Large Scale Data Mining: Models and Algorithms: Project #5

Due on March 22th, 2020 at 11:59pm

Professor Roychowdhury, Vwani

Wang, Yin, He, Kang

Question 1

In this project, we are going to work with the Twitter dataset collected by querying popular hashtags related to the 2015 Super Bowl within the period starting from 2 weeks before the game to 1 week after the game. In particular, we want to solve the problem of predicting the tweet activity in the future, i.e. given current and previous tweet activity, can we predict if it will become more popular in the future and if so by how much? More specifically, we are going to use the dataset consists of tweets containing a hashtag in a certain time window, and build models to predict number of tweets containing the hashtags within one hour immediately following the given time window.

First, we will perform some statistics analysis for each hashtag. In particular, we will calculate: 1. Average number of tweets per hour. 2. Average number of followers of users posting the tweets per tweet. 3. Average number of retweets per tweet, for each hashtag. The results are shown below as Table 1.

Hashtag	Average tweets/hour	Average followers/user	Average retweets/tweet
#nfl	397.0213	4662.3754	1.5345
#patriots	750.8942	3280.4636	1.7853
#sb49	1276.8571	10374.1603	2.5271
#superbowl	2072.1184	8814.9680	2.3912
#gohawks	292.4879	2217l9237	2.0132
#gopatriots	40.9547	1427.2526	1.4082

Table 1: Some statistics for each hashtag related to Superbowl.

Question 2

Next, for two specific hashtags: #SuperBowl and #NFL, we will plot 'number of tweets in hour' over time with one bar corresponding to 1-hour period. The resulting plots are shown as Figure 1 and 2:

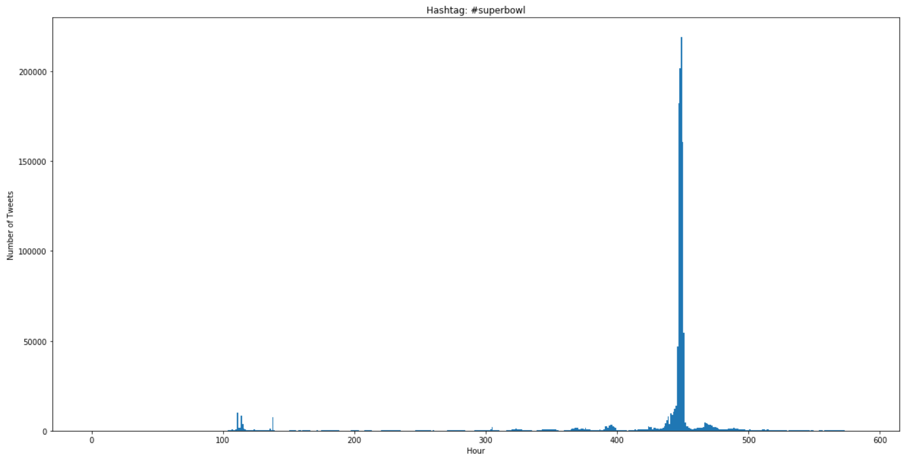


Figure 1: Number of tweets posted in one hour for #Superbowl hashtag.

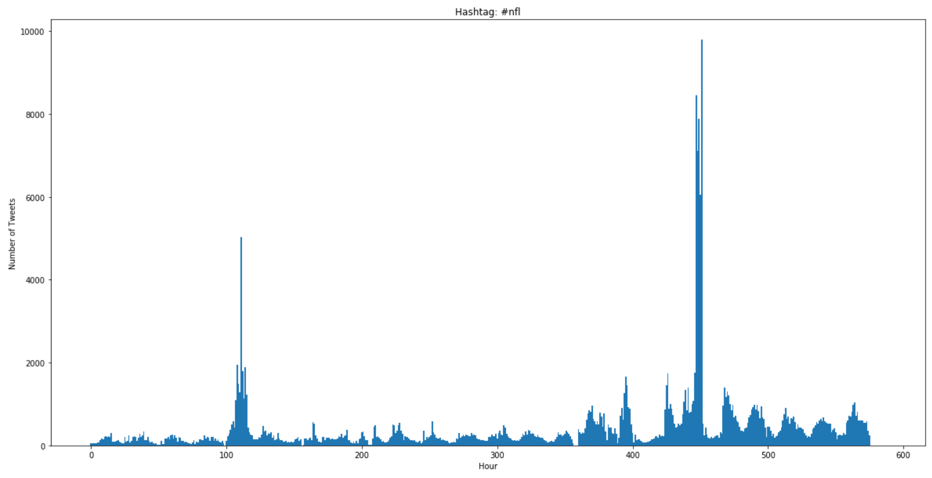


Figure 2: Number of tweets posted in one hour for #nfl hashtag.

Question 3

Then, we will first fit a linear regression model using 5 features given as: 1. Number of tweets. 2. Total number of retweets. 3. Sum of the number of followers of the users posting the hashtag. 4. Maximum number of followers of the users posting the hashtag. 5. Time of the day. In particular, we will create 1-hour windows and calculate the features in each time window, and then fit the linear regression model. In this question, we are going to first report the Mean Squared Error (MSE) and R-squared measure for each hashtag. The results are shown in the following as Table 2.

Hashtag	MSE	R-score
#nfl	6.3343×10^6	0.570
#patriots	1.5652×10^8	0.668
#sb49	8.997×10^8	0.804
#superbowl	2.6266×10^9	0.800
#gohawks	1.4731×10^7	0.476
#gopatriots	9.6673×10^5	0.627

Table 2: MSE and R-squared measure for each hashtag related to Superbowl.

Next, for each hashtag, we will analyze the significance of each feature using the t-test and p-value. First, the resulting t-test and p-value for each feature for each hashtag are shown in the Table 3.

In general, $t-statistic$ can be computed as the *coefficient* divided by its *standard error*, and the *standard error* is a measurement of the *standard deviation* of the coefficient. In short, it represents the precision of a certain coefficient. The $p-value$ is connected to the $t-statistic$, in the way that if we know the $t-statistic$ and degree of freedom, we can find out the $p-value$ easily by referring to a table. The meaning of $p-value$, assuming that everyone has had at least one statistic course, will not be discussed in detail here, but in short, the idea is that given a significance level and the null hypothesis, depending on the $p-value$, we decide whether to reject the null hypothesis or not. Now referring to the results listed above, assuming that the significance level is 0.05, then we can say that the significant features in each hashtag are shown as Table 4:

Question 3 continued on next page...

Hashtag	tweets		retweets		followers		max followers		time	
	t-test	*p*-value	*t*-test	*p*-value	*t*-test	*p*-value	*t*-test	*p*-value	*t*-test	*p*-value
#nfl	4.173	0.000	-2.578	0.010	4.473	0.000	-3.527	0.000	0.093	0.926
#patriots	12.937	0.000	-1.178	0.239	-0.417	0.677	1.340	0.181	-0.426	0.670
#sb49	12.485	0.000	-1.953	0.051	0.743	0.458	2.056	0.040	-0.666	0.506
#superbowl	28.537	0.000	-5.544	0.000	-6.265	0.000	4.889	0.000	-0.470	0.639
#gohawks	7.767	0.000	-3.113	0.002	-2.407	0.016	0.403	0.687	0.254	0.799
#gopatriots	0.937	0.349	2.223	0.027	-0.424	0.672	-0.113	0.910	-0.239	0.812

Table 3: *p*-value and *t*-test of linear regression for different hashtags and different features.

Hashtag	Tweets number	Retweets number	followers number	max followers	Time
#nfl	$\surd$	$\surd$	$\surd$	$\surd$	$\times$
#patriots	$\surd$	$\times$	$\times$	$\times$	$\times$
#sb49	$\surd$	$\times$	$\times$	$\surd$	$\times$
#superbowl	$\surd$	$\surd$	$\surd$	$\surd$	$\times$
#gohawks	$\surd$	$\surd$	$\surd$	$\times$	$\times$
#gopatriots	$\times$	$\surd$	$\times$	$\times$	$\times$

Table 4: Significance features for different hashtags.

These features are chosen because their $p - values$ are less than 0.05. And one thing needs to be noticed is that the time feature is not significant in all the dataset, thus we may say that this feature is not a significant feature. For details, i.e. detailed OLS Regression Results, please refer to our python project notebook.

Question 4

Now, in this question, we are going to add more features and train the regression model again. By reading the reference paper, http://arxiv.org/abs/1401.2018, we found out that 'mention count' and 'url count' can be the features. Thus, besides the original features, we decided to add 'number of mentions per tweet' and 'number of urls per tweet' into our feature space. And then, we will retrain the model and report the MSE and significance of each features. Thus, we will reuse the code in question 3 with these two new features added. The resulting MSEs are shown in the Table 5:

Hashtag	MSE
#nfl	6.3343×10^6
#patriots	1.5652×10^8
#sb49	8.997×10^8
#superbowl	2.6266×10^9
#gohawks	1.4731×10^7
#gopatriots	9.6673×10^5

Table 5: MSE and R-squared measure for each hashtag related to Superbowl.

Then, the resulting $t - test$ and $p - value$ for each feature and each hashtag are shown as Table 6.

Question 4 continued on next page. . .

Feature name	*t*-test						*t*-test					
	nfl	patriots	sb49	superbowl	gohawks	gopatriots	nfl	patriots	sb49	superbowl	gohawks	gopatriots
tweets	-0.092	-6.162	-4.725	-2.980	6.181	1.075	0.927	0.000	0.000	0.003	0.000	0.283
retweets	-2.806	-1.932	1.868	-12.185	-1.391	-1.391	0.005	0.054	0.062	0.000	0.165	0.000
followers	1.411	11.713	5.100	-2.144	-4.508	0.849	0.159	0.000	0.000	0.032	0.000	0.396
max_followers	-1.316	-6.692	-0.799	-0.537	1.428	-1.125	0.189	0.000	0.425	0.592	0.154	0.261
time	-0.251	0.218	-0.724	-0.485	-0.392	-0.204	0.802	0.828	0.469	0.628	0.695	0.839
mentions	4.105	-1.717	-3.799	-1.420	8.425	4.240	0.000	0.086	0.000	0.156	0.000	0.000
urls	4.574	3.958	5.738	16.323	-2.785	8.986	0.000	0.000	0.000	0.000	0.006	0.000

Table 6: p-value and t-test of linear regression for different hashtags and different features with extra features introduced.

Again, by the same logic as in Question 3, by referring to the results listed above, assuming that the significance level is 0.05, then we can say that the significant features in each hashtag can refer to Table 7:

Hashtag	Tweets	Retweets	followers	max followers	Time	Mentions/tweet	urls/tweet
#nfl	$\times$	$\surd$	$\times$	$\times$	$\times$	$\surd$	$\surd$
#patriots	$\surd$	$\times$	$\surd$	$\surd$	$\times$	$\times$	$\surd$
#sb49	$\surd$	$\times$	$\surd$	$\times$	$\times$	$\surd$	$\surd$
#superbowl	$\surd$	$\surd$	$\surd$	$\times$	$\times$	$\times$	$\surd$
#gohawks	$\surd$	$\times$	$\surd$	$\times$	$\times$	$\surd$	$\surd$
#gopatriots	$\times$	$\surd$	$\times$	$\times$	$\times$	$\surd$	$\surd$

Table 7: Significance features for different hashtags with extra features introduced.

Similarly, the time feature is not significant in all the dataset, thus we may say that this feature is not a significant feature, and for detailed OLS Regression Results, please refer to our python project notebook.

Question 5

In this question, we will first identify the top 3 features, i.e. with the smallest p-values, in each hashtag, and draw a scatter plot of predictant, i.e. number of tweets for next hour, versus value of that feature. By inspecting the results from Question 4, we can easily identify the top 3 features from each hashtag and organize them as Table 8:

Hashtag	1_{st}	2_{nd}	3_{rd}
#nfl	mentions/tweet	urls/tweet	retweets
#patriots	tweets	followers	max followers
#sb49	tweets	followers	mentions/tweet
#superbowl	tweets	followers	mentions/tweet
#gohawks	tweets	followers	mentions/tweet
#gopatriots	retweets	mentions/tweet	urls/tweet

Table 8: Top three features based on p-value for different hashtags.

Then, we can draw the scatter plots for different hashtags. Specifically, the scatter plots for #nfl are shown as Figure 3. The scatter plots for #patriots can refer to Figure 4. For hashtag #sb49, scatter plots are shown as Figure 5. Figure 6 represents the scatter plot for hashtag #superbowl. Figure 7 and 8 relate to

Question 5 continued on next page...

hashtag #gohawks and #gopatriots respectively.

In general, we can see that the as the predicted number increases, the value of the features also increases. In terms of the regression coefficients, from the OLS Regression Results, we can see that there are both positive and negative coefficients in each hashtag model. This directly implies that the regression coefficients do not always agree with the trends in the plots. Generally speaking, since all the plots indicate an increasing trend, we would say that the coefficients also need to be positive, however, the key difference is that we are separating the features and plot the trend one by one, thus when combining all the features to build a model, there are things like intercorrelations between features need to be considered, thus giving the result that some coefficients are positive while some are negative.

Question 6

Using what we learned and applied before, we train a Piece-wise linear regression model for all three periods for each hashtag. Namely, we will use several extra features which is not mentioned in requirement to help us find better relationship between data and label. Here we will use active degree, mentioned, media, author, favorite counts and title to predict the tweets posted next hour or next five minutes as well. Corresponding results are organized as Table 9

Hashtag	Before Superbowl		Between Superbowl		After Superbowl	
	MSE	R^2	MSE	R^2	MSE	R^2
#NFL	5.9789×10^5	0.5481	3.2258×10^2	0.8539	1.4845×10^4	0.8427
#patriots	2.9719×10^6	0.6260	1.4068×10^6	0.8954	5.1792×10^4	0.9319
#sb49	3.4771×10^4	0.8785	6.0349×10^6	0.8539	1.5517×10^5	0.9517
#superbowl	4.1335×10^6	0.3943	8.8328×10^4	0.9274	8.1802×10^5	0.8815
#gohawks	4.2113×10^6	0.5988	7.2076×10^2	0.9025	8.6508×10^2	0.8966
#gopatriots	1.2317×10^4	0.7091	1.3948×10^1	0.8857	2.5696×10^0	0.9818

Table 9: Metrics for linear regression for different hashtag on different periods.

In order to perform the analysis intuitively, we can plot the R^2 value of each hashtag into a histogram, which is shown as Figure 9.

For different labels, as far as training R^2 is concerned, the performance varies widely. The best result is the third period of "#gopatriots", and the worst result is the first period of "#superbowl". On average "#sb49" has the best performance, while "#superbowl" has the worst performance. We guess the reason of such result is because the distribution of "#sb49" is more uniform, and the distribution of "#superbowl" changes faster, specially for it before February 1. Generally speaking, linear models can better handle uniform distributions. Unfortunately for the Super Bowl, there is too much non-linearity around the Super Bowl date (February 1), which can lead to poor performance.

Question 7

In this question, it is similar to Question 6 but this time we consider the aggregated data for all hashtags. We still train a linear regression model for all three periods, and MSE and R2 are shows in Table 10 below.

Question 7 continued on next page. . .

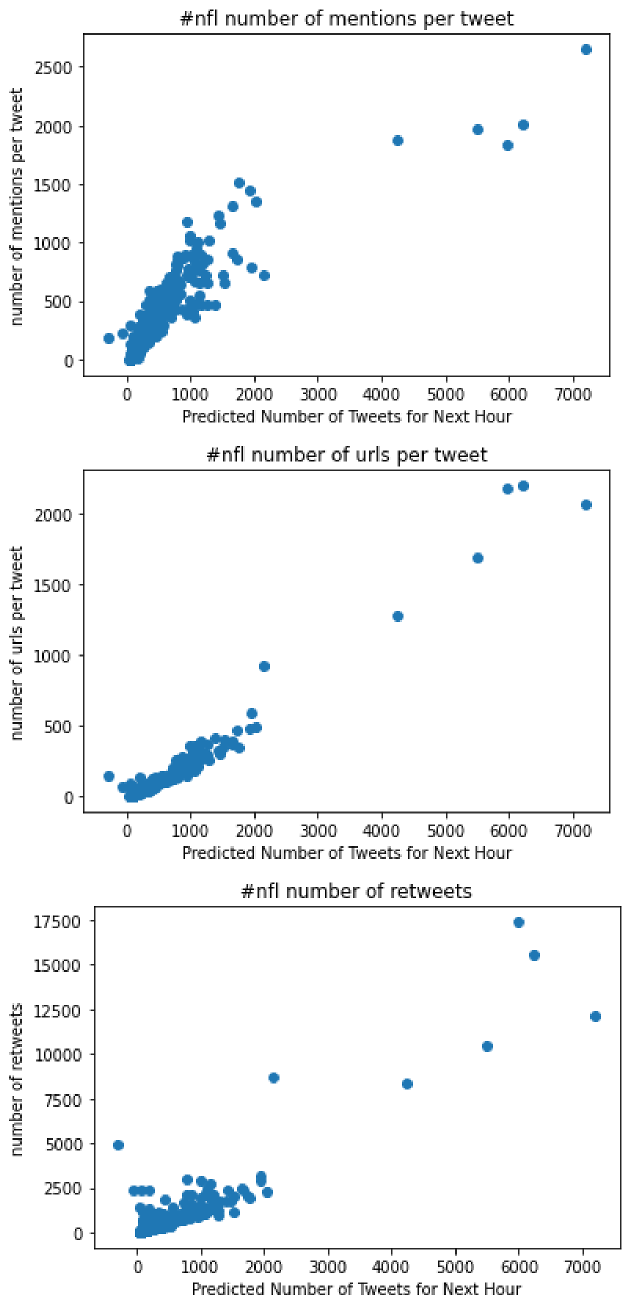


Figure 3: Scatter diagram of top three features and predicted next hour tweets for #nfl.

Data	Before Superbowl		Between Superbowl		After Superbowl	
	MSE	R^2	MSE	R^2	MSE	R^2
Entire Data	3.8458×10^7	0.4638	1.6228×10^7	0.8953	2.2780×10^6	0.9253

Table 10: Metrics for linear regression for entire aggregated data on different periods.

Question 7 continued on next page. . .

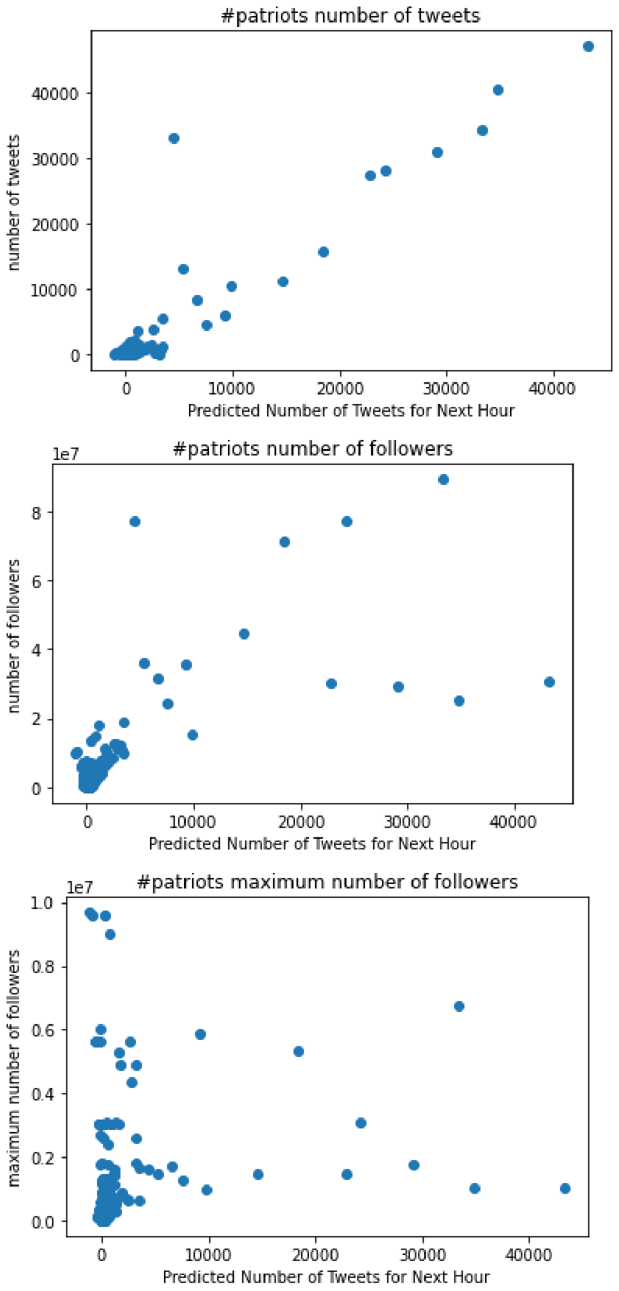


Figure 4: Scatter diagram of top three features and predicted next hour tweets for #patriots.

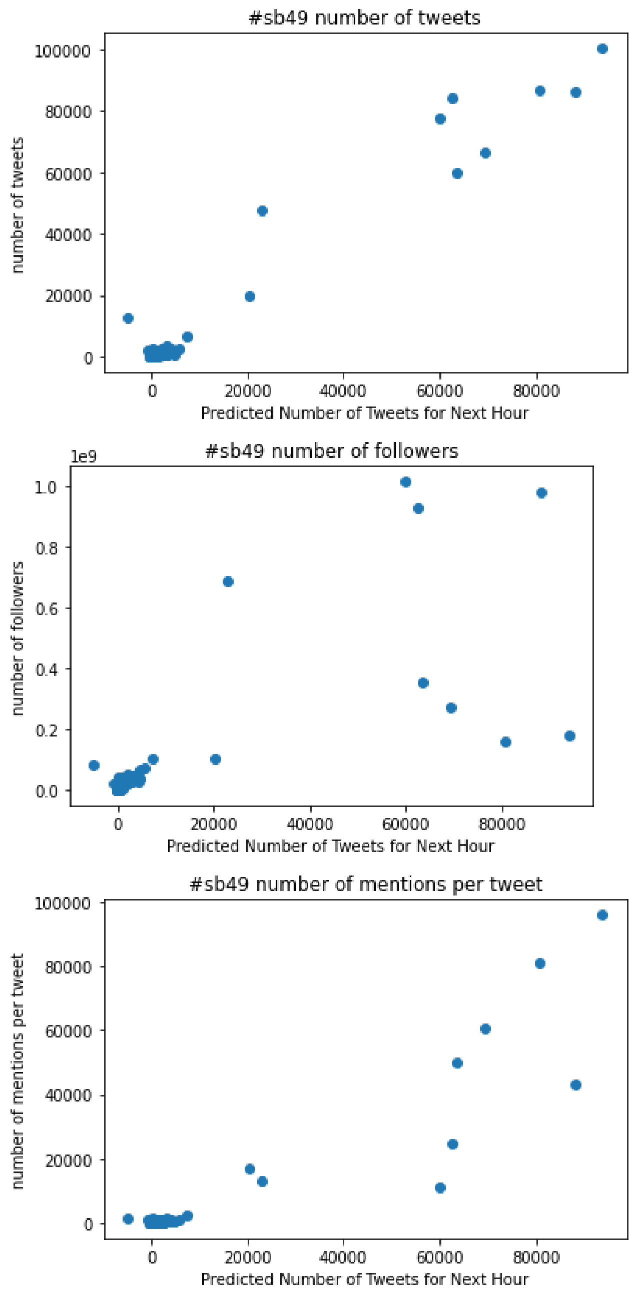


Figure 5: Scatter diagram of top three features and predicted next hour tweets for #sb49.

Question 7 continued on next page...

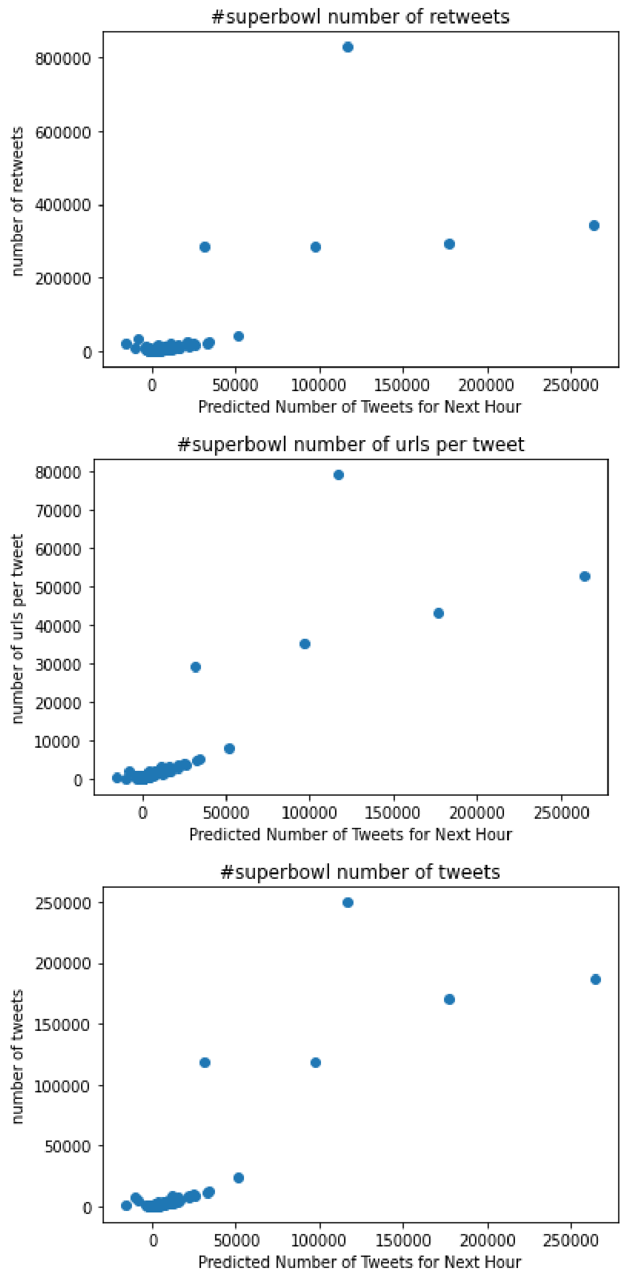


Figure 6: Scatter diagram of top three features and predicted next hour tweets for #superbowl.

Question 7 continued on next page. . .

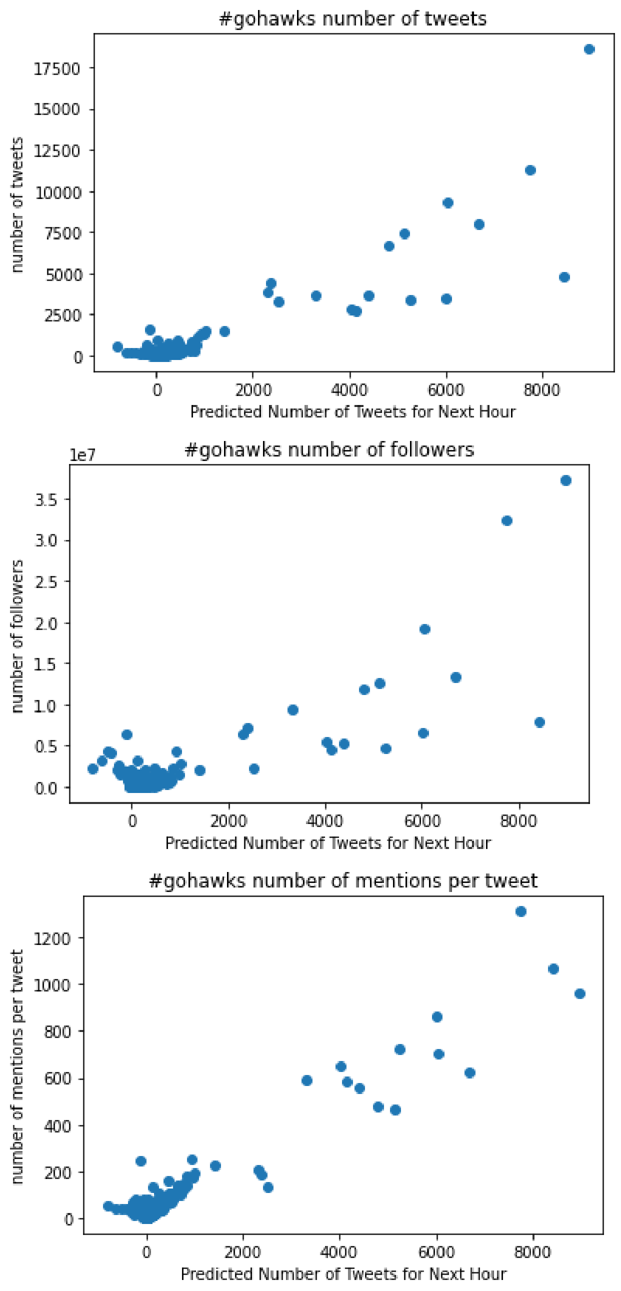


Figure 7: Scatter diagram of top three features and predicted next hour tweets for #gohawks.

Question 7 continued on next page. . .

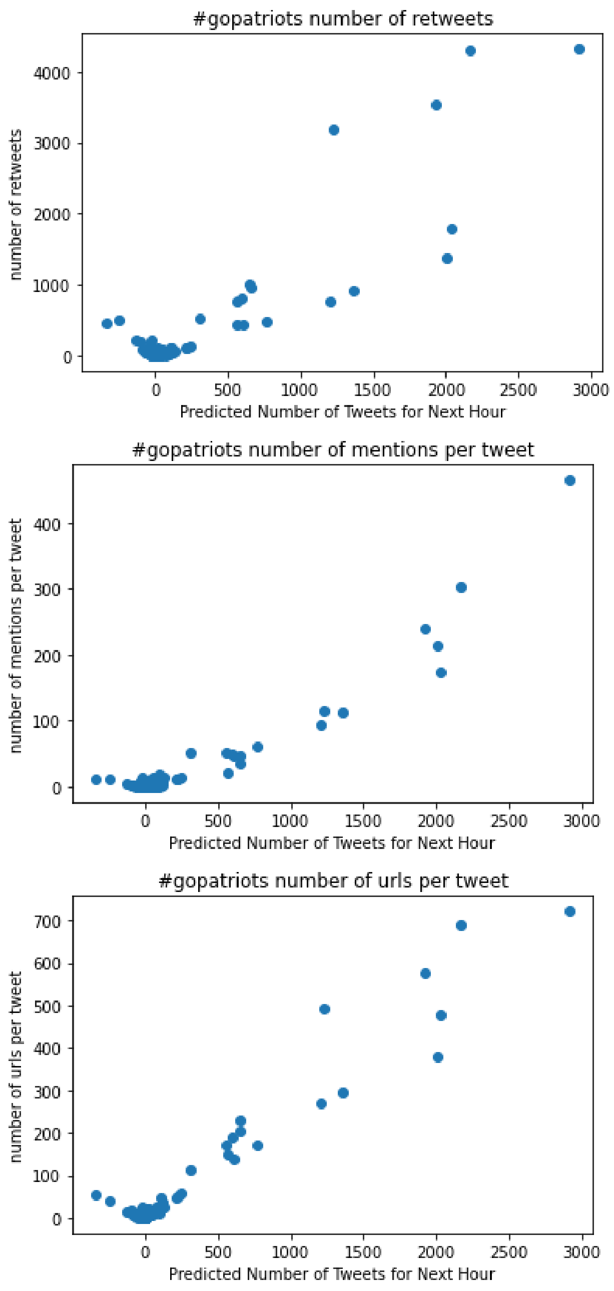


Figure 8: Scatter diagram of top three features and predicted next hour tweets for #gopatriots.

Question 7 continued on next page. . .

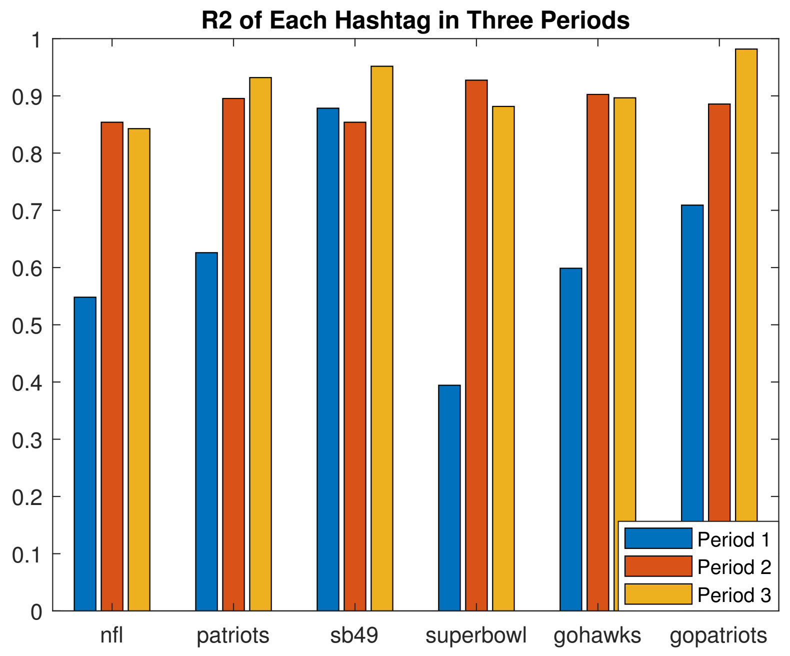


Figure 9: R^2 score for each hashtag in three time periods.

Needless to say, the results of the aggregated data have a similar pattern to the case of a single hashtag (Question 6), with a relatively low MSE in the period before and after the active time. That is because the MSE of the aggregated data should be a weighted average of the MSEs of the individual data. Since the disadvantage of the linear regression mode, we consider using a non-linear regression model in next section to train and predict the number of tweets.

Question 8

We apply two integrated methods (Random Forest, Gradient Boosting) and neural network methods for prediction. One thing needs to be noticed that the project required a grid search with 5-fold CV to find the optimal parameters set for each of these two methods. However, because of the large amount of calculation, we actually use the greedy algorithm just like Project 4. Table 11 summarizes the best parameter set and corresponding test RMSE from CV.

Model	max depth	max features	min leaf	min split	n_estimators	RMSE
Random Forest	10	sqrt	1	2	1800	4.4689×10^3
Gradient Boosting	10	sqrt	3	2	800	4.4512×10^3

Table 11: Best Model parameters for Random Forest and Gradient Boosting and their corresponding test RMSE.

In the models, the parameters max features, min samples leaf, and min samples split have a greater influence on the results. The score of train dataset is extremely good ($score = 0.9999999978871486$, which can be seen as 1), but the test RMSE of both two models are quite large. On the one hand, the high RMSE is because of the large date size. The total RMSE of a 12GB data set cannot be too low. However, on the other hand,

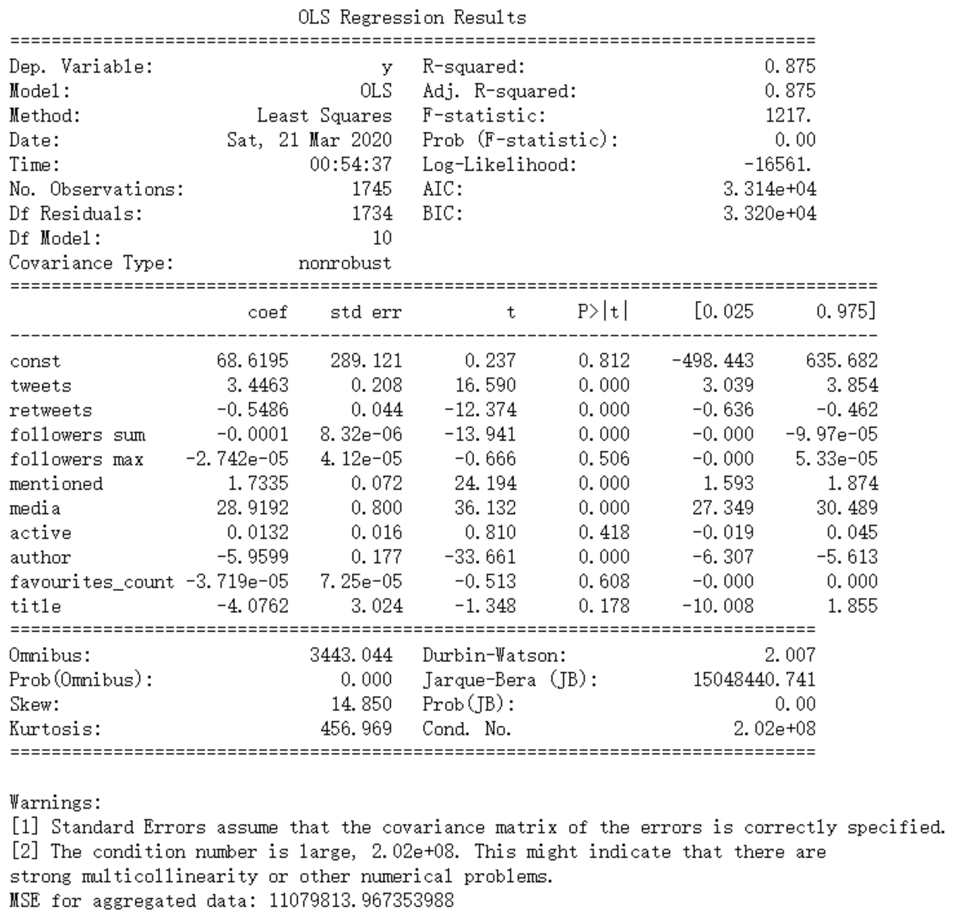
Question 8 continued on next page. . .

there are overfitting in our model, especially for the Gradient Boosting. These two reasons can explain the huge RMSE of our data.

It shows that RF is much easier to tune than GBM. There are typically two parameters in RF: number of trees and number of features to be selected at each node. And RF is harder to overfit than GBM.

Question 9

In this part, we use OLS to analysis the whole data set. Figure 10 is the detailed OLS results on the entire data set. In the OLS results, the most important feature order with the smallest p-value is consistent with the node feature order of random forest regression. The corresponding MSE for three hypothesis class is



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.875
Model:                            OLS   Adj. R-squared:                  0.875
Method:                 Least Squares   F-statistic:                     1217.
Date:                Sat, 21 Mar 2020   Prob (F-statistic):               0.00
Time:                        00:54:37   Log-Likelihood:                -16561.
No. Observations:                1745   AIC:                         3.314e+04
Df Residuals:                    1734   BIC:                         3.320e+04
Df Model:                          10
Covariance Type:            nonrobust
===================================================================================
                      coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------
const              68.6195    289.121      0.237      0.812    -498.443     635.682
tweets              3.4463      0.208     16.590      0.000       3.039       3.854
retweets           -0.5486      0.044    -12.374      0.000      -0.636      -0.462
followers sum      -0.0001   8.32e-06    -13.941      0.000      -0.000   -9.97e-05
followers max   -2.742e-05   4.12e-05     -0.666      0.506      -0.000    5.33e-05
mentioned           1.7335      0.072     24.194      0.000       1.593       1.874
media              28.9192      0.800     36.132      0.000      27.349      30.489
active              0.0132      0.016      0.810      0.418      -0.019       0.045
author             -5.9599      0.177    -33.661      0.000      -6.307      -5.613
favourites_count -3.719e-05   7.25e-05     -0.513      0.608      -0.000       0.000
title              -4.0762      3.024     -1.348      0.178     -10.008       1.855
==============================================================================
Omnibus:                     3443.044   Durbin-Watson:                   2.007
Prob(Omnibus):                  0.000   Jarque-Bera (JB):        15048440.741
Skew:                          14.850   Prob(JB):                         0.00
Kurtosis:                     456.969   Cond. No.                     2.02e+08
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.02e+08. This might indicate that there are
strong multicollinearity or other numerical problems.
MSE for aggregated data: 11079813.967353988
```

Figure 10: Verbose result for OLS method applied to the entire aggregated data.

shown below as Table 12.

Metric	OLS	Random Forest	Gradient Boosting
MSE	1.1080×10^8	1.9971×10^8	1.9813×10^8

Table 12: Comparison for different model performance based on MSE.

According to the previous result, OLS enjoys a better performance here.

Question 10

In this section, we will further explore the gradient boosting methods described for three different periods. For each period, we use 5-fold CV for the same greedy algorithm for gradient boosting. Table 13 summarizes the optimal parameter set of CV and corresponding test MSE in different periods.

Time Period	max depth	max features	min leaf	min split	n_estimators	RMSE
Before Superbowl	10	sqrt	3	2	200	1.9893×10^3
Between Superbowl	10	sqrt	4	2	600	1.7811×10^3
After Superbowl	10	sqrt	4	10	2000	5.7002×10^2

Table 13: Best Model parameters for different time period based on gradient boosting model.

Compared to the test MSE for the entire period shown in Table 8, the test MSE for all three periods signifyingly decreased. The optimal parameter settings for each period are different from each other, and also different from the parameters for the entire period. Considering that we are using a greedy algorithm, the global optimal solution is not obtained. So performance can actually be improved even more. We believe that the flexibility of choosing parameters per cycle can explain the better performance of the piecewise gradient enhancement method.

Question 11

In this question, we will use the entire aggregated data derived in Question 6 as data set. This means that we will not use any prior knowledge about the trend of topic. As mentioned, we just need to construct a DataFrame containing all entries of tweets posted during the interval we are interested, set the frequency of *pd.Grouper* as one hour and finally exert appropriate operation on the grouper separated to extract features. Inheriting the feature construction in Question 2 without introducing any extra features, we get a 587×5 matrix. The corresponding value vector is created by shifting the first column of this matrix one index backward. However, the last row vector of this data matrix does not relate to any value so that we will delete the this vector to ensure our data will not be contaminated.

To restrict the number of architecture we will explore, we just provide 3 possible depths, 3 alternative neuron number for each layer and force that every layer in the network shares the same width, namely, the number of neurons in each layer is the same. The result is organized as Table 14, which implies that the best model here is network with 3 hidden layers of width 20. This result is a little bit counter-intuitive: In general, the more sophisticated architecture is, the better the model can fit the seen data. The contradiction may result from the fact that the sophisticated model need more iterations to converge while we restrict the maximum iteration as 10000. In another aspect, this restriction can be seen as early stop to limit the model complexity. So the result is somewhat meaning.

Question 12

The standardization operation can be easily applied to our model by introducing the *StandardScaler* in *sklearn.preprocessing* into the pipeline. The mean square error (MSE) with standardization method for the best architecture found in Question 11 is 1.0230×10^6. Even though we just inspect the model performance on training data without validation or test, this result is still meaningful as mentioned. In other words,to some extent, the MSE on the entire data set can represent the model generalization ability. Now that the

Question 12 continued on next page. . .

Depth	Width	MSE
2	20	7.4579×10^{9}
2	30	1.6651×10^{10}
2	50	1.0146×10^{10}
3	20	1.9701×10^{9}
3	30	4.3186×10^{9}
3	50	1.2574×10^{12}
5	20	6.2035×10^{9}
5	30	4.2435×10^{9}
5	50	3.9150×10^{11}

Table 14: MSE for different architectures based on the entire aggregated data.

result is much better than the one shown above, the standardization method does improve the generalization ability of our model.

Question 13

The generalization ability of our model is not good enough to predict the trend of popularity for the Superbowl. To increase the correctness of our model, we will separate the entire data into three parts according to their posting time. The intuition of this operation is straightforward: users in time period holding Superbowl are much more active than other period and the corresponding data is more volatile. So we need to use shorter window to seize the possible pattern between aggregated data. To construct the training set and validation set for each period, we first use the '$timestamp_raw$' column to separate our DataFrame into three sub-DataFrame. Then we will use *pd.Grouper* to aggregate distributed data with different frequency setting. Finally, grid search with parameter grid same as Question 11 is adopted to find the best architecture for each period. Rather than using MSE, we use R^2 score to measure the performance of our model. In all, the result is shown as Table 15.

From this tabel, we can directly find the best model for different time interval: for time period before Superbowl, a network with depth 2 and width 50 will perform best; for time period between Superbowl, the best architecture is five layers within which there are 50 neurons. For time period after Superbowl, we can attain the best performance by setting hidden layers as 2 and width of layers as 30. From this result, we can see that the pattern between each period is totally different for MLP to predict, meaning that our intuition to separate the data is correct. From the result derived, we can conclude that the data between Superbowl is much harder to predict, which lives up to our expectation.

Time period	Depth=2			Depth=3			Depth=5		
	width=20	width=30	width=50	width=20	width=30	width=50	width=20	width=30	width=50
Before Superbowl	-1.4210	-1.7745	0.0065	-0.5030	-0.5422	-0.6180	-0.0742	-0.2923	-0.4983
Between Superbowl	-29.5051	-14.1764	-18.9024	-387.3015	-172.2165	-190.9701	-5.2087	-98.0898	-324.0637
After Superbowl	0.5986	0.6784	0.5486	0.4176	-0.0876	0.3241	0.3112	-0.1112	0.0316

Table 15: R^2 score for different time periods based on different network architecture setting.

Question 14

The linear regression model seems to not fit this dataset well, and we tried different models in Question 8 and 9. We applied Random Forest and Gradient Boosting method, and for this question, we used random forest and here are the results shown as Table 20.

Sample	Period 1		Period 2		Period 3	
	Prediction	Actual	Prediction	Actual	Prediction	Actual
Sample0	1073.8264	121	23056.8236	1123	894.3748	87
Sample1	5610.3484	844	10937.9374	910	487.3837	46
Sample2	1739.8463	61	285.2184	28	567.4563	43

Table 16: Random Forest prediction and correpsonding ground truth for different test set

Question 15

We determined the authors in Washington by finding the following classes in the location categories: "Seattle, Washington", "Washington", "WA", "Seattle, WA", "Kirkland, Washington". By observation of the class names, we found the authors to be in Massachusetts when their location belongs to any of the "Massachusetts", "MASS", "MA", "Boston, Massachusetts", "Boston, MA". We may have missed a few data points, but this should include sufficiently enough of them.

First, we tried logistic regression. For the confusion matrix for this hypothesis class, it can refer to 17. The corresponding roc curve is displayed as Figure 11.
It seems to have a reasonable performance. The confusion matrix we used are normalized so that we can compare between different methods.

		Predicted	
		WA	MA
Actual	WA	1	106
	MA	0	530

Table 17: Confusion Matrix of the logistic regression on twitter data.

The second model we used is K nearest neighbor, and here are the following results. For confusion matrix, it can refer to Table 18 while the corresponding ROC curve is shown as Figure 12

		Predicted	
		WA	MA
Actual	WA	9	98
	MA	48	482

Table 18: Confusion Matrix of the K-nearest neighbor on twitter data.

Finally we use the Support vector machine to fit the data. The confusion matrix of this method is shown as Table 19 and the roc curve can refer to Figure 13. The three model's accuracy the precision and recall rate for MA class is organized into one table, which is shown as Table **??**.

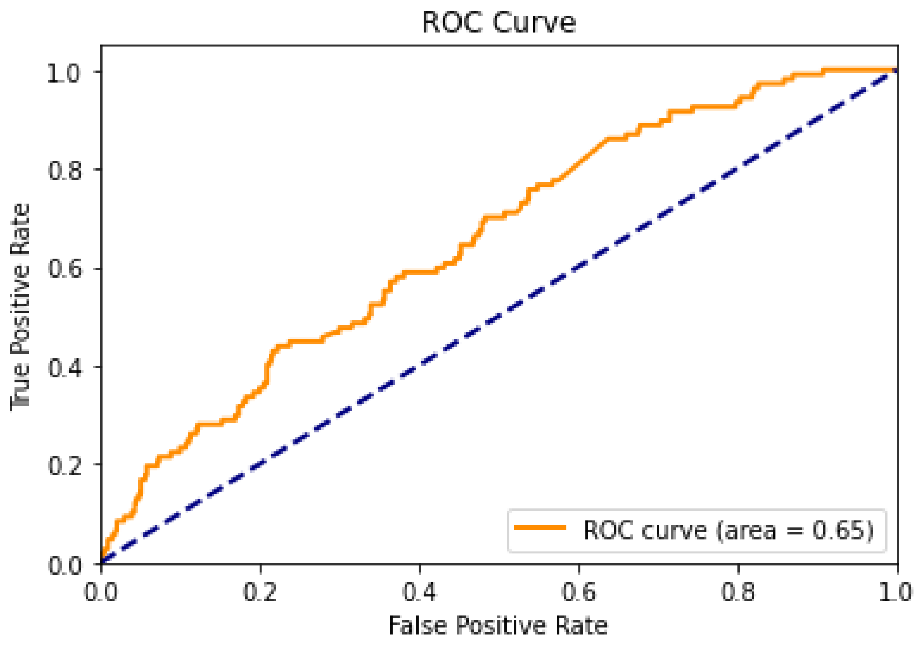


Figure 11: ROC curve for logistic regression model.

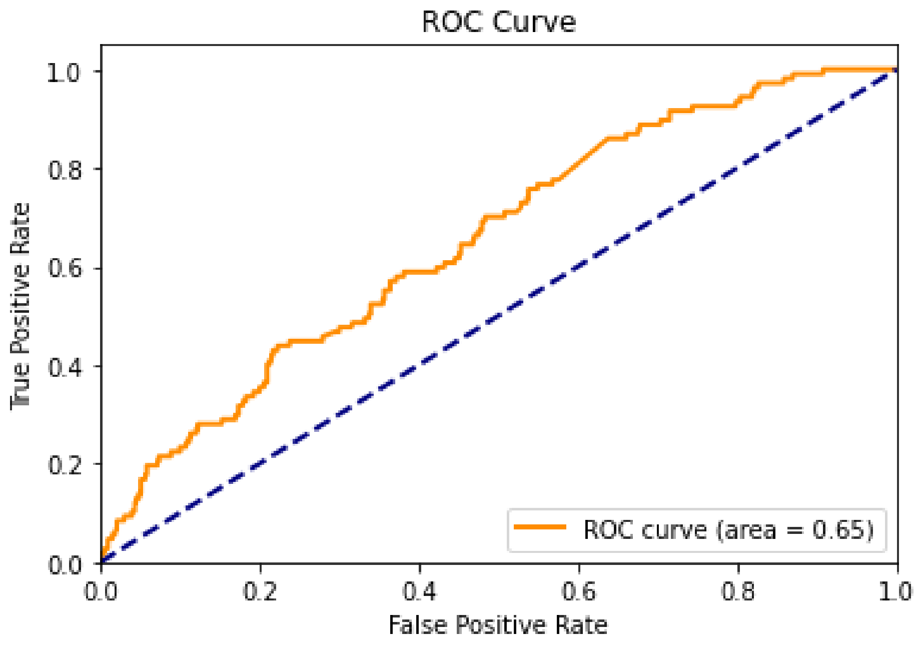


Figure 12: ROC curve for K-nearest neighbor model.

		Predicted WA	Predicted MA
Actual	WA	1	106
	MA	0	530

Table 19: Confusion Matrix of the K-nearest neighbor on twitter data.

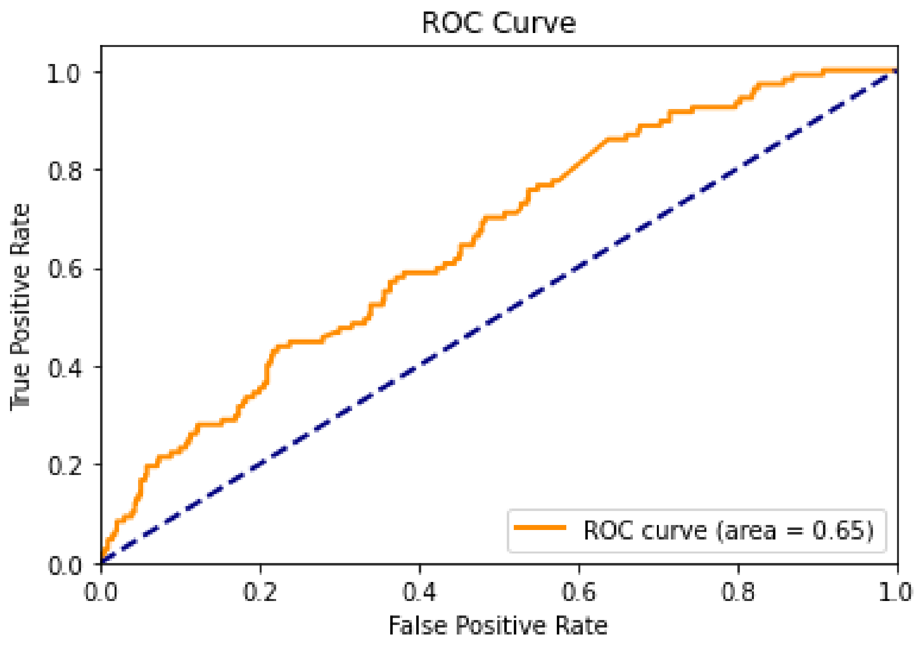


Figure 13: ROC curve for K-nearest neighbor model.

Question 16

For our own exploration, we decide to do some sort of tweet sentiment analysis as suggested in the project spec. We can get a sense of what people were thinking and talking about at the time by looking at the most frequent used words. We do this hourly analysis and hopefully we can get some insight from the results. We would expect people's tweets and emotions change as the game goes and player's performance varies. They would probably tweet some encouragement words for the team they support. We used IF-IDF matrix to gain insight about the frequency of the words in each tweet. Since we looked Massachusetts and Washington in the previous question, we are interested in the location of the people tweeting. We do a NMF transformation for the TF-IDF matrix for each hour, and we check the location of the user of each tweet. Our results for MA and WA respectively:
From 12:00:00 - 13:00:00
Patriotswin patriots ve winning nfl sb49 supperbowlxlix superbowl
Seahawkswin seahawks ve winning got nfl sb49 http superbowlxlix superbowl

From 13:00:00 - 14:00:00
patriotswin patriots winning ve got nfl sb49 http superbowlxlix superbowl
seahawkswin seahawks winning ve got nfl sb49 http superbowlxlix superbowl

From 14:00:00 - 15:00:00

Question 16 continued on next page. . .

Model	Accuracy	Precision$_{MA}$	Recall$_{MA}$
Logistic	0.834	0.833	1.000
KNN	0.771	0.831	0.909
SVM	0.834	0.833	1.000

Table 20: Metrics of different models for fitting the Twitter data.

patriotswin patriots winning ve got nfl sb49 http superbowlxlix superbowl
seahawkswin seahawks winning ve got nfl sb49 http superbowlxlix superbowl

From 15:00:00 - 16:00:00
patriotswin patriots winning ve got nfl http sb49 superbowl superbowlxlix
seahawkswin seahawks winning ve got nfl http sb49 superbowl superbowlxlix

From 16:00:00 - 17:00:00
winning ve got nfl http patriotswin sb49 patriots seahawkswin seahawks
touchdown superbowlxlix superbowl patriots seahawks sb49 http gohawks gronk seahawkswin

From 17:00:00 - 18:00:00
winning got ve nfl http sb49 patriots patriotswin seahawks seahawkswin
superbowl superbowlxlix halftime katyperry katy perry http seahawks sb49 halftimeshow

From 18:00:00 - 19:00:00
topspot2015 voted patriots seahawks http sb49 dodgewisdom jeepplays dodge katyperry
superbowl patriots superbowlxlix http sb49 seahawks touchdown game nfl catch

From 19:00:00 - 20:00:00
superbowlxlix win patriots congrats champions sb49 game http super bowl
patriotswin superbowl http game sb49 seahawks nfl play wow fight

In the first few hours we have examined, there was not many changes. But as time progresses, we can see that from 19:00:00 - 20:00:00, a winner has been selected, and people are celebrating for it. Also, there is the constant appearance of http, which is probably a noise that represents links that people are sending back and forth.