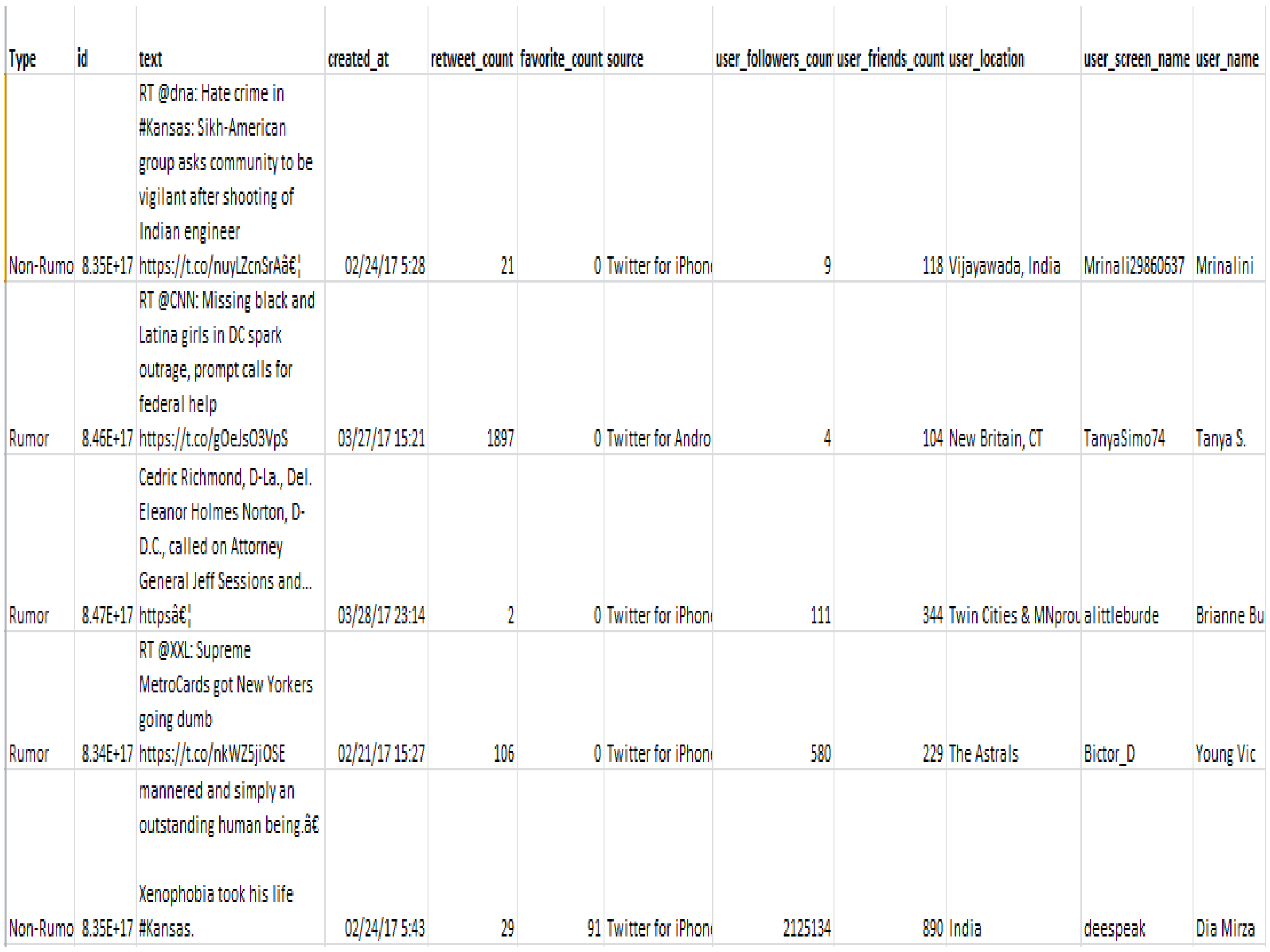
Sanchari Chowdhuri
Sohan Shah
Jung Kyu Rhys Lim

MILESTONE 2 – REPORT

RUMOR PREDICTION ON TWITTER

Data Collection

Our data collection has been ongoing, and our dataset contains more rumor and non-rumor tweets since Milestone 1. We have over 10000 tweets right now which we have used in our training and testing datasets. The code for data collection has remained the same, only the keywords to extract tweets vary from one rumor/non-rumor case to another. The figure below shows the main dataset of rumor and non-rumor cases .The dataset has rumor tweets relating to topics like missing black girls from DC, iphone7 explosion going viral, rumors of trump's allegiance to Russia and rumors of sale of Supreme metro cards in New York. Non Rumor tweets are related to the Kansas shooting incidents, ISRO launching satellites, Brexit related tweets and best picture winner at Oscars 2017.



Type	id	text	created_at	retweet_count	favorite_count	source	user_followers_coun	user_friends_count	user_location	user_screen_name	user_name
Non-Rumo	8.35E+17	RT @dna: Hate crime in #Kansas: Sikh-American group asks community to be vigilant after shooting of Indian engineer https://t.co/nuyLZcnSrAâ€¦	02/24/17 5:28	21	0	Twitter for iPhon	9	118	Vijayawada, India	Mrinali29860637	Mrinalini
Rumor	8.46E+17	RT @CNN: Missing black and Latina girls in DC spark outrage, prompt calls for federal help https://t.co/gOeJsO3VpS	03/27/17 15:21	1897	0	Twitter for Andro	4	104	New Britain, CT	TanyaSimo74	Tanya S.
Rumor	8.47E+17	Cedric Richmond, D-La., Del. Eleanor Holmes Norton, D-D.C., called on Attorney General Jeff Sessions and... httpsâ€¦	03/28/17 23:14	2	0	Twitter for iPhon	111	344	Twin Cities & MNprou	alittleburde	Brianne Bu
Rumor	8.34E+17	RT @XXL: Supreme MetroCards got New Yorkers going dumb https://t.co/nkWZ5jiOSE	02/21/17 15:27	106	0	Twitter for iPhon	580	229	The Astrals	Bictor_D	Young Vic
Non-Rumo	8.35E+17	mannered and simply an outstanding human being.â€ Xenophobia took his life #Kansas.	02/24/17 5:43	29	91	Twitter for iPhon	2125134	890	India	deespeak	Dia Mirza

Data Cleaning

We had to do additional data cleaning to run our classifiers. For every csv file that was created in the data collection process, we added a new column called 'Type' and labelled all rows in one csv as either 'Rumor' or 'Non-Rumor'. The cleaning process for each research question is explained below -

Research Question 1: All rumor and non-rumor tweets had to be merged into a single csv file. We used the pd.concat function in Pandas to merge these csv files. The rows in the file then had to be sorted in a random order. For the ***logistic regression problem***, we only needed the columns – 'retweet_count', 'favorite_count', 'user_followers_count' and 'user_friends_count' and 'Type'. We changed the 'Type' column and gave it the value of 1 for rumor and 0 for non-rumor. The same set of columns were used with ***Decision Trees*** as well, except for the 'Type' column which contained 'Rumor' and 'Non-Rumor' instead of 1 and 0. For ***Naïve Bayes classifier***, we only used the column 'text' which contained the text of the tweet.

Research Question 2: We only merged the csv files that had rumor in them (again, using the Python code). Three news columns were added to this dataset – 'First Name', 'Last Name' and 'Gender'. First and last names were extracted from the 'user_name' column and 'Gender' was populated using the 'Sex Machine Library' in Python. Gender can take one of the three values – 'Male', 'Female', or 'Andy'. 'Andy' would refer to those users who do not have a first name that could be determined male/female, or these could be news bots or other organizations/groups tweeting information. We randomized the rows in the dataset before training them using the classifiers. For ***Naïve Bayes***, only the columns 'text' and 'Gender' was used to build the classifier. For ***Decision Trees***, we used the columns 'retweet_count', 'favorite_count', 'user_followers_count' and 'user_friends_count' and 'Gender'.

Research Question 3: Only the csv files that contained rumors were merged into a single csv file. We added a new column called 'Type' where we labeled each tweet based on the industry in which the rumor was spread. As in the previous two research questions, we randomized the order of the rows in the dataset. For the ***Naïve Bayes classifier***, we used the columns 'text' and 'Type' (in this case Type refers to the type of industry – city, electronics, politics and transport). For ***Decision Trees***, we used the columns 'retweet_count', 'favorite_count', 'user_followers_count' and 'user_friends_count' and 'Type'.

Research Questions

Research Question 1: Predict whether a given tweet is a rumor or not.

1. Linear Regression

Linear and multivariable regression techniques are not applicable to the research question because the independent variable is not continuous. Our independent variable can assume only two values - 'Rumor' or 'Non-Rumor'. Subsequently, there is no requirement for regularization either.

2. Logistic Regression

First, we computed a logistic regression with the following independent features on our training dataset - 'retweet_count', 'favorite_count', 'user_followers_count' and 'user_friends_count'. The results for this model are as follows -

Intercept is -4.794e-02.

Coefficients for the features retweet_count, favorite_count1, favorite_count2, and a number of user_friends_count are statistically significant. The following is the result of the model, which shows the coefficient for each feature and that they are statistically significant (The ones that were not found to be statistically significant are not shown below) -

```
                          Estimate    Std. Error  z value   Pr(>|z|)
(Intercept)              -4.794e-02   2.027e-01   -0.237    0.813027
retweet_count            -1.526e-04   1.003e-05  -15.212   < 2e-16 ***
favorite_count1          -3.947e-01   1.803e-01   -2.189    0.028607 *
favorite_count2          -8.551e-01   3.163e-01   -2.703    0.006864 **
user_friends_count4      -2.315e+00   1.064e+00   -2.176    0.029582 *
user_friends_count11      1.663e+00   5.302e-01    3.137    0.001707 **
user_friends_count17      1.684e+00   8.011e-01    2.102    0.035565 *
user_friends_count25      1.262e+00   3.695e-01    3.416    0.000635 ***
user_friends_count27     -1.616e+00   5.275e-01   -3.063    0.002188 **
user_friends_count36     -1.505e+00   6.623e-01   -2.272    0.023084 *
user_friends_count39      1.371e+00   6.899e-01    1.987    0.046968 *
user_friends_count48      1.252e+00   6.164e-01    2.032    0.042161 *
user_friends_count51      1.219e+00   4.482e-01    2.721    0.006509 **
user_friends_count54      1.401e+00   6.084e-01    2.302    0.021318 *
user_friends_count62      1.421e+00   7.053e-01    2.014    0.044001 *
user_friends_count79      1.549e+00   7.010e-01    2.210    0.027116 *
user_friends_count94      1.037e+00   4.274e-01    2.427    0.015234 *
user_friends_count103    -1.679e+00   7.871e-01   -2.133    0.032926 *
user_friends_count124     1.302e+00   4.881e-01    2.667    0.007649 **
user_friends_count125     1.252e+00   5.648e-01    2.216    0.026695 *
user_friends_count136     2.452e+00   7.834e-01    3.130    0.001747 **
user_friends_count161     3.167e+00   9.051e-01    3.499    0.000467 ***
user_friends_count217    -2.434e+00   1.058e+00   -2.301    0.021410 *
user_friends_count229     9.329e-01   4.567e-01    2.043    0.041080 *
user_friends_count277     1.587e+00   7.599e-01    2.088    0.036814 *
user_friends_count303     1.579e+00   6.498e-01    2.429    0.015128 *
```

```
user_friends_count306         1.607e+00   5.696e-01   2.821    0.004793 **
user_friends_count311        -1.631e+00   7.930e-01  -2.057    0.039672 *
user_friends_count343        -1.682e+00   7.907e-01  -2.127    0.033382 *
user_friends_count403         1.846e+00   9.276e-01   1.990    0.046592 *
user_friends_count467         2.389e+00   1.112e+00   2.149    0.031631 *
user_friends_count598         3.290e+00   9.938e-01   3.310    0.000932 ***
user_friends_count703         2.244e+00   1.108e+00   2.026    0.042781 *
user_friends_count905        -3.189e+00   1.039e+00  -3.068    0.002152 **
user_friends_count1689        3.605e+00   1.104e+00   3.266    0.001090 **
```

Log Odds: For every unit change in retweet_count, the log odds of rumor vs. non-rumor decreases by 0.00015. Similarly, for every unit change in the feature favorite_count1, the log odds of rumor vs. non-rumor decreases by 0.394. In the case of a unit change in the feature user_friends_count11, the log odds of rumor vs. non-rumor increases by 1.663. The same explanation applies to all other features.

Odd Ratios: For 1 unit increase in retweet_count, the odds of a tweet being a rumor increases by 0.99. Similarly, for 1 unit increase in favorite_count1, the odds of a tweet being a rumor increases by 0.673. The same explanation applies to all other features. A few of them are listed below.

```
      Retweet_count             favorite_count1
       9.998474e-01                6.738641e-01
 user_friends_count4         user_friends_count11
        9.872157e-02               5.275712e+00
user_friends_count17         user_friends_count25
       5.385567e+00                3.533871e+00
.                                .
.                                .
```

Per the training dataset, ‘retweet_count’, ‘favorite_count1’, ‘favorite_count2’ and ‘user_friends_count’ are the most predictive features.

We used the trained model to predict on our training dataset. A snippet of the result is below, and the column RumorP shows the probability of the tweet being a rumor, based on the corresponding values of the other columns –

```
  retweet_count user_followers_count user_friends_count favorite_count      RumorP
1            21                    9                118              0 0.49375081
2          1897                    4                104              0 0.42986157
3             2                  111                344              0 0.49496314
4           106                  580                229              0 0.49109958
5           314                 2204               2638              0 0.48993557
6         19520                  110                 87              0 0.06300564
```

The mean value of the probability in the last column above is 0.458.

3. Naïve Bayes

We are analyzing the text in the tweets and predicting whether a tweet is a rumor or not. The dataset contains tweets that are labeled either 'Rumor' or 'Non-Rumor'. There are 5157 tweets that are rumors and 5718 tweets that are not rumors.

We randomly ordered the rows in our dataset and divided it into training and testing in the ratio of 75:25. The following table describes what percentage of our training and testing dataset contains rumor and non-rumor tweets. As you can see below, we have a close percentage of rumors and non-rumors in our training and testing datasets.

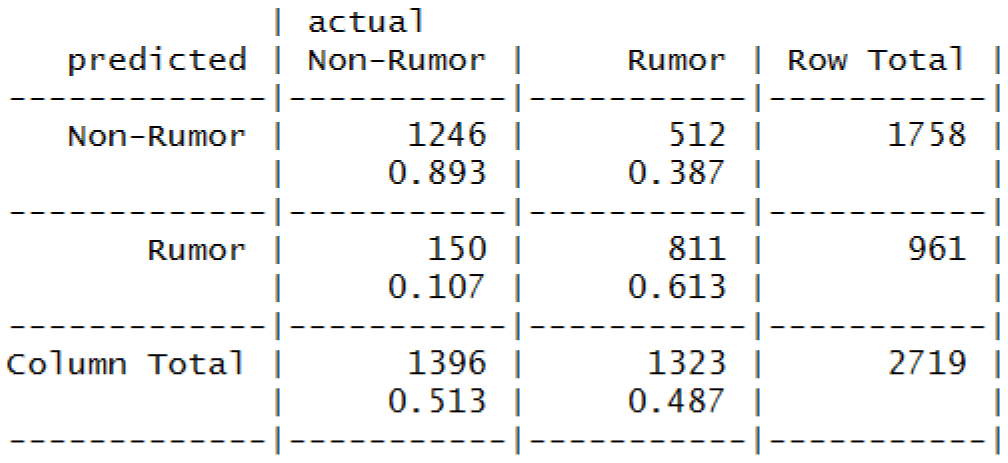
	Rumor	**Non-Rumor**
Training	0.4700834	0.5299166
Testing	0.4865759	0.5134241

The following is the confusion matrix that is generated.

```
             | actual 
   predicted | Non-Rumor |      Rumor | Row Total | 
-------------|-----------|------------|-----------|
   Non-Rumor |      1276 |        483 |      1759 | 
             |     0.914 |      0.365 |           | 
-------------|-----------|------------|-----------|
       Rumor |       120 |        840 |       960 | 
             |     0.086 |      0.635 |           | 
-------------|-----------|------------|-----------|
Column Total |      1396 |       1323 |      2719 | 
             |     0.513 |      0.487 |           | 
-------------|-----------|------------|-----------|
```

From the confusion matrix above we can infer that 91.4% of all tweets that are not rumors are classified correctly, and 8.6% of non-rumors are classified as rumors. On the other hand, 63.5% of tweets that are rumors are classified correctly and 36.5% of rumors are classified incorrectly as non-rumors.

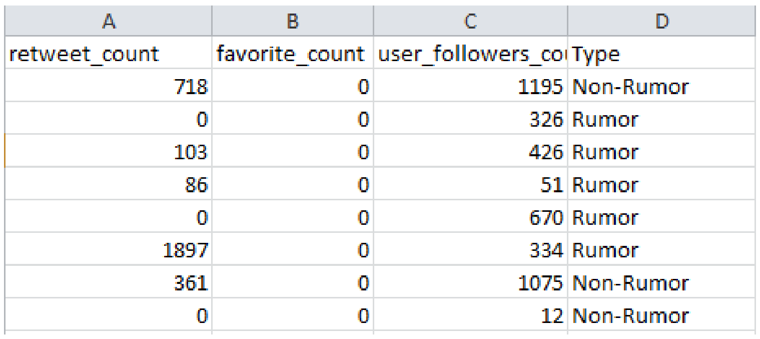
Confusion Matrix with Laplace Estimator does not change the above results drastically.



```
             | actual
   predicted | Non-Rumor |     Rumor | Row Total |
-------------|-----------|-----------|-----------|
   Non-Rumor |      1246 |       512 |      1758 |
             |     0.893 |     0.387 |           |
-------------|-----------|-----------|-----------|
       Rumor |       150 |       811 |       961 |
             |     0.107 |     0.613 |           |
-------------|-----------|-----------|-----------|
Column Total |      1396 |      1323 |      2719 |
             |     0.513 |     0.487 |           |
-------------|-----------|-----------|-----------|
```

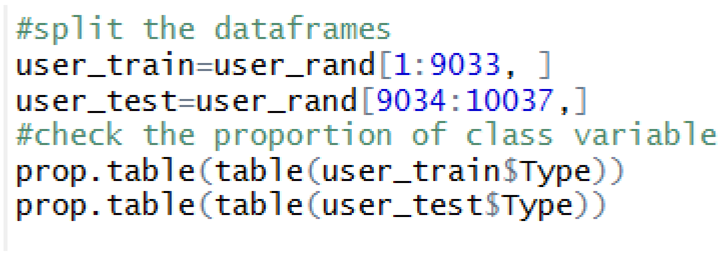
4. Decision Trees

In order to predict the rumor and non-rumor status of a given tweet we have considered factors like ***retweet counts, favorite counts and user follower counts.*** Following figure shows the small snippet of the data set which we will be using to train our **decision tree**. The dataset contains 100037 rows of tweets which are rumors and non rumors. For **rumor cases** tweets related to iphone7 explosion, sale of supreme metro card in NYC metro, Trump's allegiance to Russia and missing report of black girls in DC were collected while for **non-rumor cases tweets** related to Brexit, Kansas shooting incident, Oscar best picture fiasco were collected.

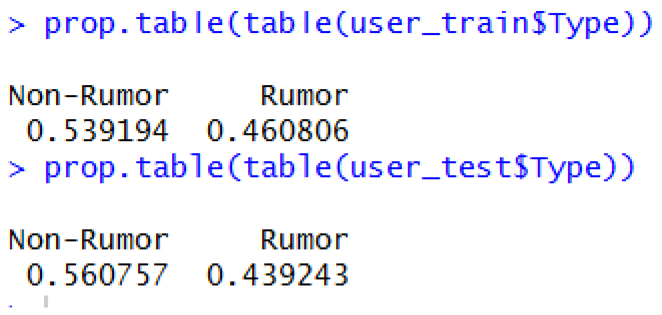


A	B	C	D
retweet_count	favorite_count	user_followers_co	Type
718	0	1195	Non-Rumor
0	0	326	Rumor
103	0	426	Rumor
86	0	51	Rumor
0	0	670	Rumor
1897	0	334	Rumor
361	0	1075	Non-Rumor
0	0	12	Non-Rumor

In order to start with decision tree we have to first divide the entire dataset into training and testing data. Training data consists of 90% of the data set and testing data consists of 10 % of the dataset .

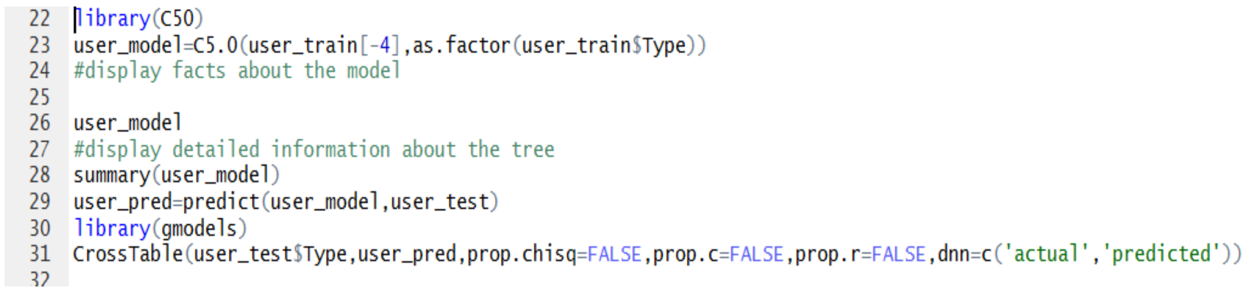


```
#split the dataframes
user_train=user_rand[1:9033, ]
user_test=user_rand[9034:10037,]
#check the proportion of class variable
prop.table(table(user_train$Type))
prop.table(table(user_test$Type))
```



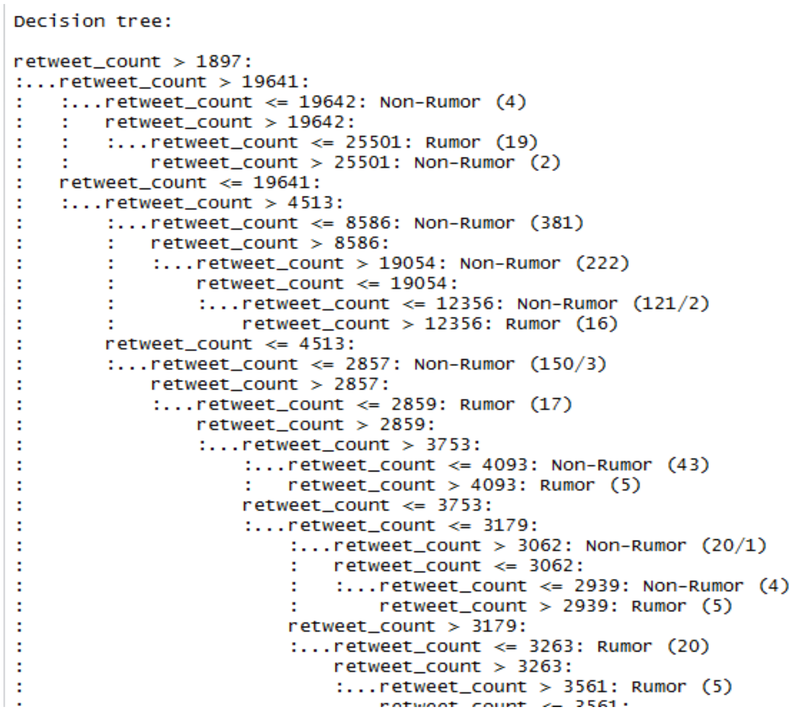
```
> prop.table(table(user_train$Type))

Non-Rumor     Rumor 
 0.539194  0.460806 
> prop.table(table(user_test$Type))

Non-Rumor     Rumor 
 0.560757  0.439243 
```

The above figure shows that the portion of rumors and non-rumors in the training and testing dataset is almost equal, thus there is no bias in the testing and training dataset. Following commands are executed in order to train the decision tree. The column ***Type*** (rumor, non-rumor) is used as factors



```
library(C50)
user_model=C5.0(user_train[-4],as.factor(user_train$Type))
#display facts about the model

user_model
#display detailed information about the tree
summary(user_model)
user_pred=predict(user_model,user_test)
library(gmodels)
CrossTable(user_test$Type,user_pred,prop.chisq=FALSE,prop.c=FALSE,prop.r=FALSE,dnn=c('actual','predicted'))
```

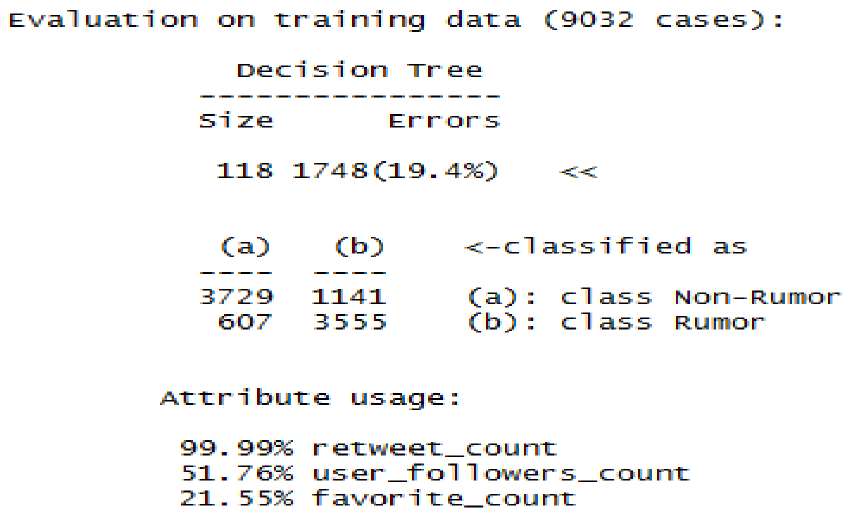
The figure below shows the Decision tree .Details of the decision tree is in Appendix A.



```
Decision tree:

retweet_count > 1897:
:...retweet_count > 19641:
:   :...retweet_count <= 19642: Non-Rumor (4)
:   :   retweet_count > 19642:
:   :   :...retweet_count <= 25501: Rumor (19)
:   :       retweet_count > 25501: Non-Rumor (2)
:   retweet_count <= 19641:
:   :...retweet_count > 4513:
:       :...retweet_count <= 8586: Non-Rumor (381)
:       :   retweet_count > 8586:
:       :   :...retweet_count > 19054: Non-Rumor (222)
:       :       retweet_count <= 19054:
:       :       :...retweet_count <= 12356: Non-Rumor (121/2)
:       :           retweet_count > 12356: Rumor (16)
:       retweet_count <= 4513:
:       :...retweet_count <= 2857: Non-Rumor (150/3)
:           retweet_count > 2857:
:           :...retweet_count <= 2859: Rumor (17)
:               retweet_count > 2859:
:               :...retweet_count > 3753:
:                   :...retweet_count <= 4093: Non-Rumor (43)
:                   :   retweet_count > 4093: Rumor (5)
:                   retweet_count <= 3753:
:                   :...retweet_count <= 3179:
:                       :...retweet_count > 3062: Non-Rumor (20/1)
:                       :   retweet_count <= 3062:
:                       :   :...retweet_count <= 2939: Non-Rumor (4)
:                       :       retweet_count > 2939: Rumor (5)
:                       retweet_count > 3179:
:                       :...retweet_count <= 3263: Rumor (20)
:                           retweet_count > 3263:
:                           :...retweet_count > 3561: Rumor (5)
:                               retweet_count <= 3561:
```

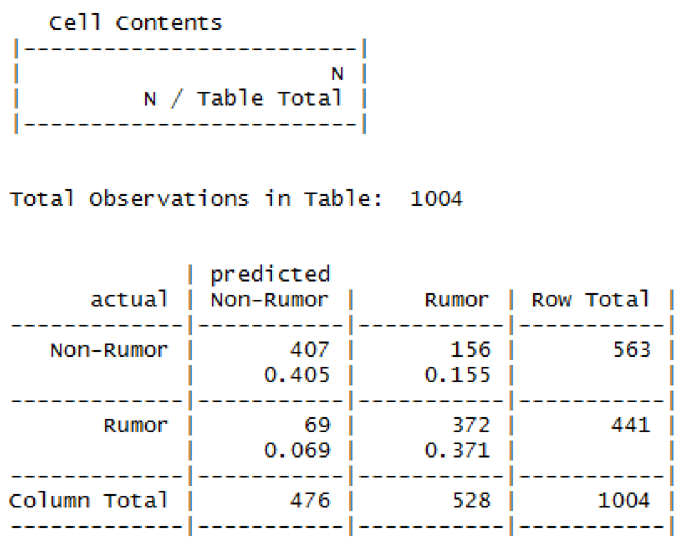
Interpreting the decision tree: If retweets count is greater than 19641 and if retweet count is less than or equal to 19642 it's a non-rumor .Such an incident has happened 4 times. Then if the retweet count is less than equal to 25501 it's a rumor. Such an incident has happened 19 times.

The following figure shows error rate on training set. Displaying the effectivity of the training on the training set of data. The decision tree was able correctly trained on the training set of the data with an error rate of 19.4%. 3729 Rumors were correctly labelled as Rumors, while 1141 were wrongly labeled as non-rumors, inspite of being rumors. On the other hand 3555 non-rumors were correctly classified as non-rumors while 607 were wrongly labelled as rumors in spite of being non-rumors. The figure below shows the summary of the performance of the decision tree. The model heavily relies on retweet counts (99.9%) followed by user follower count (51.76%) and favorite count (21.55%)



```
Evaluation on training data (9032 cases):

          Decision Tree
        ----------------
        Size      Errors

         118 1748(19.4%)   <<


         (a)   (b)    <-classified as
        ----  ----
        3729  1141    (a): class Non-Rumor
         607  3555    (b): class Rumor


      Attribute usage:

       99.99% retweet_count
       51.76% user_followers_count
       21.55% favorite_count
```

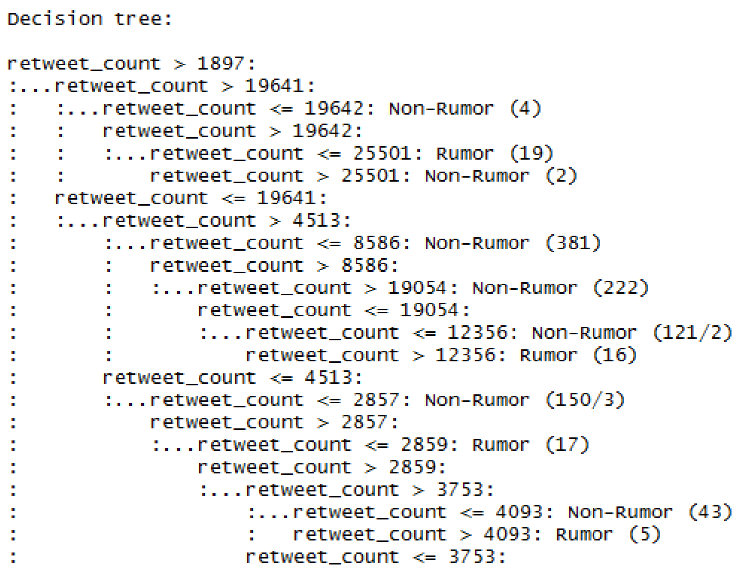
The following figure shows the **confusion matrix**.



```
   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  1004


             | predicted
      actual | Non-Rumor |     Rumor | Row Total |
-------------|-----------|-----------|-----------|
   Non-Rumor |       407 |       156 |       563 |
             |     0.405 |     0.155 |           |
-------------|-----------|-----------|-----------|
       Rumor |        69 |       372 |       441 |
             |     0.069 |     0.371 |           |
-------------|-----------|-----------|-----------|
Column Total |       476 |       528 |      1004 |
-------------|-----------|-----------|-----------|
```

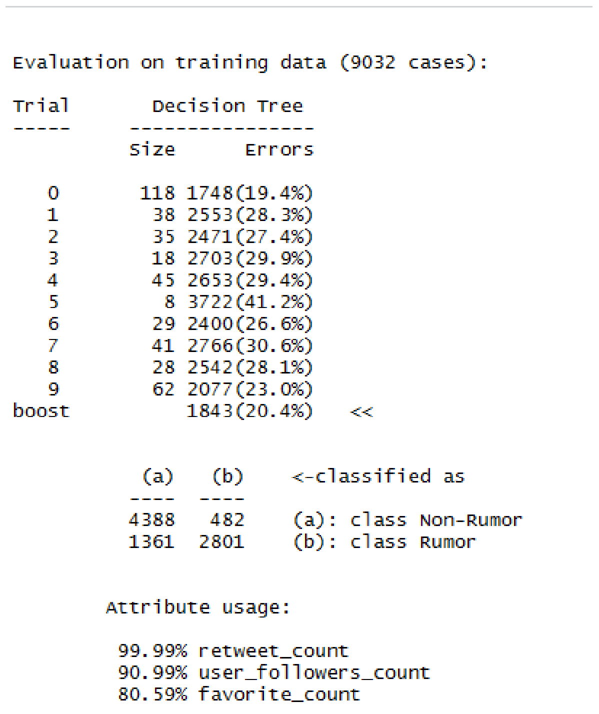
The confusion matrix tells that the classifier predicted 407 times a non-rumor when it was actually a non-rumor while 156 times it predicted a non-rumor as a rumor. While 372 times it correctly predicted a rumor as a rumor while 156 times it wrongly predicted Rumor as a non-rumor.

Next we apply ***boosting*** and get the following new decision tree. The figure shows a small snippet of the decision tree obtained by boosting, Details of the result is available in Appendix A.



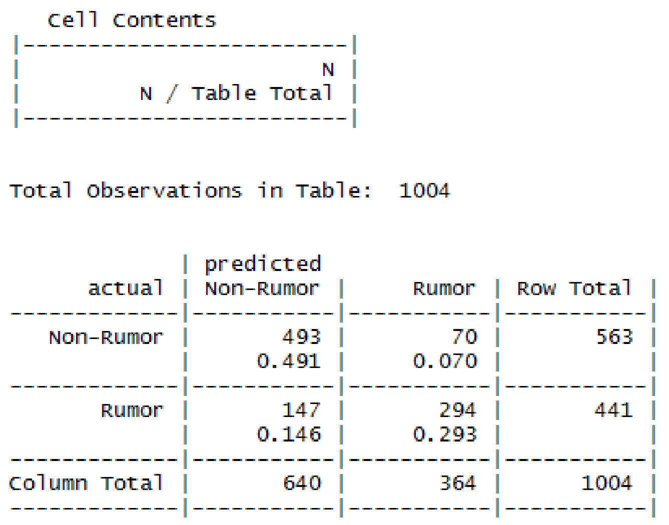
```
Decision tree:

retweet_count > 1897:
:...retweet_count > 19641:
:   :...retweet_count <= 19642: Non-Rumor (4)
:   :   retweet_count > 19642:
:   :   :...retweet_count <= 25501: Rumor (19)
:   :       retweet_count > 25501: Non-Rumor (2)
:   retweet_count <= 19641:
:   :...retweet_count > 4513:
:       :...retweet_count <= 8586: Non-Rumor (381)
:       :   retweet_count > 8586:
:       :   :...retweet_count > 19054: Non-Rumor (222)
:       :       retweet_count <= 19054:
:       :       :...retweet_count <= 12356: Non-Rumor (121/2)
:       :           retweet_count > 12356: Rumor (16)
:       retweet_count <= 4513:
:       :...retweet_count <= 2857: Non-Rumor (150/3)
:           retweet_count > 2857:
:           :...retweet_count <= 2859: Rumor (17)
:               retweet_count > 2859:
:               :...retweet_count > 3753:
:                   :...retweet_count <= 4093: Non-Rumor (43)
:                   :   retweet_count > 4093: Rumor (5)
:                   retweet_count <= 3753:
```

Following figure shows evaluation of training set after boosting. The classifier is able to correctly classify 4388 Non-Rumors and 2801 Rumors. The figure mentions the error in each trial.



```
Evaluation on training data (9032 cases):

Trial       Decision Tree
-----     ----------------
          Size      Errors

   0       118 1748(19.4%)
   1        38 2553(28.3%)
   2        35 2471(27.4%)
   3        18 2703(29.9%)
   4        45 2653(29.4%)
   5         8 3722(41.2%)
   6        29 2400(26.6%)
   7        41 2766(30.6%)
   8        28 2542(28.1%)
   9        62 2077(23.0%)
boost          1843(20.4%)   <<


           (a)   (b)    <-classified as
          ----  ----
          4388   482    (a): class Non-Rumor
          1361  2801    (b): class Rumor


      Attribute usage:

       99.99% retweet_count
       90.99% user_followers_count
       80.59% favorite_count
```

After boosting the classifier makes optimum usage of the attributes. Following ***figure displays confusion matrix after boosting.*** Out of 563 cases of non-rumor 493 were correctly classified and out of 441 cases of rumors,294 were correctly labelled.



```
   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  1004


             | predicted
      actual | Non-Rumor |     Rumor | Row Total |
-------------|-----------|-----------|-----------|
   Non-Rumor |       493 |        70 |       563 |
             |     0.491 |     0.070 |           |
-------------|-----------|-----------|-----------|
       Rumor |       147 |       294 |       441 |
             |     0.146 |     0.293 |           |
-------------|-----------|-----------|-----------|
Column Total |       640 |       364 |      1004 |
-------------|-----------|-----------|-----------|
```

Random Forest and bagging don't apply as we are predicting categorical variables (Rumor, Non Rumor) based on numerical factors like user's friends, user's followers, retweets and favorite counts.

Research Question 2: Predict user demographic information of those who are spreading rumors.

1. Linear Regression

Linear and multivariable regression techniques are not applicable to the research question because the independent variable is not continuous. Our independent variable can assume only three values - 'Male', 'Female' or 'Andy'. Subsequently, there is no requirement for regularization either.

2. Logistic Regression

Our Independent variable is 'Gender' which can take one of three values – 'Male', 'Female' or 'Andy'. Since logistic regression can only be applied if it is binomial, we cannot use this technique train and test our data.

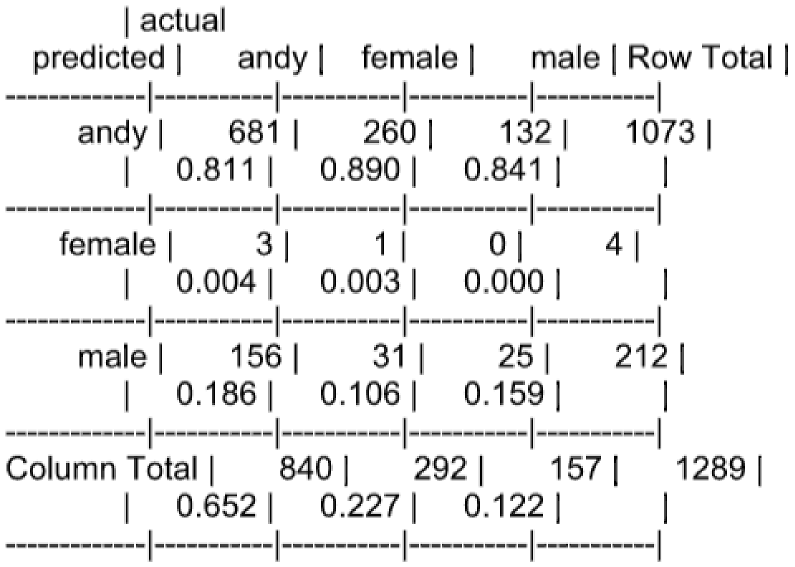
3. Naïve Bayes

We are analyzing the text in the tweets and predicting the gender of the user who tweeted. The dataset contains tweets and the corresponding gender of the user.

We randomly ordered the rows in our dataset and divided it into training and testing in the ratio of 75:25. The following table describes what percentage of our training and testing dataset contains tweets by males, females and others.

	Male	**Female**	**Andy**
Training	0.1652017	0.1171148	0.7176836
Testing	0.1217998	0.2265322	0.6516680

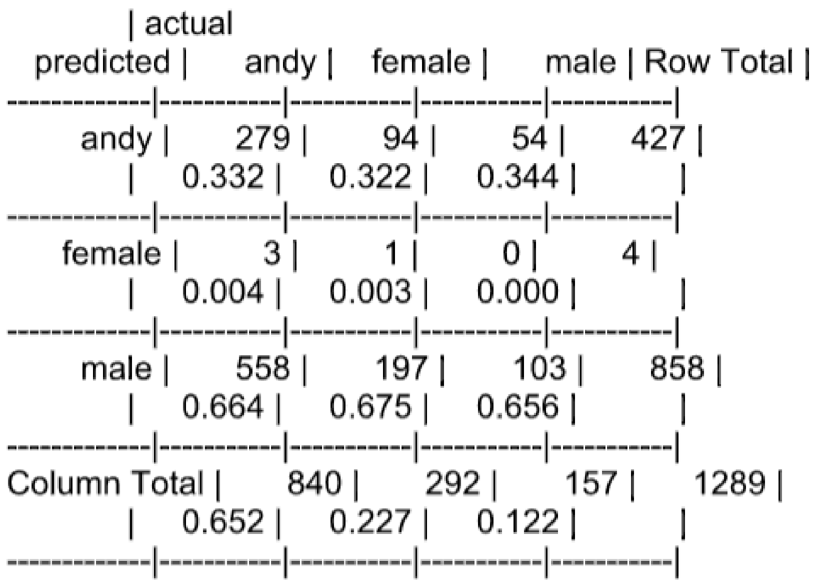
The following is the confusion matrix that is generated.



```
             | actual
   predicted |      andy |    female |      male | Row Total |
-------------|-----------|-----------|-----------|-----------|
        andy |       681 |       260 |       132 |      1073 |
             |     0.811 |     0.890 |     0.841 |           |
-------------|-----------|-----------|-----------|-----------|
      female |         3 |         1 |         0 |         4 |
             |     0.004 |     0.003 |     0.000 |           |
-------------|-----------|-----------|-----------|-----------|
        male |       156 |        31 |        25 |       212 |
             |     0.186 |     0.106 |     0.159 |           |
-------------|-----------|-----------|-----------|-----------|
Column Total |       840 |       292 |       157 |      1289 |
             |     0.652 |     0.227 |     0.122 |           |
-------------|-----------|-----------|-----------|-----------|
```

From the confusion matrix above we can infer that 15.9% of users who are males are classified correctly, and the remaining 84.1% are classified as Andy. 0.3% of users who are females are classified correctly as females, 10.6% of the remaining who are females are classified as males and the others as Andy. 81.1% of users who are neither male nor female are classified correctly. The classifier does not do a very good job in classifying males and females accurately.

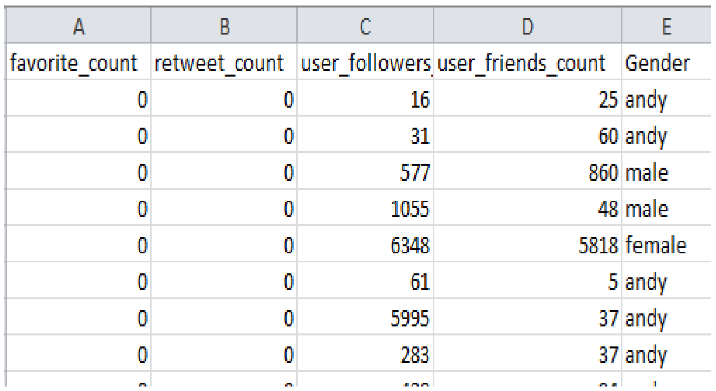
With Laplace Estimator in place, there is a drastic change with respect to classification of males and Andy. As we can see in the confusion matrix below, 65.6% of users who are males are classified correctly, in contrast to 15.9% without the Laplace Estimator. 33.2% of users who are neither male nor female are classified correctly as Andy, in contrast to 81.1% without the Laplace Estimator. Classification of females does not change. It has done a better job in classifying males accurately.



```
             | actual
   predicted |      andy |    female |      male | Row Total |
-------------|-----------|-----------|-----------|-----------|
        andy |       279 |        94 |        54 |       427 |
             |     0.332 |     0.322 |     0.344 |           |
-------------|-----------|-----------|-----------|-----------|
      female |         3 |         1 |         0 |         4 |
             |     0.004 |     0.003 |     0.000 |           |
-------------|-----------|-----------|-----------|-----------|
        male |       558 |       197 |       103 |       858 |
             |     0.664 |     0.675 |     0.656 |           |
-------------|-----------|-----------|-----------|-----------|
Column Total |       840 |       292 |       157 |      1289 |
             |     0.652 |     0.227 |     0.122 |           |
-------------|-----------|-----------|-----------|-----------|
```

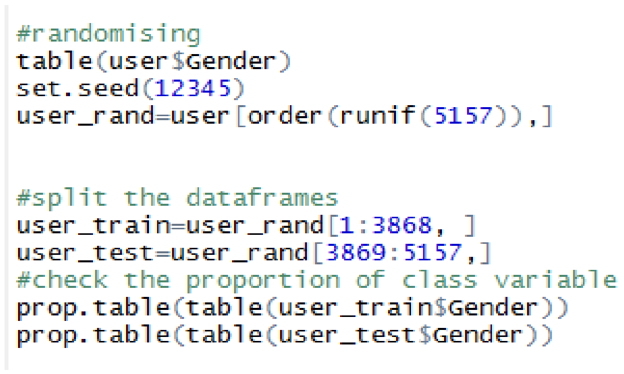
4. Decision Trees

To predict gender of the user based on favorite counts, retweets, user followers and user's friends we develop a classifier to predict gender based on above mentioned variables. Following figure shows the small snippet of the data set which we will be using to train our **decision tree**. The dataset contains 5157 rows of rumor tweets like explosion of iphone 7, supreme metro card in New York, Trump's allegiance to Russia and missing news of black girls from DC. The texts of the tweets are removed thus subsetting the main dataset.



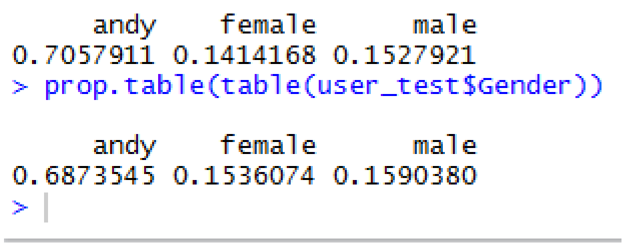
A	B	C	D	E
favorite_count	retweet_count	user_followers	user_friends_count	Gender
0	0	16	25	andy
0	0	31	60	andy
0	0	577	860	male
0	0	1055	48	male
0	0	6348	5818	female
0	0	61	5	andy
0	0	5995	37	andy
0	0	283	37	andy

In order to start with decision tree we have to first divide the entire dataset into training and testing data and also check the proportion of the variables in it to determine if it's been equally distributed among training and testing dataset.



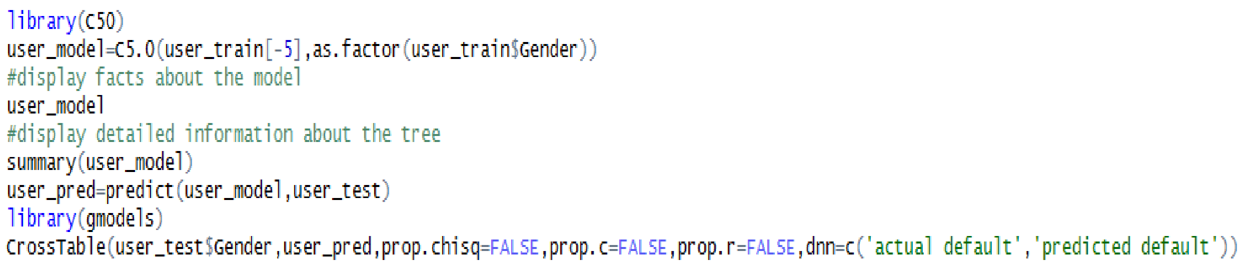
```
#randomising
table(user$Gender)
set.seed(12345)
user_rand=user[order(runif(5157)),]


#split the dataframes
user_train=user_rand[1:3868, ]
user_test=user_rand[3869:5157,]
#check the proportion of class variable
prop.table(table(user_train$Gender))
prop.table(table(user_test$Gender))
```

The following figure shows the division of variables in training and testing datasets.



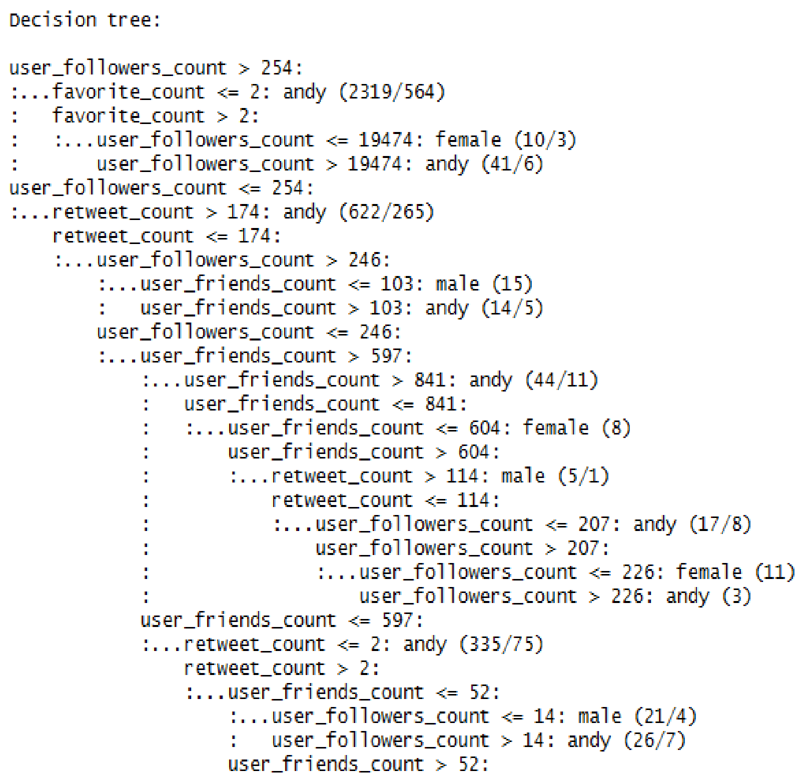
```
     andy    female      male 
0.7057911 0.1414168 0.1527921 
> prop.table(table(user_test$Gender))

     andy    female      male 
0.6873545 0.1536074 0.1590380 
> 
```

The above figure shows that the portion of andy *(androgynous)*, ***female and male in the training and testing dataset is almost equal***, Thus there is no bias in the testing and training dataset. Following commands are executed in order to train the decision tree. The column ***Gender*** (andy, female, male) is used as factors.



```
library(C50)
user_model=C5.0(user_train[-5],as.factor(user_train$Gender))
#display facts about the model
user_model
#display detailed information about the tree
summary(user_model)
user_pred=predict(user_model,user_test)
library(gmodels)
CrossTable(user_test$Gender,user_pred,prop.chisq=FALSE,prop.c=FALSE,prop.r=FALSE,dnn=c('actual default','predicted default'))
```

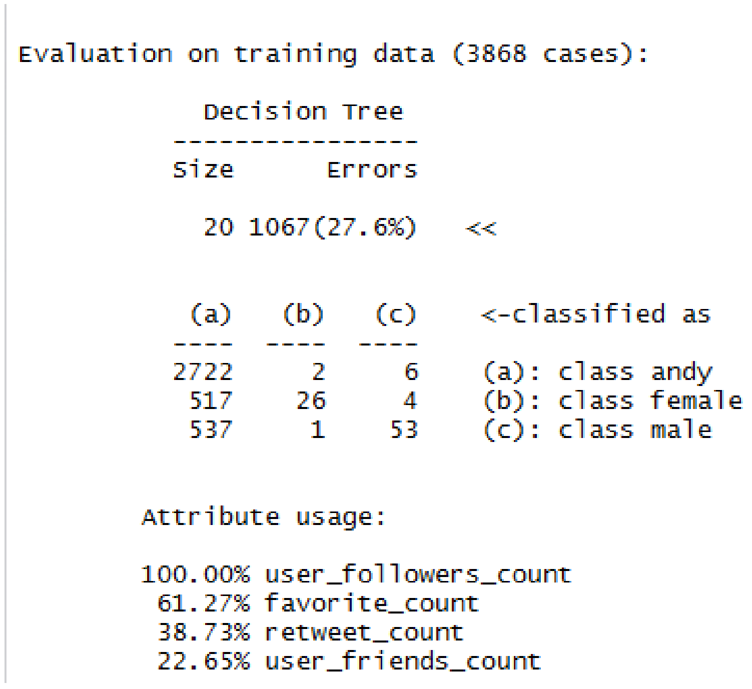
The figure below shows the decision tree. A detailed decision tree is available in Appendix –B.



```
Decision tree:

user_followers_count > 254:
:...favorite_count <= 2: andy (2319/564)
:   favorite_count > 2:
:   :...user_followers_count <= 19474: female (10/3)
:       user_followers_count > 19474: andy (41/6)
user_followers_count <= 254:
:...retweet_count > 174: andy (622/265)
    retweet_count <= 174:
    :...user_followers_count > 246:
        :...user_friends_count <= 103: male (15)
        :   user_friends_count > 103: andy (14/5)
        user_followers_count <= 246:
        :...user_friends_count > 597:
            :...user_friends_count > 841: andy (44/11)
            :   user_friends_count <= 841:
            :   :...user_friends_count <= 604: female (8)
            :       user_friends_count > 604:
            :       :...retweet_count > 114: male (5/1)
            :           retweet_count <= 114:
            :           :...user_followers_count <= 207: andy (17/8)
            :               user_followers_count > 207:
            :               :...user_followers_count <= 226: female (11)
            :                   user_followers_count > 226: andy (3)
            user_friends_count <= 597:
            :...retweet_count <= 2: andy (335/75)
                retweet_count > 2:
                :...user_friends_count <= 52:
                    :...user_followers_count <= 14: male (21/4)
                    :   user_followers_count > 14: andy (26/7)
                    user_friends_count > 52:
```

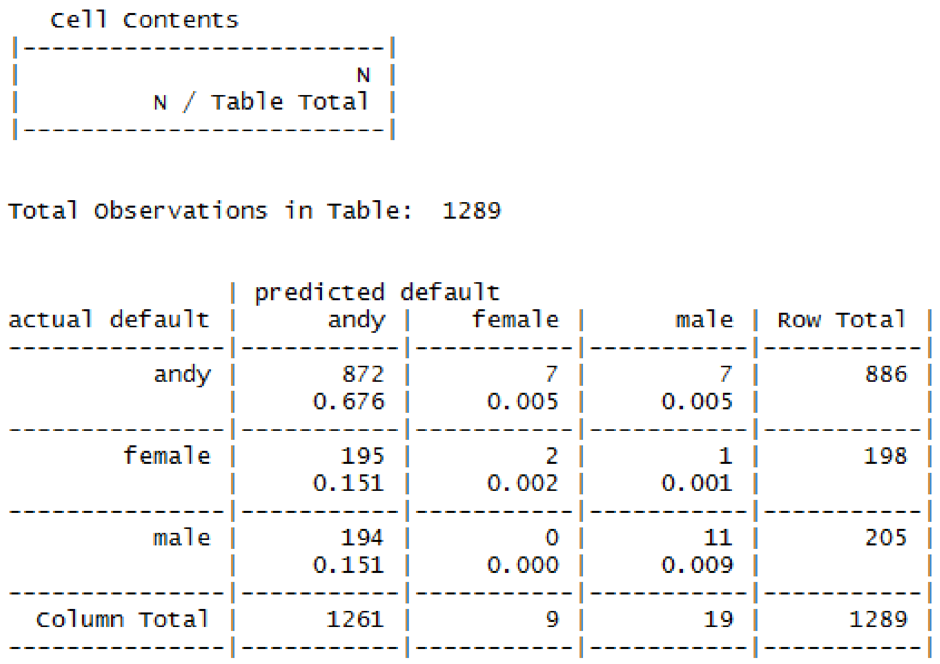
Interpreting the decision tree: If user follower count is greater than 254 then favourite count is less or equal to 2 it's an Andy (i.e can be an institute or bot or names which are common for male and female)

The following figure shows error rate on training set and displays the effectivity of the training on the training set of data. The decision tree was able correctly trained on the training set of the data with an error rate of 27.6 %. 2722 were rightly classified as andy (neutral)



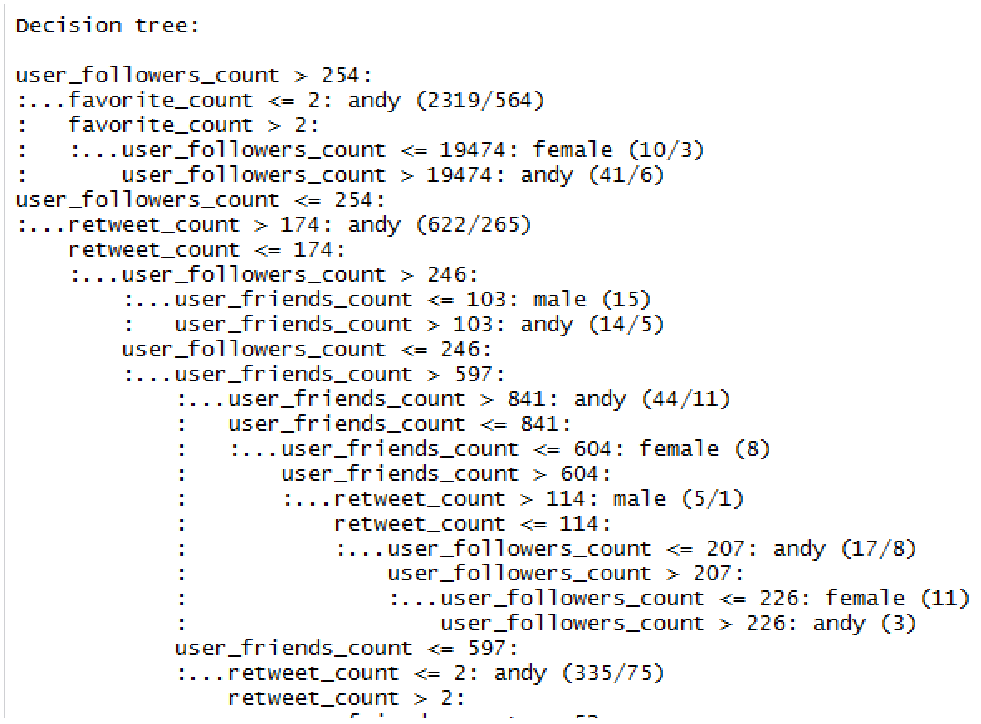
```
Evaluation on training data (3868 cases):

            Decision Tree
          ----------------
          Size      Errors

            20 1067(27.6%)   <<


           (a)   (b)   (c)    <-classified as
          ----  ----  ----
          2722     2     6    (a): class andy
           517    26     4    (b): class female
           537     1    53    (c): class male


        Attribute usage:

        100.00% user_followers_count
         61.27% favorite_count
         38.73% retweet_count
         22.65% user_friends_count
```

The following figure shows the **confusion matrix**.

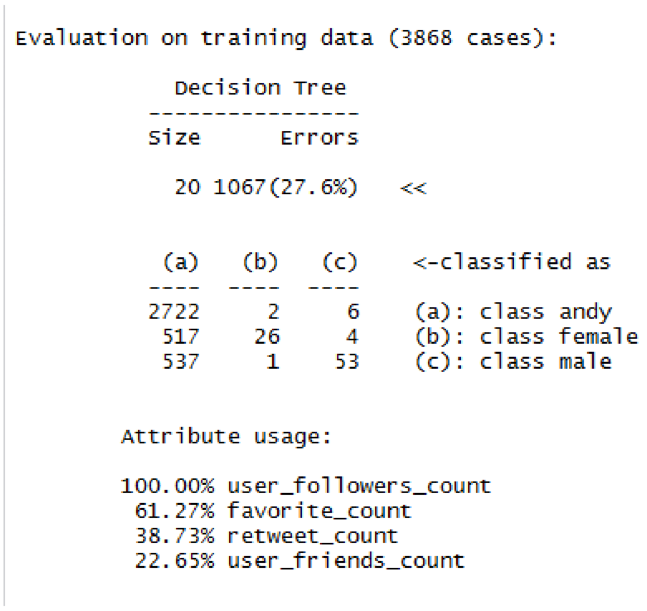


```
   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  1289


               | predicted default
actual default |      andy |    female |      male | Row Total |
---------------|-----------|-----------|-----------|-----------|
          andy |       872 |         7 |         7 |       886 |
               |     0.676 |     0.005 |     0.005 |           |
---------------|-----------|-----------|-----------|-----------|
        female |       195 |         2 |         1 |       198 |
               |     0.151 |     0.002 |     0.001 |           |
---------------|-----------|-----------|-----------|-----------|
          male |       194 |         0 |        11 |       205 |
               |     0.151 |     0.000 |     0.009 |           |
---------------|-----------|-----------|-----------|-----------|
  Column Total |      1261 |         9 |        19 |      1289 |
---------------|-----------|-----------|-----------|-----------|
```

Next we apply ***boosting*** and get the following new decision tree. The figure shows a small snippet of the decision tree obtained by boosting, Details of the result is available in Appendix B.

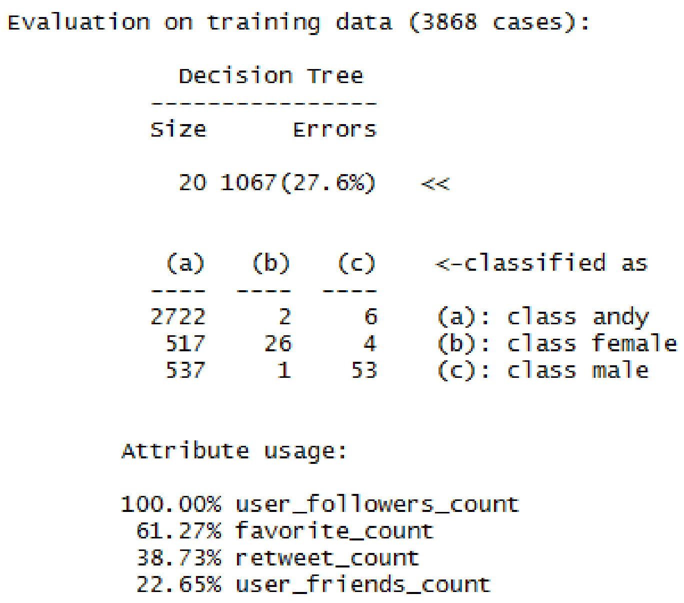


```
Decision tree:

user_followers_count > 254:
:...favorite_count <= 2: andy (2319/564)
:   favorite_count > 2:
:   :...user_followers_count <= 19474: female (10/3)
:       user_followers_count > 19474: andy (41/6)
user_followers_count <= 254:
:...retweet_count > 174: andy (622/265)
    retweet_count <= 174:
    :...user_followers_count > 246:
        :...user_friends_count <= 103: male (15)
        :   user_friends_count > 103: andy (14/5)
        user_followers_count <= 246:
        :...user_friends_count > 597:
            :...user_friends_count > 841: andy (44/11)
            :   user_friends_count <= 841:
            :   :...user_friends_count <= 604: female (8)
            :       user_friends_count > 604:
            :       :...retweet_count > 114: male (5/1)
            :           retweet_count <= 114:
            :           :...user_followers_count <= 207: andy (17/8)
            :               user_followers_count > 207:
            :               :...user_followers_count <= 226: female (11)
            :                   user_followers_count > 226: andy (3)
            user_friends_count <= 597:
            :...retweet_count <= 2: andy (335/75)
                retweet_count > 2:
```



```
Evaluation on training data (3868 cases):

	    Decision Tree
	  ----------------
	  Size      Errors

	    20 1067(27.6%)   <<


	   (a)   (b)   (c)    <-classified as
	  ----  ----  ----
	  2722     2     6    (a): class andy
	   517    26     4    (b): class female
	   537     1    53    (c): class male


	Attribute usage:

	100.00% user_followers_count
	 61.27% favorite_count
	 38.73% retweet_count
	 22.65% user_friends_count
```

Inspite of boosting there is not much of improvement in the error rate. Confusion matric after boosting.



```
Evaluation on training data (3868 cases):

	    Decision Tree
	  ----------------
	  Size      Errors

	    20 1067(27.6%)   <<


	   (a)   (b)   (c)    <-classified as
	  ----  ----  ----
	  2722     2     6    (a): class andy
	   517    26     4    (b): class female
	   537     1    53    (c): class male


	Attribute usage:

	100.00% user_followers_count
	 61.27% favorite_count
	 38.73% retweet_count
	 22.65% user_friends_count
```

Random Forest and bagging don’t apply as we are predicting categorical variables (Gender) based on numerical factors like user’s friends, user’s followers, retweets and favorite counts.

Research Question 3: Predict the topic/industry of rumors

1. Linear Regression

Linear and multivariable regression techniques are not applicable to the research question because the independent variable is not continuous. Our independent variable can assume one of 4 values - ‘City’, ‘Electronics’, ‘Politics’ or ‘Transport. Subsequently, there is no requirement for regularization either.

2. Logistic Regression

Our Independent variable is ‘Type’ which can take one of 4 values – ‘City’, ‘Electronics’, ‘Politics’ or ‘Transport. Since logistic regression can only be applied if it is binomial, we cannot use this technique train and test our data.

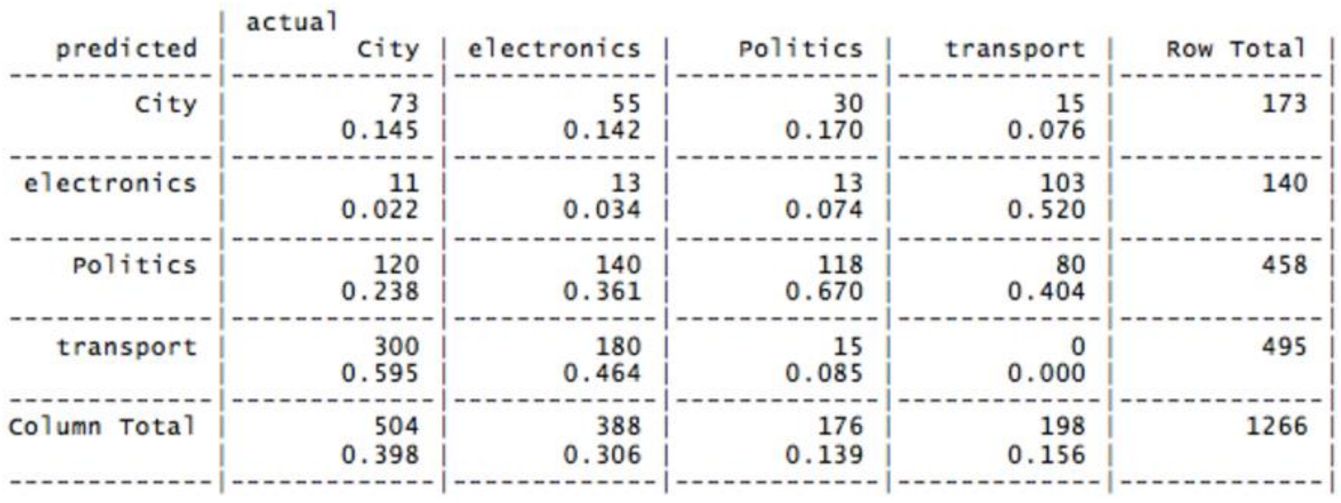
3. Naïve Bayes

Using **Naive Bayes**, we attempt to identify topics/industries of rumors. NB makes strong assumptions about conditional independence of the attributes. Our data contains 4 types of rumors: *electronics, transport, politics, and city*.

We randomly ordered the rows in our dataset and divided it into training and testing in the ratio of 75:25. The following table describes what percentage of our training and testing dataset contains tweets in the topic of City, Electronics, Politics and Transport.

	City	**Electronics**	**Politics**	**Transport**
Training	0.3867632	0.3073940	0.1602896	0.1455533
Testing	0.3981043	0.3064771	0.1390205	0.1563981

The following is the confusion matrix that is generated.



```
             | actual
   predicted |        City | electronics |    Politics |   transport |   Row Total |
-------------|-------------|-------------|-------------|-------------|-------------|
        City |          73 |          55 |          30 |          15 |         173 |
             |       0.145 |       0.142 |       0.170 |       0.076 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
 electronics |          11 |          13 |          13 |         103 |         140 |
             |       0.022 |       0.034 |       0.074 |       0.520 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
    Politics |         120 |         140 |         118 |          80 |         458 |
             |       0.238 |       0.361 |       0.670 |       0.404 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
   transport |         300 |         180 |          15 |           0 |         495 |
             |       0.595 |       0.464 |       0.085 |       0.000 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
Column Total |         504 |         388 |         176 |         198 |        1266 |
             |       0.398 |       0.306 |       0.139 |       0.156 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
```

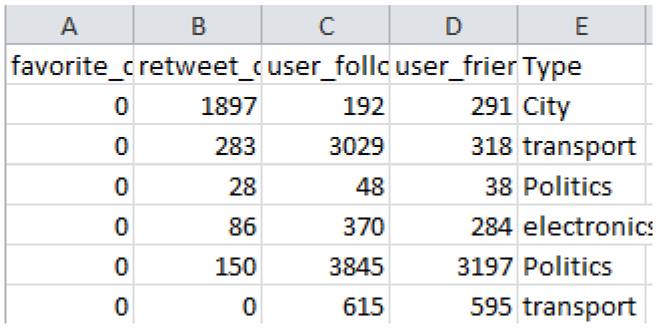
The classifier could fairly classify most tweets that were related to politics (67% of tweets classified correctly). However, it did not classify tweets in the other industries very well - city (14.5%), electronics (3.4%), and transport (0%).

After repeating the process with Laplace estimator, prediction on tweets related to politics improved from 67% to 74.4%, which prediction on tweets in other industries did not improve. The confusion matrix is show below.

```
             | actual
   predicted |        City | electronics |    Politics |   transport |   Row Total |
-------------|-------------|-------------|-------------|-------------|-------------|
        City |          63 |          41 |          22 |          15 |         141 |
             |       0.125 |       0.106 |       0.125 |       0.076 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
 electronics |           4 |          13 |           8 |         102 |         127 |
             |       0.008 |       0.034 |       0.045 |       0.515 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
    Politics |         137 |         148 |         131 |          81 |         497 |
             |       0.272 |       0.381 |       0.744 |       0.409 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
   transport |         300 |         186 |          15 |           0 |         501 |
             |       0.595 |       0.479 |       0.085 |       0.000 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
Column Total |         504 |         388 |         176 |         198 |        1266 |
             |       0.398 |       0.306 |       0.139 |       0.156 |             |
-------------|-------------|-------------|-------------|-------------|-------------|
```

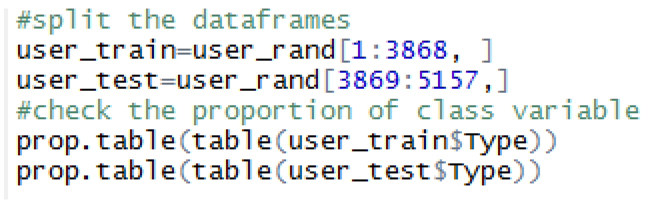
4. Decision Trees

To predict type of tweet based on favorite counts, retweets, user followers and user's friends we develop a classifier to predict type based on above mentioned variables. Following figure shows the small snippet of the data set which we will be using to train our **decision tree**. The dataset contains 5157 rows of rumor tweets like explosion of iphone 7, supreme metro card in New York, Trump's allegiance to Russia and missing news of black girls from DC. The texts of the tweets are removed thus subsetting the main dataset.

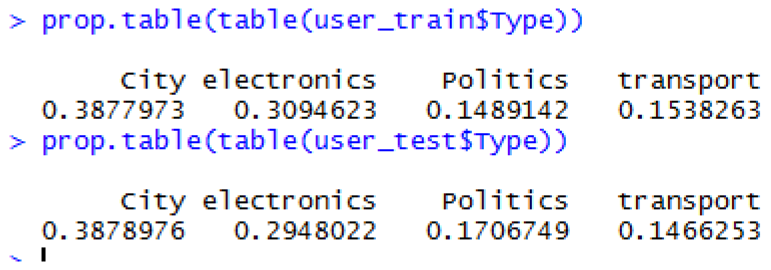


A	B	C	D	E
favorite_c	retweet_c	user_follc	user_frier	Type
0	1897	192	291	City
0	283	3029	318	transport
0	28	48	38	Politics
0	86	370	284	electronic:
0	150	3845	3197	Politics
0	0	615	595	transport

In order to start with decision tree we have to first divide the entire dataset into training and testing data.

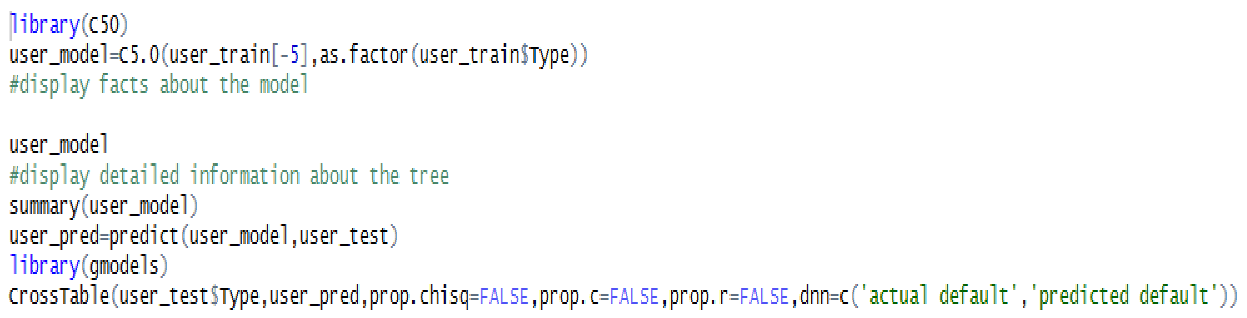


```
#split the dataframes
user_train=user_rand[1:3868, ]
user_test=user_rand[3869:5157,]
#check the proportion of class variable
prop.table(table(user_train$Type))
prop.table(table(user_test$Type))
```



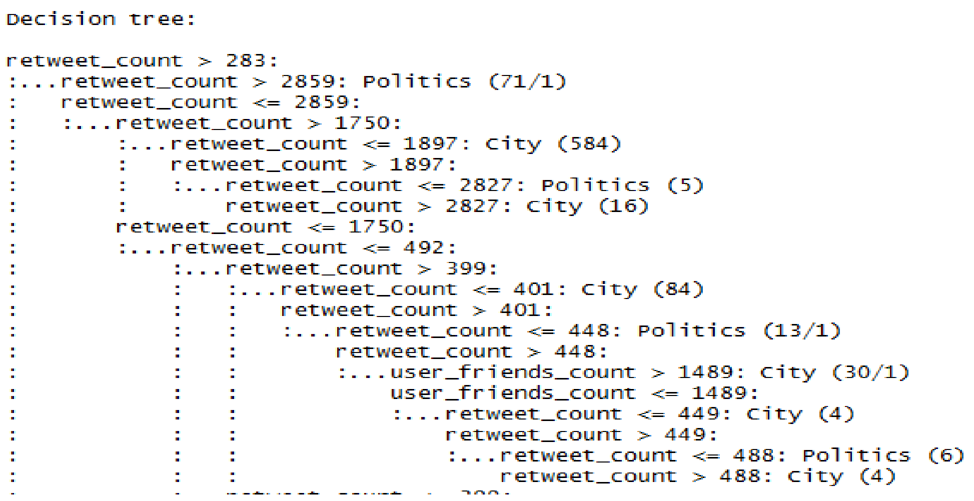
```
> prop.table(table(user_train$Type))

      City electronics    Politics   transport
 0.3877973   0.3094623   0.1489142   0.1538263
> prop.table(table(user_test$Type))

      City electronics    Politics   transport
 0.3878976   0.2948022   0.1706749   0.1466253
```

The above figure shows that the portion of tweets with types city, electronics ,politics, transport in the training and testing dataset is almost equal, thus there is no bias in the testing and training dataset. Following commands are executed in order to train the decision tree. The column ***Type*** (city, electronics ,politics, transport) is used as factors



```
library(C50)
user_model=C5.0(user_train[-5],as.factor(user_train$Type))
#display facts about the model

user_model
#display detailed information about the tree
summary(user_model)
user_pred=predict(user_model,user_test)
library(gmodels)
CrossTable(user_test$Type,user_pred,prop.chisq=FALSE,prop.c=FALSE,prop.r=FALSE,dnn=c('actual default','predicted default'))
```

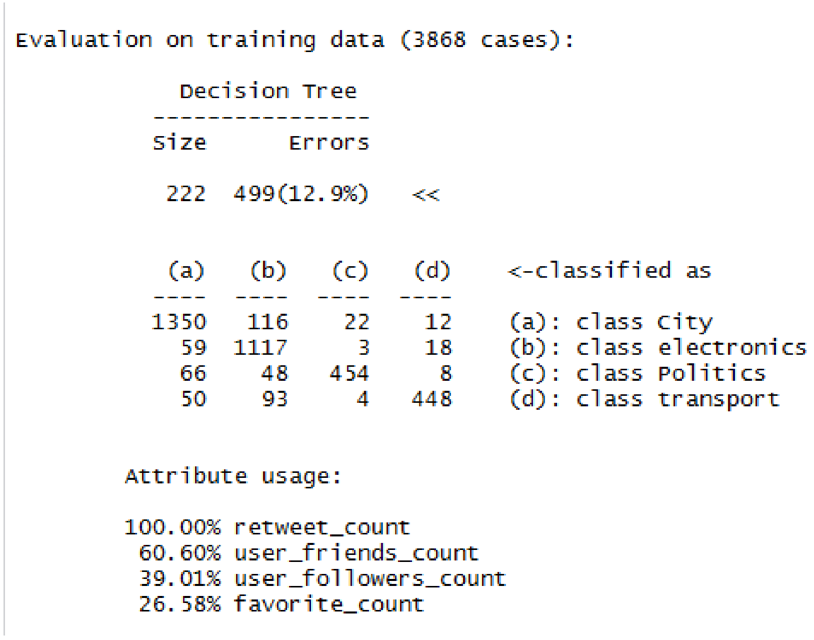
The figure below shows the Decision tree .Details of the decision tree is in Appendix C.



```
Decision tree:

retweet_count > 283:
:...retweet_count > 2859: Politics (71/1)
:   retweet_count <= 2859:
:   :...retweet_count > 1750:
:       :...retweet_count <= 1897: City (584)
:       :   retweet_count > 1897:
:       :   :...retweet_count <= 2827: Politics (5)
:       :       retweet_count > 2827: City (16)
:       retweet_count <= 1750:
:       :...retweet_count <= 492:
:           :...retweet_count > 399:
:           :   :...retweet_count <= 401: City (84)
:           :   :   retweet_count > 401:
:           :   :   :...retweet_count <= 448: Politics (13/1)
:           :   :       retweet_count > 448:
:           :   :       :...user_friends_count > 1489: City (30/1)
:           :   :           user_friends_count <= 1489:
:           :   :           :...retweet_count <= 449: City (4)
:           :   :               retweet_count > 449:
:           :   :               :...retweet_count <= 488: Politics (6)
:           :   :                   retweet_count > 488: City (4)
```

Interpreting the decision tree: If retweets count is greater than 283 and if retweet count is greater than 2859 then its politics related tweet.

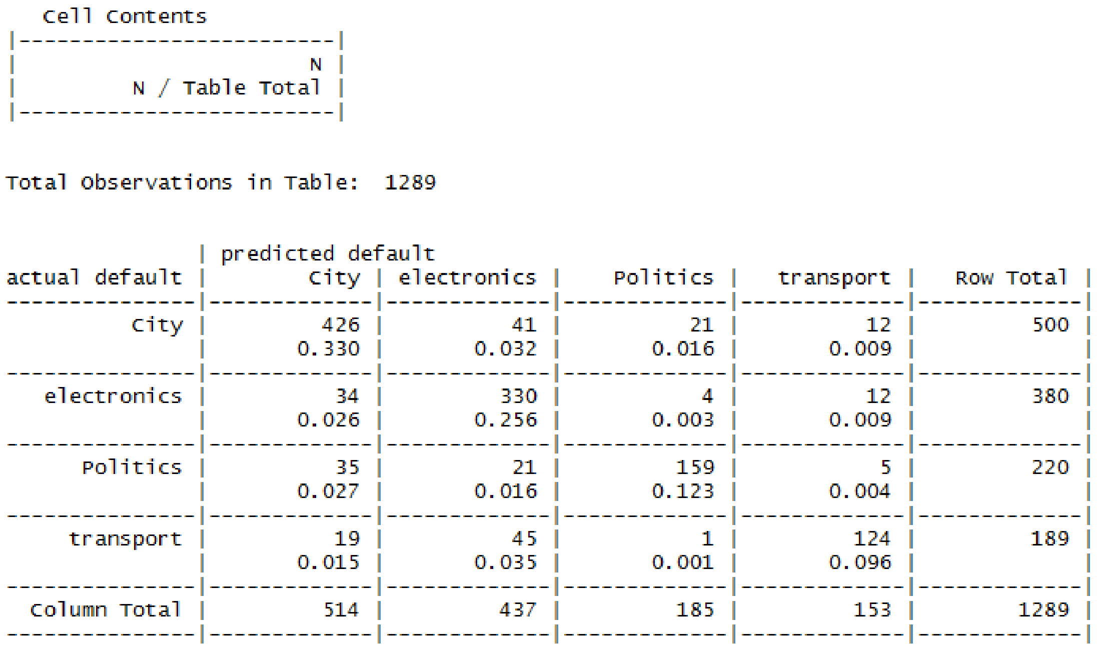
The following figure shows error rate on training set. Displaying the effectivity of the training on the training set of data. The decision tree was able correctly trained on the training set of the data with an error rate of 12.9%. 1350 rumors were classified as city related rumor while 116 of those were mislabeled as tweets belonging to electronics, 22 of city related tweets were mislabeled as politics and 12 were mislabeled as transport related tweets.



```
Evaluation on training data (3868 cases):

            Decision Tree
          ----------------
          Size      Errors

           222  499(12.9%)   <<


           (a)   (b)   (c)   (d)    <-classified as
          ----  ----  ----  ----
          1350   116    22    12    (a): class City
            59  1117     3    18    (b): class electronics
            66    48   454     8    (c): class Politics
            50    93     4   448    (d): class transport


        Attribute usage:

        100.00% retweet_count
         60.60% user_friends_count
         39.01% user_followers_count
         26.58% favorite_count
```

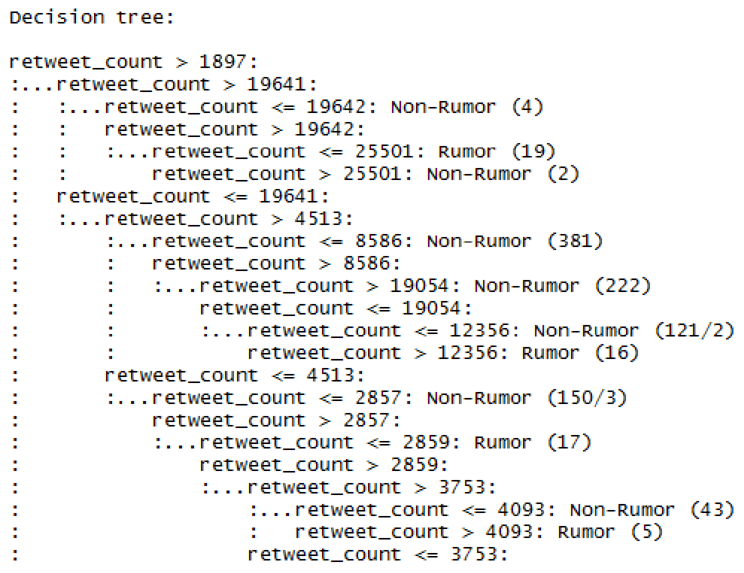
The following figure shows the **confusion matrix**.

The confusion matrix tells that the classifier predicted 426 times city related tweets correctly while mislabeled 41 of those tweets as electronics ,21 of those tweets as politics, 12 of those tweets as transport. The classifier correctly predicts 330 tweets related to electronics.159 of politics related tweets were correctly predicted.124 of transport related tweets were correctly determined.



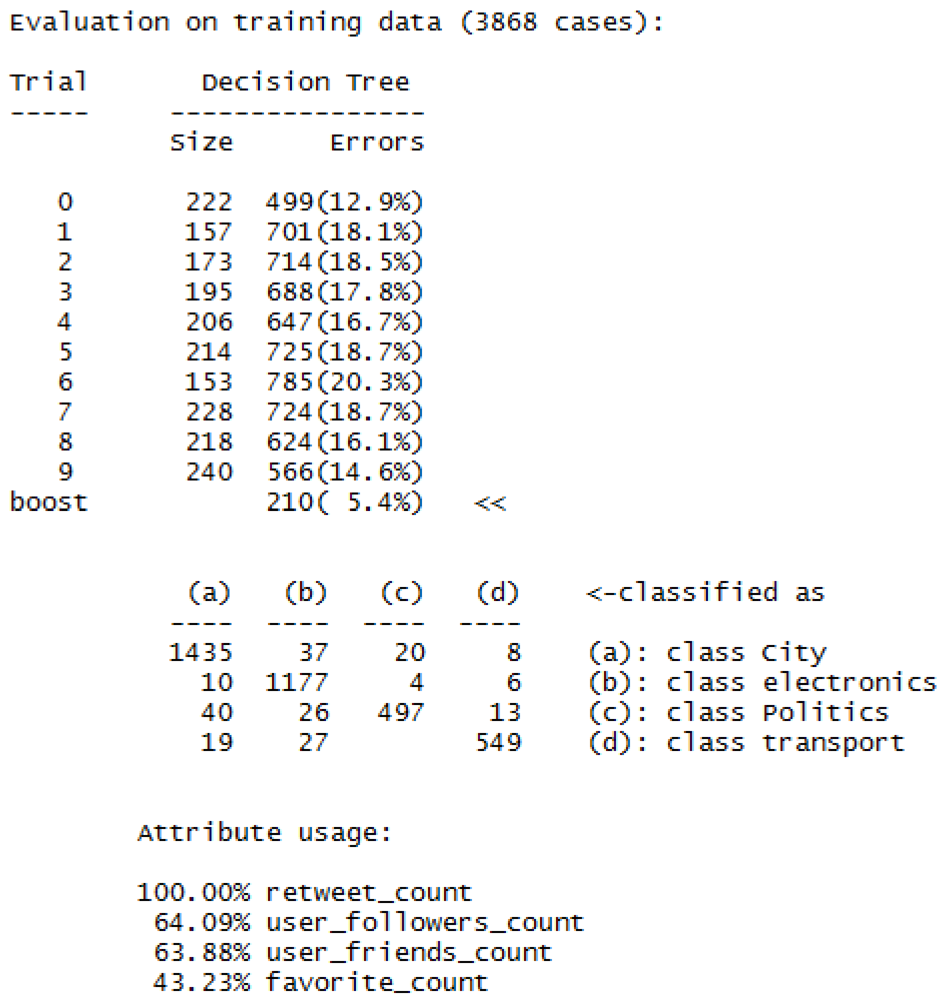
```
   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  1289


               | predicted default
actual default |         City |  electronics |     Politics |    transport |    Row Total |
---------------|--------------|--------------|--------------|--------------|--------------|
          City |          426 |           41 |           21 |           12 |          500 |
               |        0.330 |        0.032 |        0.016 |        0.009 |              |
---------------|--------------|--------------|--------------|--------------|--------------|
   electronics |           34 |          330 |            4 |           12 |          380 |
               |        0.026 |        0.256 |        0.003 |        0.009 |              |
---------------|--------------|--------------|--------------|--------------|--------------|
      Politics |           35 |           21 |          159 |            5 |          220 |
               |        0.027 |        0.016 |        0.123 |        0.004 |              |
---------------|--------------|--------------|--------------|--------------|--------------|
     transport |           19 |           45 |            1 |          124 |          189 |
               |        0.015 |        0.035 |        0.001 |        0.096 |              |
---------------|--------------|--------------|--------------|--------------|--------------|
  Column Total |          514 |          437 |          185 |          153 |         1289 |
---------------|--------------|--------------|--------------|--------------|--------------|
```

Next we apply ***boosting*** and get the following new decision tree. The figure shows a small snippet of the decision tree obtained by boosting, Details of the result is available in Appendix C.



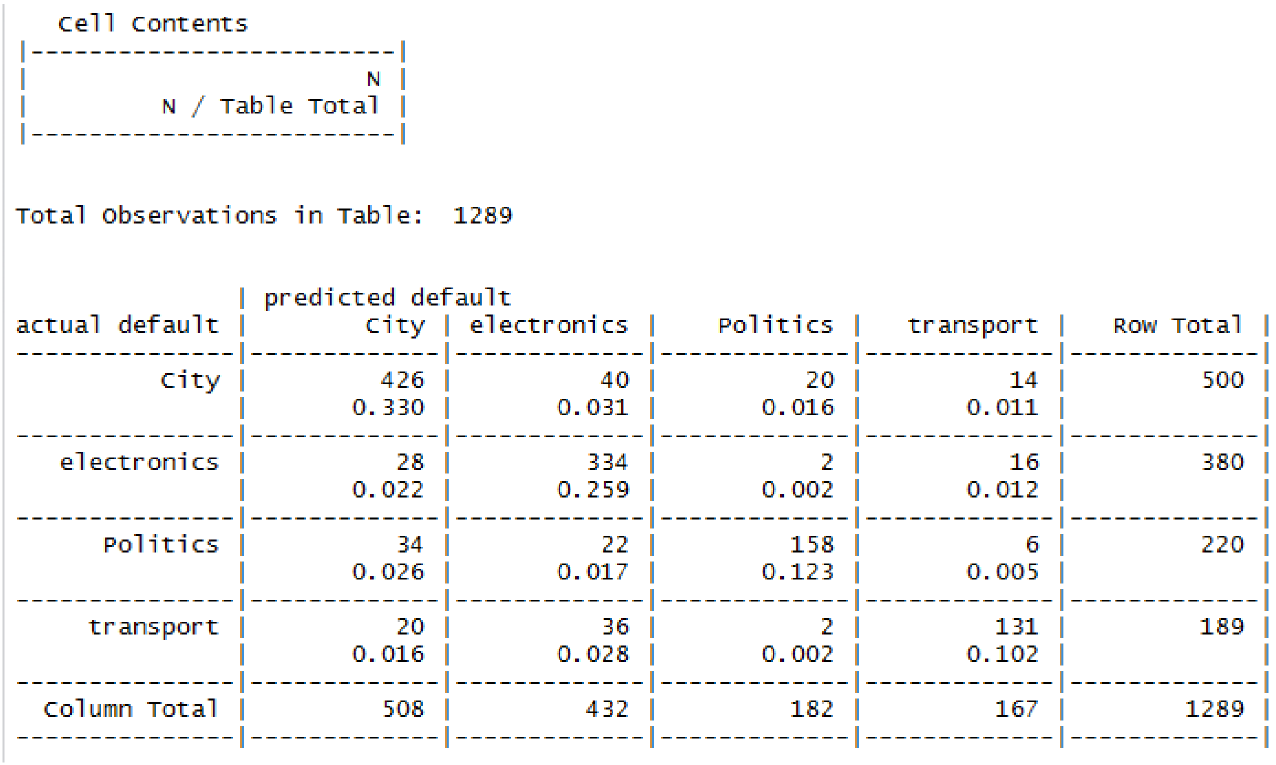
```
Decision tree:

retweet_count > 1897:
:...retweet_count > 19641:
:   :...retweet_count <= 19642: Non-Rumor (4)
:   :   retweet_count > 19642:
:   :   :...retweet_count <= 25501: Rumor (19)
:   :       retweet_count > 25501: Non-Rumor (2)
:   retweet_count <= 19641:
:   :...retweet_count > 4513:
:       :...retweet_count <= 8586: Non-Rumor (381)
:       :   retweet_count > 8586:
:       :   :...retweet_count > 19054: Non-Rumor (222)
:       :       retweet_count <= 19054:
:       :       :...retweet_count <= 12356: Non-Rumor (121/2)
:       :           retweet_count > 12356: Rumor (16)
:       retweet_count <= 4513:
:       :...retweet_count <= 2857: Non-Rumor (150/3)
:           retweet_count > 2857:
:           :...retweet_count <= 2859: Rumor (17)
:               retweet_count > 2859:
:               :...retweet_count > 3753:
:                   :...retweet_count <= 4093: Non-Rumor (43)
:                   :   retweet_count > 4093: Rumor (5)
:                   retweet_count <= 3753:
```

Following figure shows evaluation of training set after boosting. The error reduces to 5.4%



```
Evaluation on training data (3868 cases):

Trial       Decision Tree
-----     ----------------
          Size      Errors

   0       222  499(12.9%)
   1       157  701(18.1%)
   2       173  714(18.5%)
   3       195  688(17.8%)
   4       206  647(16.7%)
   5       214  725(18.7%)
   6       153  785(20.3%)
   7       228  724(18.7%)
   8       218  624(16.1%)
   9       240  566(14.6%)
boost           210( 5.4%)   <<


           (a)   (b)   (c)   (d)    <-classified as
          ----  ----  ----  ----
          1435    37    20     8    (a): class City
            10  1177     4     6    (b): class electronics
            40    26   497    13    (c): class Politics
            19    27         549    (d): class transport


        Attribute usage:

        100.00% retweet_count
         64.09% user_followers_count
         63.88% user_friends_count
         43.23% favorite_count
```

After boosting the classifier makes optimum usage of the attributes. Following ***figure displays confusion matrix after boosting.***



```
   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  1289


               | predicted default 
actual default |        city | electronics |    Politics |   transport |   Row Total | 
---------------|-------------|-------------|-------------|-------------|-------------|
          city |         426 |          40 |          20 |          14 |         500 | 
               |       0.330 |       0.031 |       0.016 |       0.011 |             | 
---------------|-------------|-------------|-------------|-------------|-------------|
   electronics |          28 |         334 |           2 |          16 |         380 | 
               |       0.022 |       0.259 |       0.002 |       0.012 |             | 
---------------|-------------|-------------|-------------|-------------|-------------|
      Politics |          34 |          22 |         158 |           6 |         220 | 
               |       0.026 |       0.017 |       0.123 |       0.005 |             | 
---------------|-------------|-------------|-------------|-------------|-------------|
     transport |          20 |          36 |           2 |         131 |         189 | 
               |       0.016 |       0.028 |       0.002 |       0.102 |             | 
---------------|-------------|-------------|-------------|-------------|-------------|
  Column Total |         508 |         432 |         182 |         167 |        1289 | 
---------------|-------------|-------------|-------------|-------------|-------------|
```

Random Forest and bagging don't apply as we are predicting categorical variables (type of tweet) based on numerical factors like user's friends, user's followers, retweets and favorite counts.

Comparative analysis

Methods

Multiple methods were available for the team to construct the prediction models needed for each research question. In particular, the characteristics of data and the research questions guided the team to the certain research methods (See Table below).

Methods for Research Questions				
	Linear Regression	**Logistic Regression**	**Naïve Bayes**	**Decision Trees & Random Forest**
RQ1. To predict if tweets are rumor		O	O	O
RQ2. To predict who are spreading rumors			O	O
RQ3. To predict types of rumors			O	O

RQ1. To predict if tweets are rumor

For RQ1, logistic regression, naïve bayes, and decision trees were utilized. Collectively, retweet_count was the most significant predictor of rumor tweets (identified by logistic regression and decision trees). Decision trees (84.35%) showed better capability in predicting rumors when compared to Naïve Bayes (63.5%).

	Linear regression	Logistic regression	Naive bayes	Decision Trees and Random Forest
RQ1. To predict if Tweets are rumor		☐	☐	☐

Logistic regression. Logistic regressions results showed that retweet_count, favorite_count, and a number of user_friends_count are statistically significant. Logistic regression helped us to identify the significant predictor of rumor tweets. The mean value of probability of the tweets being rumor using training model is 0.458. In other words, based on mean of all parameters, the probability of a tweet being a rumor is 45.8%.

Naïve Bayes. Results from our Naïve Bayes model with Laplace estimator revealed that 63.5% of rumor tweets were correctly classified as rumor, while the other 36.5% are incorrectly classified as non-rumors.

Decision Trees. Results revealed the factors that criteria used for deciding rumor or non-rumors, as well as error rate of training set. The most impactful factor was retweet counts (99.99%): If retweets is greater than 19641 or less than 19643, it's non-rumor; then, less than 25500, it's a rumor. Following retweet_count, user_followers_count (51.76%) and favorite_count (21.55%) were used. Error rate was 19.4%. 372 rumor tweets out of 441 tweets (84.35%) were correctly categorized as rumors.

Moreover, applying boosting changed error rate (from 19.4% to 20.4%) as well as attribute usage (99.99% retweet_count; 90.99% user_followers_count; 90.59% favourite_count). Out of 441 tweets, 294 (66.66%) were correctly labelled as rumors.

RQ2. To predict who are spreading rumors.

For RQ2, the team particularly looked at the gender attribute. Naïve bayes and decision trees were used to build the prediction models. For decision trees, User_followers_count (100%) was the most used attribute, followed by favorite_count (61.27%), retweet_count (38.73%), and user_friends_count (22.65%).

	Linear regression	Logistic regression	Naive bayes	Decision Trees and Random Forest
RQ2. To predict who are spreading rumors			☐ ☐	☐ ☐

Collectively, prediction model was good for identifying N/A accounts (Naïve Bayes without Laplace Estimator: 81.1%, Decision trees: 67.6%). Also, Naïve Bayes with Laplace Estimator showed great improvement on prediction of male account (65.6%). However, most prediction on male and female was very low (Naïve Bayes without Laplace Estimator: 15.9% of males, 0.03% of female; Decision trees: 0.2% of female, and 0.9%).

Naïve Bayes. Naïve Bayes results revealed that 15.9% of males, 0.03% of female, and 81.1% of N/A were correctly labelled. Moreover, interestingly, use of Laplace estimator greatly improved identification of male. Specifically, with a dramatic improvement in man with Laplace Estimator, results from our Naïve Bayes model revealed that *65.6% of men*, 0.03% of female, and 33.2% of N/A were correctly classified.

Decision trees. Results revealed the factors that the criteria used for deciding gender, as well as error rate of training set. Decision trees correctly predicted 67.6% of N/A; 0.2% of female, and 0.9% of males Twitter users spreading rumors. The most used attribute for prediction was user_followers_count (100.00%), followed by favorite_count (61.27%), retweet_count (38.73%), and user_friends_count (22.65%).

RQ3. To predict the topics/industries of rumors

For RQ3, the team particularly looked at the topic of rumors, such as technology or politics. Naïve bayes and decision trees were used to build the prediction models. For decision trees, retweet_count (100.00%) was the most used attribute, followed by user_friends_count (60.60%), user_followers_count (39.01%), and favorite_count (26.58%).

	Linear regression	Logistic regression	Naive bayes	Decision Trees and Random Forest
RQ3. To predict types of rumors			□ □	□

Prediction model showed different results of prediction, Still overall, decision trees showed much better prediction. Naïve bayes model predicted politics very well (67%), while have unsatisfactory prediction on the other types, such as *city* (14.5%), *electronics* (3.4%), and *transport* (0%). This remained similar with Laplace estimator, as prediction politics improved (from 67% to 74.4%), while other industries did not improve.

On the other hand, decision trees showed much better results: 85.2% of city; 86.84% of electronics; 72.27% of politics; 65.60% of transport were correctly predicted. Applying boosting, City remained same; Electronics (87.89% <- 86.84%) and Transport (69.31% <- 65.60%) increased; and politics dropped (71.81% <- 72.27%)

Naïve Bayes. Naïve Bayes results revealed that 14.5% of city, 3.4% of electronics, 67% of politics, and 0% of transport were correctly labelled. Moreover, interestingly, use of Laplace estimator greatly improved identification of politics. Specifically, with a dramatic improvement in politics (from 67% to 74.4%) with Laplace Estimator, results from our Naïve Bayes model revealed that *12.5 of city, 3.4% of electronics, 74.4% of politics*, and 0% of transport were correctly classified.

Decision trees. Results revealed the factors that the criteria used for deciding rumor types, as well as error rate of training set. Decision trees correctly predicted 85.2% of city; 86.84% of electronics, 72.27% of politics and 65.60% of transport. The most used attribute for prediction was retweet_count (100.00%), followed by user_friends_count (60.60%), user_followers_count (39.01%), and favorite_count (26.58%).

After boosting, the prediction on electronics (from 86.87% to 87.89%) and transport increased (from 65.60% to 69.31%); politics decreased (from 72.27% to 71.81%); while the city remained same(85.2%). The most used attribute for prediction after boosting was retweet_count (100.00%), followed by user_ followers _count (from 39.01% to 64.09%), user_friends_count (from 60.60% to 63.88%), and favorite_count (from 26.58% to 43.23%).