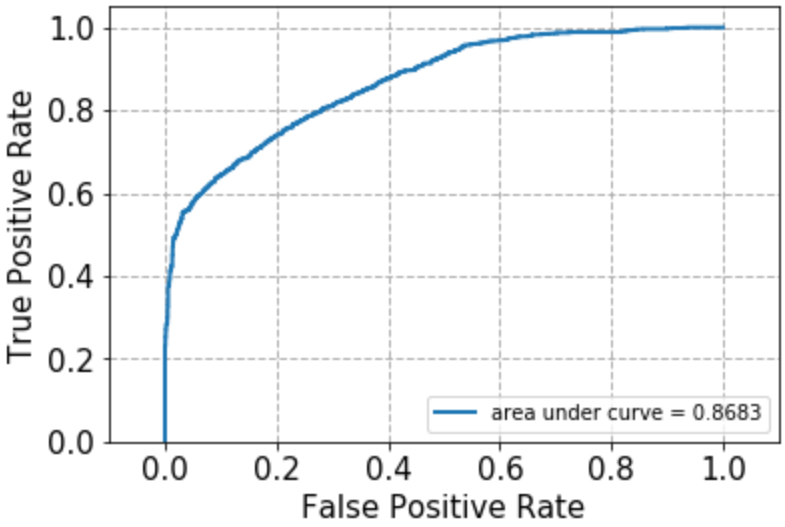
Neural Network

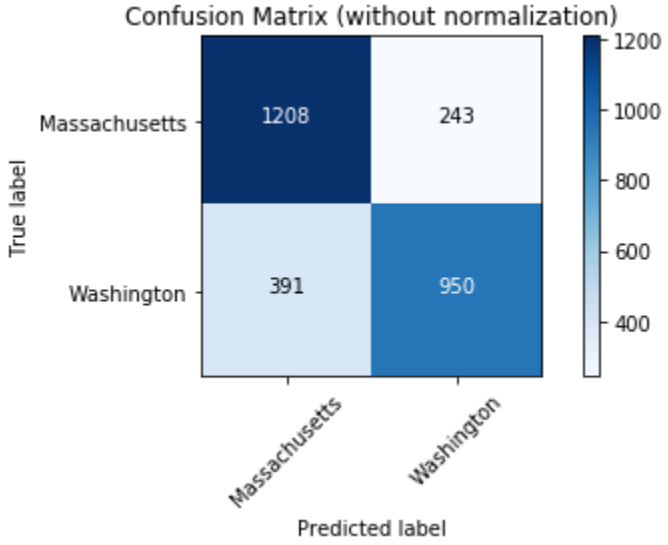
Accuracy: 0.7729226361031518

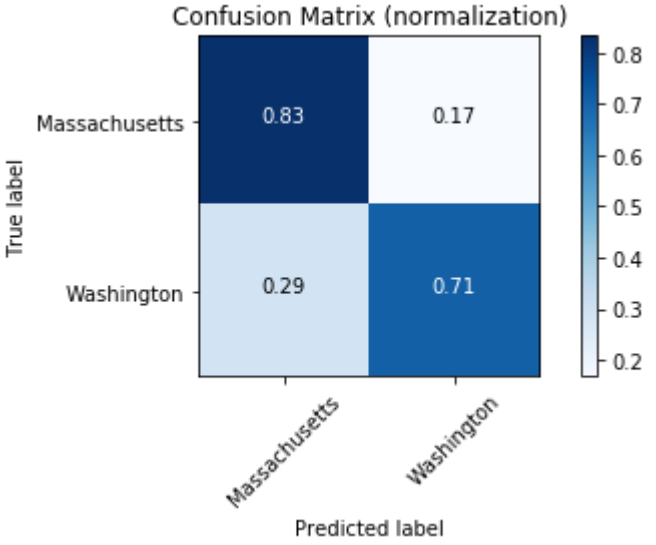
Recall: 0.7084265473527218

Precision: 0.7963118189438391

ROC Curve


True Positive Rate
False Positive Rate
area under curve = 0.8683
0.0
0.2
0.4
0.6
0.8
1.0


Confusion Matrix (without normalization)
Massachusetts
Washington
True label
1208
243
391
950
1200
1000
800
600
400
Massachusetts
Washington
Predicted label

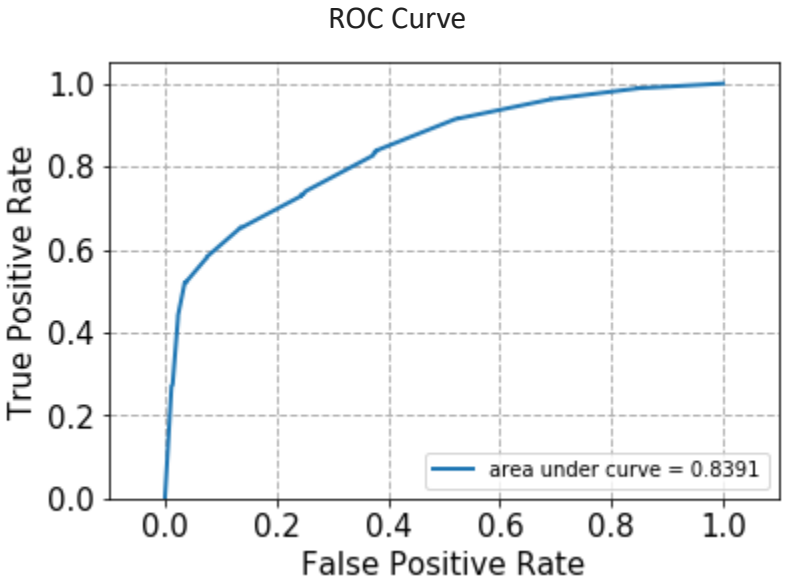

Confusion Matrix (normalization)
Massachusetts
Washington
True label
0.83
0.17
0.29
0.71
0.8
0.7
0.6
0.5
0.4
0.3
0.2
Massachusetts
Washington
Predicted label

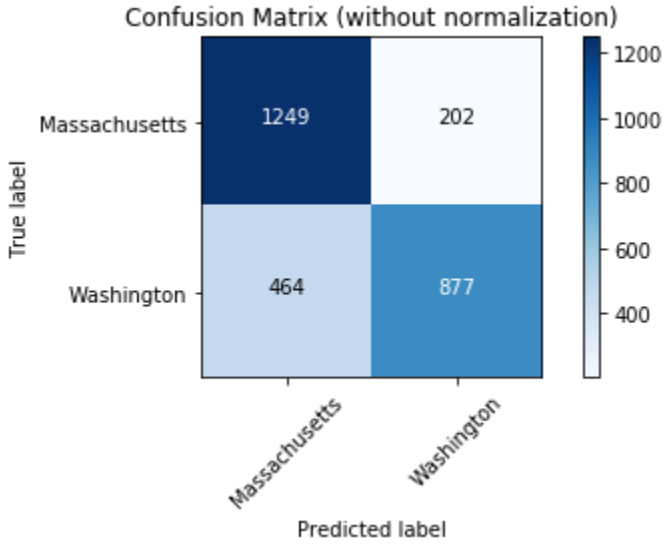
Random Forest

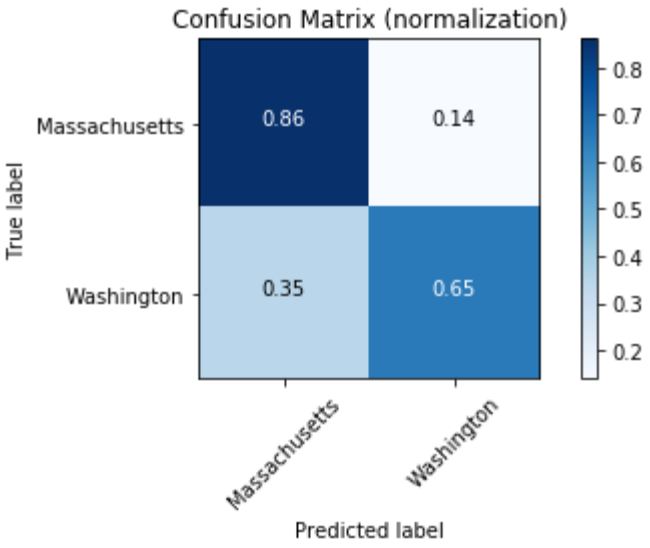
Accuracy: 0.7614613180515759

Recall: 0.6539895600298284

Precision: 0.8127896200185357






Confusion Matrix (normalization)
Massachusetts
Washington
True label
0.86
0.14
0.35
0.65
0.8
0.7
0.6
0.5
0.4
0.3
0.2
Massachusetts
Washington
Predicted label

Part 3: Fan Sentiment over Time

For this part, we aimed to see what information could be obtained from observing the average sentiment of fans of the two participating teams over time. It is logical to suppose that when a team is doing well, their fans would be more likely to post positive tweets, and when their team is doing poorly, they would be more inclined to post negative tweets. While this intuition seems logical, we aimed to see if it was true in practice, and whether fan sentiment could be used to determine things like which team won.

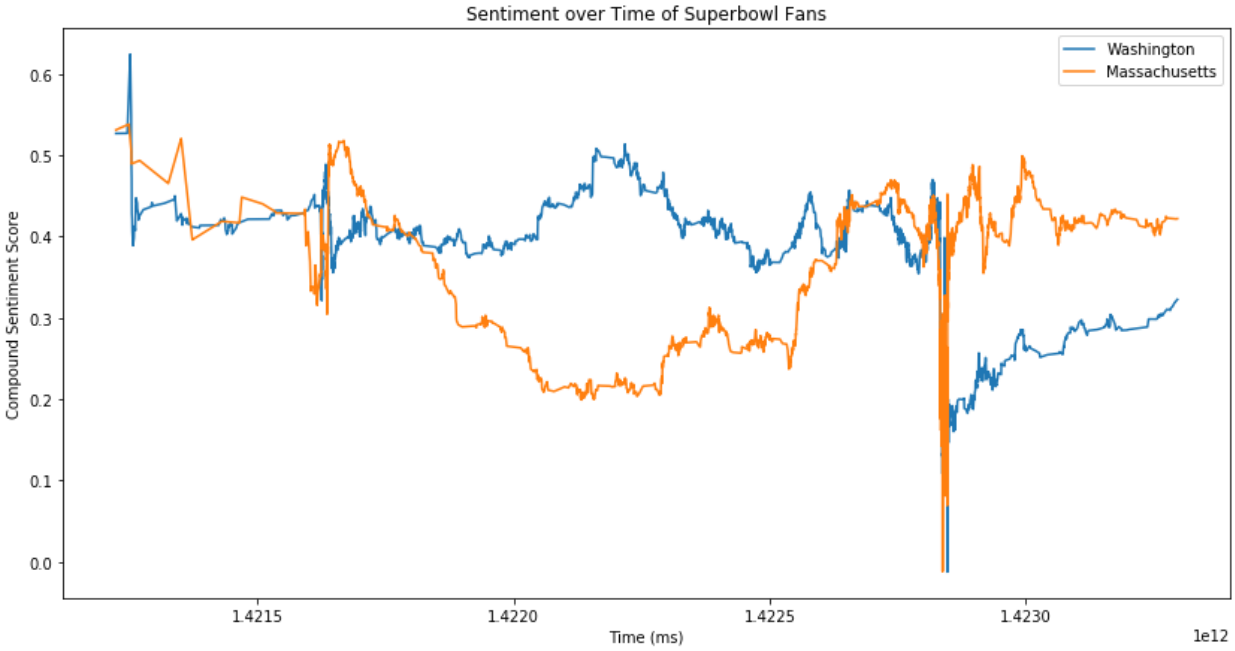
Method:

For our definition of a "fan" of a team, we decided to use the same definition from part 2, the self reported location of the tweeting user. This method was chosen over others such as using hashtags to determine fan affiliation because it would remain constant as a users sentiment changes. For example, a user might be more inclined to use the #gopatriots hashtag when the Patriots are doing well than when they are doing bad, but a user with Massachusetts set as their location is unlikely to change it based on how the team is doing. It also seemed logical to assume the majority of people from Massachusetts would be bigger fans of the Patriots than the Seahawks, and vice versa for people from Washington.

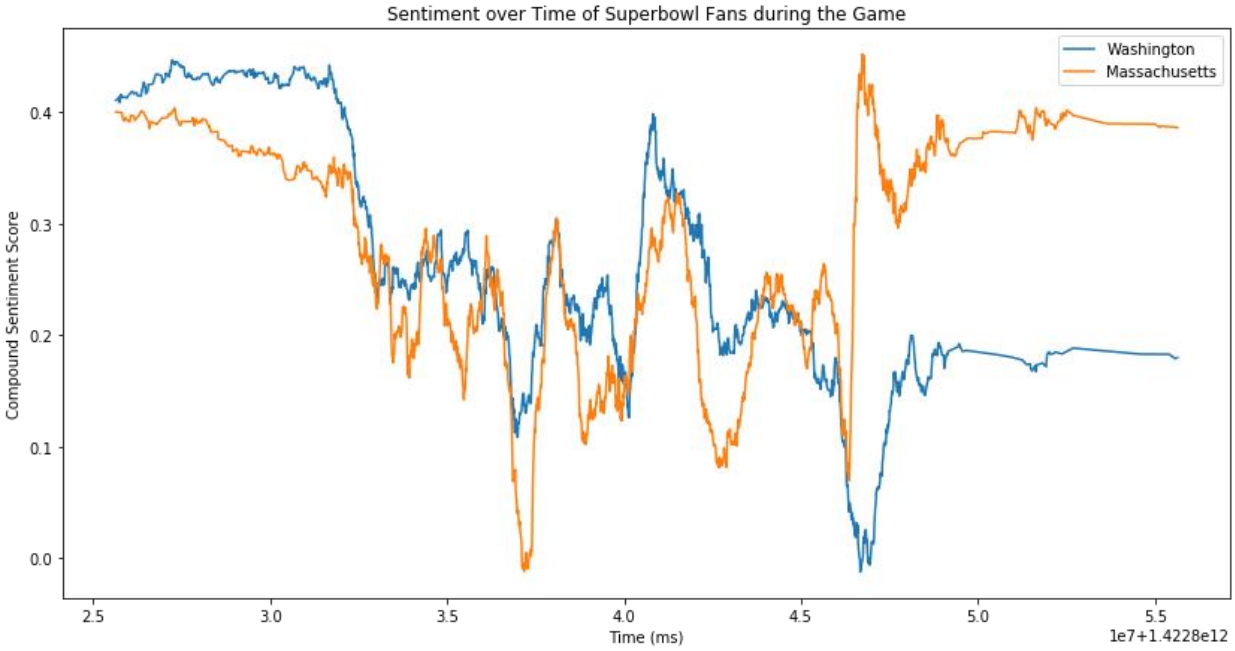
Once the location of the user was determined, two pieces of data were taken from the tweets, the time of the tweet and the sentiment. The time of the tweet was taken in the form of the timestamp, and was used to relate the sentiment across tweets. To obtain the sentiment, we used the VADER polarity score method found in the nltk package. This function returns scores for given text in the positive, negative, and neutral sentiment categories, along with a composite score for the whole text, where a higher score indicates more positive, and a lower score indicates more negative. VADER is particularly good for this task as it takes into account text emoticons, capitalization, and other methods commonly used to convey emotion in tweets. Tweets with a mostly neutral composite score were discarded, as they were assumed to be unrepresentative of fan sentiment.

After obtaining the desired data, we attempted to remove some of the noise from the sentiment by taking a moving average over a window of 200 tweets. The sentiment for each team was then interpolated at the same set of points in order to facilitate comparison between the two data sets.

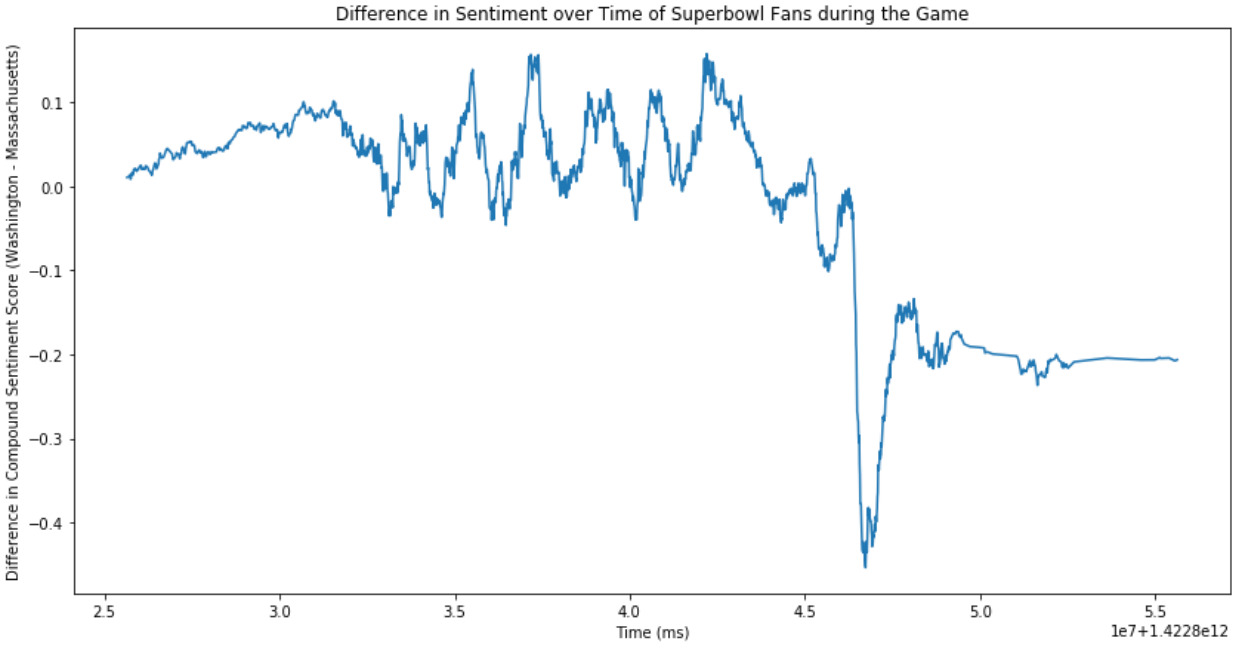
Results:



Taking a look at the data over the entire period, we notice a few interesting features. Most notably there is a period of very high variability in the sentiment of both teams. Perhaps unsurprisingly, this time period corresponds to the time that the actual game was being played. The high variability can likely be attributed emotional impact of watching the game, as well as the increase in tweets during that time, resulting in higher variation in users. Another notable fact is that leading up to the game, the sentiment of Washington and Massachusetts fans seem loosely negatively correlated, but begins to converge before the game. It is unclear why this is and if it has an significance. Additionally, we notice that after the game there is a clear separation between Washington and Massachusetts fans, with Massachusetts fans having clearly higher positive sentiment than Washington fans. This corresponds to our intuition, as the Patriots won the game.



Taking a closer look at the change in sentiment during the game, we can make some more interesting observations. For most of the game, the sentiment of the two fanbases actually seems rather positively correlated. This is slightly contrary to intuition, as one would expect positive events for one team to be negative for the other. However, it is possible that these spikes are due to more impartial events, e.g. the spike for both teams in the middle could be a reaction to the halftime show rather than one team's performance in the game. At the end of the game, the observed sentiment lines up well with intuition. Massachusetts fans see a large positive spike and Washington fans see a large negative spike before settling to fairly positive and fairly negative respectively, which is consistent with the Patriots winning the game.



We also observed the difference between the sentiment score of the two fanbases in order to get a better idea of the relation between the two. As seen in the graph, for most of the game the difference swings back and forth, being centered around zero. At the end, the difference trends heavily negative, signifying the Seahawks losing.

Conclusion:

Overall, our analysis seemed to show that there is information that can be taken from fan sentiment. From the data from this single game, it seems as if the winner of a football game could be inferred solely from sentiment analysis of the fans. However, determining the status of the game during the game itself seems much harder. To get a better idea of the usefulness of sentiment analysis we would have to obtain similar data from other games in order to be able to recognize more trends in the data.