EE219 - UCLA

WINTER 2019

PROJECT 5: REPORT

Application - Twitter Data

GROUP MEMBERS

Anchal Goyanka
Nandan Parikh
Pratik Mangalore
Srishti Majumdar

Part 1: Popularity Prediction

Understanding the Data

Q1. Statistics for each hashtag

Following are the required statistics of the data from each of the hashtag files:

Hashtags	Average tweets per hour	Average number of followers	Average retweets
gohawks	292	2217.92	2
gopatriots	40	1427.25	1
nfl	397	4662.37	1
patriots	750	3280.46	1
sb49	1276	10374.16	2
superbowl	2072	8814.96	2

Q2. Number of Tweets in hour

Following are two plots of histogram for the “number of tweets in hour” over time for #SuperBowl and #NFL

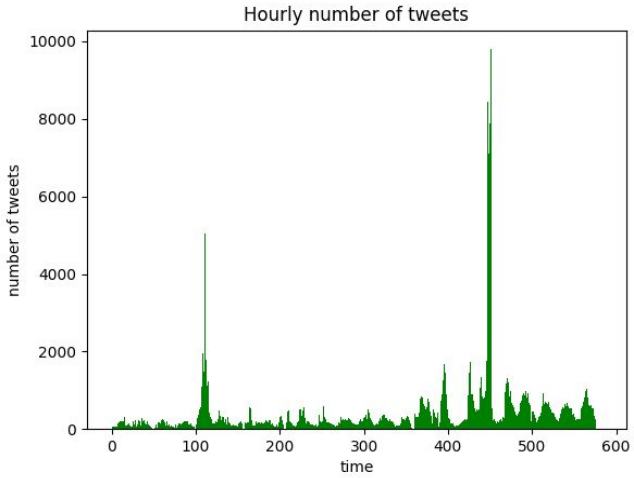


Figure 1.2.1: Number of tweets in hr (NFL)

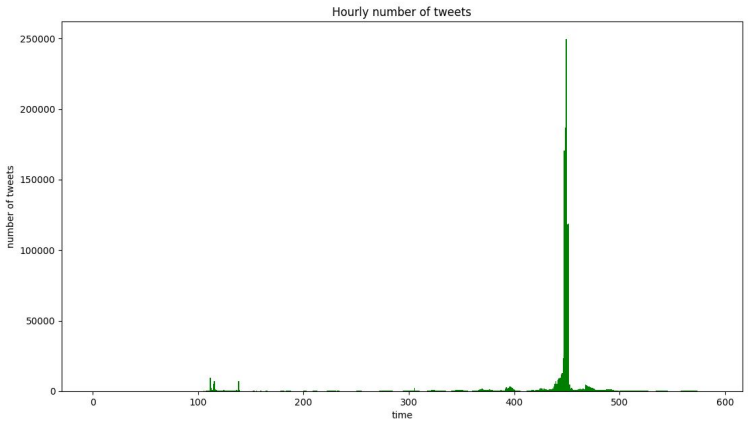


Figure 1.2.2: Number of tweets in hr (SuperBowl)

Linear Regression

Q3. Analysis of Models

Here we train a linear regression model where we try to predict the number of tweets in an hour based on the features of tweets in the previous hour. The following five features are used to train the model.

- 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

We train one regression model for each of the six hashtag files.

Below we report the Mean Squared Error (MSE) and R-squared measure for each model.

Hashtags	MSE	R-squared
gohawks	766490	0.476
gopatriots	27880	0.629
nfl	272751	0.570
patriots	5234485	0.668
sb49	16358652	0.804
superbowl	53026404	0.799

We also report the statics for training the regressor from library statsmodels.

'Current Hashtag: ', 'tweets_#gohawks.txt'

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          no_of_tweets   R-squared:                       0.476
Model:                            OLS   Adj. R-squared:                  0.472
Method:                 Least Squares   F-statistic:                     104.1
Date:                Thu, 21 Mar 2019   Prob (F-statistic):           4.98e-78
Time:                        13:03:12   Log-Likelihood:                -4733.0
No. Observations:                 578   AIC:                             9478.
Df Residuals:                     572   BIC:                             9504.
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         93.8768     70.004      1.341      0.180     -43.619     231.373
hours          1.7403      5.299      0.328      0.743      -8.667      12.147
followers_sum    -0.0002      8e-05     -2.427      0.016      -0.000    -3.7e-05
followers_max  6.106e-05      0.000      0.410      0.682      -0.000       0.000
no_of_tweets     1.2825      0.164      7.829      0.000       0.961       1.604
total_retweets    -0.1365      0.043     -3.140      0.002      -0.222      -0.051
==============================================================================
Omnibus:                      916.643   Durbin-Watson:                   2.216
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           783056.471
Skew:                           8.691   Prob(JB):                         0.00
Kurtosis:                     182.478   Cond. No.                     5.09e+06
==============================================================================
```

'Current Hashtag: ', 'tweets_#gopatriots.txt'

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          no_of_tweets   R-squared:                       0.629
Model:                            OLS   Adj. R-squared:                  0.626
Method:                 Least Squares   F-statistic:                     192.9
Date:                Thu, 21 Mar 2019   Prob (F-statistic):          7.06e-120
```

Time: 13:03:12 Log-Likelihood: -3749.1
No. Observations: 574 AIC: 7510.
Df Residuals: 568 BIC: 7536.
Df Model: 5
Covariance Type: nonrobust

	coef	std err	t	P>|t|	[0.025	0.975]
const	6.0229	10.869	0.554	0.580	-15.325	27.371
hours	0.1144	0.938	0.122	0.903	-1.728	1.956
followers_sum	-0.0001	0.000	-0.504	0.614	-0.001	0.000
followers_max	-2.36e-05	0.000	-0.108	0.914	-0.000	0.000
no_of_tweets	0.3073	0.285	1.079	0.281	-0.252	0.866
total_retweets	0.4892	0.192	2.550	0.011	0.112	0.866

Omnibus: 485.015 Durbin-Watson: 1.908
Prob(Omnibus): 0.000 Jarque-Bera (JB): 290846.979
Skew: 2.515 Prob(JB): 0.00
Kurtosis: 113.161 Cond. No. 6.00e+05

'Current Hashtag: ', 'tweets_#nfl.txt'

OLS Regression Results
Dep. Variable: no_of_tweets R-squared: 0.571
Model: OLS Adj. R-squared: 0.567
Method: Least Squares F-statistic: 154.3
Date: Thu, 21 Mar 2019 Prob (F-statistic): 4.54e-104
Time: 13:03:12 Log-Likelihood: -4495.8
No. Observations: 586 AIC: 9004.
Df Residuals: 580 BIC: 9030.
Df Model: 5
Covariance Type: nonrobust

	coef	std err	t	P>|t|	[0.025	0.975]
const	123.0774	42.635	2.887	0.004	39.339	206.816
hours	0.4876	3.139	0.155	0.877	-5.678	6.653

followers_sum	0.0001	2.5e-05	4.580	0.000	6.54e-05	0.000
followers_max	-0.0001	3.31e-05	-3.526	0.000	-0.000	-5.17e-05
no_of_tweets	0.5669	0.135	4.201	0.000	0.302	0.832
total_retweets	-0.1654	0.064	-2.593	0.010	-0.291	-0.040

Omnibus:	670.119	Durbin-Watson:	2.373
Prob(Omnibus):	0.000	Jarque-Bera (JB):	350887.669
Skew:	4.596	Prob(JB):	0.00
Kurtosis:	122.525	Cond. No.	8.55e+06

'Current Hashtag: ', 'tweets_#patriots.txt'

OLS Regression Results

Dep. Variable:	no_of_tweets	R-squared:	0.668
Model:	OLS	Adj. R-squared:	0.666
Method:	Least Squares	F-statistic:	233.8
Date:	Thu, 21 Mar 2019	Prob (F-statistic):	1.91e-136
Time:	13:03:12	Log-Likelihood:	-5361.4
No. Observations:	586	AIC:	1.073e+04
Df Residuals:	580	BIC:	1.076e+04
Df Model:	5		
Covariance Type:	nonrobust		

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

Omnibus:	887.682	Durbin-Watson:	1.998
Prob(Omnibus):	0.000	Jarque-Bera (JB):	690539.222
Skew:	7.937	Prob(JB):	0.00
Kurtosis:	170.420	Cond. No.	1.60e+07

==========
==========

'Current Hashtag: ', 'tweets_#sb49.txt'

OLS Regression Results

==========
==========

Dep. Variable:	no_of_tweets	R-squared:	0.805
Model:	OLS	Adj. R-squared:	0.803
Method:	Least Squares	F-statistic:	474.3
Date:	Thu, 21 Mar 2019	Prob (F-statistic):	1.71e-201
Time:	13:03:12	Log-Likelihood:	-5656.4
No. Observations:	582	AIC:	1.132e+04
Df Residuals:	576	BIC:	1.135e+04
Df Model:	5		
Covariance Type:	nonrobust		

==========
==========

	coef	std err	t	P>|t|	[0.025	0.975]
const	174.8572	302.524	0.578	0.563	-419.328	769.042
hours	-14.2385	23.398	-0.609	0.543	-60.195	31.718
followers_sum	9.742e-06	1.25e-05	0.779	0.437	-1.48e-05	3.43e-05
followers_max	9.538e-05	4.38e-05	2.177	0.030	9.33e-06	0.000
no_of_tweets	1.1364	0.087	13.019	0.000	0.965	1.308
total_retweets	-0.1606	0.079	-2.041	0.042	-0.315	-0.006

==========
==========

Omnibus:	1179.266	Durbin-Watson:	1.673
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2205322.879
Skew:	14.582	Prob(JB):	0.00
Kurtosis:	303.151	Cond. No.	1.44e+08

==========
==========

'Current Hashtag: ', 'tweets_#superbowl.txt'

OLS Regression Results

==========
==========

Dep. Variable:	no_of_tweets	R-squared:	0.800
Model:	OLS	Adj. R-squared:	0.798
Method:	Least Squares	F-statistic:	463.5

```
Date:               Thu, 21 Mar 2019   Prob (F-statistic):         6.72e-200
Time:                       13:03:12   Log-Likelihood:                -6039.9
No. Observations:                586   AIC:                         1.209e+04
Df Residuals:                    580   BIC:                         1.212e+04
Df Model:                          5
Covariance Type:           nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       -149.5572    605.382     -0.247      0.805   -1338.565    1039.451
hours        -20.4965     43.624     -0.470      0.639    -106.177      65.184
followers_sum  -0.0001    2.2e-05     -6.265      0.000      -0.000   -9.47e-05
followers_max   0.0007      0.000      4.889      0.000       0.000       0.001
no_of_tweets    2.2766      0.080     28.537      0.000       2.120       2.433
total_retweets -0.2543      0.046     -5.544      0.000      -0.344      -0.164
==============================================================================
Omnibus:                      973.862   Durbin-Watson:                   2.283
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1787388.254
Skew:                           9.272   Prob(JB):                         0.00
Kurtosis:                     272.925   Cond. No.                     2.21e+08
==============================================================================
```

Significance Analysis of features

From the above statistics, we see that number of tweets and total retweets are the features with least p values (their p values are close to 0 always) and thus they are more significant and helps the regressor. Number of followers helped in regression for some files and not for others while the other features like time of the day and maximum number of followers had a high significance value and thus not much help to the regressor.

Feature Analysis

Q4. Regression Model

Here we again train a regression model that predicts the number of tweets in an hour by using features from previous hours as we did in Q3. But we use only the features that proved useful in Q3 and also some new features that we think can be of help to the regressor.

We added 5 new features with the intention that it will improve the regression process.

1. Number of hashtags: We sum total hashtags for each time period from each tweet. This gives information about how popular or widespread the tweets are.
2. Number of replies: We sum over replies for each tweet that measures the involvement due to the tweet
3. Number of urls: This tells something about the amount of information that the tweet contains
4. Number of user mentions: Again user mentions tells about the engagement that the tweet is causing
5. Total ranking scores: We sum the ranking scores for each tweet over the time period

In addition to these new features we keep the following old features which turned out to be significant in the previous questions.

1. Number of tweets
2. Total retweets

We also add hours as a feature for convenience.

The MSE and other statistics summary for each file are as follows:

Current Hashtag: '', 'tweets_#gohawks.txt'

OLS Regression Results

Dep. Variable:	no_of_tweets	R-squared:	0.601
Model:	OLS	Adj. R-squared:	0.595
Method:	Least Squares	F-statistic:	107.0
Date:	Thu, 21 Mar 2019	Prob (F-statistic):	3.43e-108
Time:	21:10:56	Log-Likelihood:	-4654.8
No. Observations:	578	AIC:	9328.
Df Residuals:	569	BIC:	9367.
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-67.4501	62.497	-1.079	0.281	-190.203	55.303
hours	1.9277	4.681	0.412	0.681	-7.267	11.122
ranking_scores_sum	8.0613	0.904	8.914	0.000	6.285	9.837
hashtags_sum	0.6962	0.357	1.952	0.051	-0.004	1.397
user_mentions_sum	1.7260	0.513	3.362	0.001	0.718	2.734
no_of_tweets	-39.2020	4.335	-9.042	0.000	-47.717	-30.687

total_retweets	0.0078	0.045	0.174	0.862	-0.081	0.096
urls_sum	7.3642	1.576	4.673	0.000	4.269	10.460
replies_sum	-5.3335	7.036	-0.758	0.449	-19.154	8.487

Omnibus:	1021.037	Durbin-Watson:	2.309
Prob(Omnibus):	0.000	Jarque-Bera (JB):	954130.400
Skew:	10.963	Prob(JB):	0.00
Kurtosis:	200.831	Cond. No.	1.32e+04

MSE : 587865.076627
R-squared : 0.600593030562

Current Hashtag: '', 'tweets_#gopatriots.txt'

OLS Regression Results

Dep. Variable:	no_of_tweets	R-squared:	0.868
Model:	OLS	Adj. R-squared:	0.866
Method:	Least Squares	F-statistic:	465.6
Date:	Thu, 21 Mar 2019	Prob (F-statistic):	4.94e-243
Time:	21:10:56	Log-Likelihood:	-3452.1
No. Observations:	574	AIC:	6922.
Df Residuals:	565	BIC:	6961.
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-4.0651	6.506	-0.625	0.532	-16.844	8.714
hours	0.4525	0.572	0.792	0.429	-0.670	1.575
ranking_scores_sum	6.6307	0.310	21.397	0.000	6.022	7.239
hashtags_sum	1.8992	0.352	5.391	0.000	1.207	2.591
user_mentions_sum	4.1539	0.423	9.809	0.000	3.322	4.986
no_of_tweets	-34.8612	1.834	-19.008	0.000	-38.463	-31.259
total_retweets	-1.2947	0.141	-9.157	0.000	-1.572	-1.017
urls_sum	8.0644	0.690	11.688	0.000	6.709	9.420
replies_sum	-22.9495	2.558	-8.972	0.000	-27.974	-17.925

Omnibus: 400.623 Durbin-Watson: 1.864
Prob(Omnibus): 0.000 Jarque-Bera (JB): 42158.443
Skew: 2.202 Prob(JB): 0.00
Kurtosis: 44.753 Cond. No. 2.37e+03
==========
==========

MSE : 9959.4856274
R-squared : 0.86828911305

Current Hashtag: '', 'tweets_#nfl.txt'

OLS Regression Results
==========
==========
Dep. Variable: no_of_tweets R-squared: 0.683
Model: OLS Adj. R-squared: 0.679
Method: Least Squares F-statistic: 155.5
Date: Thu, 21 Mar 2019 Prob (F-statistic): 1.26e-138
Time: 21:10:56 Log-Likelihood: -4406.9
No. Observations: 586 AIC: 8832.
Df Residuals: 577 BIC: 8871.
Df Model: 8
Covariance Type: nonrobust
==========
==========

	coef	std err	t	P>|t|	[0.025	0.975]
const	-35.4732	39.589	-0.896	0.371	-113.229	42.282
hours	-3.5700	2.728	-1.309	0.191	-8.927	1.787
ranking_scores_sum	0.2877	0.331	0.869	0.385	-0.362	0.938
hashtags_sum	1.1298	0.093	12.158	0.000	0.947	1.312
user_mentions_sum	1.7830	0.572	3.116	0.002	0.659	2.907
no_of_tweets	-4.6554	1.586	-2.935	0.003	-7.771	-1.540
total_retweets	-0.0872	0.060	-1.443	0.149	-0.206	0.031
urls_sum	1.2609	0.123	10.263	0.000	1.020	1.502
replies_sum	-2.7885	3.171	-0.879	0.380	-9.017	3.440

==========
==========
Omnibus: 699.273 Durbin-Watson: 2.386
Prob(Omnibus): 0.000 Jarque-Bera (JB): 151559.555
Skew: 5.354 Prob(JB): 0.00
Kurtosis: 81.055 Cond. No. 1.09e+04

==========
==========

MSE : 202416.534139
R-squared : 0.683185064191

'Current Hashtag: ', 'tweets_#patriots.txt'

OLS Regression Results

==========
==========

Dep. Variable:	no_of_tweets	R-squared:	0.813
Model:	OLS	Adj. R-squared:	0.811
Method:	Least Squares	F-statistic:	314.4
Date:	Thu, 21 Mar 2019	Prob (F-statistic):	9.77e-205
Time:	21:10:56	Log-Likelihood:	-5193.0
No. Observations:	586	AIC:	1.040e+04
Df Residuals:	577	BIC:	1.044e+04
Df Model:	8		
Covariance Type:	nonrobust		

==========
==========

	coef	std err	t	P>|t|	[0.025	0.975]
const	-246.5527	144.494	-1.706	0.088	-530.350	37.245
hours	-1.3707	10.393	-0.132	0.895	-21.783	19.041
ranking_scores_sum	12.3663	0.681	18.171	0.000	11.030	13.703
hashtags_sum	3.7903	0.350	10.828	0.000	3.103	4.478
user_mentions_sum	6.2347	0.826	7.546	0.000	4.612	7.857
no_of_tweets	-66.0268	3.454	-19.116	0.000	-72.811	-59.243
total_retweets	-0.2180	0.046	-4.772	0.000	-0.308	-0.128
urls_sum	-3.3721	1.585	-2.128	0.034	-6.484	-0.260
replies_sum	-12.6561	4.617	-2.741	0.006	-21.724	-3.588

==========
==========

Omnibus:	1063.549	Durbin-Watson:	1.782
Prob(Omnibus):	0.000	Jarque-Bera (JB):	1148612.803
Skew:	11.594	Prob(JB):	0.00
Kurtosis:	218.649	Cond. No.	4.36e+04

==========
==========

MSE : 2960845.43125
R-squared : 0.81341693005

Current Hashtag: '', 'tweets_#sb49.txt'

OLS Regression Results

Dep. Variable:	no_of_tweets	R-squared:	0.849
Model:	OLS	Adj. R-squared:	0.847
Method:	Least Squares	F-statistic:	403.2
Date:	Thu, 21 Mar 2019	Prob (F-statistic):	1.06e-229
Time:	21:10:56	Log-Likelihood:	-5581.0
No. Observations:	582	AIC:	1.118e+04
Df Residuals:	573	BIC:	1.122e+04
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-52.6197	268.549	-0.196	0.845	-580.079	474.840
hours	3.5606	20.737	0.172	0.864	-37.170	44.291
ranking_scores_sum	18.9183	1.832	10.328	0.000	15.320	22.516
hashtags_sum	2.6363	0.337	7.823	0.000	1.974	3.298
user_mentions_sum	3.4574	0.537	6.439	0.000	2.403	4.512
no_of_tweets	-96.0500	9.074	-10.585	0.000	-113.872	-78.228
total_retweets	0.8182	0.090	9.077	0.000	0.641	0.995
urls_sum	6.5131	1.249	5.216	0.000	4.061	8.966
replies_sum	-48.3097	6.452	-7.488	0.000	-60.981	-35.638

Omnibus:	1043.350	Durbin-Watson:	2.156
Prob(Omnibus):	0.000	Jarque-Bera (JB):	934574.556
Skew:	11.368	Prob(JB):	0.00
Kurtosis:	197.993	Cond. No.	9.20e+04

MSE : 12691514.991
R-squared : 0.849170205296

'Current Hashtag: ", 'tweets_#superbowl.txt'

OLS Regression Results

Dep. Variable:	no_of_tweets	R-squared:	0.886
Model:	OLS	Adj. R-squared:	0.885
Method:	Least Squares	F-statistic:	563.0
Date:	Thu, 21 Mar 2019	Prob (F-statistic):	7.76e-267
Time:	21:10:56	Log-Likelihood:	-5873.8
No. Observations:	586	AIC:	1.177e+04
Df Residuals:	577	BIC:	1.180e+04
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-537.8380	453.439	-1.186	0.236	-1428.431	352.755
hours	3.3129	33.169	0.100	0.920	-61.833	68.459
ranking_scores_sum	8.0032	1.286	6.221	0.000	5.476	10.530
hashtags_sum	3.4390	0.423	8.136	0.000	2.609	4.269
user_mentions_sum	3.8787	1.582	2.452	0.015	0.772	6.986
no_of_tweets	-41.4167	6.226	-6.652	0.000	-53.645	-29.188
total_retweets	-0.7175	0.052	-13.801	0.000	-0.820	-0.615
urls_sum	-1.9741	1.082	-1.825	0.069	-4.099	0.151
replies_sum	-102.5591	21.147	-4.850	0.000	-144.093	-61.025

Omnibus:	1072.396	Durbin-Watson:	1.864
Prob(Omnibus):	0.000	Jarque-Bera (JB):	1439266.959
Skew:	11.692	Prob(JB):	0.00
Kurtosis:	244.659	Cond. No.	1.85e+05

MSE : 30241396.4295
R-squared : 0.886437706138

We look at the p-values for each feature over each file as follows to infer about the usefulness of the features:

P-values

Features: ['hours','ranking_scores_sum', 'hashtags_sum', 'user_mentions_sum', 'no_of_tweets', 'total_retweets', 'urls_sum', 'replies_sum']

('Current Hashtag: ', 'tweets_#gohawks.txt')
['0.280931', '0.680648', '0.000000', '0.051454', '0.000826', '0.000000', '0.861927', '0.000004', '0.448767']
('Current Hashtag: ', 'tweets_#gopatriots.txt')
['0.532328', '0.428836', '0.000000', '0.000000', '0.000000', '0.000000', '0.000000', '0.000000', '0.000000']
('Current Hashtag: ', 'tweets_#nfl.txt')
['0.370603', '0.191093', '0.385001', '0.000000', '0.001926', '0.003471', '0.149465', '0.000000', '0.379557']
('Current Hashtag: ', 'tweets_#patriots.txt')
['0.088486', '0.895114', '0.000000', '0.000000', '0.000000', '0.000000', '0.000002', '0.033745', '0.006311']
('Current Hashtag: ', 'tweets_#sb49.txt')
['0.844726', '0.863733', '0.000000', '0.000000', '0.000000', '0.000000', '0.000000', '0.000000', '0.000000']
('Current Hashtag: ', 'tweets_#superbowl.txt')
['0.236059', '0.920475', '0.000000', '0.000000', '0.014506', '0.000000', '0.000000', '0.068541', '0.000002']

From this p-values data we find that the following features have p-scores very close to zero in almost all cases:

- 'Hashtags_sum'
- 'User_mentions_sum'
- 'No_of_tweets'
- 'urls_sum'

Thus we can say that total number of hashtags is a useful feature. This also makes sense as the number of hashtags in a tweet describes something about it's scope and/or spread of the topic it is describing. We can also say that user mentions can be justified since the number of mentions on a tweet tells about the how many people are interested in it. Regarding no of tweets it is obvious that the number of tweets in an hour before will tell the most about the activity that is taking place in that hour and also about the possible activity that can take place in the next hour. For number of urls we can say that more the number of urls in a tweet, more is the content that is covered in it and more is the possibility that people get engaged by it. We can also look at it in a different way and say that more urls implies more activity of the users.

Q5. Top 3 features

From analysis in Q4, we get the top three features with least p values as

- number of hashtag
- number of user mentions
- number of tweets

Next we draw a scatter plot of predicant vs feature values for each hashtags and the top 3 feat features.

Patriots

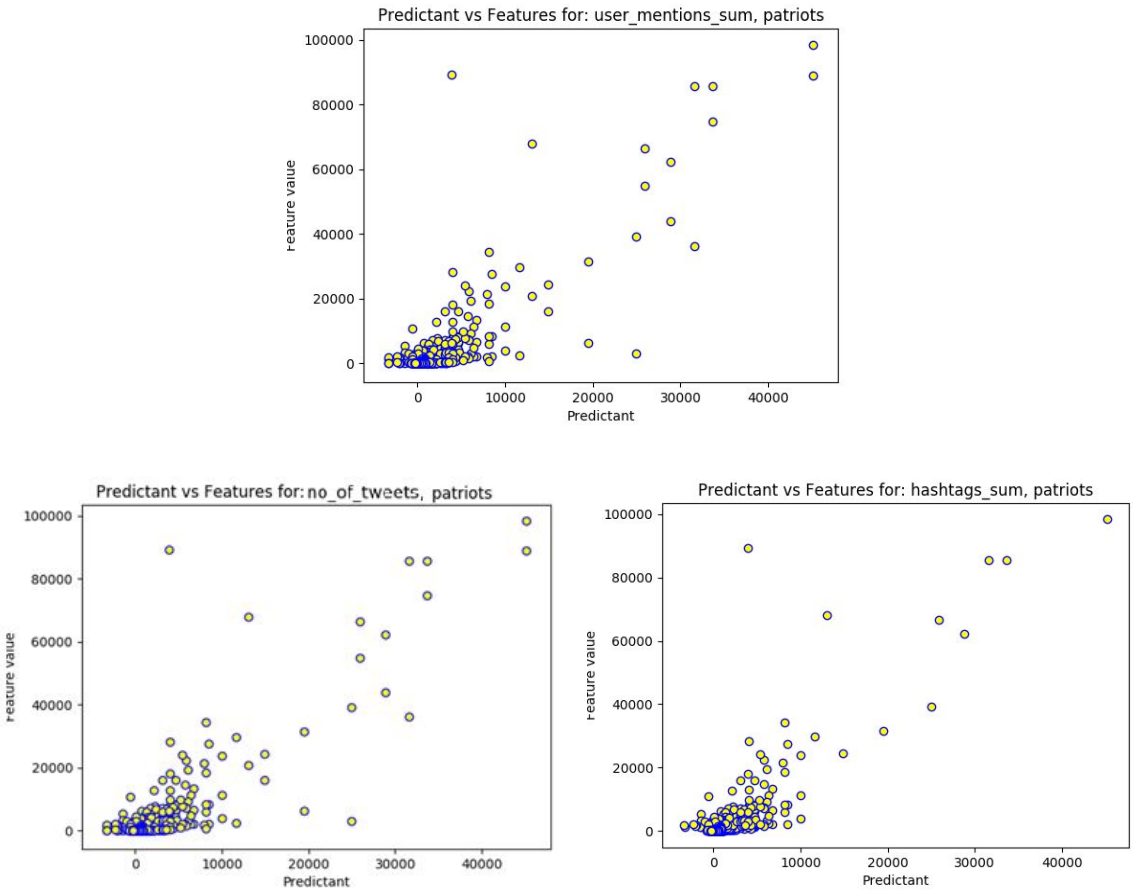


Figure 1.5.2: Predictant vs feature values for number of hashtags, tweets and user_mentions (Patriots)

NFL

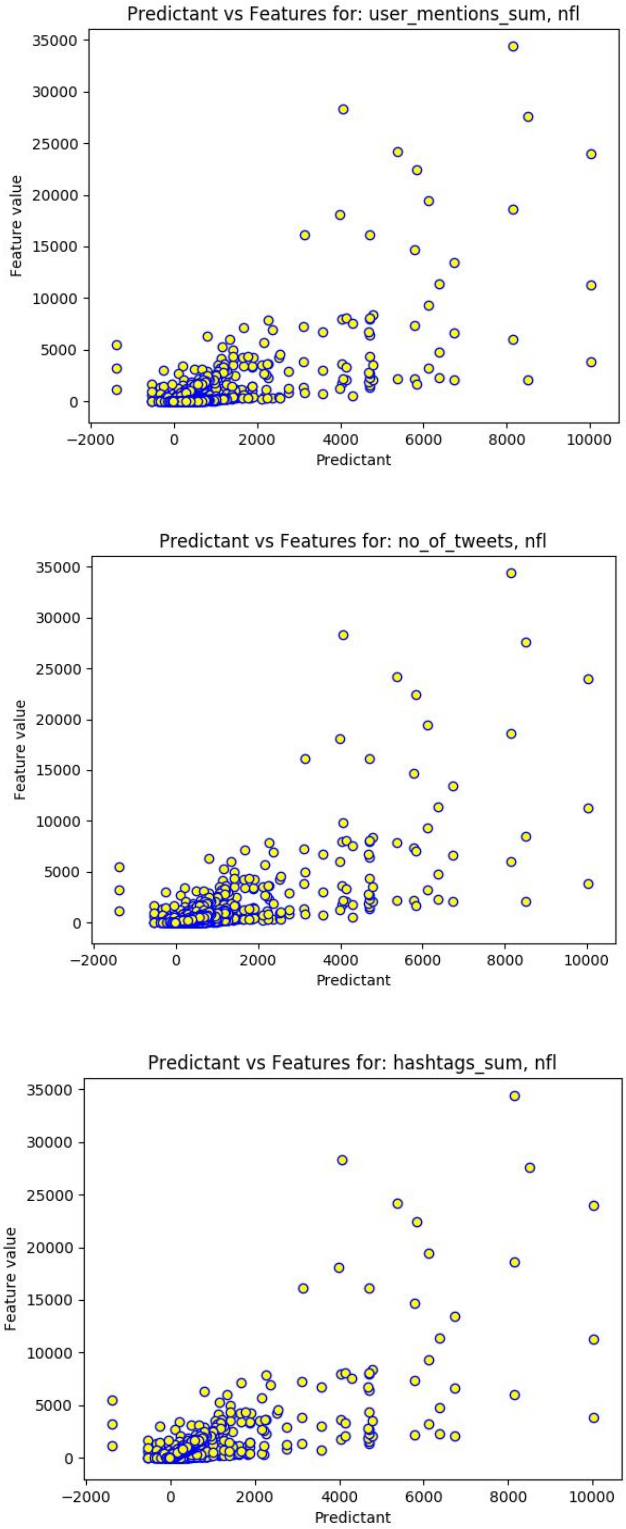


Figure 1.5.2: Predictant vs feature values for number of hashtags, tweets and user_mentions (NFL)

GoPatriots

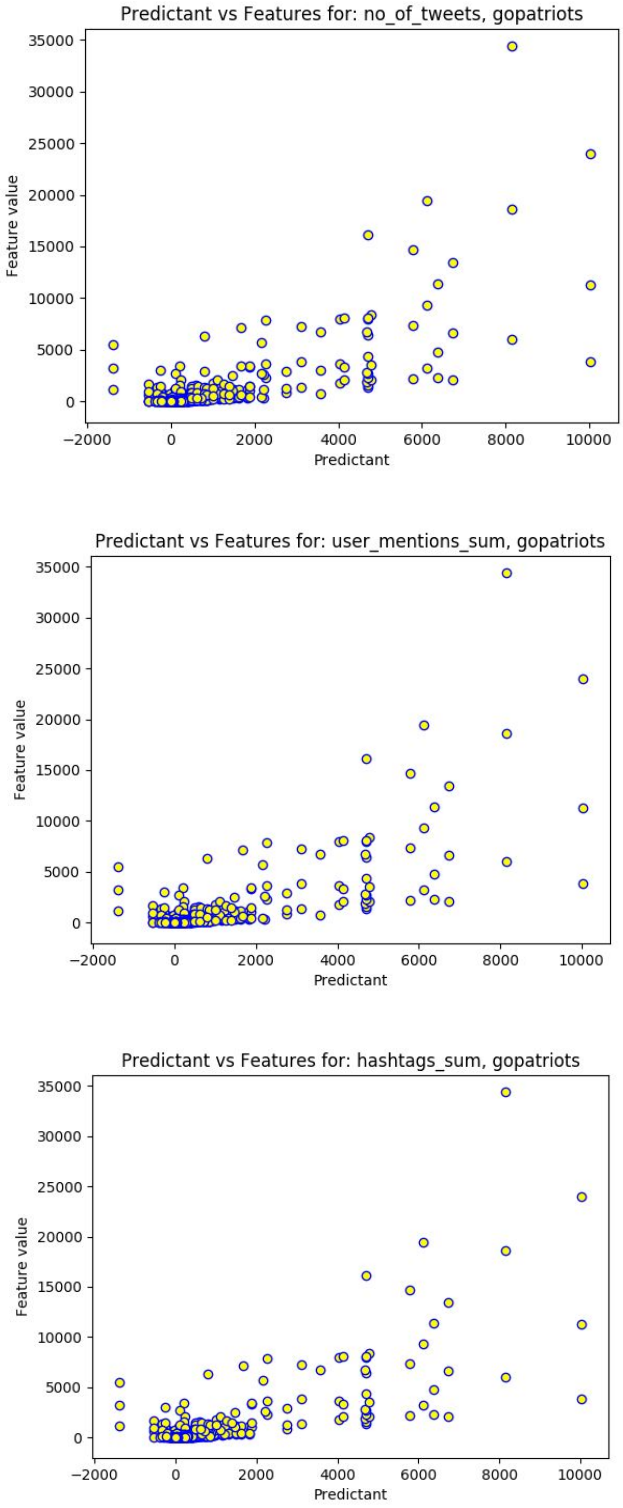


Figure 1.5.3: Predictant vs feature values for number of hashtags, tweets and user_mentions go_partriots

Superbowl

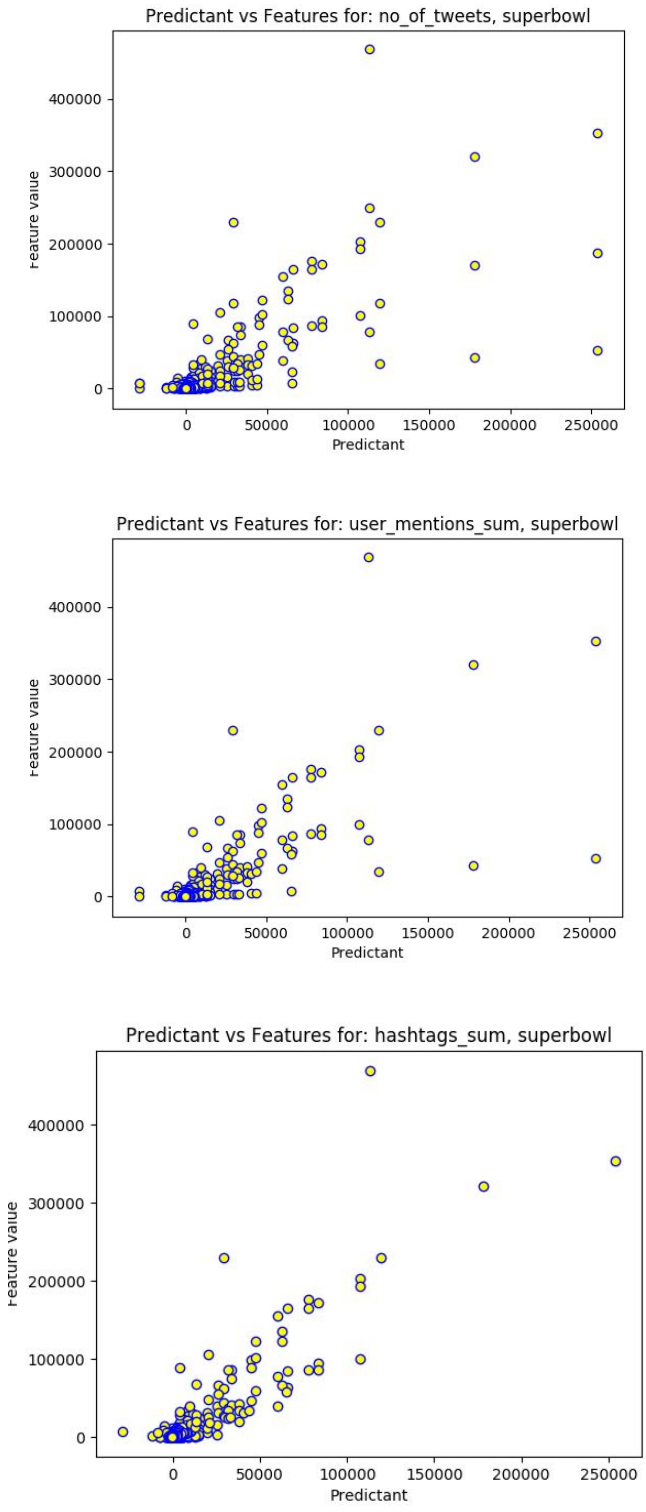


Figure 1.5.4: Predictant vs feature values for number of hashtags, tweets and user_mentions (SuperBowls)

Sb49

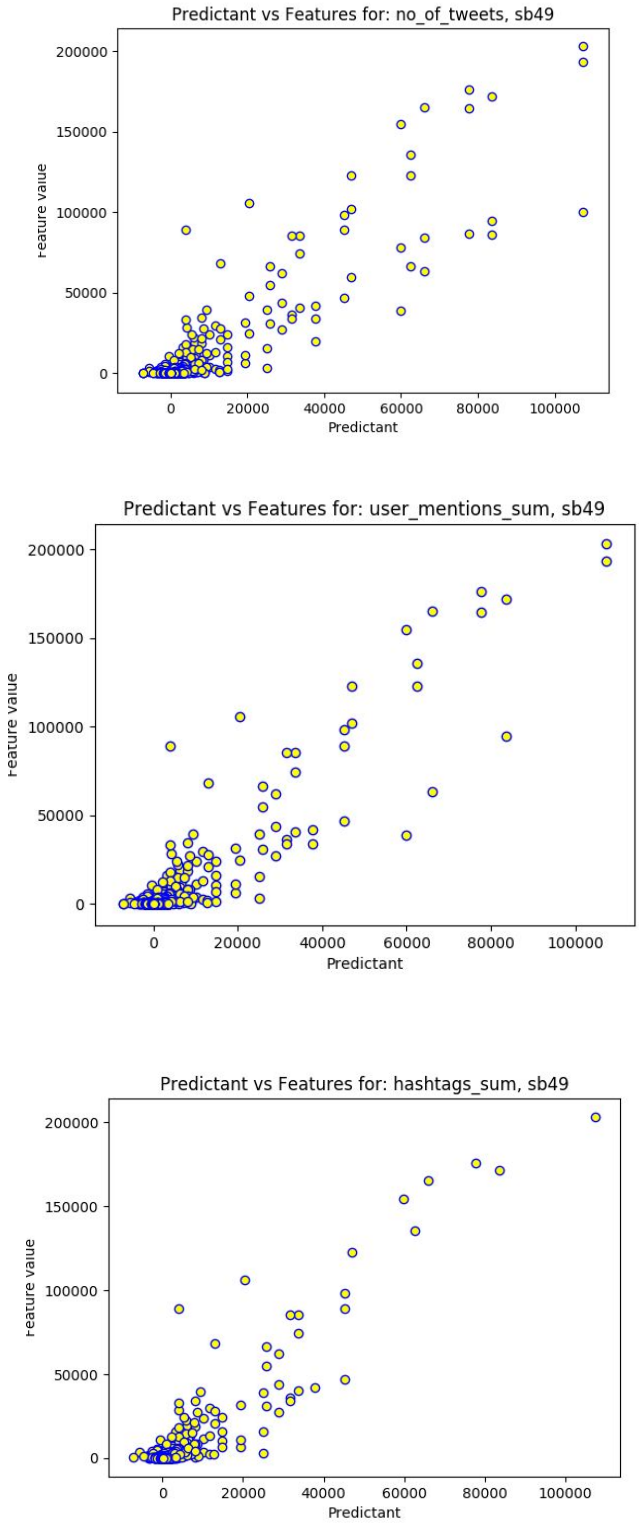
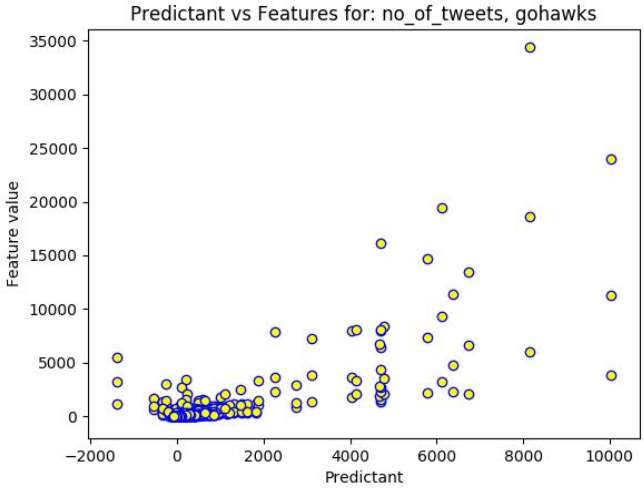
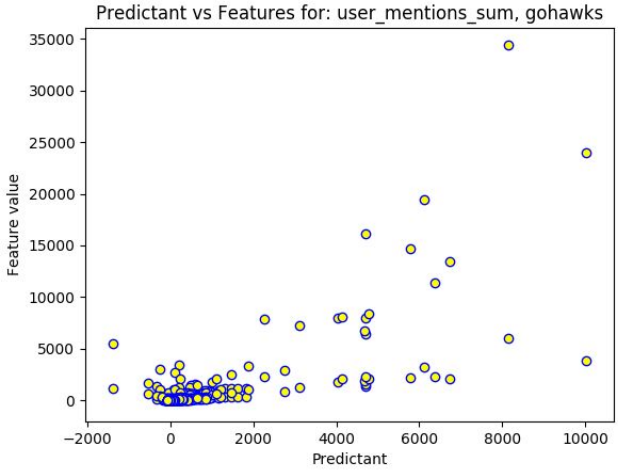


Figure 1.5.5: Predictant vs feature values for number of hashtags, tweets and user_mentions(sb49)





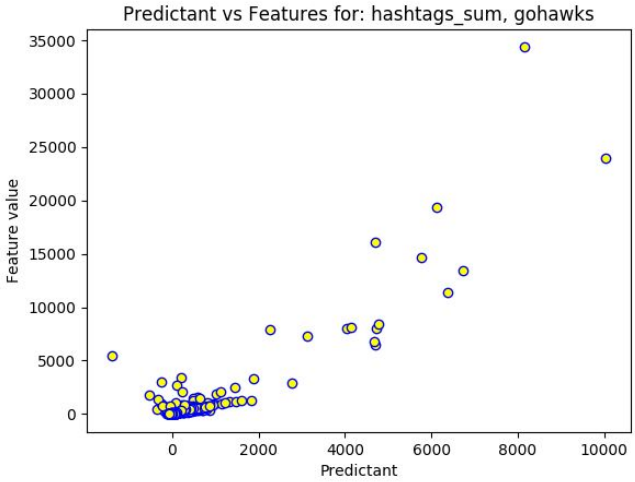


Figure 1.5.6: Predictant vs feature values for number of hashtags, tweets and user_mentions (gohawks)

As we see from the above scatter plots, for each of the files, every feature almost fits with the values of predictant. This explains why these three features had lesser p values, since a line can be drawn to represent relation between the values of the plot and the slope of the line could be viewed as the weights of the features in regression. So we can say that the regression coefficients agree with the trends in the plot. Also we see that the fitting is better for Superbowls and sb49 compared to the others.

Piecewise Linear Regression

Q6. Regression on Time Periods for Each Hashtag

We divided our data in three time periods and window sizes:

1. Before Feb. 1, 8:00 a.m. with 1-hour window (Referred as TP1 in Table)
2. Between Feb. 1, 8:00 a.m. and 8:00 p.m. with 5-minute window (Referred as TP2 in Table)
3. After Feb. 1, 8:00 p.m. with 1-hour window (Referred as TP1 in Table)

We, then, used 10-fold cross validation to train and test our data using Linear Regression model. The outcomes are noted in the table below.

Measures obtained using OLS			
Hashtag	**Time Pd.**	**MSE**	**R2 Score**
#gohawks	TP1	728194.2920771493	0.3033004780550576 7
	TP2	77548.48163358681	0.4763044683405422
	TP3	1956.4497852878683	0.8315986336579579
#gopatriots	TP1	1772.7115645004142	0.5736103278744 42
	TP2	15316.552549938042	0.4286701338151852 6
	TP3	47.270075356513566	0.7454261245537939

#nfl	TP1	66832.61787215127	0.5117006851825123
	TP2	22618.727103423305	0.8124150936761714
	TP3	17406.081440494963	0.8060917903895648
#patriots	TP1	342862.3252091237	0.5671551897037115
	TP2	730407.1176304751	0.6981141486433596
	TP3	11650.804579675618	0.8762553872117086
#sb49	TP1	7001.190231438968	0.8676140764426818
	TP2	1359322.3835582242	0.8644737604577095
	TP3	76934.31783248139	0.7995613101434234
#superbowl	TP1	523008.50278626056	0.4021952358270693
	TP2	7169481.433046755	0.8900038530677978
	TP3	117813.98637606586	0.8418057152511749

We also used Linear Regression with 10-fold cross validation and checked the results.

Measures obtained using Linear Regression with 10 Fold CV					
Hashtag	**Time Pd.**	**Training MSE**	**Testing MSE**	**Training R2 Score**	**Testing R2 Score**
#gohawks	TP1	353720.76951350085	490879.8854072023	0.19835413057233087	-2.8978110789800384
	TP2	36733.697519558846	44602.520786039604	0.2393202141505041	-0.7197316749758949

	TP3	922.9103248862223	1971.8023568113974	0.3635634372384096	-7.8707880132526
#gopatriots	TP1	868.3394891959919	983.1691422727921	0.2776006810084986	-0.5929046167959601
	TP2	6995.063102772379	13991.074349419205	0.2254816499024905	-5.643504030984717
	TP3	22.170865866649695	4324.581912223975	0.3115291277182709	-19.936440312135694
#nfl	TP1	32811.651577875266	36068.64955603435	0.2672983757657822	0.16683956399174882
	TP2	10674.993140413284	14575.500141023504	0.4063457839713618 4	-1.0437587455077244
	TP3	8217.759542847973	9587.405458204541	0.4034555942958316	0.3699978098183217 5
#patriots	TP1	165964.39673990285	241996.8113790912	0.2628659344897205 7	-0.4682996073528487 6
	TP2	342337.4260956473	445122.64797789976	0.3431852589848171 7	-7.4890816285735 1
	TP3	5467.199642219915	92983.25106658487	0.4058859193450577 4	-0.2714626046321040 3
#sb49	TP1	3429.9273611615267	5222.932551777553	0.4072825950369468 6	-0.0305902935208431 27
	TP2	638509.4897081117	852317.9070372216	0.4256788507775940 4	-4.4280523692503 82
	TP3	35724.33960734388	70146.47764786413	0.4064424548671666	-2.2483643193764684
#superbowl	TP1	251728.43232911863	381865.6533577031	0.2184938142889141 2	-4.2198211023460 99
	TP2	3327882.121316061	12022411.692853898	0.4451221045098785 5	-0.9489408781148242
	TP3	53453.236645462595	94535.05692593768	0.4243119082005941 7	-0.0221956810968001

Q7. Regression on Time Periods over Aggregated Data

We aggregated the data from each time period over all hashtags. The outcomes are noted in the table below.

Measures obtained using OLS		
Time Pd.	**MSE**	**R2 Score**
TP1 (Model 1)	4487889.566901041	0.40968242537748556
TP2 (Model 2)	18230230.27577118	0.8466583238059244
TP3 (Model 3)	467315.2272885898	0.8612429686023604

The performance of these 3 combined models is compared with those of the individual hashtags needs to be compared. We use R2 score as it is a normalised version of MSE, hence, we can ignore the scale of data.
Now, we see based on R2 score in OLS, that:

- The combined Model 1 performs better than model 1 for #gohawks, comparable to model 1 for #superbowl and worse than all other individual TP 1 models except #patriots.
- The combined Model 2 performs better than all individual TP 2 except #sb49 and #superbowl.
- The combined Model 3 performs better than all except the individual TP 3 model for #patriots.

Overall, we see that combined models, tend to perform better.

Measures obtained using Linear Regression with 10 Fold CV				
Time Pd.	**Training MSE**	**Testing MSE**	**Training R2 Score**	**Testing R2 Score**
TP1 (Model 1)	2196879.501021846	2610293.0359374126	0.2353094180314835	-0.026023751174752086
TP2 (Model 2)	8524125.240707096	20090324.400135167	0.41880562960954804	-1.0868447715318426

TP3 (Model 3)	215572.88421701686	562053.1017601157	0.43154095608943377	0.13270433446614419

Using Linear Regression with 10 fold cross validation, we notice testing R2 score is negative in many cases, indicating our chosen model performs worse than the null hypothesis in both individual and combined models.

Now, we see based on training R2 score, that overall our combined model perform better than corresponding individual TP models. On considering testing R2 score:

- The combined Model 1 performs better than all individual TP 1 models except #nfl.
- The combined Model 2 performs worse than #superbowl and #gohawks, comparable to #nfl and better than all others.
- The combined Model 3 performs better than all except #nfl.

Non-Linear Regression

Q8. Grid Search- Random Forest Regressor | Gradient Boosting Regressor (Using the hourly data)

The data with the aggregated hourly data created before was used for the Grid Search. We report the best model parameters we find for each and also the best 5 results of Random Forest Regressor and 5 best results of Gradient Boosting Regressor. Sorted according to the negative mean squared error.

The test errors from the model do not look good. There are various reasons that we feel lead to these results. First is that the total data points for training the model are very less (~600). The data is divided according to different hours on different days. The day on which the tweets are more is really important. The days on the day of the match, and on previous and next days will always have more tweets rather than the other days. Hence this leads to inconsistent data points.
To get better performance we need more data over more days as well as divide the results over varying time periods.

Random Forest Regressor :

The best parameters are with max depth : 20, number of estimators as 400 and min leaf samples as 1. The absolute error is in the range of 10^8 reported below.

Random Forest Regressor : Top 5 Results		
Model Parameters	**Mean Test Squared Error (Neg)**	**Mean Train Squared Error (Neg)**
'max_depth': 20, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 400	-728148061.51	-80753070.74
'max_depth': 200, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 600	-728998088.59	-81305844.02
‘max_depth': 20, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 1000	-729124730.52	-83465583.94
'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 400	-729537447.01	-83859476.17
'max_depth': 20, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 1800	-729694662.45	-91548153.64

Gradient Boosting Regressor :

The best parameters are with max depth : 40, number of estimators as 1800 and min leaf samples as 1. The absolute error is in the range of 10^8 reported below.

Random Forest Regressor : Top 5 Results		
Model Parameters	**Mean Test Squared Error (Neg)**	**Mean Train Squared Error (Neg)**
'max_depth': 40, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 1800	-727076900.45	-9.977583250033469e-08
'max_depth': 20, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 1400	-732628591.91	-9.965143036371504e-08
'max_depth': 60, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 1800	-733556750.06	-9.92155893834781e-08

‘max_depth': 200, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 10, 'n_estimators': 1800	-733939633.10	-9.978589130841706e-08
'max_depth': 60, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 1600	-734518115.00	-9.903719631249428e-08

Q9. Grid Search vs OLS

The results of best Grid Search result and OLS are mentioned below.
Negative test mean squared errors are reported for both :

OLS : -137866122.90
Grid Search : -727076900.45

Q10. Grid Search- Using the data from Q6

The results of Grid Search on the data created in Q6 are mentioned below. We ran all the three datasets created for different window lengths. We report the top 3 from each of the three aggregated datasets.

Yes, the cross validation has certainly improved. The test error has reduced by a factor of 100. With the train error with see that there is a lot of over-fitting in these models. With some more features in the data, we can hope to have better test results as well.

The reason for this is that during the actual game time on Feb 1, there were a lot more tweets and hence that data is treated separately.
Now we have segregated data according to pre post and during the match. So the model understands the conditions accordingly.

Gradient Boosting Regressor			
Data Set	**Model Parameters**	**Mean Test Squared Error (Neg)**	**Mean Train Squared Error (Neg)**
Before Feb 1, 8 am	'max_depth': 40, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 1600	-4660179.79	-9.938205535718068e-08

Before Feb 1, 8 am	'max_depth': None, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 600	-4685859.71	-1.5343854744152312e-07
Between Feb 1, 8 am to Feb 1, 8 pm	'max_depth': 40, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 1200	-26624709.77	-9.851394582230266e-084
Between Feb 1, 8 am to Feb 1, 8 om	'max_depth': 60, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 400	-26791090.39	-0.0488
After Feb 1, 8 pm	'max_depth': 100, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200	-1267615.53	-675.80
After Feb 1, 8 pm	'max_depth': 20, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200	-1267993.70	-675.81

Neural Network

Q11. Using MLPRegressor

The results of Grid Search with MLPRegressor are as below. The different architectures that we used are as follows.

```
{
'MLPRegressor': {'hidden_layer_sizes': [(1,),(2,),(5,),
           (10,),(20,),(25,),(30,),(40,),(50,),(75,),
           (100,),(200,),(250,),(500,),
           (1,1,),(2,2,),(5,5,),(10,10,),
           (10,10,10,),
           (10,10,10,10,),
           (100,100,100,100,100,100,100,100,),
          ],
    'activation': ['relu','tanh'],
    'solver': ['lbfgs','adam'],
    'learning_rate': ['constant','adaptive'],
    }, }
```

Figure 11.1 : Architecture for using MLP Regressor : Varying hidden layers and sizes

MLP Regressor : Top 5 Results		
Model Parameters	**Mean Test Squared Error (Neg)**	**Mean Train Squared Error (Neg)**
{'activation': 'relu', 'hidden_layer_sizes': (40,), 'learning_rate': 'adaptive', 'solver': 'lbfgs'}	-281241850.55	-277039729.17
{'activation': 'relu', 'hidden_layer_sizes': (10, 10), 'learning_rate': 'constant', 'solver': 'lbfgs'}	-283866021.74	-286541970.53
{'activation': 'relu', 'hidden_layer_sizes': (10,), 'learning_rate': 'adaptive', 'solver': 'lbfgs'}	-295421290.07	-338266856.48
{'activation': 'relu', 'hidden_layer_sizes': (30,), 'learning_rate': 'adaptive', 'solver': 'lbfgs'}	-296231774.13	-284097874.58
{'activation': 'relu', 'hidden_layer_sizes': (25,), 'learning_rate': 'constant', 'solver': 'lbfgs'}	-300025802.80	-307008694.01

Q12. Scaling Data

We applied Standard Scalar before feeding to the MLPRegressor. The performance by using the standard scalar on the aggregated data and then using the best architecture above improves by a factor of 4.

The range remains the same but the absolute mean squared error reduces by a factor of 4. The negative test squared error comes out to be -210708958.89.

Q13. MLPRegressor : Aggregated Data of Q6

Reporting the best parameters (top 2) found using GridSearch on the three different time based data sets. We see that the best parameters do change, the mean test errors come in the range of 10^5 - 10^6. This gives slightly better results without scaling it.

MLP Regressor			
Data Set	**Model Parameters**	**Mean Test Squared Error (Neg)**	**Mean Train Squared Error (Neg)**
Before Feb 1, 8 am	'activation': 'tanh', 'hidden_layer_sizes': (500,), 'learning_rate': 'constant', 'solver': 'lbfgs'	-3782486.13	-2177728.13
Before Feb 1, 8 am	‘activation': 'tanh', 'hidden_layer_sizes': (500,), 'learning_rate': 'adaptive', 'solver': 'lbfgs'	-4019328.24	-2118259.67
Between Feb 1, 8 am to Feb 1, 8 pm	'activation': 'relu', 'hidden_layer_sizes': (2,), 'learning_rate': 'adaptive', 'solver': 'lbfgs'	-25172125.05	-16062286.60
Between Feb 1, 8 am to Feb 1, 8 om	'activation': 'relu', 'hidden_layer_sizes': (5,), 'learning_rate': 'constant', 'solver': 'lbfgs'	-26287181.80	-12066482.77
After Feb 1, 8 pm	'activation': 'relu', 'hidden_layer_sizes': (100, 100, 100, 100, 100, 100, 100, 100), 'learning_rate': 'adaptive', 'solver': 'adam'	-637025.77	-163170.98
After Feb 1, 8 pm	activation': 'relu', 'hidden_layer_sizes': (5,), 'learning_rate': 'constant', 'solver': 'lbfgs'	-776333.26	-162577.01

6x Window

Q14. Using new window to predict

For this question we have used Gradient Boosting Regressor, as out model for each time period since it performed the best as per the Grid Search we ran. For each time period, the parameters of the model were changed based on the best model obtained as per the last question. Here are the parameters of each model for each period -

Model for Period 1 -

```
'max_depth': 40,
'max_features': 'sqrt',
'min_samples_leaf': 2,
'min_samples_split': 5,
'n_estimators': 1600
```

Model for Period 2 -

```
'max_depth': 40,
'max_features': 'sqrt',
'min_samples_leaf': 2,
'min_samples_split': 5,
'n_estimators': 1200
```

Model for Period 3 -

```
'max_depth': 100,
'max_features': 'auto',
'min_samples_leaf': 4,
'min_samples_split': 10,
'n_estimators': 200
```

Following predictions and MSEs were obtained for each test file using Gradient Boosting Regressor -

- Model ==> Sample 0 - Period 1
  MSE = 253487.433446841

	True	Predictions
0	79.0	190.579017
1	94.0	468.853045
2	101.0	1042.471759
3	122.0	486.000107
4	120.0	429.204356

- Model ==> Sample 0 - Period 2
  MSE ==> 2527841.9165648757

	True	Predictions
0	3834.0	1492.401626
1	2258.0	1249.655084
2	1455.0	2504.161351
3	1235.0	3323.048155
4	1123.0	1946.821903

- Model ==> Sample 0 - Period 3
  MSE ==> 1355.3454273025704

	True	Predictions
0	48.0	16.740099
1	94.0	99.980466
2	45.0	29.337950
3	77.0	34.493310
4	87.0	26.076595

- Model ==> Sample 1 - Period 1
  MSE ==> 7327.851739978627

	True	Predictions
0	180.0	206.881594
1	202.0	252.499001
2	294.0	344.067538
3	555.0	730.083917
4	846.0	860.330149

- Model ==> Sample 1 - Period 2
  MSE ==> 427048.5505388725

	True	Predictions
0	995.0	465.485659
1	870.0	1629.977785
2	960.0	2037.696233
3	861.0	537.287393
4	903.0	797.776154

- Model ==> Sample 1 - Period 3
  MSE ==> 136363.8236369305

	True	Predictions
0	87.0	910.641290
1	43.0	0.016898
2	27.0	61.008605
3	44.0	23.651091
4	46.0	49.991551

- Model ==> Sample 2 - Period 1
  MSE ==> 24915.66885756762

	True	Predictions
0	141.0	268.412331
1	102.0	268.894218
2	144.0	111.456338
3	104.0	121.044483
4	61.0	342.320383

- Model ==> Sample 2 - Period 2
  MSE ==> 101503.79182439715

	True	Predictions
0	19.0	252.143541
1	25.0	252.902838
2	27.0	393.691856
3	29.0	393.691856
4	28.0	393.732523

- Model ==> Sample 2 - Period 3
  MSE ==> 136840.63104960407

	True	Predictions
0	90.0	46.447262
1	40.0	1.713887
2	58.0	83.053524
3	87.0	910.672199
4	43.0	0.846494

Part 2: Fan Base Prediction

Q15. Model based on Location

1. Location Determination

This is an interesting problem where we wish to predict the location of a tweet based on its content's textual context. We are constrained to a binary classification in this problem, namely between two locations - Washington and Massachusetts. There is also the added constraint that only those tweets be considered, which have #superbowl mentioned somewhere in them. Following are the steps and conditions which we have checked for so as to extract/mine the relevant data out of the corpus.

- Extract all tweets from all the training data, and iterate over them. Only go forward with the processing if the 'text' attribute contains #superbowl in it. (We use the lowercase form of the 'text' attribute to extract samples such as #SuPeRBOwl, etc).

- First check if any mention of Washington exists in the location attribute of the tweet. Any of the following conditions could be met for this purpose -

    - If ", WA" or ", wa" exist at the end of the location string.
    - If " WA" exists at the end of the location string.
    - If any case-form of "Seattle","Redmond","Kirkland" or "Bellevue" exist within the location string.
    - If "Washington" exists in the location string AND "DC" or "dc" or "D.C." or "d.c." do not exist in the location string.
    - If only "WA" is the location string.

- In case any of the above met, this sample is selected and added to the dataset as a tweet whose target variable is 'WA'. (which is to be used ahead).

- In the alternate case, check if any mention of Massachusetts exists in the location attribute of the tweet. Any of the following conditions could be met for this purpose -

    - If ", MA" or ", ma" exist at the end of the location string.
    - If " MA" exists at the end of the location string.
    - If any case-form of "Boston","Springfield" or "Gillette" exist within the location string.
    - If "Massachusetts" exists in the location string.
    - If only "MA" is the location string.

- In case any of the above is true, then add the tweet to the dataset, and make its target variable as 'MA'.

2. Binary Classifier

Before classification, the data was preprocessed to convert it into a more sound format, and we also performed dimensionality reduction on the data so as to decrease run times and only use the best features to perform the classification.
Since we need to use the textual context of the tweet to predict the location, it makes sense to create the count vector and eventually the TF-IDF (Term Frequency - Inverse Document Frequency) matrix, so as to obtain word level statistics of the data before performing best feature selection using SVD. The same was performed, and after SVD and a train-test split of 20% we were left with training data of size - .(27516, 50), and testing data size of (6879, 50).

The following three classifiers were used -

- Linear SVM (Using the LinearSVC python library with 100 iterations for ensuring convergence). We noticed that a harder margin provided better results during the testing phase. (The C parameter in LinearSVC was set to 0.1 to model a hard margin).

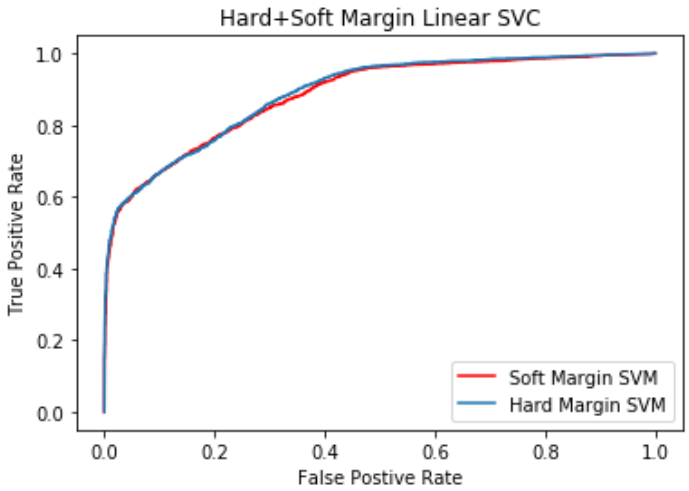
- Logistic Regression using the L2 Norm (logical fit for performing binary classification).

- Gaussian NB as our third choice and baseline model.

As predicted, we observed that Gaussian NB performs the worst. Linear SVM and Logistic Regression (L2) performed almost similarly, but Logistic Regression was our best model by a slight edge.

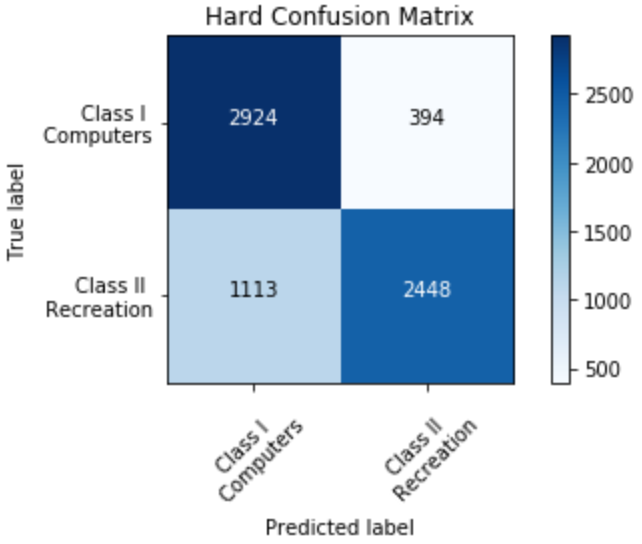
Here our are observations, i.e. our ROC curves, confusion matrices and our prediction statistics.

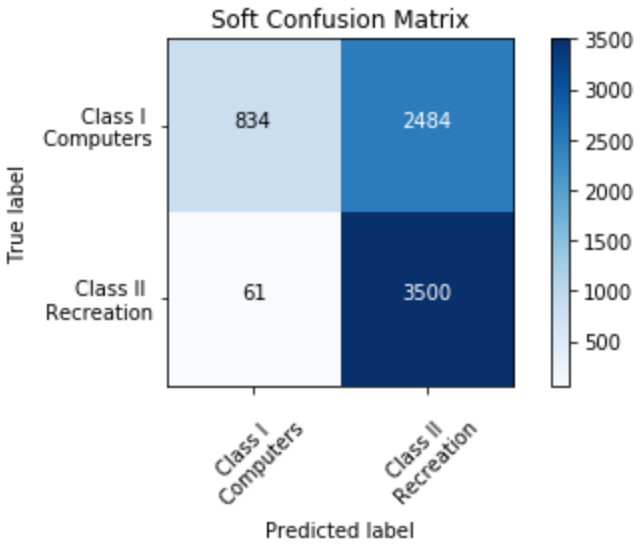
1. Linear SVM -



Area under hard margin ROC 0.8837378563125846

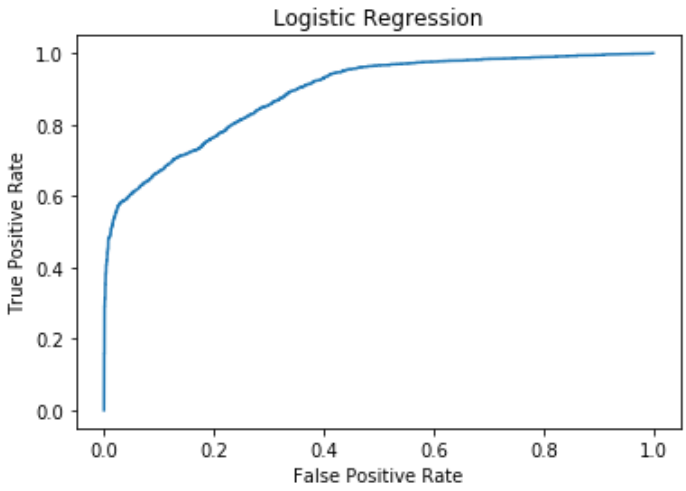
Area under soft margin ROC 0.8837378563125846



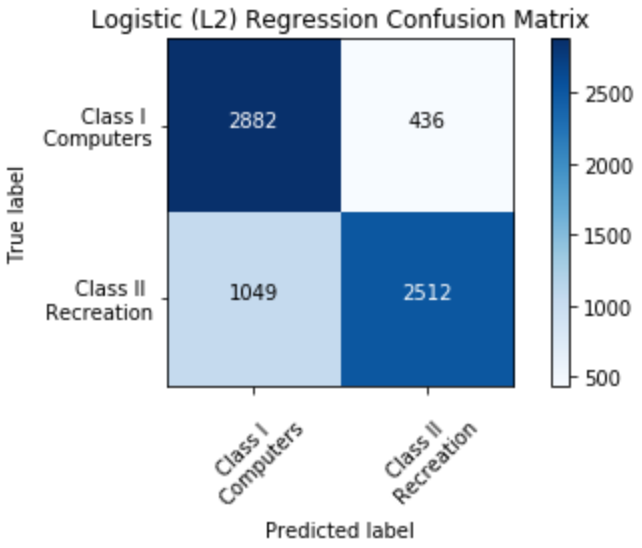


Hard SVM accuracy ==> **0.7809274603866841**
Soft SVM accuracy ==> **0.6300334350923099**
Hard SVM recall ==> **0.6874473462510531**
Soft SVM recall ==> **0.9828699803426004**
Hard SVM precision ==> **0.8613652357494722**
Soft SVM precision ==> **0.5848930481283422**
Hard SVM F1 score ==> **0.7646415742620647**
Soft SVM F1 score ==> **0.7333682556312204**

2. Logistic (L2) Regression -

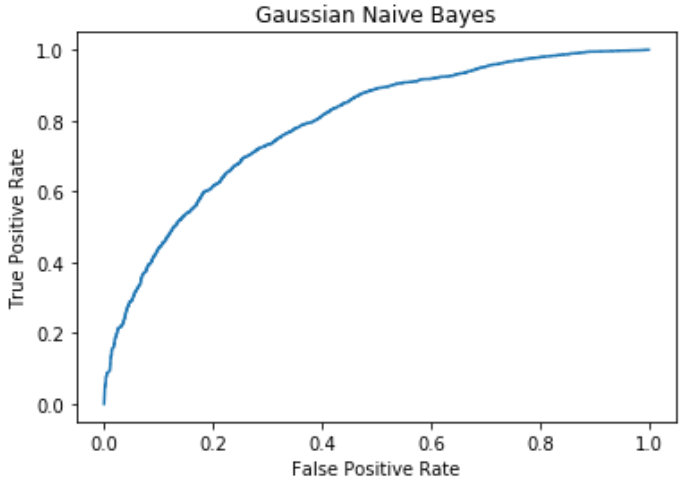


Area under the Logistic (L2) ROC curve 0.8892448227304742

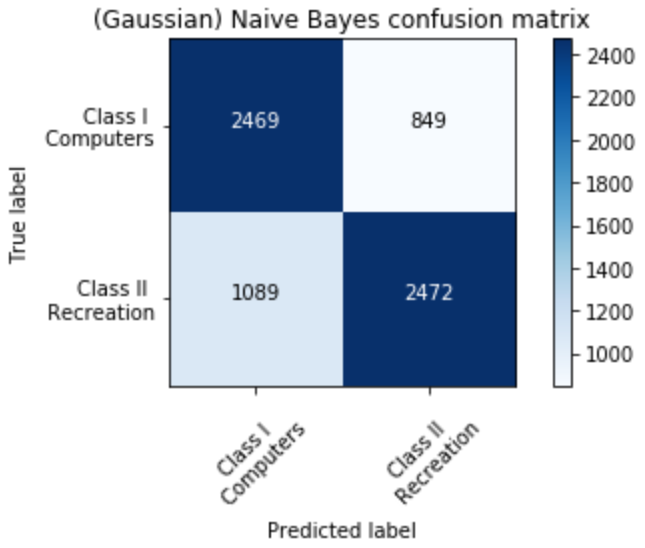


Accuracy for Logistic Regression ==> **0.7841255996511121**
Recall for Logistic Regression ==> **0.7054198258916035**
Precision logistic ==> **0.8521031207598372**
F1 score for Logistic ==> **0.7718543555077585**

3. Gaussian NB -



Area under the Gaussian NB ROC curve 0.790951477047155



Accuracy for Gaussian Naive Bayes ==> **0.7182730047972089**
Recall for Gaussian Naive Bayes ==> **0.6941870261162595**
Precision for Gaussian Naive Bayes ==> **0.7443541102077688**
F1 score for Gaussian Naive Bayes ==> **0.7183958151700087**

Part 3: Own Project

Q16. Project Proposal and Implementation

In this part we have tried to implement our own project idea based on social network graphs. Social network graphs are the graphs where each node represent users and edges represent relations between the users or the two nodes connected. It could represent the act of **'repinnig'** in case of **pinterest** or the act of **'retweeting'** in **twitter**. So if we create a directed graph, we can say that there is an edge from user 1 to user 2 if user 1 has retweeted a tweet from user 2.

We extracted such a graph from the data set provided. But the graph we extracted is not directed. We converted each user name into a unique id starting from 0. We also assume that two users in twitter will not have similar names.(twitter does not allow it). So if user 1 is author and user 2 is original author in the dataset, then we create an edge in between them. Also since it is an undirected graph, we create an edge in between them if user 2 is author and user 1 is original author. So each edge in out social network graph represent the action of retweeting. For each tweet in the dataset, we get an edge in the graph, if it was a retweet. We also combine the data from all the hashtags to get a graph.

Following are some statistics of the graph that we get:

The following table shows how many retweet relation exist between users,for each file.

Hashtags	Number of unique edges
gohawk	69009
gopatriot	16235
nfl	67213
patriot	261361
sb49	445830
superbowl	582568

From the above table we get a sense of how many unique people were actually involved in the hashtag activities of each file and also in total. I tells us about the spread of the event. If the number is small, that means most of the retweets happened with same group of people amongst themselves. But if it is high, it tells us about the spread of data in the network.

In the complete graph there were 1032137 unique edges and 1029235 unique nodes representing number of total users involved. (these edges can include edges between user1 to user1, since many people retweet their own tweet and thus the ratio of edges to nodes is like this)

We further tried to find if the graph we got was connected or not. We used networkx package to do the connected analysis on the graph. We found that the graph was heavily disconnected as expected. The following table tells about number of nodes in the disconnected components and frequency of components with that many nodes:
Note: We do not display all the components here, but only some of them

Length of connected component (no. of node in that part of graph)	Frequency (How many components in the graph with this length)
2	8333
3	1041
25	1
29	1
46	1
48	1
2943	1

We can see that there are around 8333 components with only two users in them. This and the frequency of the edges between them tells us that those two users just tweet and retweet from each other.

Also, there are many components with length between 25 and 48. Such groups represent various communities in the social networking graph. We can use this information to learn about the various subtle communities of users that are connected with each other on twitter.
The largest subcomponent of the graph had 2934 users in it. This specifies the group of people that are probably most connected and active during football matches.

Viral Marketing: This information can be used to do viral marketing during future football matches where we target a small group of people and ask them to promote our product or service and reply on mouth to mouth publicity of the product. So we will find those people who are at the center of the graph and connected to many other people.

Following are some plots of the graph that represents the shape of the graph of the largest connected subcomponent.

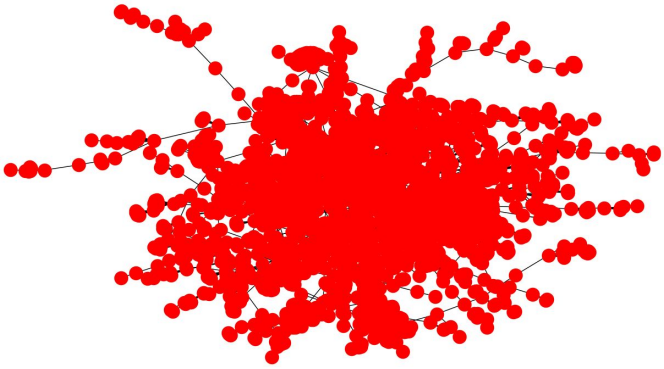


Figure 3.16.1: Largest Connected Subcomponent Zoom = 100

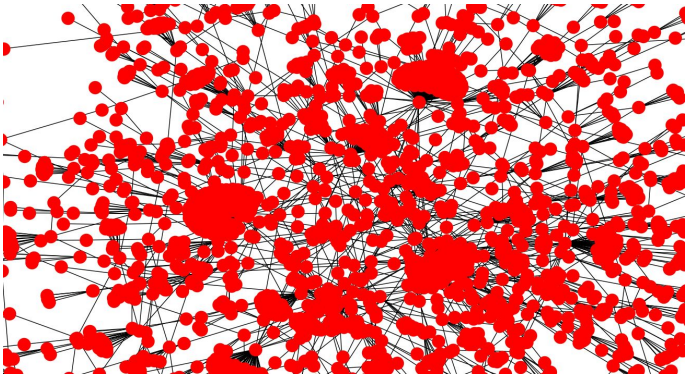


Figure 3.16.2: Largest Connected Subcomponent Zoom = 50

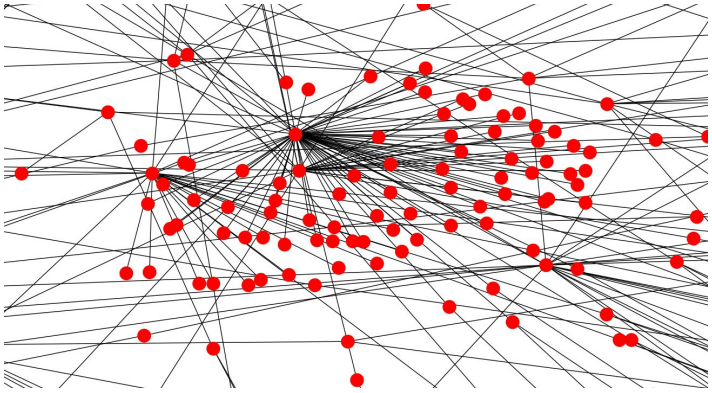


Figure 3.16.3: Largest Connected Subcomponent Zoom = 25

As we can see that the graph looks pretty good in the sense that the dataset is spread out and we can infer many interesting facts about the dataset. If we look at zoom level 25, we can see that there are some very famous and connected people who have a lot of connectivity with others and are very influential. All this information can be used in numerous analysis like the following:

Link Prediction: Link prediction is the task of predicting a link between two users. For example in facebook, it can be used to give friend suggestions to people. Here also we can use it for the same purpose. We can learn a distribution on the edges and predict probabilities for unseen edges in the graph.

Viral Marketing: As mentioned before, we learn a distribution on this graph and ask MPE queries to do approximate viral marketing predictions.

Fake News Detection: We can also do detection of content that is going to viral looking at the structure of the graph it has followed till now. We can learn a model that learns structures of graphs for viral content and use it for future predictions. If we detect that some content is going viral, we can look into it and if it is fake, we can take precautions to stop it.