Computer Science and Software Engineering Département

COMP6731 & COMP473
Pattern Recognition

Team Project Report On

Fake News Detection on Social Media

Name of Your Examiner on Demo Day: Dittimi Tamara Finide

ALL Team members:

Name	Family Name	StudentID#	Email ID
Prasanth	Ambalam Jawaharlal	40042116	santhtce@gmail.com
Anil	Heggodu Raghavendra	40042818	anilconcordia@gmail.com
Simarpreet	Singh	40049885	simar.admissions@gmail.com

Team Leader Full Name: Simarpreet Singh

Name of Your Professor: Javad Sadri

Name of Your Evaluator on Demo Day: Dittimi Tamara Finide

Submission Date of this report: 12-16-2017

TABLE OF CONTENTS

List of Diagrams

1. Abstract

The ease of access and rapid improvements in social media has enabled more than half of the world's population using it in their daily life. These sites not only act as a platform for staying connected with friends and exchanging opinions but also help to share and disseminate information. With the huge availability of information over the Social media platform, People consider them as the primary source of news. Are all the information over the social media credible and trustworthy? With the flexibility of anyone can share anything over the platform, Social Media is more prone to the spread of irrelevant and misleading information. This extensive spread of spam has the potential for extremely negative impacts on individuals and the society. Therefore, detecting the spam or false information on social media has gained attention of many researchers.

2. INTRODUCTION

"*A lie can travel halfway around the world before the truth has got its boots on*". According to the recent studies that compared how falsehoods and truths spread, on an average, it takes more than 12 hours for a false claim to be debunked online. A study on this topic over the social media found that a rumor which turns out to be true is often resolved within two hours of first emerging. But a rumor that proves false takes closer to 14 hours to be debunked[19].

The spread of rumors and false information among public can have serious impacts on our society. For example during the London riots in 2011[8], Twitter was used as a medium to spread rumors. Rioters spread rumors about certain incidents like London eye being set on fire, police beating up a 16 year old, rioters breaking into McDonalds, rioters attacking London Zoo and the animals being freed, attacking the children's hospital at Birmingham and army being deployed in bank which other users further tweeted, leading to the spread of these rumors. This led to panic in the city and the government had to take immediate steps to halt it. So, it is very important to identify the misleading information spreading across social media for the benefit of the society. Unreliable evidence of hoaxes, conspiracy theories and fake news on social media is so abundant that massive digital misinformation has been ranked among the top global risks for our society. To mitigate the negative effects caused by fake news/rumor–both to benefit the public and the news ecosystem. It's critical that we develop methods to automatically detect fake news on social media. The Fake News detection on social media is a classification problem that can be solved using various classification algorithms. This classification problem can be solved using algorithms such as Naïve Bayes, SVM, KNN [11][12][13] etc. We will be classifying this problem through all the above mentioned classifiers and also have designed a cascaded classifier by combining few classifiers. We use the Twitter dataset to extract various features and classify it using a relevant classifier to detect the hoax. The various features that are considered to classify are Length of the tweet, Number of words, number of hashtags, Number of retweets, number of swear language words, number of special words, number of URLs. Malicious users spreading rumors on Social media can be tracked down and the further spreading of these fake news can be identified and mined using the above suggested approach in a more effective way.

3. Development Environment

- **Programming language:** Python[20]
  Libraries: [9][10][11][12][13] Numpy, Scipy, sklearn , pandas, matplotlib, Beautiful Soup 4, Requests
- **IDE:** Pycharm

4. Short Literature Review

This paper[3] deals in ranking the tweets based on the credibility during high impact events using SVM, Naive Bayes and K-Nearest Neighbour algorithms. They use the dataset collected from Twitter website by scraping the twitter pages. On feeding the dataset, data is classified using the models derived using different classifiers. As a part of Post-processing steps, performance of the existing output is combined with other classifiers in a cascading way and also ranked the tweets based on the Naive Bayes output. Finally, implemented the maximum voting by feeding each test point to all classifiers. With the data analyzed, 28% of total tweets posted about an event was spam on an average. Around 67% of the total tweets posted about the event contained situational awareness information that was credible.

The paper completely deals with detection of spam and non-spam tweets mainly based on the features set. Training the model is done using SVM, Naive Bayes and KNN classifiers.We concluded to use the following features like number of URLs, number of swear words, number of spam words, length of tweet text, favorites, retweets which seems to be more reliable for the dataset, therefore we chose retweets, favorites and the sum of all other features to detect the class for the tweets dataset using different classifying algorithm. We evaluated its accuracy by comparing with the output of different classifying algorithms such as Naïve Bayes, KNN and SVM. The implementation in the paper resulted in an accuracy of 81%, so we aimed in gaining a much higher accuracy with the above mentioned feature set using only SVM classifying algorithm.

Note : All of the Project Members (Prasanth, Anil, Simarpreet) have taken Data Mining Course and did Data Mining project with the similar concept but with a totally different algorithm in a different language. Basically we have implemented SVM classifier from scratch in R Language.

5. Data Collection

Twitter has been used as the source of dataset. It is one of the major platforms used to spread news every day, and a number of fake news have been spreading over the social media. To collect the dataset, We have chosen one such high impact event occured in UK in 2011[8]. The UK riot occurred between 6th and 11th of August 2011, when thousands of people rioted in several cities and towns across England. We tried to collect the data by using the streaming API of twitter but unfortunately twitter will provide only the past seven days of data which is not useful in our case because the UK riot happened in 2011, so we scrapped the Twitter HTML web page by scrolling the tweets infinitely to get the historical tweets during August 2011.

How ?:

We have used a open source python script[7] to scrap the historic data from the Twitter web page. We searched the tweets based on various hash tags like #**ukriot**, #**londonriot**, #**ukriot2011** and also filtered it for specific date during which the riot happened.We got around 80,000 tweets as raw data which included the following features extracted in a csv file.

Raw data:

- Tweet Text
- Tweet Date/Time
- Re-tweets.
- Location of Tweet
- User Details
- No of favorites

The extracted raw data is filtered with the date between Aug 6 to Aug 11 2011[7][8], which corresponds to the exact duration of London riots 2011. The filtered raw data corresponds to 10,000 datasets.

We have divided the 10,000 scrapped tweets into two csv files:

- RawTrainingDataSet.csv
- RawTestDataSet.csv

Data preprocessing, feature selection, feature extraction has been performed on both training and test data set. The training dataset is used for modelling the classifiers and the model is used for predicting the test data.

Snapshot of the data set:-

	A	B	C	D	E	F	G	H
1	INFO:root:1-[2011-08-06 19:59:59]	Facebook Roundup: #####londonsSchoo	100000000000000000	64576598	Frank Barrett	64576598	56	67
2	INFO:root:2-[2011-08-06 19:59:59]	we face racism in #####londonRiots####	100000000000000000	65070907	C.	65070907	7	1
3	INFO:root:3-[2011-08-06 19:59:59]	1 o'clock a.m. (#####londonRiots http:/	100000000000000000	311554864	New Hour In ###londo	3.12E+08	2	1
4	INFO:root:4-[2011-08-06 19:59:57]	@AmyyStretton if I ever go to London_ t	100000000000000000	80579173	Courtney	80579173	2	3
5	INFO:root:5-[2011-08-06 19:59:54]	Lol #LondonRiotsjus spit a he'll of ALOT c	100000000000000000	168852959	She.	1.69E+08	4	3
6	INFO:root:6-[2011-08-06 19:59:54]	We are at #LondonRiotsZoo Attacking th	100000000000000000	66514060	Henry Kwabena Pecku	66514060	76	86
7	INFO:root:8-[2011-08-01 19:59:59]	@Alphs121 watch @jaywonjuwonlo 's liv	98200000000000000	246832103	Dj Lanre Factory78	2.47E+08	3	2
8	INFO:root:9-[2011-08-07 19:59:59]	1 o'clock a.m. (#LondonRiots http://che	100000000000000000	311554864	New Hour In London	3.12E+08	1	0
9	INFO:root:10-[2011-08-03 19:59:54]	RT @joeashtonsing New Guidelines for P	98900000000000000	67898811	The London Bike Bot	67898811	2	3
10	INFO:root:11-[2011-08-10 19:59:59]	Check this video out -- VIVA BROTHER -	101000000000000000	58164582	ICantFindMyMind	58164582	1	0
11	INFO:root:12-[2011-08-02 19:59:38]	HIV epidemics emerging in M.East_ N.Afr	98500000000000000	66619037	Somdutt Bose	66619037	5	3
12	INFO:root:13-[2011-08-08 19:59:59]	RT @Rockstar_Yoni @LondonMayor: 17(	101000000000000000	199443532	not so spooky Nate	1.99E+08	67	57
13	INFO:root:14-[2011-08-04 19:59:53]	London is a TT !! :)	99300000000000000	24529046	Life of Leesha	24529046	2	2
14	INFO:root:15-[2011-08-09 19:59:59]	RT :Lol #LondonRiotsjus spit a he'll of AL(	101000000000000000	311554864	New Hour In London	3.12E+08	4	3
15	INFO:root:16-[2011-08-05 19:59:59]	1 o'clock a.m. (#LondonRiots http://che	99600000000000000	311554864	New Hour In London	3.12E+08	2	1
16	INFO:root:17-[2011-08-06 19:59:53]	1 o'clock a.m. (#LondonRiots http://che	100000000000000000	215313949	Jack Maple	2.15E+08	1	1
17	INFO:root:18-[2011-08-01 19:59:59]	RaminKPhan haha nice_ I'm pretty much	98200000000000000	235797095	London Classifieds	2.36E+08	4	6
18	INFO:root:19-[2011-08-07 19:59:59]	LONDON>For Sale>Tickets>V Festival tic	100000000000000000	174223944	Florence Herbert	1.74E+08	98	69
19	INFO:root:20-[2011-08-03 19:59:49]	Stop Racism @Pier59Studios Follow @Da	98900000000000000	7267552	sas	7267552	65	81
20	INFO:root:21-[2011-08-10 19:59:59]	RT :Lol #LondonRiotsjus spit a he'll of AL(	101000000000000000	31530002	Meech Golden	31530002	4	3
21	INFO:root:22-[2011-08-02 19:59:37]	RT @SooriMadsoos: "@cnnbrk: Death to	98500000000000000	17211615	Amanda	17211615	76	59
22	INFO:root:23-[2011-08-08 19:59:59]	I wonder what's going on in London Zoo	101000000000000000	42107321	CaptainCartmill	42107321	80	76
23	INFO:root:24-[2011-08-04 19:59:53]	Etsy Storque: In May 2010_ London-base	99300000000000000	278052879	Jordi Ollier-Howard	2.78E+08	5	2
24	INFO:root:25-[2011-08-09 19:59:59]	RT :Lol #LondonRiotsjus spit a he'll of AL(	101000000000000000	194786054	Leslie Brodie	1.95E+08	4	3

Fig. 1

Tweet Annotation :

We manually annotated the raw dataset into two categories - Spam and non-Spam. The annotations are done based on the rumors that were spread on twitter during the UK riots 2011. We have taken the below article [8] from "The Guardian" as reference for annotating the tweets and categorizing them as Spam or Non Spam. Approximately 10 different fake news have spread widely in twitter during that period.

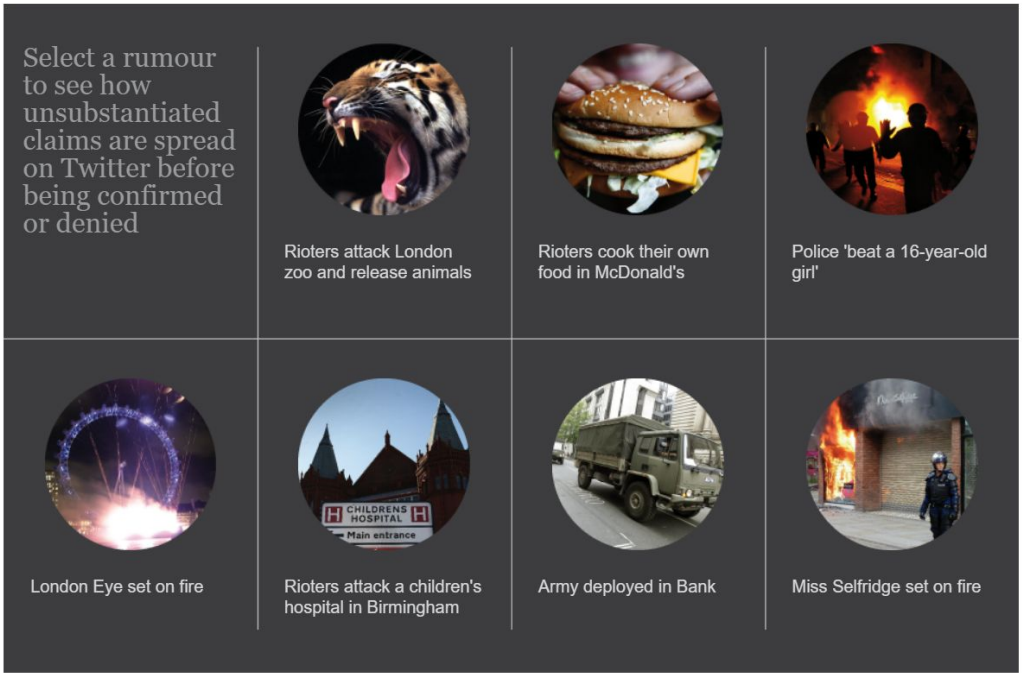


Fig. 2 Fake news spread on Twitter[8]

We introduced a new column named Class for representing the category(spam and non-spam) for each tweet in the raw dataset.

- If the tweet is spam, the class is assigned value -1
- If the tweet is non-spam, the class is assigned value 1

6. Methodology

1) Data preprocessing/Cleaning:-

The extracted raw dataset is processed in python using **read_csv()**[18] function of pandas library. Once the file is loaded, data types of required columns such as Date ect has been modified using **astype()** function in python. Then we checked for fields having missing values in our dataset, and removed the missing rows using **dropna()** function in python. We have also parsed the date column in the rawdata set using the regex expression in find function to obtain the date in the desired format.

At the end of preprocessing, we got two preprocessed csv files from the raw dataset

- Training_Cleaned.csv
- Test_Cleaned.csv

2) Feature Extraction :

Features are extracted from the cleaned data set and extracted features are used for the feature selection to model the classifier. The below features are obtained from the cleaned dataset.

a) **No of URLs:** We found the number of occurrences of URLs in each tweet in our extracted data using **count()** function in Python**.**

b) **No of Emoticons**: We found the number of occurrences of Emoticons such as smiley etc. in each tweet in our extracted dataset using **count()** function in Python.

c) **Length of Tweet**: We found the length of each tweet in our raw data using **translate()** function in Python.

d) **No of @ Mentions** : Found the number of occurrences of '@' symbol in each of the tweet using **count()** function.

e) **No of Hash tags :** Found the number of occurrences of '#' symbol in each of the tweet using **count()** function.

f) **Length of User Screen Name:** Found the length of the username that appears on the user's screen on his/her twitter account using **len()** function.

In order to identify a tweet content has Spam / Swear words[3], we have used two set of words, as inputs, containing Spam / Swear words with respect to the context of 2011 UK riots.

- Swear_list
- Spam_list

We have found the occurrence of the swear word and spam word in the tweet text using pattern matching function in python.

g) **Frequency of spam words**[3]: We looked for the spam words (ex: London Zoo etc.) corresponding to the event UK Riots and found the occurrence of these words using **count()** function in Python .

h) **Frequency of swear words**[3]**:** We looked for the spam words (ex: burnt etc.) corresponding to the event UK Riots and found the occurrence of these words using **count()** function in Python .

The outputs of the feature extractions are added as separate columns in the cleaned data sets for both test and training data and we got the feature extracted files.

- Training_Feature_Extraction_Dataset.csv
- Test_Feature_Extraction_Dataset.csv

3) FEATURE SELECTION/EXTRACTION:

The following features has been extracted from the preprocesses data set

1) Retweets on each Tweet.
2) Favorites.
3) Number of URLs in each Tweet.
4) Number of Spam words in each Tweet.
5) Number of Swear Words.
6) Number of Hash tags in each Tweet.
7) Number of @ Mentioned in each Tweet.
8) Combined new feature.

We have combined the Number of URLs in each Tweet, Number of Spam words in each Tweet, Number of Swear Words, Number of Hash tags in each Tweet, Number of @ Mentioned in each Tweet to form a new feature. After a number of trails, we finalized the below three features which results in good modeling for the classifier as below.

- No of retweets.
- No of favorites.
- Combined new feature.

The selected features after feature selection are added to two new csv files as testing and training data files.

- Training_Dataset.csv
- Test_Dataset.csv

4) CLASSIFICATION/MODELING:

SVM[12][21]**:**

- Training data is used to model the SVM classifier and is used for classify the test data.
- We have currently implemented Linear SVM Classifier.
- SVM hyperplane is used to segregate the classes as Spam or Non-Spam news in the given training data, i.e., given the labeled training data (*supervised learning*), the algorithm outputs an optimal hyperplane which categorizes the test data as Spam or Non-Spam. The notation used to define a hyperplane is:

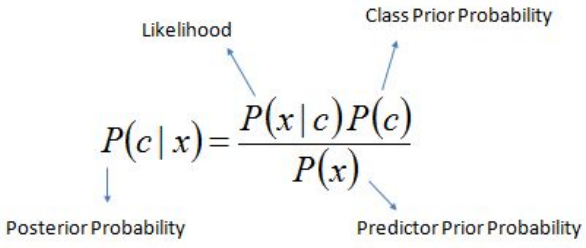
$$f(x) = \beta_0 + \beta^T x,$$

where β is known as the *weight vector* and β_0 as the *bias*.

- Our aim was to obtain a model with **minimum weight and maximum bias** by developing the fit method appropriately for 3 features set.
- Predict the new dataset using the obtained model.

Naive Bayes:

Naive Bayes[11][21] is the classification technique used to derive the classification model for extracted data. Using prior probability and likelihood for the feature, we will be computing the posterior probability of each class for each dataset. Based on the calculated posterior probability, the test data is classified. Naive Bayes by default uses **Gaussian distribution** internally, as it classifies data based on every input feature(Retweets, Favorites, New_Feature). We can change the type of distribution using density function.



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

KNN:

KNN[13][21] can be used for both classification and regression predictive problems. For each data points we take the voting among its nearest neighbours. K denotes the number of neighbours used for voting. Euclidean distance is used to compare and find the nearest neighbours. The data point is assigned to the class which has got more votes from the neighbour nodes. In our project, we are classifying the data into Spam and Non Spam. The data will classified as credible if the data is surrounded by credible data points or else classified as non-credible.

Post Processing

In the post-processing we have implemented three approaches:-

1. **Maximum Voting:**
   Each test data is fed to all the classifiers and each classifier will assign a Class for that test data. Based on the maximum voting, the test data will be assigned with a Class (Spam or Non-Spam)
2. **Assigning Weights to tweets:**
   The error rate from Naive Bayes classifier is used to assign weights to each tweet. If the error rate is less, the weight will be more and Vice versa. This weight is used to further classify the tweets into Fully Spam, Partially spam, Fully credible, Partially credible.
3. **Cascading:**
   We have designed a cascaded classifier based on the confidence of the naive bayes classifier output. If the confidence is low, the same data is passed to classifiers such as SVM or KNN and output of those classifiers will be used as the final result. Confidence level has been checked by setting up the threshold value.

7. IMPLEMENTATION

The implementation of the project is done in python language. Pattern recognition steps followed in our project are - data collection, data preprocessing, feature extraction, feature selection,modelling,evaluation and then post processing on our dataset and designed the classifier using SVM, KNN and Naive Bayes classification algorithms.

As a part of post processing, we have implemented maximum voting, cascading and assigning the credibility weights to each tweets.

Flow diagram:-

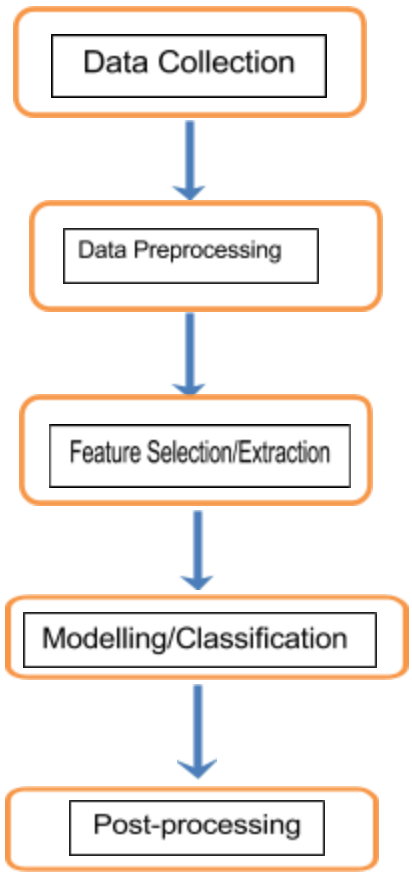


Fig. 3 Pattern Recognition Steps

Architecture of project:

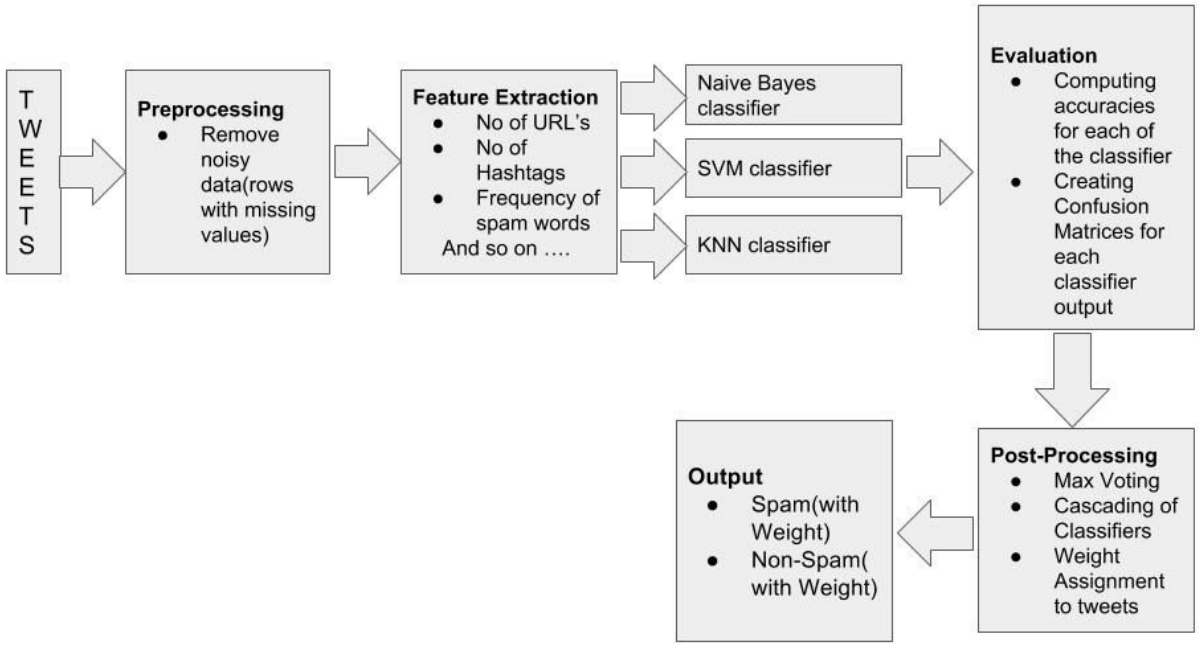


Fig 4 Project Architecture

Data collection:-

We are using Twitter as our data source and opted the datasets related to an event "UK riots". Initially by using twitter API', tried to extract the dataset however we couldn't extract due to restriction of the API's on number of days.[22]

We have used a open source python script[7] to scrap the historic data from the Twitter web page. We searched the tweets based on various hash tags like #ukriot, #londonriot, #ukriot2011 and also filtered it for specific date during which the riot happened.We got around 80,000 tweets as raw data which included the following features extracted in a csv file.

Data Preprocessing:-

Data set obtained in the Data collection steps had noisy data in terms of null or missing values. We wanted to remove these noises before feeding our data to classifiers for classification and also before identifying features. We loaded the data set using **pandas read_csv()**[18] function which in turn created the data frame of the dataset. Using the data frame created, we removed the null or missing values using **dropna()** function and restricted our data set to the Aug month by extracting the date using **pattern matcher**. We extracted around 10000 tweets as a cleaned dataset in the data preprocessing step. We have used 1500 tweets as training dataset and remaining around 8500 tweets as test dataset.

Feature Selection\Extraction:-

We are having the below 9 features extracted from the scrapped tweets :

1) Re-tweets on each Tweet.
2) Favorites.
3) Number of URLs in each Tweet.
4) Number of Spam words in each Tweet.
5) Number of Swear Words.
6) Number of Hash tags in each Tweet.
7) Number of @ Mentioned in each Tweet.
8) Combined new feature.

We have combined the Number of URLs in each Tweet, Number of Spam words in each Tweet, Number of Swear Words, Number of Hash tags in each Tweet, Number of @ Mentioned in each Tweet to form a new feature.

After a number of trails, we finalized the below three features which results in good modeling for the classifier as below

- No of retweets.
- No of favorites.
- Combined new feature.

Classification\Modeling:-

We classified our data by using three classifiers such as Support Vector Machine(SVM), Naive Bayes and K-Nearest Neighbour using inbuilt python libraries.

1. **Support Vector Machine(SVM):-**

SVM[12][21] is the said to be the best classifier for the binary classification problem. We have used the SVM inbuilt function defined in the python inbuilt library `sklearn`. Initially , we created an object of SVM using below syntax and parameters

```
SVM = svm.SVC(kernel='linear', C=1.0, gamma=2)
```

Then we defined the model using fit function of SVM object by passing the feature vector and class labels.

```
SVM.fit(FeatureVectorTraining, ClassLabelTraining)
```

Using the predict function of the SVM object, we predicted the class label for the test data and also computed the confusion matrix for the test data. We got the accuracy of around 88% for our test dataset of around 8500 tweets.

```
SVM.predict=(TestFeatureVector)
```

2. **Naive Bayes(NB)**[11][21]**:-**

We have used the Multivariate Gaussian Naive Bayes[11] inbuilt function defined in the python inbuilt library `sklearn.naive_bayes.` Initially , we created an object of Naive Bayes using below syntax:-

```
NB = GaussianNB()
```

Then we defined the model using fit function of NB object by passing the feature vector and class labels.

```
NB.fit(FeatureVectorTraining, ClassLabelTraining)
```

Using the predict function of the NB object, we predicted the class label for the test data and also computed the confusion matrix for the test data. We got the accuracy of around 80% for our test dataset of around 8500 tweets.

```
NB.predict=(TestFeatureVector)
```

3. **K-Nearest Neighbour(KNN)**[13][21]**:-**

We have used the K-Nearest Neighbour[13] inbuilt function defined in the python inbuilt library `sklearn.neighbors`. Initially , we created an object of KNN using below syntax and parameter:-

```
KNN = KNeighborsClassifier(n_neighbors=101)
```

We have used number of nearest neighbour to participate in deciding the class label for a given point as 101.Then we defined the model using fit function of KNN object by passing the feature vector and class labels.

```
KNN.fit(FeatureVectorTraining, ClassLabelTraining)
```

Using the predict function of the KNN object, we predicted the class label for the test data and also computed the confusion matrix for the test data. We got the accuracy of around 90% for our test dataset of around 8500 tweets.

```
KNN.predict=(TestFeatureVector)
```

Post processing:-

Assigning the weight:

- The error rate from Naive Bayes classifier is used as a benchmark to assign weights to each tweet.
- If the error rate is less, the weight assigned to that tweet will be more and Vice versa.
- Weight = 1 - Error_Rate
- This weight is used to further classify the tweets into Fully Spam, Partially spam, Fully credible, Partially credible.

Maximum Voting:

- Each classifier is trained with the same training data set.
- Each test data is fed to all the classifiers.
- Each classifier will assign a Class for that test data.
- Based on the maximum voting, the test data will be assigned with a Class (Spam or Non-Spam)

Cascading :

- We have designed a cascaded classifier based on the confidence of the naive bayes classifier output.
- If the confidence is low, the same data is passed to classifiers such as SVM or KNN and output of those classifiers will be used as the final result.
- Confidence level has been checked by setting up some threshold value.
- For eg; If the error rate is greater than 20%, the data will be passed to other classifiers and class label will be assigned based on its output.

8. EXPERIMENTAL RESULTS

Modelling/Classification

1. **Support Vector Machine(SVM) :-**

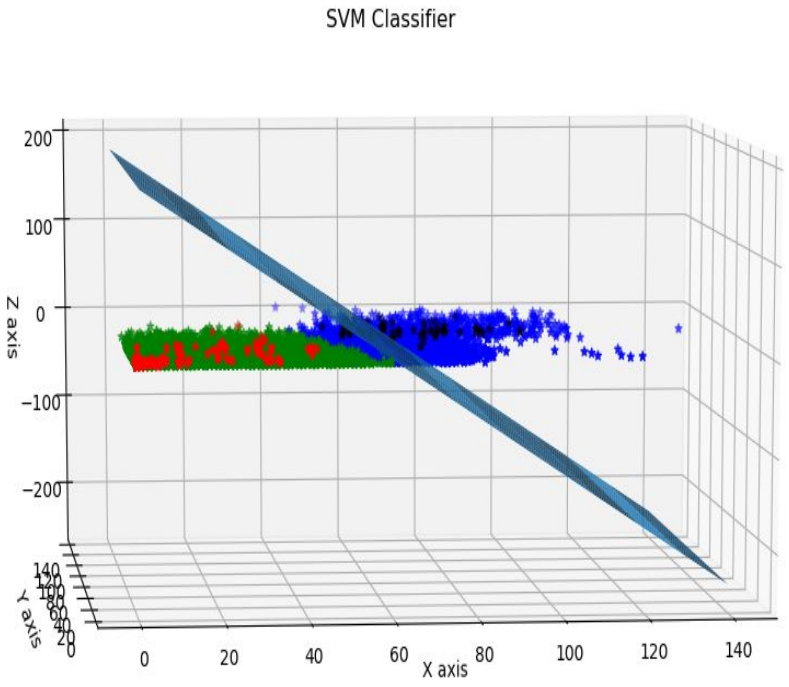


Fig. 5 SVM Classifier with Hyper Plane

```
Accuracy for SVM
88.4384023099
Confusion Matrix for SVM
[[ 894  105]
 [ 856 6457]]
```

Fig. 6 Confusion Matrix for SVM Classifier

Where,

- **Red** : Non-Spam for training data set
- **Black** : Spam for training data set
- **Green** : Non-Spam for test data set
- **Blue** : Spam for test data set

2. Naive Bayes:

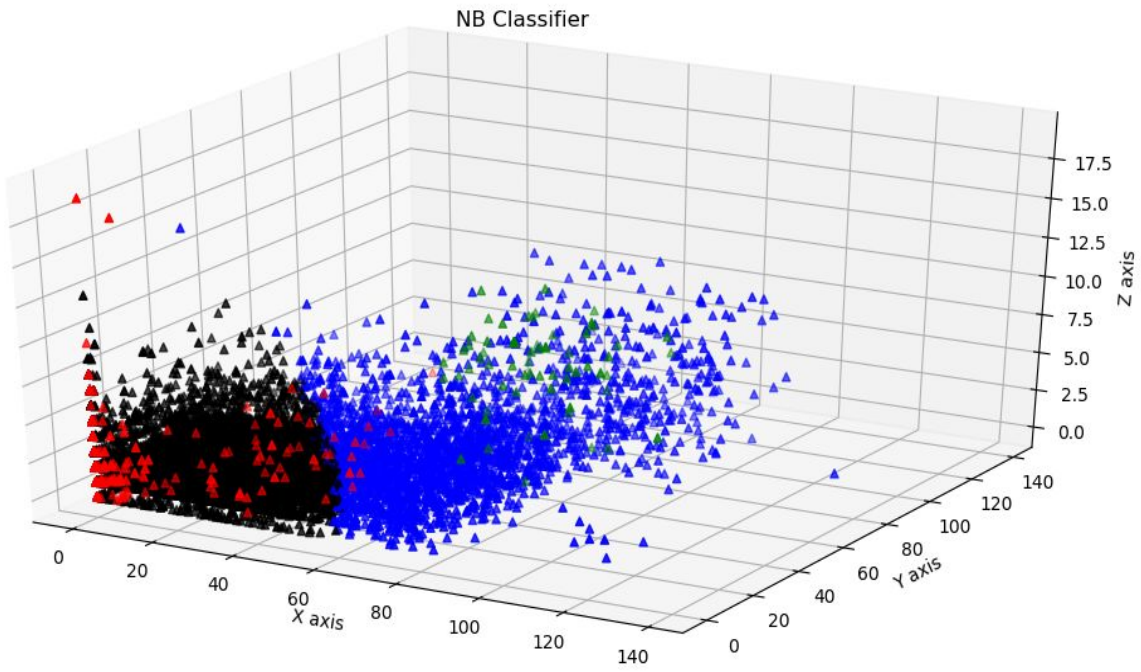


Fig.7 Naive Bayes Classifier

```
Accuracy for Naive Bayes
80.0769971126
Confusion Matrix for Naive Bayes
[[ 932   67]
 [1589 5724]]
```

Fig. 8 Confusion Matrix - Naive Bayes Classifier

Where,

- **Red** : Non-Spam for training data set
- **Green** : Spam for training data set
- **Black** : Non-Spam for test data set
- **Blue** : Spam for test data set

3. K-Nearest Neighbour(KNN):

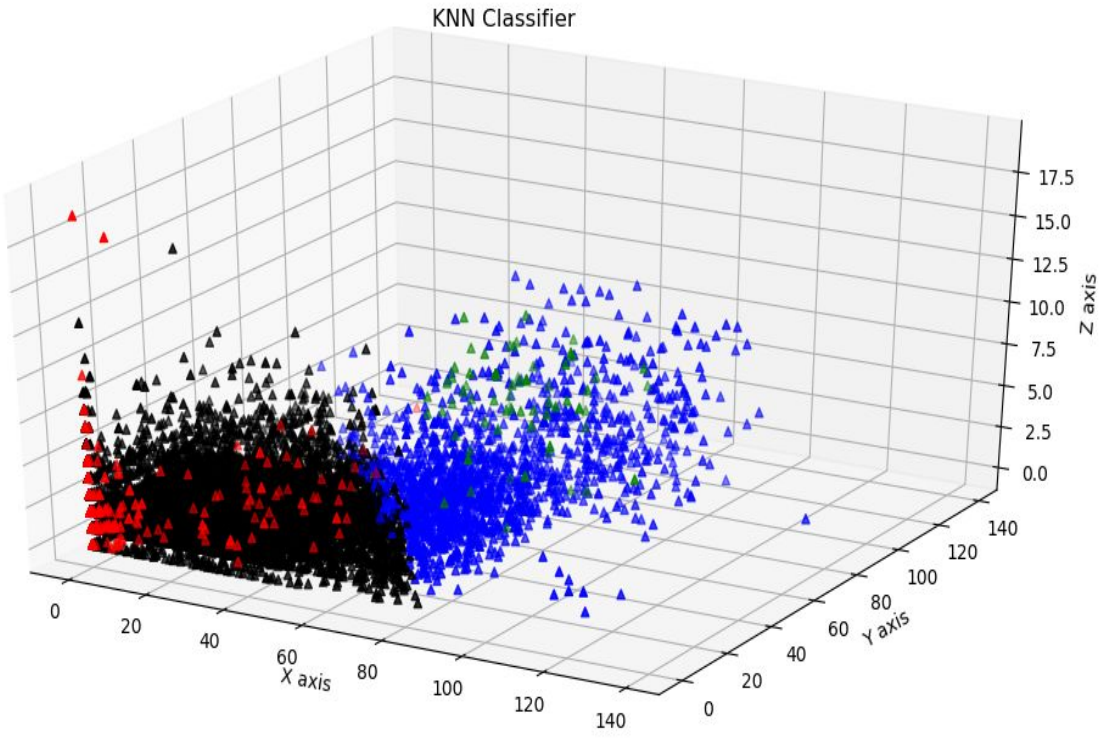


Fig.9 KNN Classifier

```
Accuracy for KNN
90.6159769009
Confusion Matrix for KNN
[[ 894  105]
 [ 675 6638]]
```

Fig.10 Confusion Matrix - KNN Classifier

Where,

- **Red** : Non-Spam for training data set
- **Green** : Spam for training data set
- **Black** : Non-Spam for test data set
- **Blue** : Spam for test data set

Post processing:

1. **Assigning the weight :-**

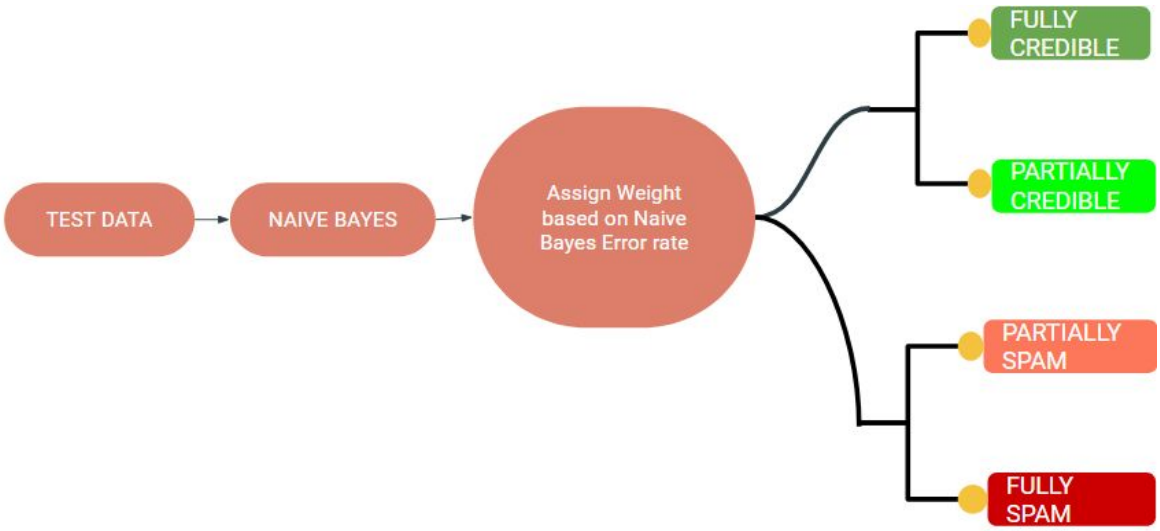


Fig.11 Assigning the Weights to Tweets

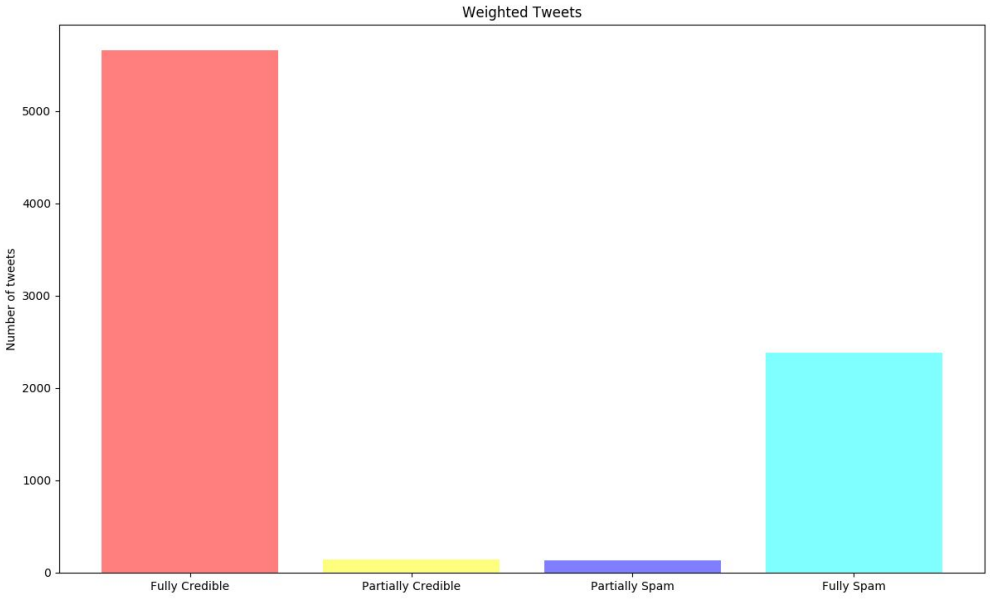


Fig. 12 Classifying tweets based on Weights

```
$$$$$$$$$$$$$$$$$$ Post Process: Analysis $$$$$$$$$$$$$$$$$$$$$$$$

Total Postprocessed Tweets:  8312

Fully Credible Tweets:  5633
Partially Credible Tweets:  158
Partially Spam Tweets:  127
Fully Spam Tweets:  2394
```

2. Maximum Voting :-

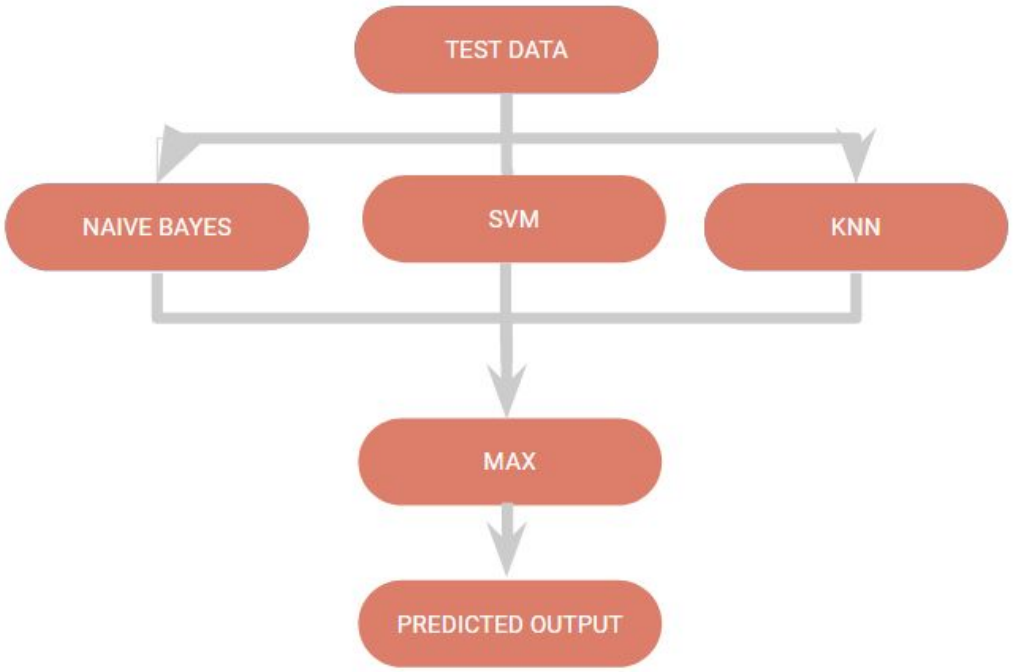


Fig. 13 Classifying tweets based on Maximum Voting classifier

```
Accuracy for Max Voting Classifier
87.2714148219
Confusion Matrix for Max Voting Classifier
[[ 907   92]
 [ 966 6347]]
```

3. Cascading :-

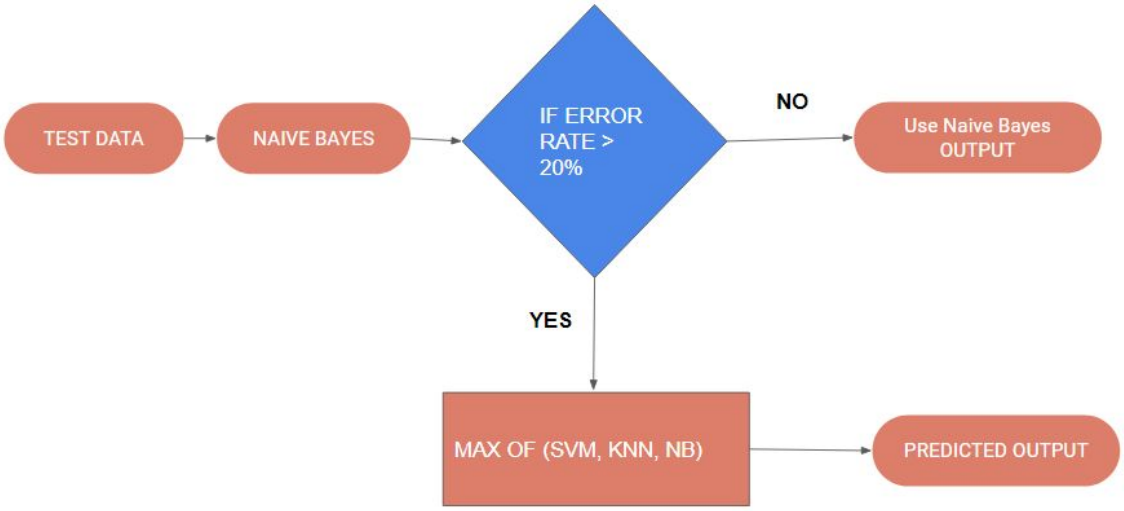


Fig. 14 Cascaded Classifier

```
Accuracy for Cascaded Classifier
81.5086621752
Confusion Matrix for Cascaded Classifier
[[ 928   71]
 [1466 5847]]
```

9. Discussion

With the comparison of the results among all the classifiers like Support Vector Machine, K Nearest Neighbor (KNN), Naïve Bayes, Decision tree and neural networks,we have found that KNN has the maximum accuracy followed by SVM. Since KNN is dependent on the value of the cluster K and is susceptible to noise, when we go for a larger dataset the accuracy could vary based on the cluster and noise attributes.

We could state that SVM as a better binary classifier than KNN.The below comparison states the performance comparison and accuracy evaluation among different classifiers with our extracted dataset.

10. Personal Conclusions and Suggestions

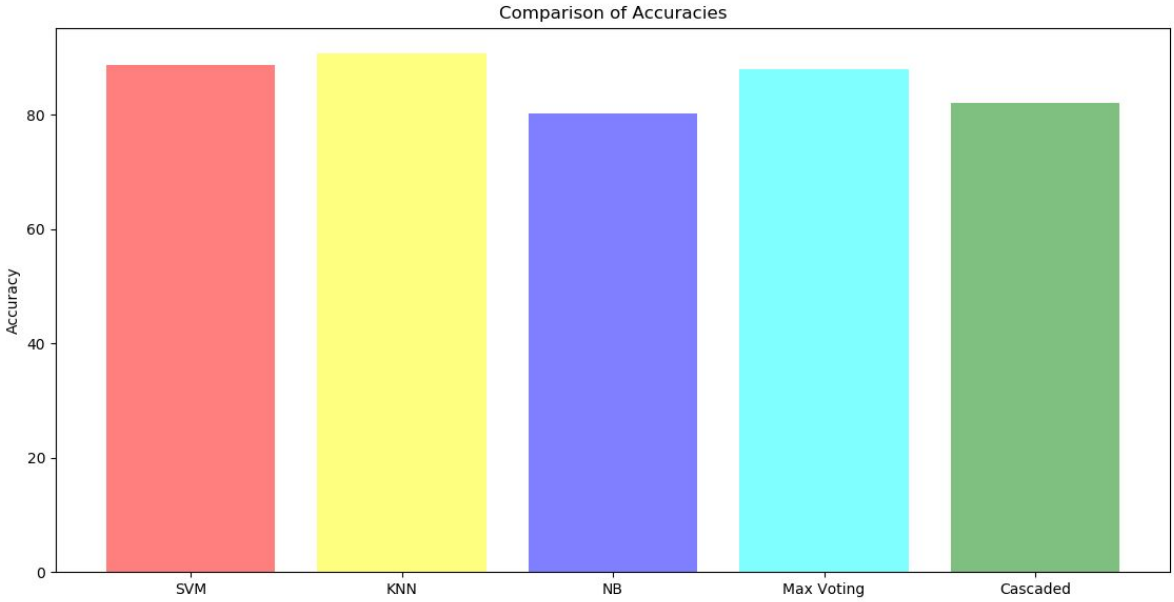


Fig.15 Comparison of Classifier Accuracy

We have obtained the maximum accuracy for K-Nearest Neighbour algorithm and Max Voting classifier with our test data, which is followed by SVM classifier with an accuracy rate of 84 percent. Naive Bayes gave a fair classification rate of around 80 percent, which was the lowest among all the algorithms that we have used.

First approach to improve is to try the same set of classifiers for a different data set , basically fetching the tweets for a different event and then trying to classify the data as Spam and Non Spam and finally observe the behaviour and accuracy of those classifiers, for that given dataset.

We can also improve our results by using different set of features to train our models. In addition to that we can use more number of features and apply some feature extraction technique such as PCA to reduce dimensions, at the same time preserving the information to obtain better classification rate. One more way is to apply Fisher Linear Discriminant on the feature set which will increase the gap between the classes to obtain better classification.The results with the cascaded classifier will also change, if we use any other classification algorithm apart from Naive Bayes, that we have used currently.

The cascading classifier can be permuted to get different combinations, so instead of using Naive Bayes Error rate as benchmark, we can use SVM error rate as confidence level and predict the data with other classifiers if the error rate of SVM exceeds the threshold limit.

11. References

[1] Gupta, A. and Kaushal, R., 2015, March. Improving spam detection in online social networks. In *Cognitive Computing and Information Processing (CCIP), 2015 International Conference on* (pp. 1-6). IEEE.

[2] Shu, K., Sliva, A., Wang, S., Tang, J. and Liu, H., 2017. Fake News Detection on Social Media: A Data Mining Perspective. *ACM SIGKDD Explorations Newsletter*, *19*(1), pp.22-36.

[3] Gupta, A. and Kumaraguru, P., 2012, April. Credibility ranking of tweets during high impact events. In *Proceedings of the 1st workshop on privacy and security in online social media* (p. 2). ACM.

[4] Rajdev, M. and Lee, K., 2015, December. Fake and spam messages: Detecting misinformation during natural disasters on social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2015 IEEE/WIC/ACM International Conference on* (Vol. 1, pp. 17-20). IEEE.

[5] Mendoza, M., Poblete, B. and Castillo, C., 2010, July. Twitter Under Crisis: Can we trust what we RT?. In *Proceedings of the first workshop on social media analytics* (pp. 71-79). ACM.

[6] Sahana, V.P., Pias, A.R., Shastri, R. and Mandloi, S., 2015, December. Automatic detection of rumoured tweets and finding its origin. In *Computing and Network Communications (CoCoNet), 2015 International Conference on* (pp. 607-612). IEEE.

[7] https://github.com/tomkdickinson/Twitter-Search-API-Python

[8] https://www.theguardian.com/uk/interactive/2011/dec/07/london-riots-twitter

[9]https://pythonprogramming.net/svm-optimization-python-machine-learning-tutorial/?completed=/svm-in-python-machine-learning-tutorial/

[10] Data Mining Project 6180, Fake News detection by implementing SVM from scratch using R

[11] http://scikit-learn.org/stable/modules/naive_bayes.html#multinomial-naive-bayes

[12] http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[13]http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

[14]https://stackoverflow.com/questions/36232334/plotting-3d-decision-boundary-from-linear-svm

[15] https://stackoverflow.com/questions/1985856/how-to-make-a-3d-scatter-plot-in-python

[16] https://matplotlib.org/gallery/mplot3d/scatter3d.html

[17] https://pythonspot.com/matplotlib-bar-chart/

[18] https://pandas.pydata.org/

[19]https://firstdraftnews.com/recent-research-reveals-false-rumours-really-do-travel-faster-and-further-than-the-truth/

[20] https://www.python.org/doc/

[21] Pattern Classification, 2nd Edition - Richard O. Duda, Peter E. Hart, David G. Stork

[22] https://developer.twitter.com/en/docs

12. Details of Division of the work among team members

Stages of the project	**Contributor**
Data Collection	Prasanth A J
Data Cleaning/Pre-processing	Anil H R
Feature Extraction and Selection	Simarpreet Singh
Classifier Implementation	Everyone
Post-processing	Everyone