Concordia Institute for Information Systems Engineering
Fall 2017

INSE6180: Security and Privacy Implications of Data Mining

Project Report
Fake News Detection on Social Media

Team

Simarpreet Singh	**(40049885)**
Prasanth Ambalam Jawaharlal	**(40042116)**
Anil Heggodu Raghavendra	**(40042818)**
Vaishnavi Venkatraj	**(40049798)**

INSE6180: Security and Privacy Implications of Data Mining

Vigneswar Mourouguessin (40057918)

Table of Contents

List of Diagrams

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.

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.

The spread of rumors and false information among public can have serious impacts on our society. For example during the London riots in 2011, 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 data mining classification algorithms. This classification problem can be solved using algorithms such as Naïve Bayes, SVM, decision trees etc. We will be classifying this problem through Support Vector Machines algorithm. 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

Related Work

Paper Summary:

Paper 1: Improving Spam Detection in Online Social Networks [1]

This paper deals with detection of spammers in a social media like Twitter based on number of features at tweet-level and user-level like followers, URLs, spam words, replies and hashtags. It is a ***combined approach of three classification algorithms***, such as ***Naive Bayes approach, General Activity Detection Clustering algorithm and Decision tree algorithm***, and finally comparing the results of the three algorithms. This approach seems to be a very accurate approach in identifying an account as spammer or non-spammer with 87.9% accuracy and detection of non-spammers was higher compared to the detection of spammers with 68.4%.This integrated algorithm was then compared with each of the learning algorithm, Naive Bayes, Clustering and Decision Trees. The results showed that Clustering algorithm performs better in detection of non-spam accounts but was very poor in detecting spam accounts. This algorithm was ***able to maintain the high accuracy of Clustering algorithm in detecting non-spam and at***

the same time, retain the accuracy of Naive Bayes in detecting Spammers accounts thereby, increasing the overall accuracy.

Paper 2: Fake News Detection on Social Media: A Data Mining Perspective [2]

This paper discusses various methods to identify false news spreading across social media and ***throws light on the existing general data mining framework*** inclusive of feature extraction using various features like News Content features and Social context features and model construction based on Knowledge and Style. They have taken ***four datasets*** BuzzFeed News, LIAR, BS Detector, CREDBANK and have ***summarized and compared*** the approaches of feature extraction process on all datasets. Generally, news data with annotations can be gathered in the following ways, Expert journalists, Fact-checking websites, Industry detectors, and Crowd-sourced workers. Finally they ***proposed evaluation metrics to compare the accuracies of the various algorithms***. They have also suggested future approaches using Data-oriented, Feature-oriented, Model-oriented and Application-oriented methods.

Paper 3: Credibility ranking of tweets during high impact events [3]

This paper deals in ranking the tweets based on the credibility during high impact events using ***SVM ranking algorithm and Blind Relevance Feedback*** algorithms. They use the dataset collected from Twitter using the ***Streaming API and Trends API***. On feeding the dataset, they rank the tweets based on the SVM ranking algorithm. Again they evaluate an enhancement to the above ranking technique by using pseudo feedback relevance re-ranking scheme, and finally evaluate the performance of Rank using the ***NDCG evaluation metric***. Based on the learned model, the algorithm predicts a ranking score and also performed ***four-fold cross validation*** of the obtained results. With the data analyzed, 30% of total tweets posted about an event contained situational information about the event while 14% was spam on an average. Only 17% of the total tweets posted about the event contained situational awareness information that was credible.

Discussion of paper

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 classifiers and the ranking is done with a variant known as SVM rank algorithm. 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 credibility of the tweets dataset using SVM classifying algorithm. We evaluated its accuracy by comparing with R language's inbuilt SVM and other classifying algorithms such as Naïve Bayes, Decision tree, KNN and Neural Networks. 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.

Implementation

The implementation of the project is done in R language. We have performed data mining operation like data collection, data preprocessing, feature extraction, feature selection on our dataset and modeled our classifier using Support Vector Machine.

We implemented SVM on our own without using the inbuilt libraries and compared our accuracy with other classification algorithms like KNN, Naïve Bayes, Decision Tree and neural networks using inbuilt library function in R.

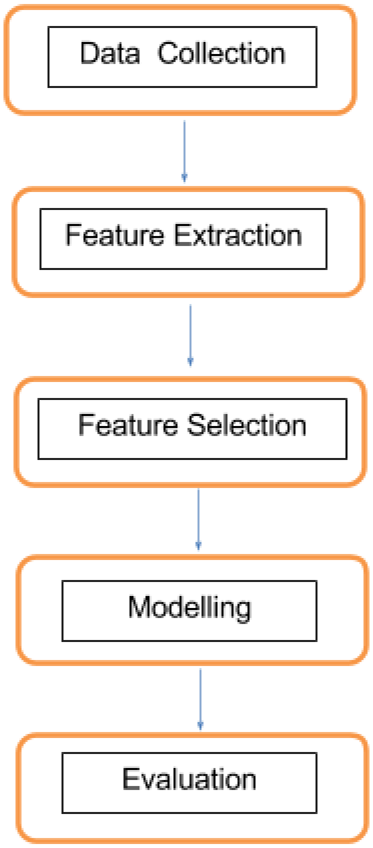


Fig 1.1

1) Dataset Collection:

Twitter has been used as the dataset source. 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. We have chosen one such event UK riots. The riots occurred between 6th and 11th of August 2011, when thousands of people rioted in several cities and towns across England. We collected the dataset of 2011 UK riots, using Streaming API, Trends API and scraping the HTML web page by scrolling the tweets infinitely to get the historical tweets during August 2011.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

How?

- Using the Streaming API and Trends API over a given period.
- Scrape the HTML web page by scrolling infinitely to get the historical tweets

The extracted raw data is filtered with the date between Aug 6 to Aug 11 2011, which corresponds to the 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:

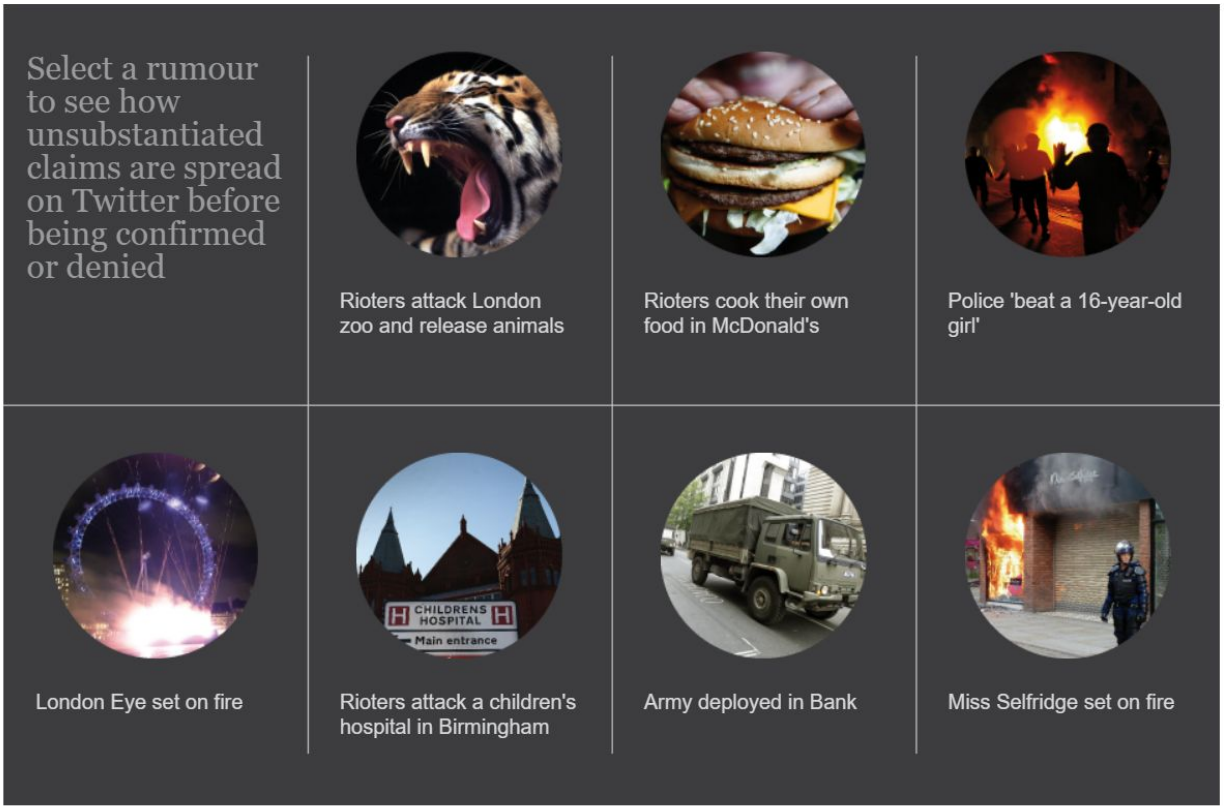
- RawTrainingDataSet.csv
- RawTestDataSet.csv

For both the training dataset and test data set we perform data preprocessing, feature selection, feature extraction and data annotations. The training dataset is used for modelling using SVM classifier and the model is tested for the test dataset.

2) Tweet Annotation :

We manually annotated the raw dataset obtained from scrapping the tweet text into two categories - Spam and non-Spam.

The annotations are done based on the rumors that were spread on twitter during the UK riots. We have taken the below article [8] from “The Guardian” as reference for annotating the tweets and categorizing them as Spam or Non Spam.



Select a rumour to see how unsubstantiated claims are spread on Twitter before being confirmed or denied

Rioters attack London zoo and release animals
Rioters cook their own food in McDonald's
Police 'beat a 16-year-old girl'
London Eye set on fire
Rioters attack a children's hospital in Birmingham
Army deployed in Bank
Miss Selfridge set on fire

Fig 1.2

We introduce new columns as Class for representing spam and non-spam tweets 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

3) Data preprocessing/Cleaning:-

The extracted raw dataset is loaded in R using the **read_csv()** function. Once the file is loaded, we have modified the data types of each column of our dataset using **as.datatype()** function in R and removed unwanted characters/white spaces from each tweet using **gsub() pattern replacement** function in R.

Then we checked for fields having missing values in our dataset, and removed the missing rows using **na.omit()** function in R. We have also parsed the date column in the rawdata set using the functions **regexpr()** to obtain the date in the desired format.

At the end of preprocessing, we obtain two cleaned csv files from the raw dataset

- Training_Cleaned.csv
- Test_Cleaned.csv

4) 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 using the R functions.

a) **No of URL's:** We found the number of occurrences of URL's in each tweet in our extracted data using **str_count()** function in R **.**

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

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

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

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

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

- Swear_words.csv
- Spam_words.csv

We will find the occurrence of the contents of the swear_words.csv / spam_words.csv with the tweet text using pattern matching function in R.

g) **Frequency of spam words**: We looked for the spam words (ex: London Zoo etc.) corresponding to the event UK Riots and found the occurrence of these words using **coll() pattern searching function** in R .

h) **Frequency of swear words:** We looked for the spam words (ex: burnt etc.) corresponding to the event UK Riots and found the occurrence of these words using **coll() pattern searching function** in R .

The outputs of the feature extractions are added as separate columns in the cleaned data sets for both test and training data and we obtain the below two csv files

- Training_Feature_Extraction_Dataset.csv
- Test_Feature_Extraction_Dataset.csv

5) Feature selection:

We are having the below 9 features extracted from the scrapped tweets, that is cleaned after preprocessing

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

We have combined the Length of each Tweet, Number of URL's 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 three features that gives good modeling for the classifier as below

- No of re-tweets.
- No of favorites.
- Combined new feature.

The outputs of the feature selection are added to a new csv file for both test and training data and we obtain the below two csv files

- Training_Dataset.csv
- Test_Dataset.csv

6) Modeling:

SVM:

- SVM is the major algorithm used for modeling our preprocessed data.
- SVM is a classification algorithm used for classifying the data and then rank the data using SVM rank algorithm.
- We currently use linear kernel for the SVM algorithm implementation.
- Find the hyper plane 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 hyper plane which categorizes new examples(test data).The notation used to define a hyper plane is:

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

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

- We have performed SVM classification in 2D (Re-tweets + Favorites) and 3D (Re-tweets + Favorites + New Feature) space.
- Our aim was to obtain a model with **minimum weight and maximum bias** by developing the fit method appropriately for both 2 and 3 features set.
- Predict the new dataset using the obtained model.

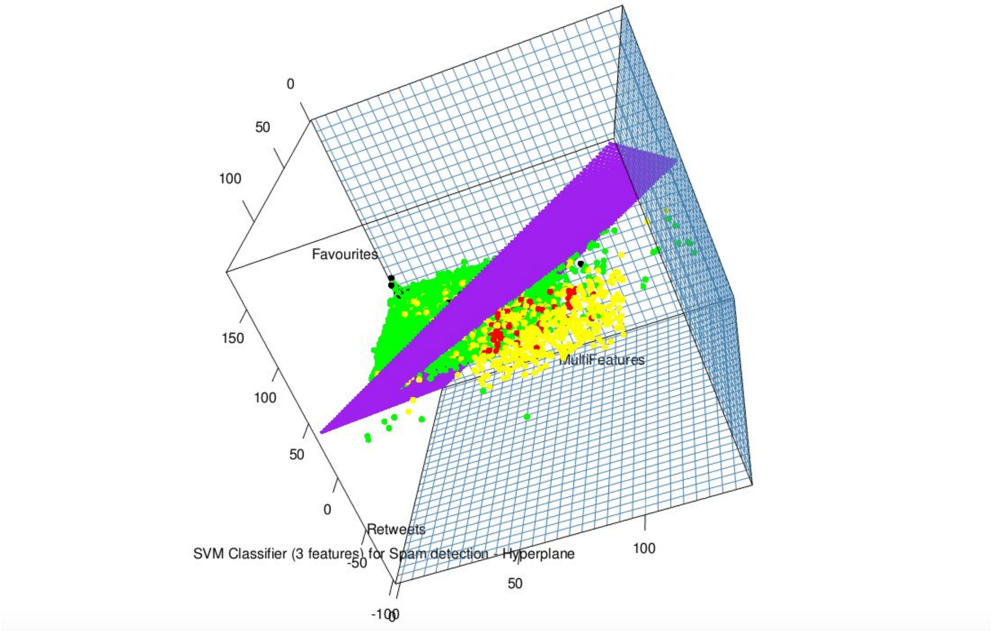

0
50
100
Favourites
150
100
50
0
MultiFeatures
Retweets
SVM Classifier (3 features) for Spam detection - Hyperplane
-50
-100
0
50
100

Fig 2.1

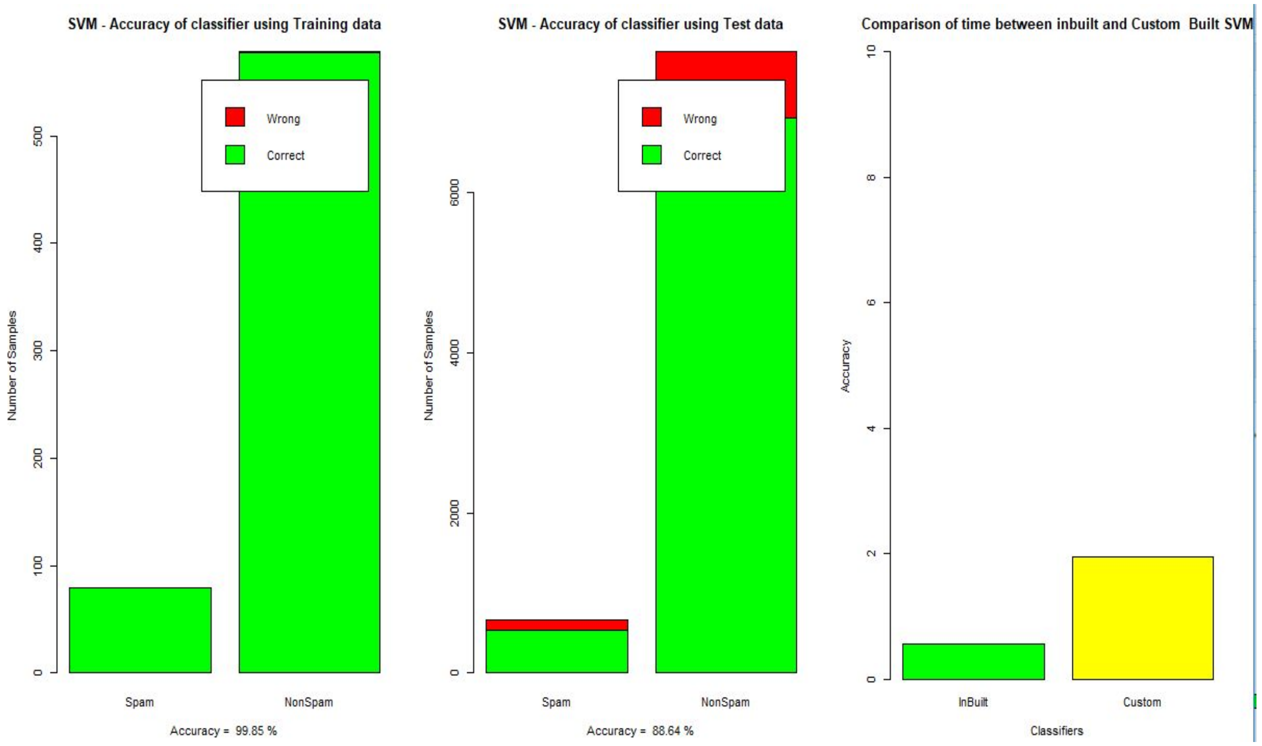

SVM - Accuracy of classifier using Training data
🟥 Wrong
🟩 Correct
Number of Samples
0
100
200
300
400
500
Spam
NonSpam
Accuracy = 99.85 %
SVM - Accuracy of classifier using Test data
🟥 Wrong
🟩 Correct
Number of Samples
0
2000
4000
6000
Spam
NonSpam
Accuracy = 88.64 %
Comparison of time between inbuilt and Custom Built SVM
Accuracy
0
2
4
6
8
10
InBuilt
Custom
Classifiers

Fig2.2

Naive Bayes:

Naive Bayes 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 probability of each class for the data using posterior probability. Based on the calculated posterior probability, data in the data set is classified.Naive Bayes by default uses **Gaussian univariate 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.

$$\underset{\text{Posterior Probability}}{P(c\mid x)} = \frac{\overset{\text{Likelihood}}{P(x\mid c)}\,\overset{\text{Class Prior Probability}}{P(c)}}{\underset{\text{Predictor Prior Probability}}{P(x)}}$$

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

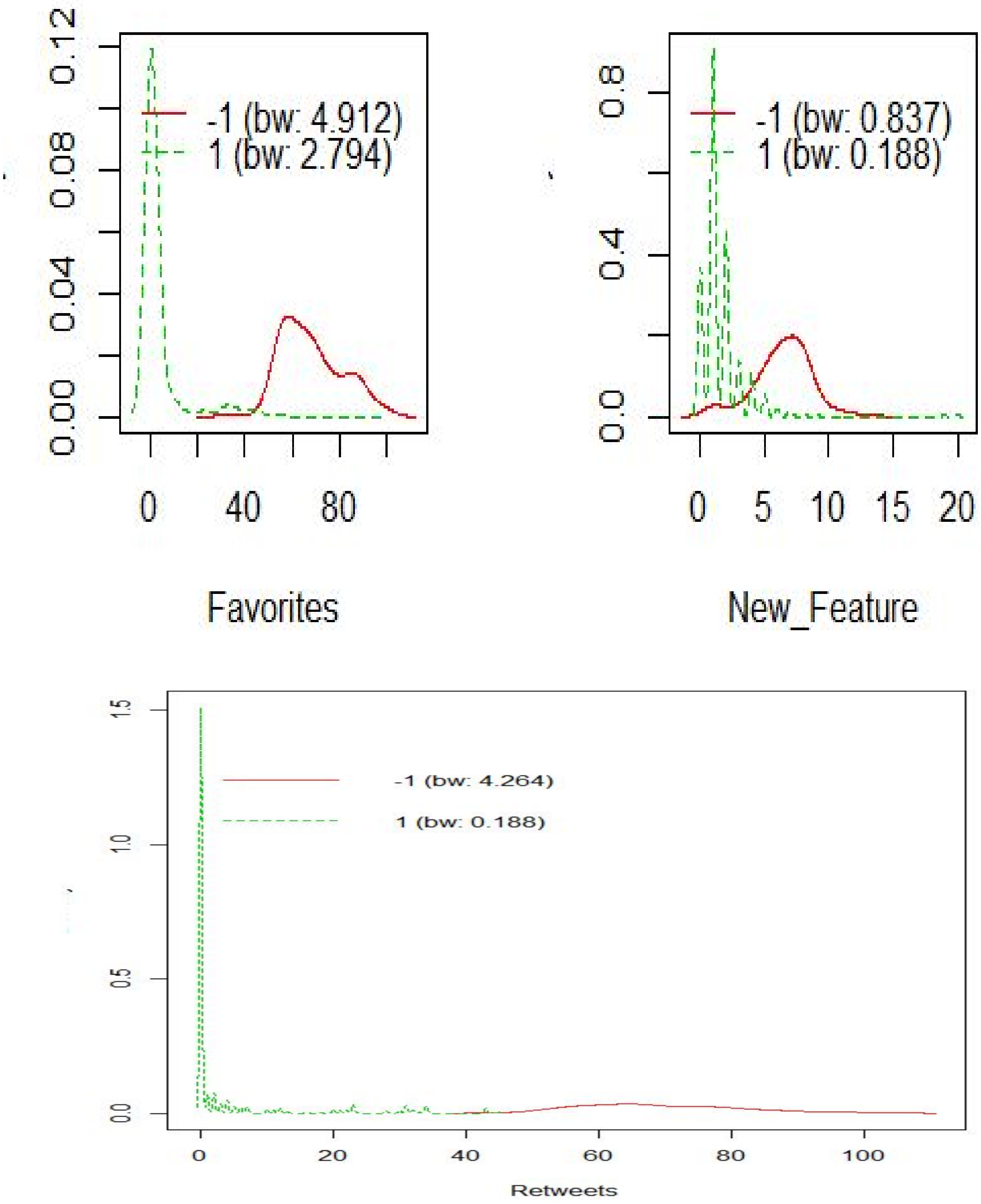

0.00
0.04
0.08
0.12
-1 (bw: 4.912)
1 (bw: 2.794)
0
40
80
Favorites
0.0
0.4
0.8
-1 (bw: 0.837)
1 (bw: 0.188)
0
5
10
15
20
New_Feature
0.0
0.5
1.0
1.5
-1 (bw: 4.264)
1 (bw: 0.188)
0
20
40
60
80
100
Retweets

Fig2.3

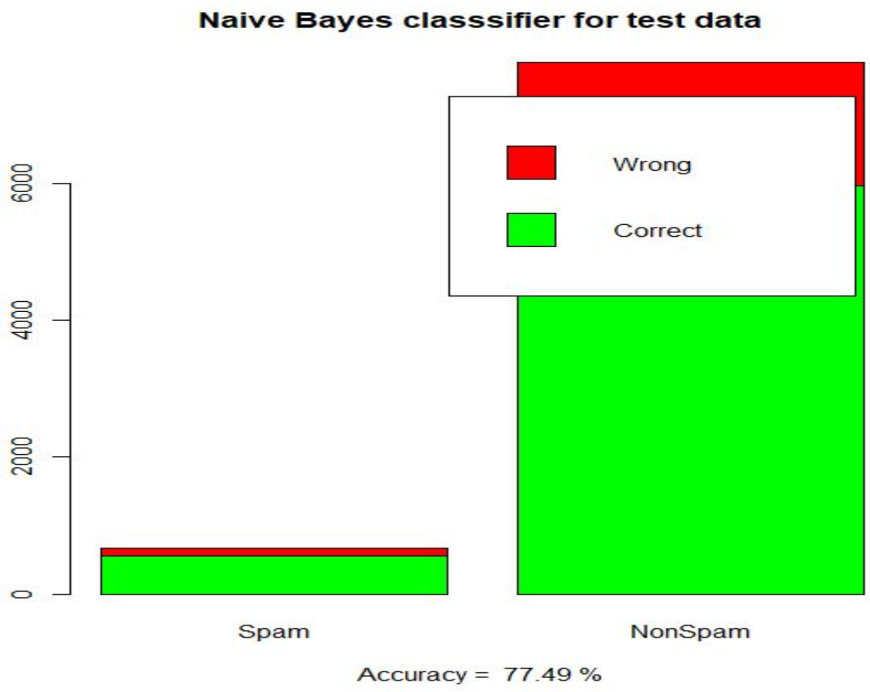


Fig2.4

Decision tree :

Decision Tree is another supervised learning algorithm that is used for classification. The tree is constructed by computing the Information gain of each of the attributes with respect to the target variable and selecting the attribute having the maximum Information gain. In our project, we are using Decision Tree to compare our model output results i.e accuracy with that of the **decision tree** inbuilt function in R. We split the sample into two homogeneous sets (or sub-populations) based on the most significant splitter / differentiator which was retweets in our case. Used rPart method which models the training data based on the feature with the highest gain. Decision trees resulted in an accuracy of 80%.

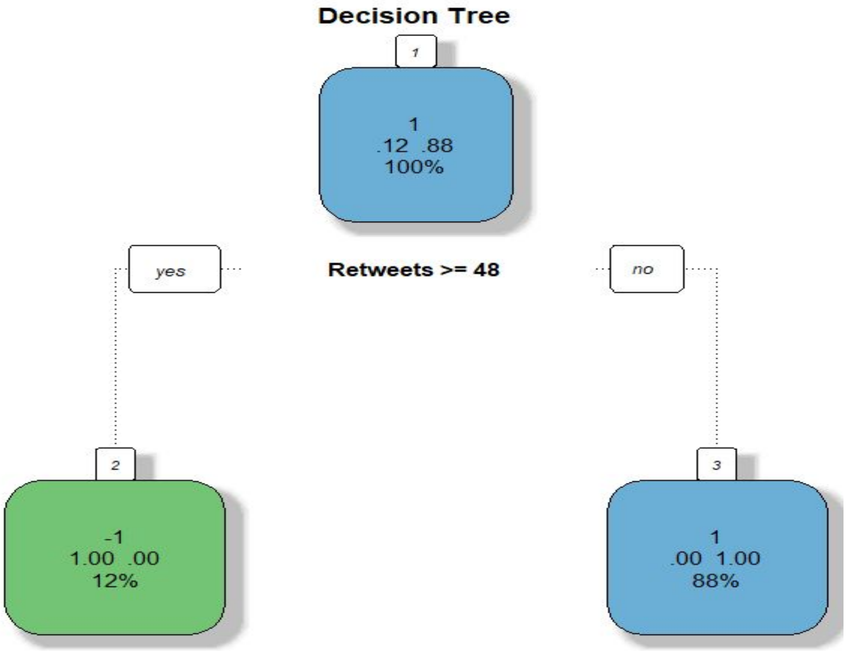

Decision Tree
1
1
.12 .88
100%
yes
Retweets >= 48
no
2
-1
1.00 .00
12%
3
1
.00 1.00
88%

Fig 2.5

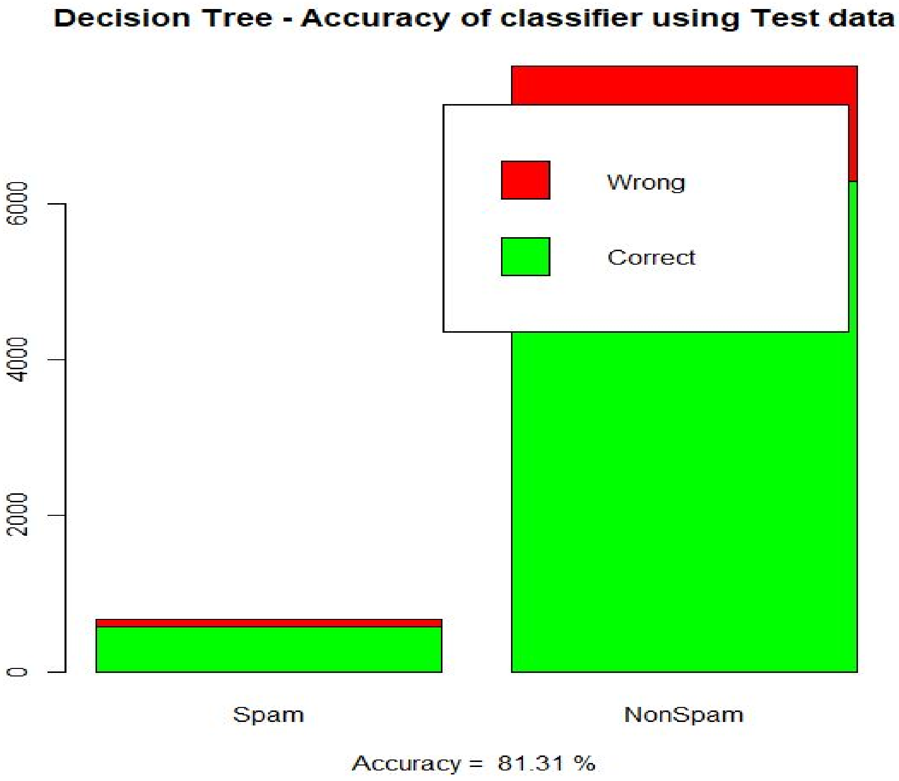

Decision Tree - Accuracy of classifier using Test data
🟥 Wrong
🟩 Correct
6000
4000
2000
0
Spam
NonSpam
Accuracy = 81.31 %

Fig 2.6

KNN:

KNN can be used for both classification and regression predictive problems. For each data points we take the voting among its 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 credible and non-credible, the data is classified as credible if the data is surrounded by credible data points or else classified as non-credible.

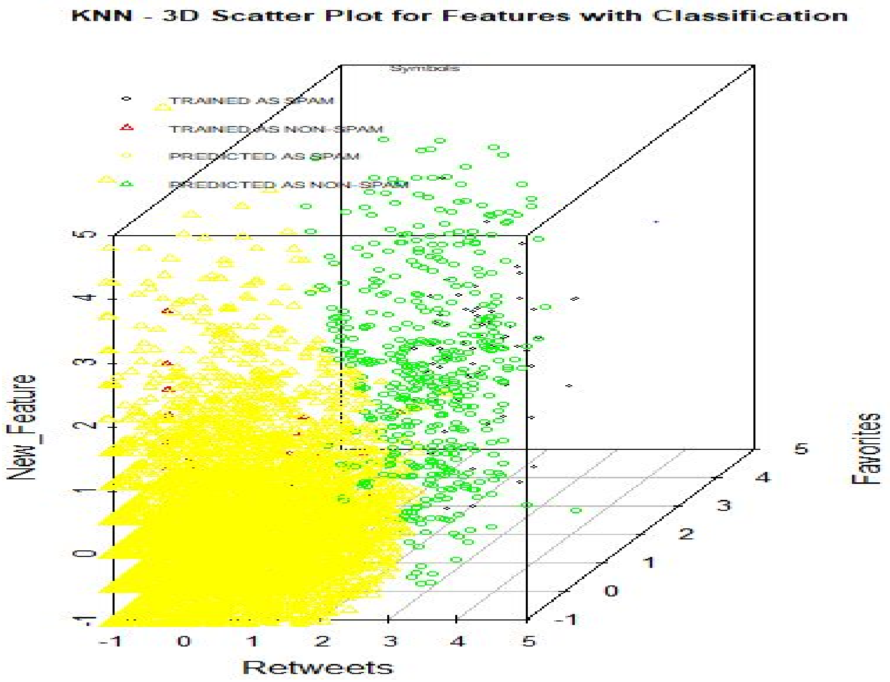


Fig 2.7

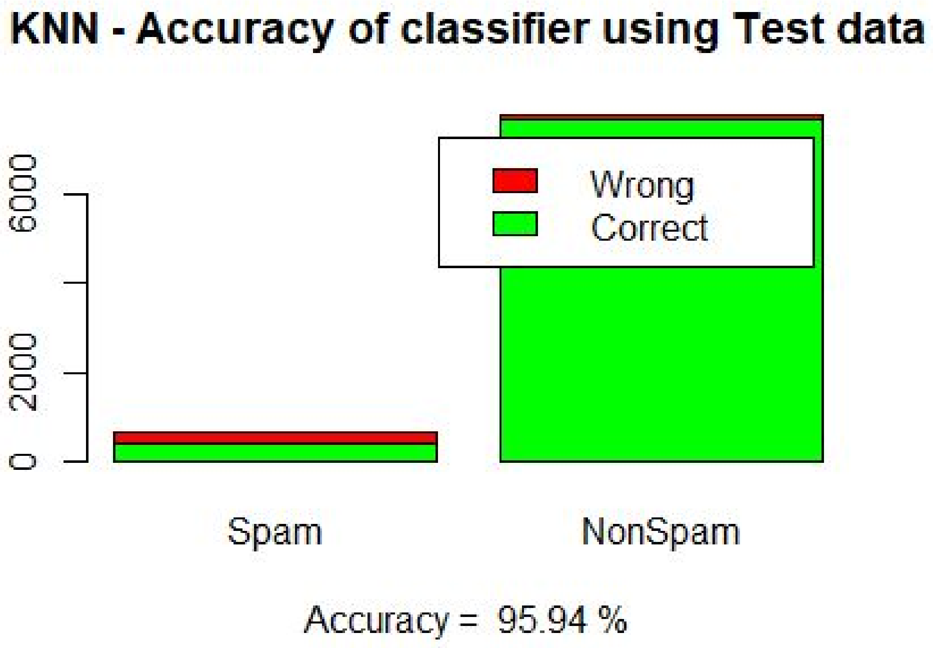


Fig 2.8

6) Neural Networks:

Neural Network is based on a collection of connected units called artificial neurons. We have used Probabilistic neural networks. The first layer computes the **distance from the input vector to the training input vectors**. The second layer sums the contribution for each class of inputs and produces its net output as a vector of probabilities. Finally, a compete transfer function on the output of the second layer picks the maximum of these probabilities, and produces a 1 (positive identification) for that class and a 0 (negative identification) for non-targeted classes. This produces a vector where its elements indicate how close the input is to the training input. Neural Networks resulted in an accuracy of 74.75%.

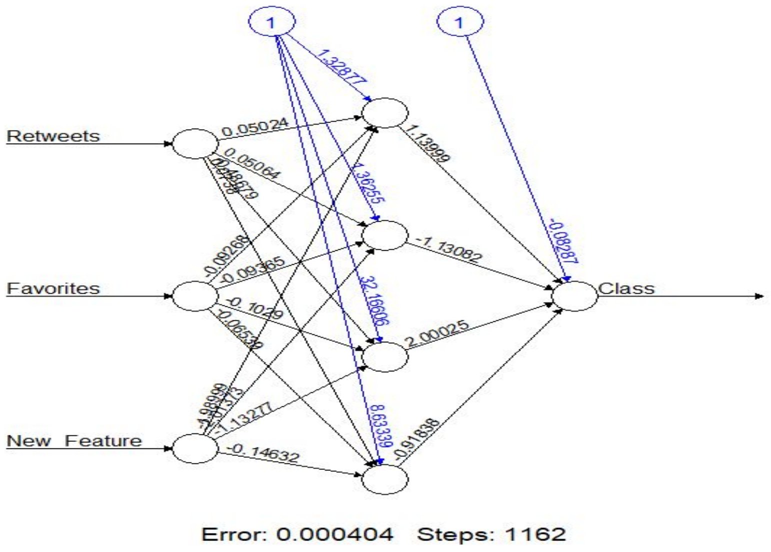


Fig 2.9

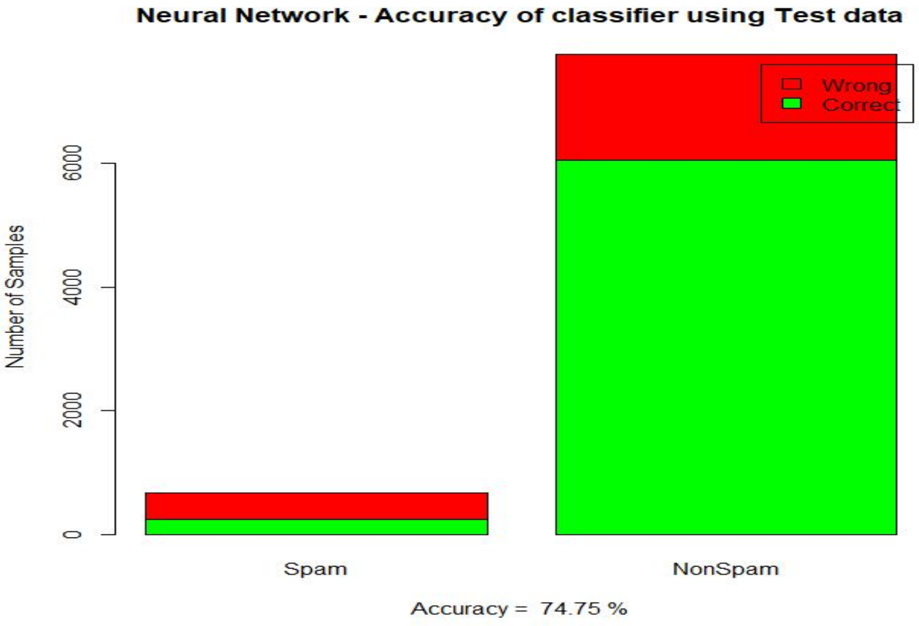


Fig 2.10

Performance Evaluation

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.

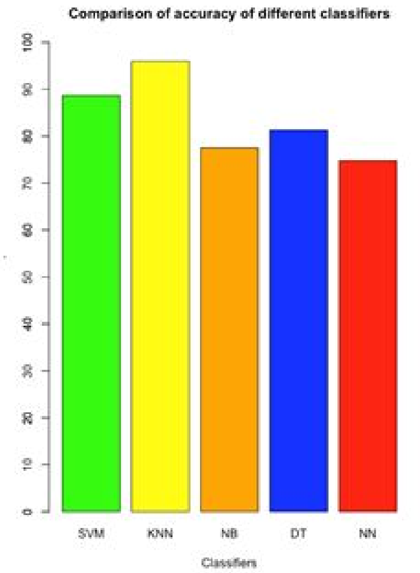


Fig 3.0

Post Processing

Based on the rank that we get from the output of SVM algorithm, we have planned to assign weights to the classified data. This is used to tell how much credible a given tweet is.

CONCLUSION

Support Vector Machine are a better classifier when it comes to binary classification. The SVM for our implementation takes around 2 seconds for executing our dataset with 10,000 tweets while the inbuilt SVM library takes around 0.5 seconds for the same dataset.

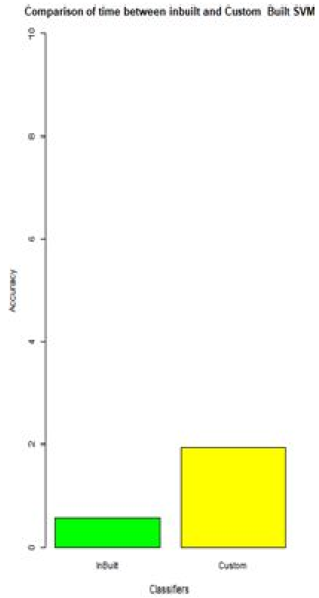


Fig 3.1

Reference

[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/