# 

# Machine Learning Cyber Security for Tweets email Classifications

**Abstract**

This project attempts to accurately classify e-mails tweets for Cyber Security. Every year, the increasing number of cyber-security comes with evolved tactics and highly skilled hackers in search of rapid financial gain. According to the "Cost of a Data Breach Study" by the Ponemon Institute LLC, the average cost of a data breach is US$3.86 million with about 30% likelihood of a recurrent breach within the next 24 months. Across various industries, heavily regulated sectors such as healthcare and financial organizations suffer from the highest data breach costs per capita (US$408 and US$206, respectively compared to an average per capita cost of US$148). Understanding the landscape of cyber threats and identifying predictive factors that contribute to increasing vulnerabilities is crucial to protect the sensitive information of individuals and enterprises. I have used five plus machine learning classifier and machine learning models to classify the given data of tweets.

**Project Description:** The popularity of social media networks, such as Twitter, leads to an increasing number of spamming activities. Researchers employed various machine learning methods to detect Twitter spams. In this assignment, you are required to classify spam tweets by using provided datasets. In given train , test 1 and test2 data we have these columns - "account\_age" "no\_followers" , "no\_folowing", "no\_userfavirate", "No\_lists" ,"no\_tweets", "no\_retweets" "no\_tweetsfaviorate", "no\_hashtag", "no\_usermention", "no\_urls" , "no\_char", "no\_digits" "spam".

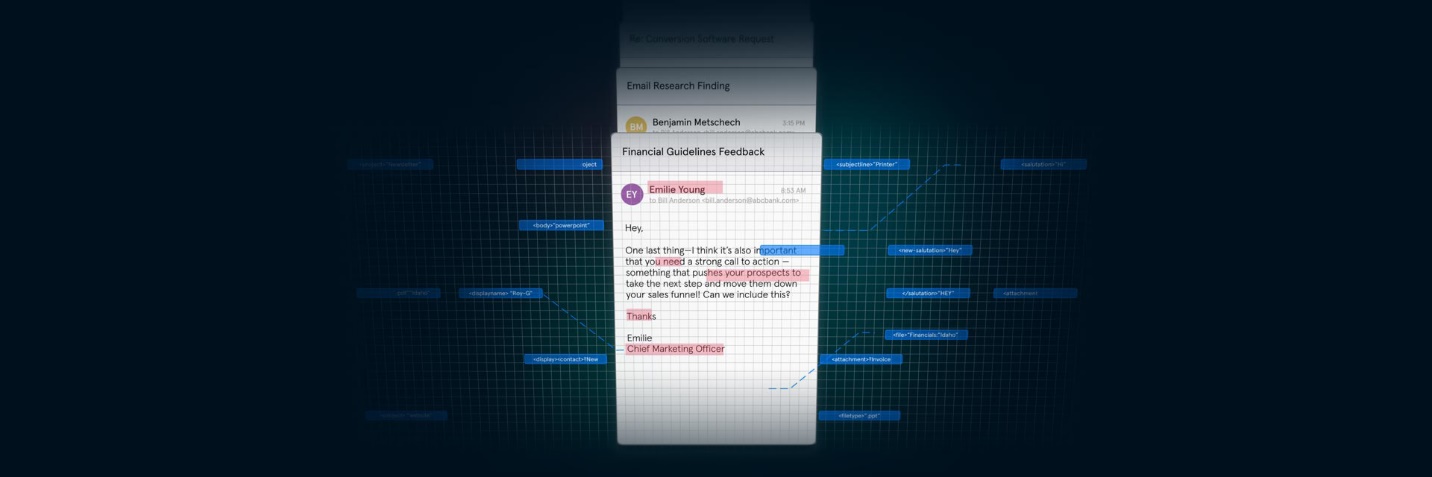
Data set contain train has 1998 rows and 14 columns and test1 and test two both has 2000 rows and 14 variables. By this data set we have to classify the Tweets mails that are spam or non-spammer. Most businesses, government organizations, and academic institutions house dedicated network security teams which help defend the integrity of internal data or client-facing sites against advanced threats or intrusion attacks. Even at home, you may have a firewall to protect your personal network from outside hackers or attacks. However, as we see on the news quite often, even the most sophisticated organizations and individuals often will have their reputable websites or twitter account passwords compromised.

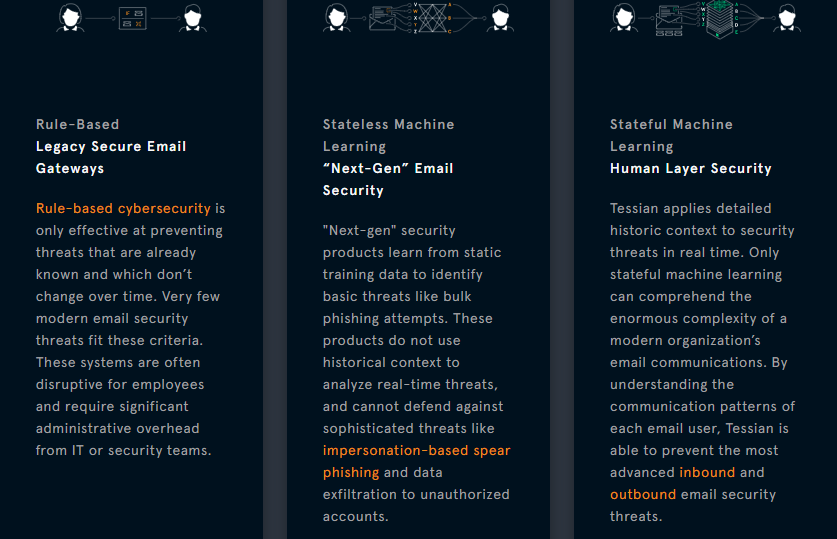
Recognizing this challenge, our project focused on developing machine learning (ML) algorithms to tackle detecting tweets spams. We also placed emphasis on balancing model accuracy with interpretability and computational resources at our disposal.

## What is a cyber-attack?

Cyber-attacks are malicious Internet operations launched mostly by criminal organizations whose goal may be to steal money, financial data, intellectual property, or to simply disrupt the operations of a certain company. Countries also get involved in so-called state-sponsored cyber-attacks, where they seek to learn classified information on a geopolitical rival, or simply to “send a message.”

The global cost of cybercrime for 2015 was $500 billion (BOLD NUMBER).That’s more than 5 times Google’s yearly cash flow of 90 billion dollars. And that number is set to grow tremendously, to around 2 trillion dollars by 2019. In this article we want to explore the types of attacks used by cybercriminals to drive up such a huge figure and help you understand how they work and affect you.





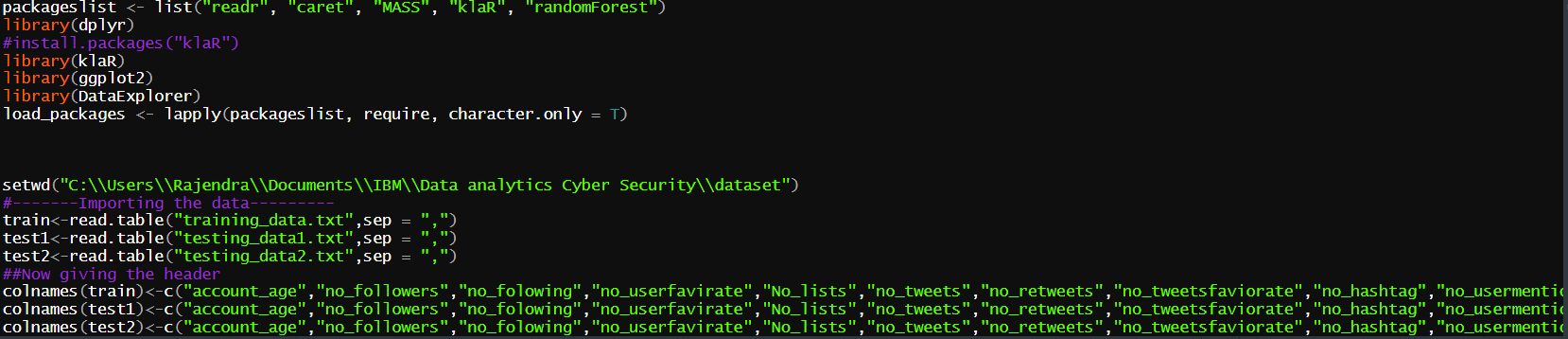
**Objective:** The our main goal in this project is to use five plus different classification machine learning model and algorithms that potency putatively can be used as spams filtration in the tweets . Essentially we are so curious to see which classifier algorithms will give the best accuracy to predicting a spam tweets in given dataset for the cyber security.

****Feature Engineering, Model Selection, and Tuning****

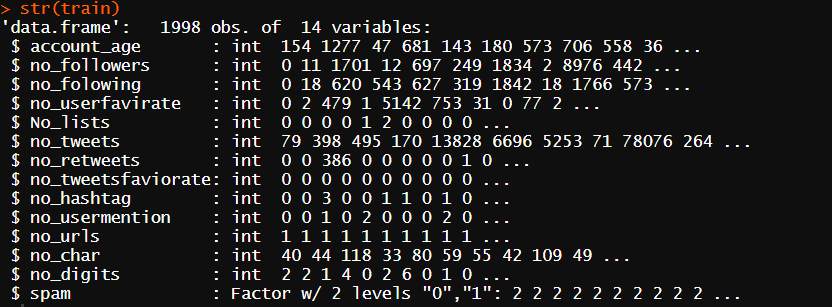
Python's scikit-learn can deal with numerical data only. First, I started with a cross validation Classifier model. I have trained the model with Logistic Regression (LR), Random Forest (RF), and neural network by using nnet package and Support Vector (SVC) classifiers, decision tree classifier. Among them, the LR model outperformed all other models. But, the score was not up to the mark. To improve the score, I switched to use binary classification model for each class. In this case, I considered the combined train and the test sets given in the data. I fit the predictor variable in the entire data set (train and test sets) with cross validation.

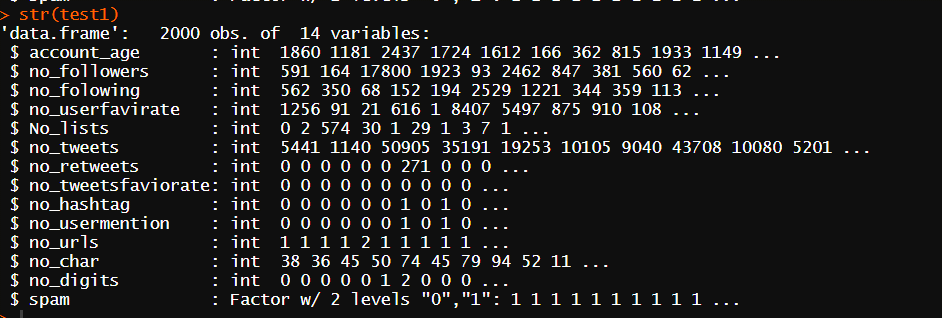
**We have perform the email –classification by using R language, the steps below:**

**1: We have Imported the all data set and have installed all packages which we have required for classification, data manipulations, data visualization, in R studio:**

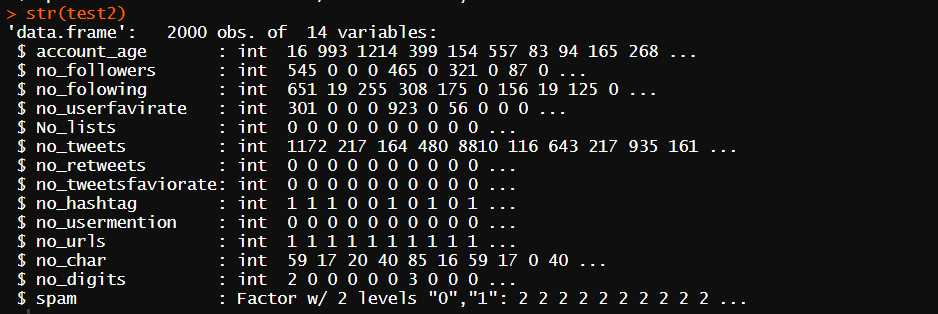


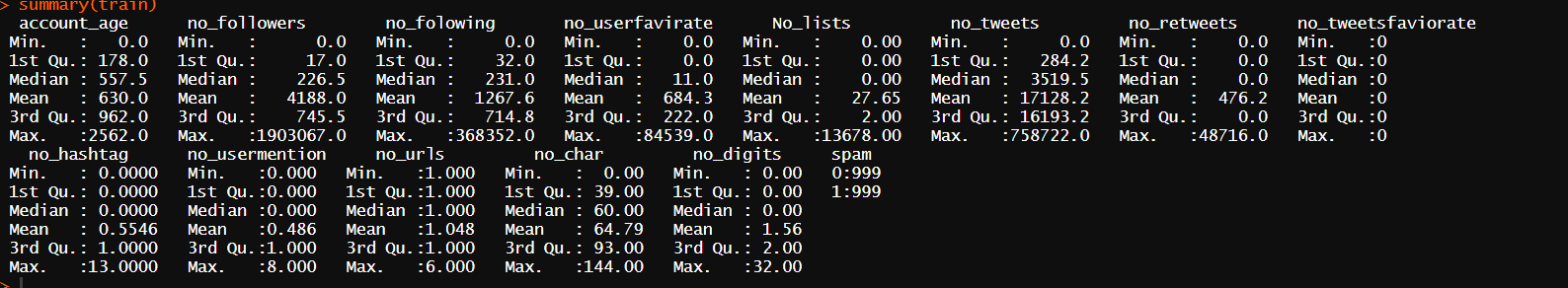
**2. Then perform the Explanatory data analysis:**

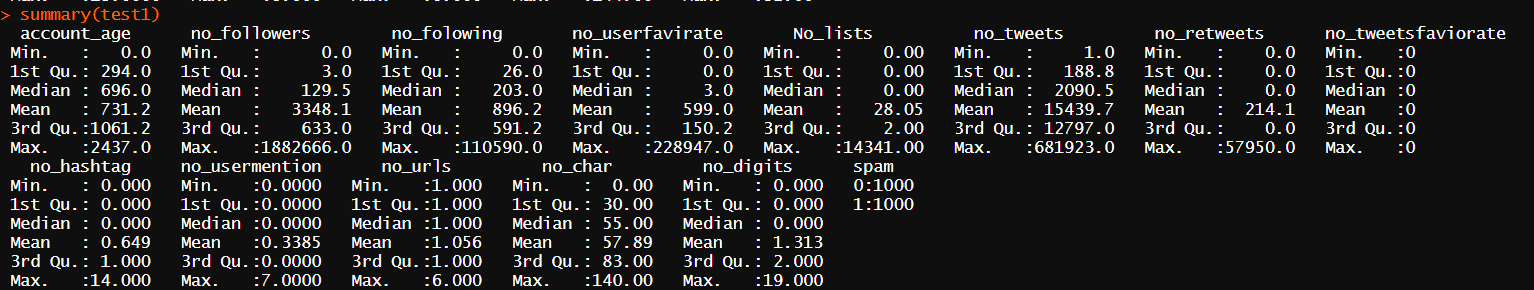


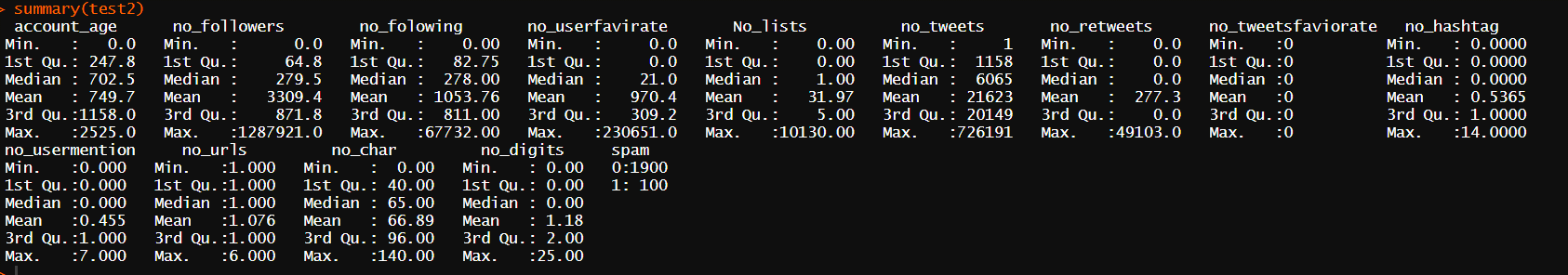


**Here we can see the spams column only is factor data type**, **and all other are integers in all three data set:**

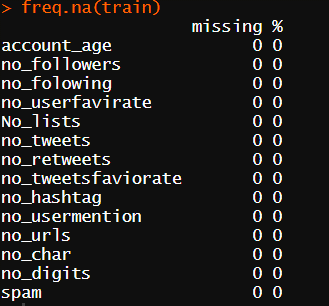


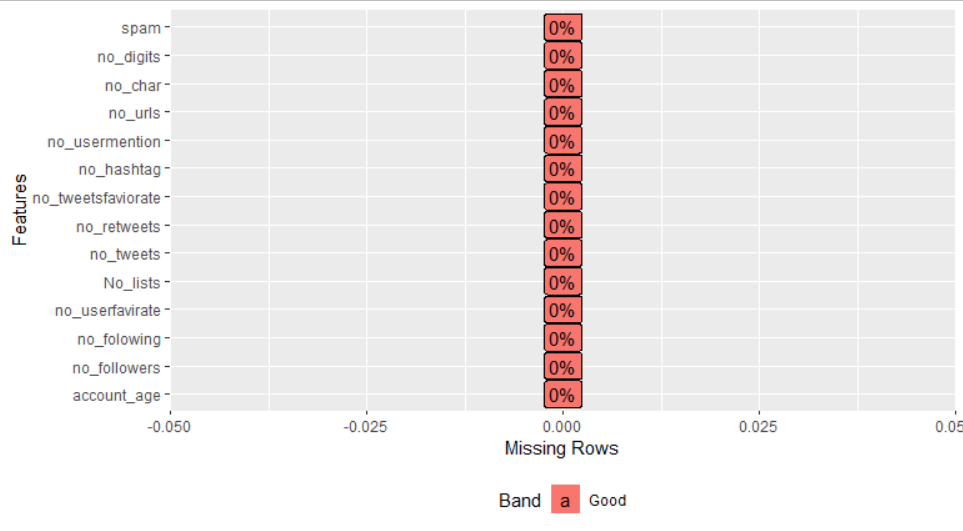


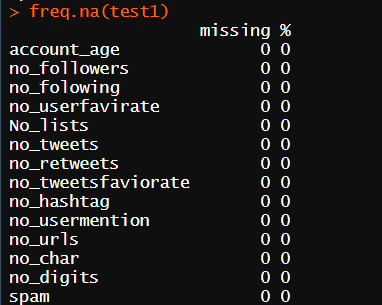


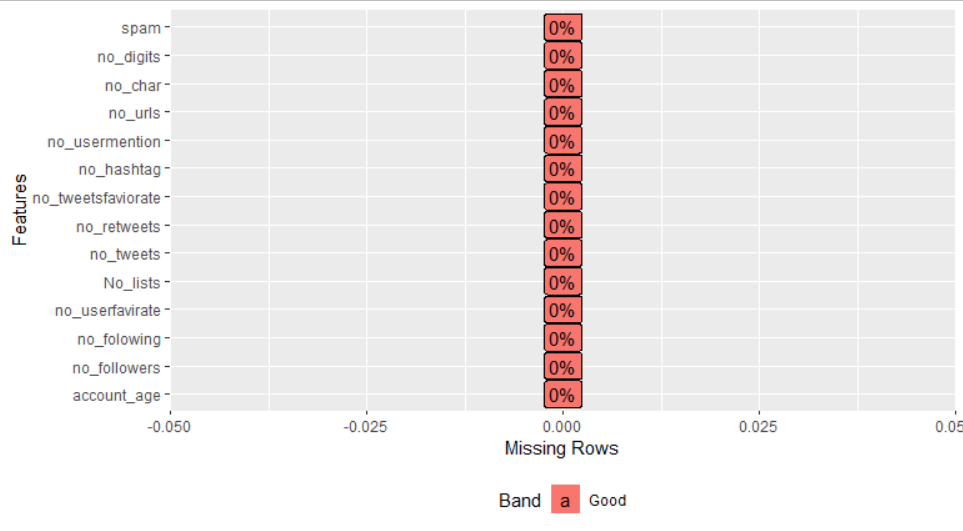


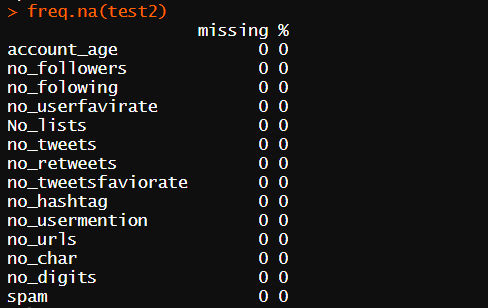
**Data set has cleaned, here none of missing values:**

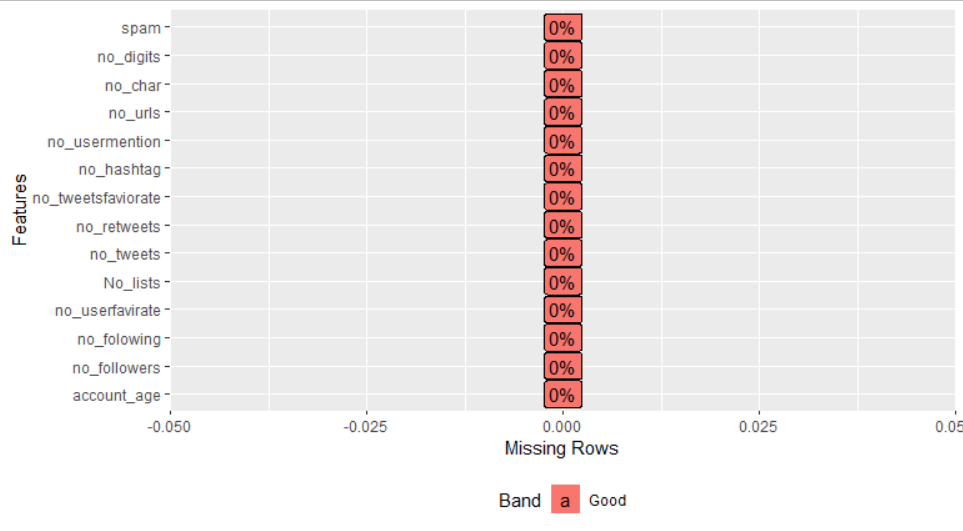




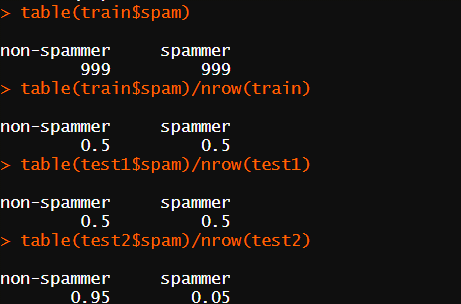




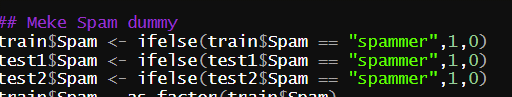




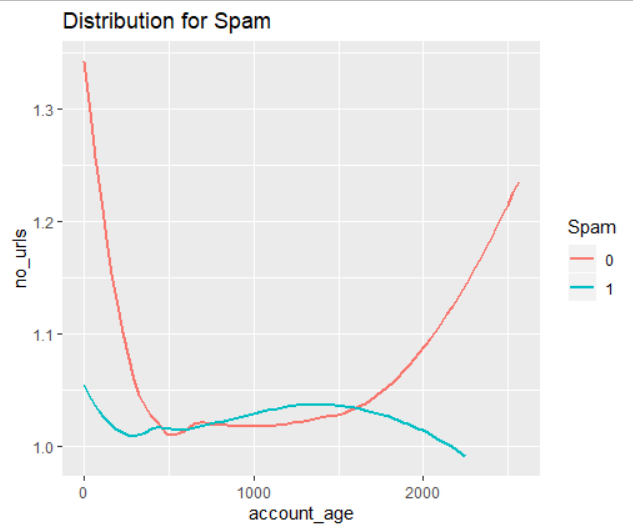
**With the help of table function we can see that 5% emails are spam given dataset:**

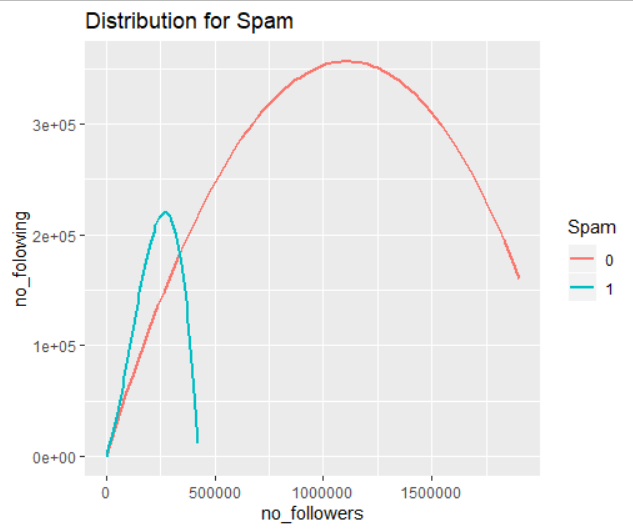


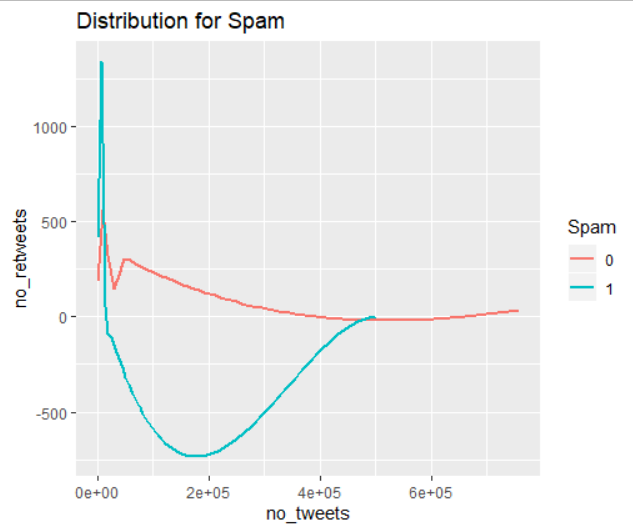
Then we have created dummy of Spam column:

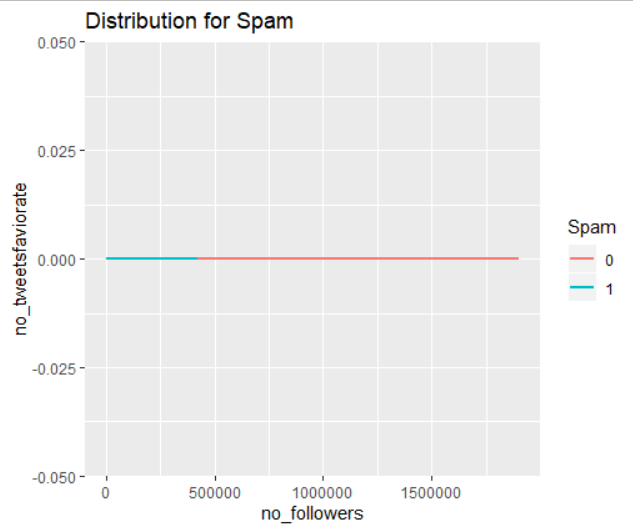


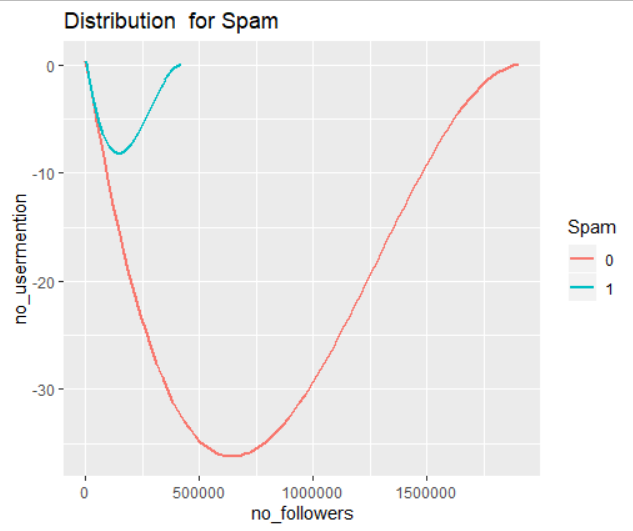
**Data visualizations:** I started visualizing the data. Preliminary exploration and analysis of the data revealed some interesting observations.

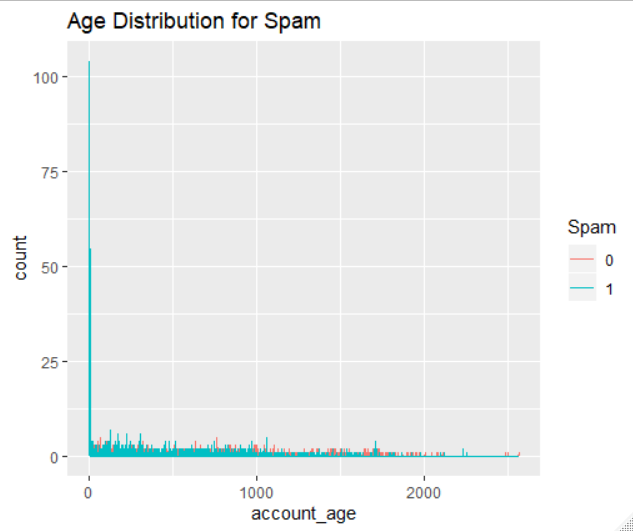


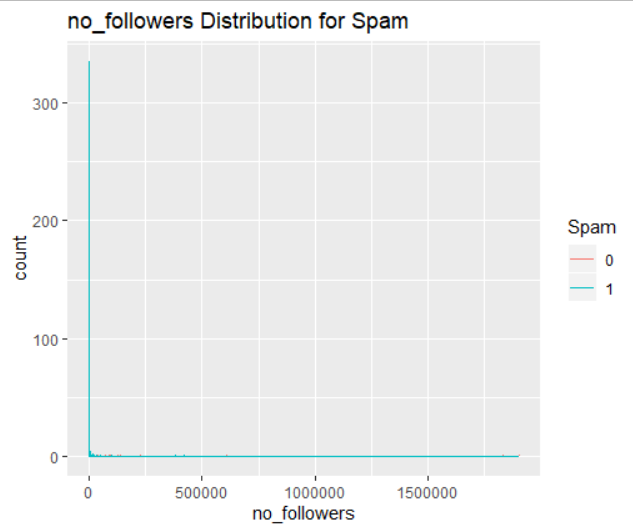


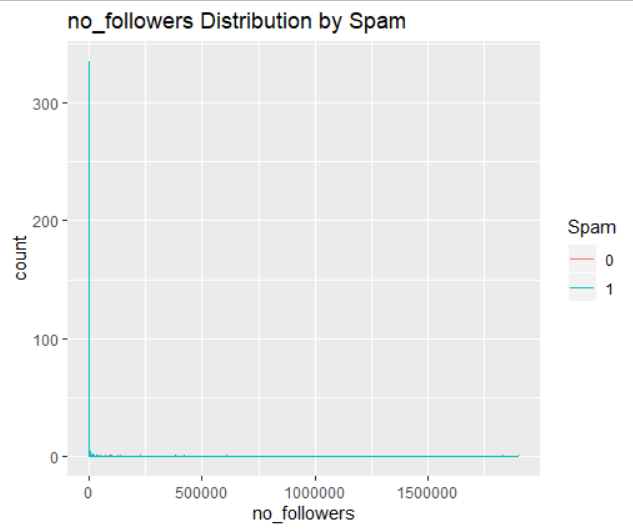


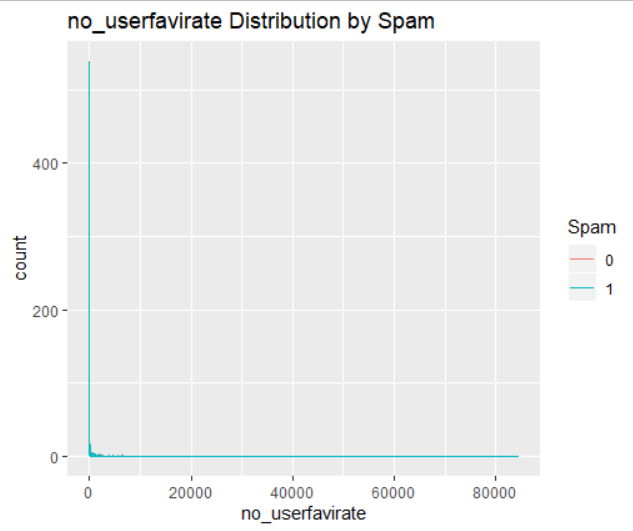


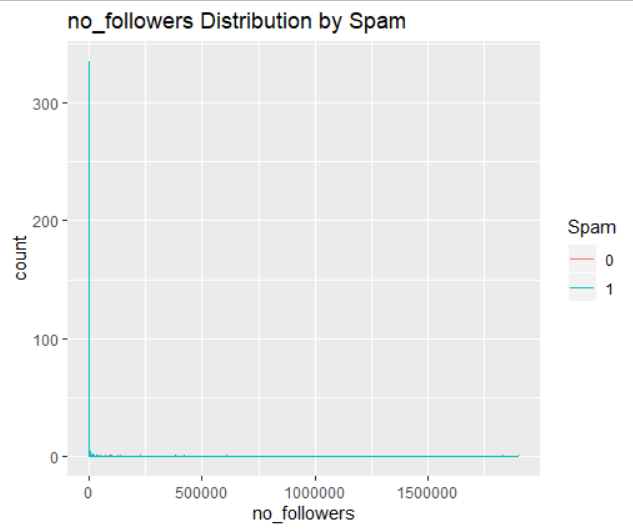


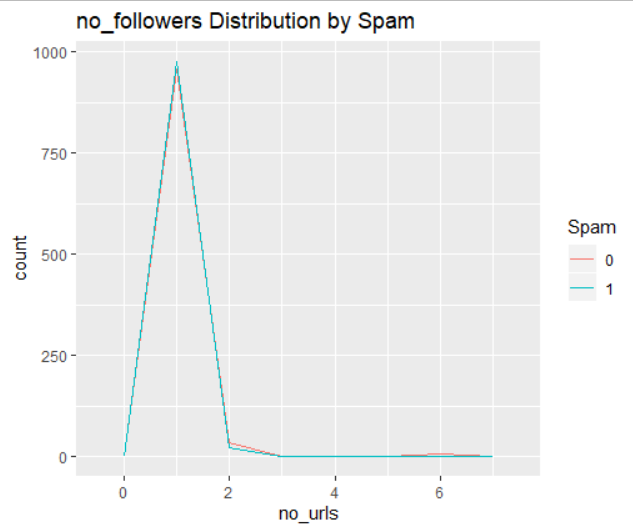


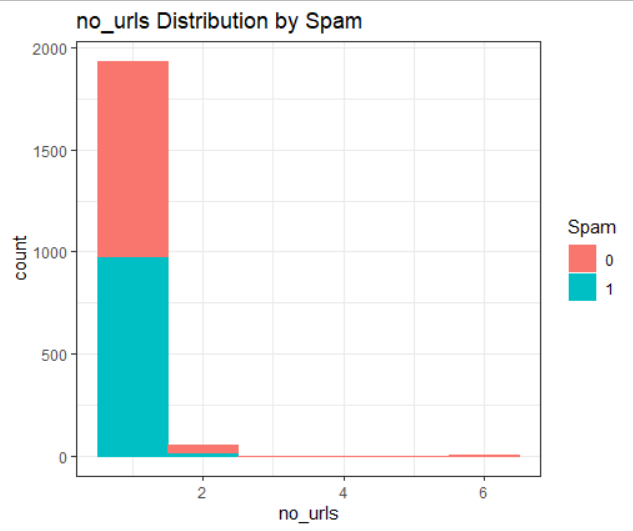


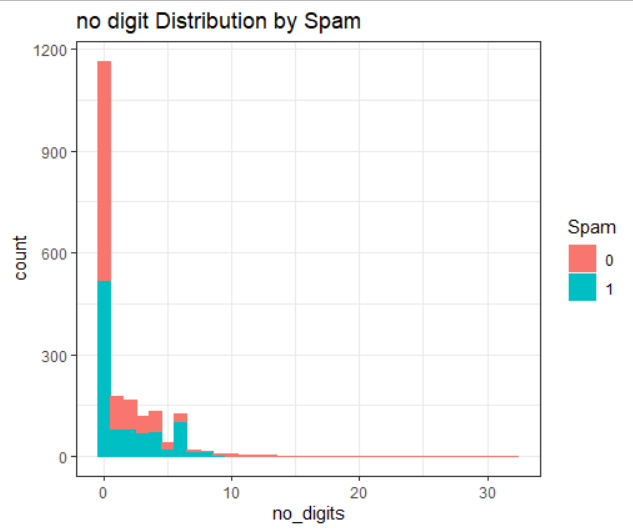


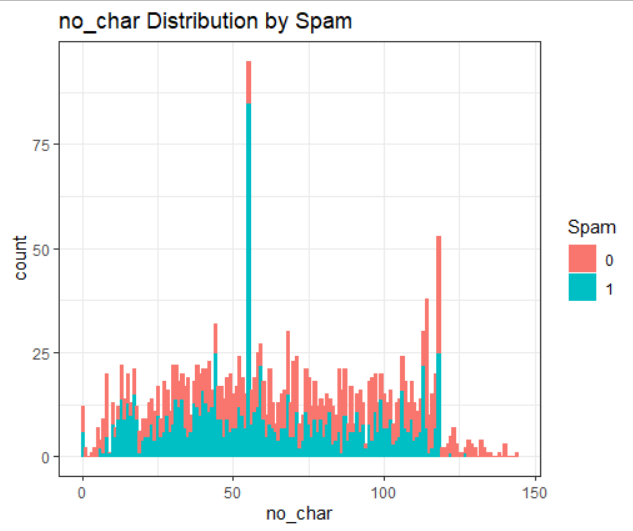


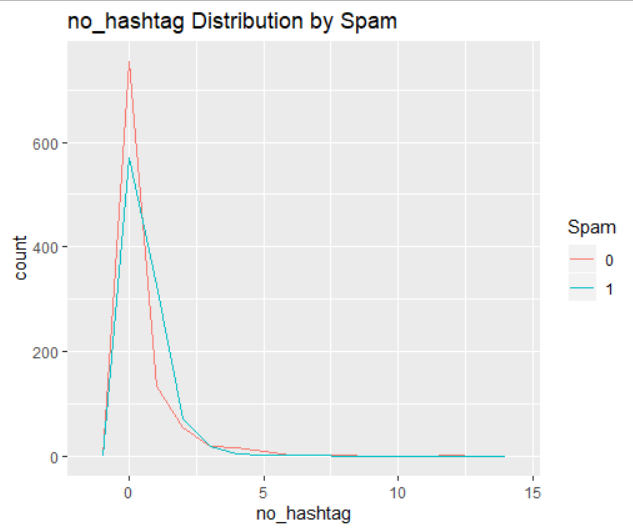




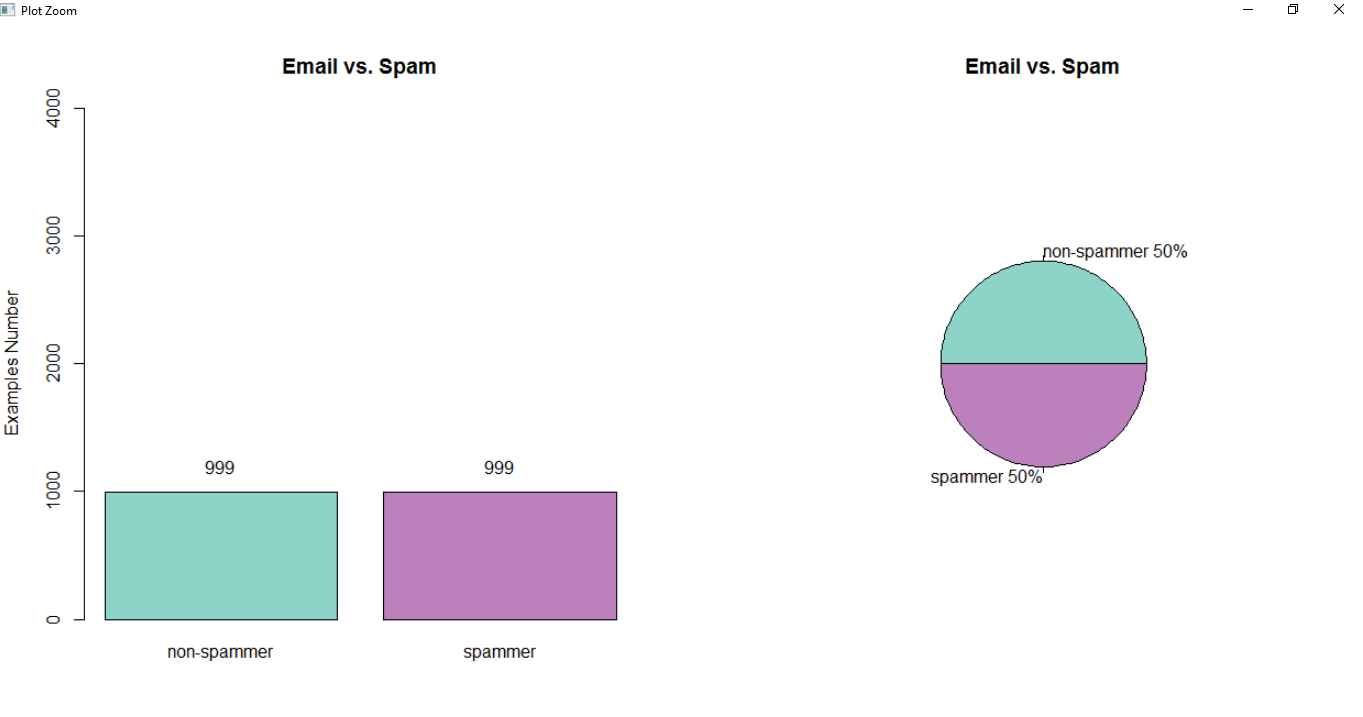




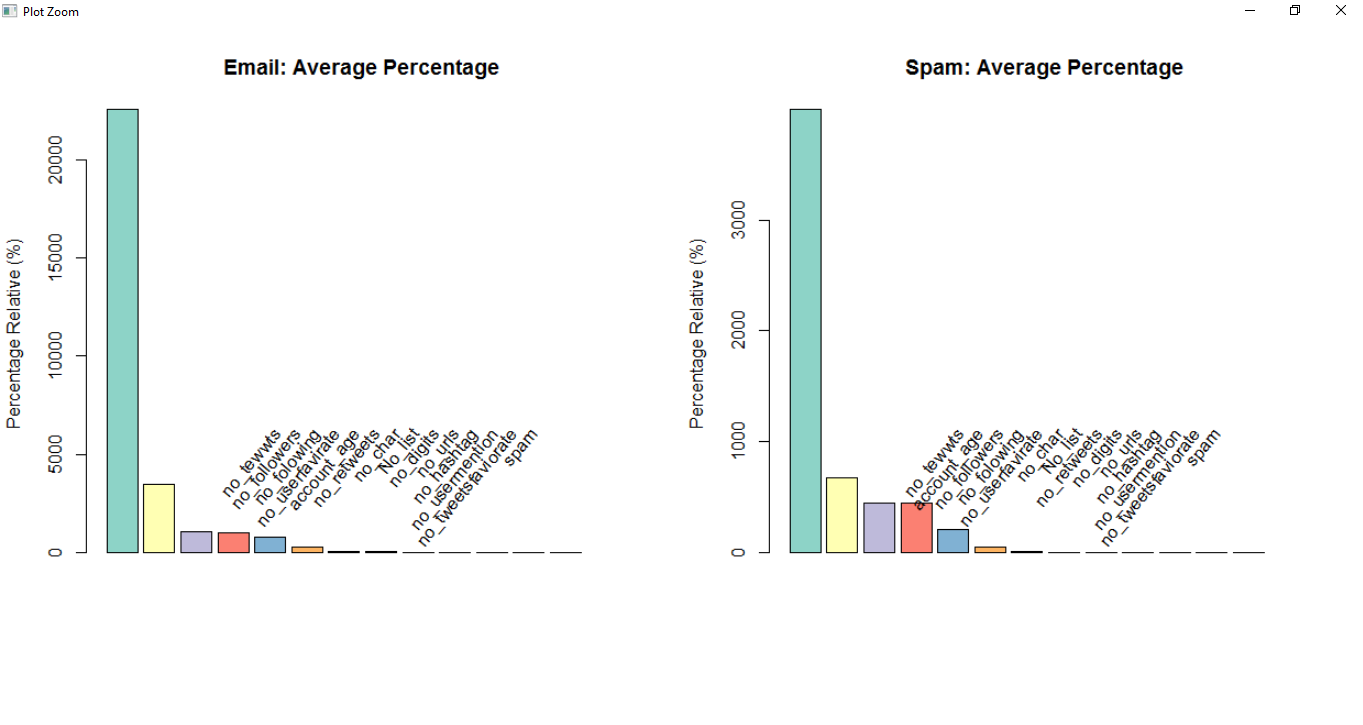


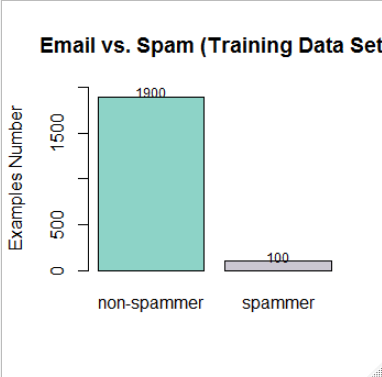


**Then we check Spams percentage:**

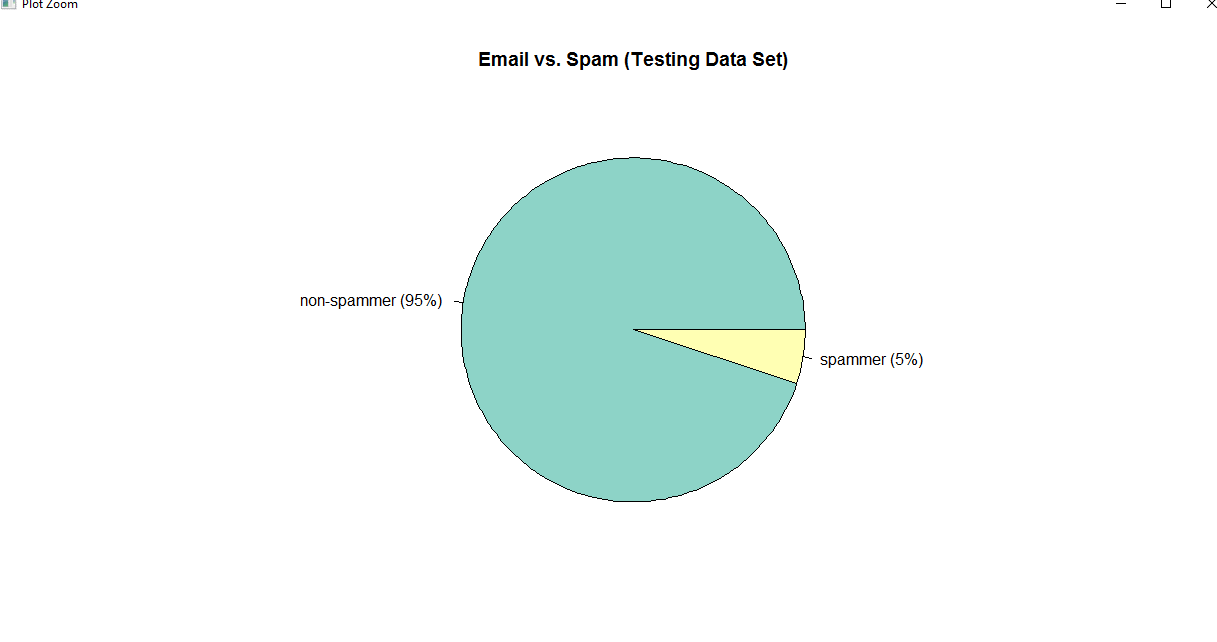


**Spam: Average Percentages:**

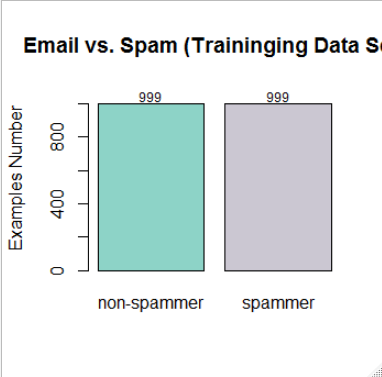




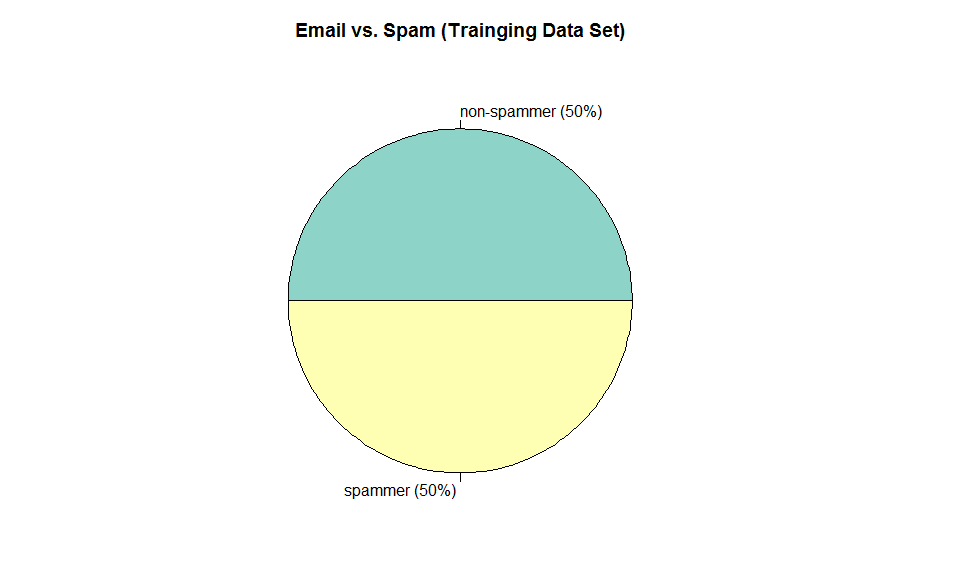
**Test2 data:**



**In Train-data set:**



**Pie Chart for non-spammer and Spams in tweets:**



## Train vs Test Data

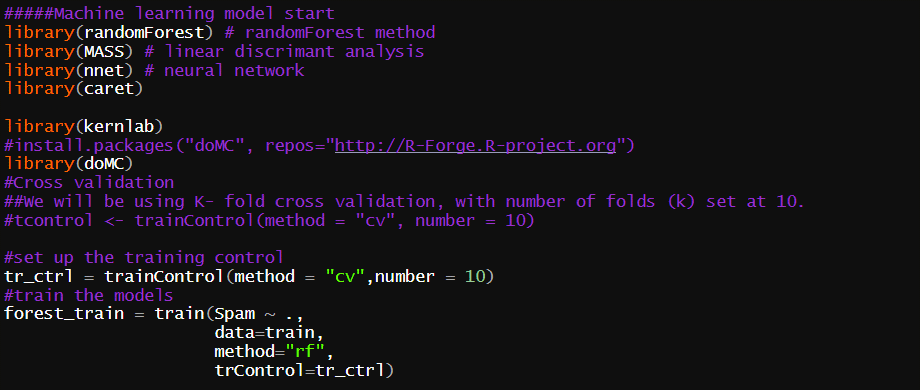
We went back to the original data containing the tweets mails span significant variables (which we had transformed initially to binary 0/1 for spammer/non spammer connections).

We have used k fold cross validation, we have used 10 fold cross, and repeat cross validation up to 5. That will increase the model accuracy. Have we have use randomForest, nnet packages in R studio to apply the classifier model. For naïve bias we have used caret package.We have used the training control. This training control function repeated 10 times to perform the cross validations. For fast execution of our model we have used cross validation method “CV” that work without receptions.



**Tweets spam classifier model:-**

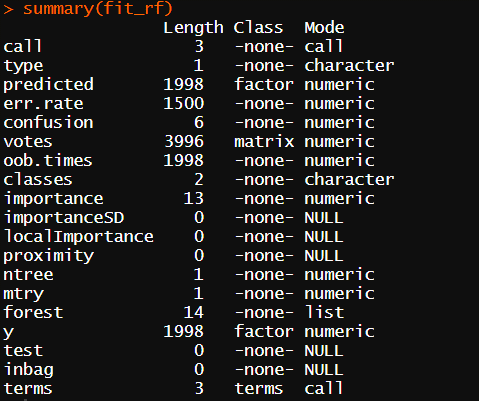
**1.Random forest :**



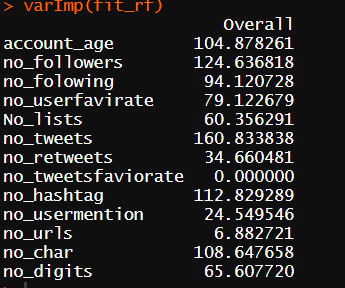
Output:

|  |
| --- |
| forest\_train$results  mtry Accuracy Kappa AccuracySD KappaSD  1 2 0.8528166 0.7056264 0.03187601 0.06375675  2 7 0.8518141 0.7036185 0.03472269 0.06945834  3 13 0.8443241 0.6886429 0.02726889 0.05454216 |
|  |
| |  | | --- | | > | |

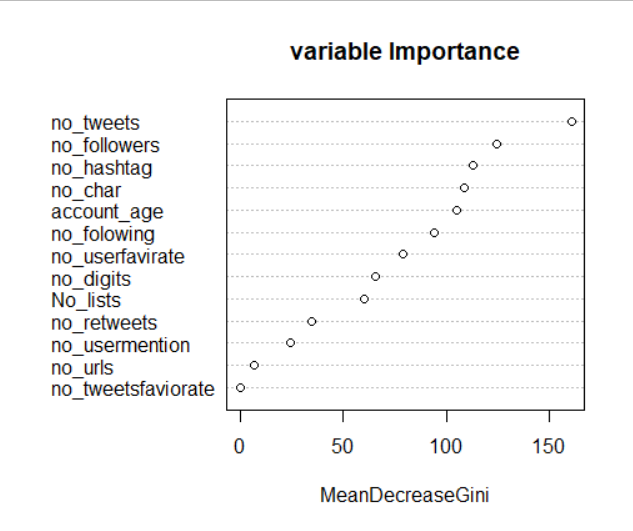
**Random forest Output:**



**Variable importance by random forest:**

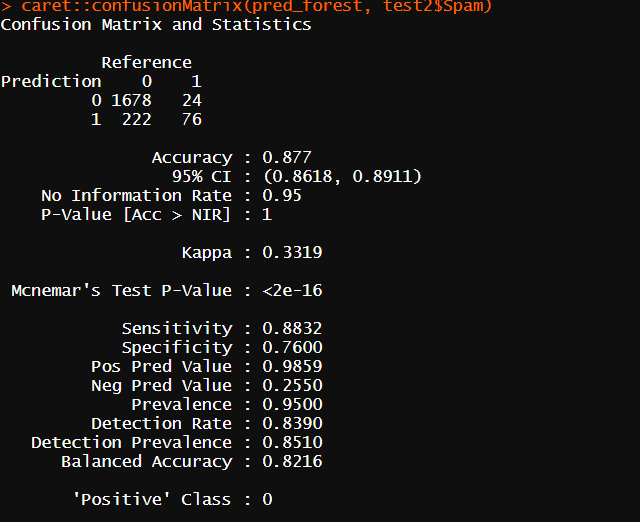


**Variable Importance plot:**

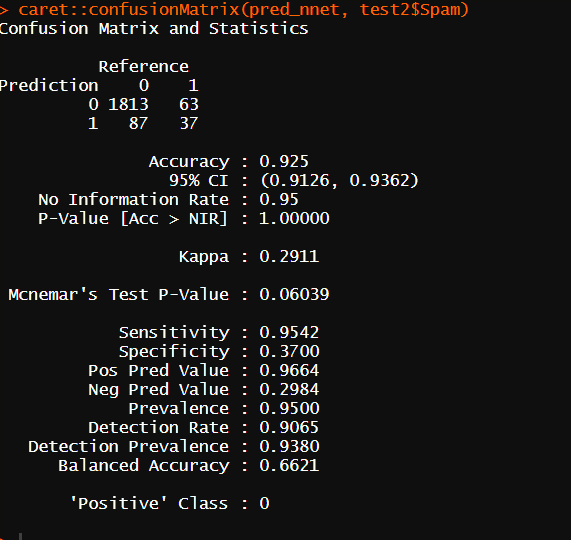


The confusion Matrix () shows the accuracy ,kappa values etc the result.

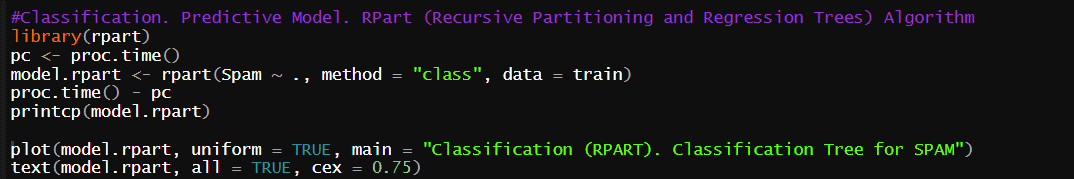
**Confusion-Matrix for Random-forest Model: We got 87.70% accuracy.**



**Confusion Matrix classification for Neural-Network: we got 91% model accuracy**



**Classification RPart Algorithm:**



printcp(model.rpart)

Classification tree:

rpart(formula = Spam ~ ., data = train, method = "class")

Variables actually used in tree construction:

[1] no\_char no\_digits no\_hashtag No\_lists no\_tweets no\_userfavirate

Root node error: 999/1998 = 0.5

n= 1998

CP nsplit rel error xerror xstd

1 0.288288 0 1.00000 1.04505 0.022349

2 0.176176 1 0.71171 0.71171 0.021422

3 0.020020 2 0.53554 0.53554 0.019812

4 0.019019 3 0.51552 0.52052 0.019633

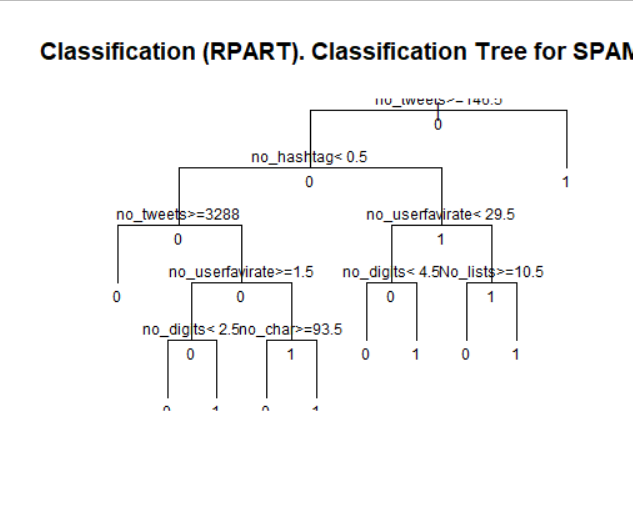
5 0.018519 4 0.49650 0.51552 0.019571

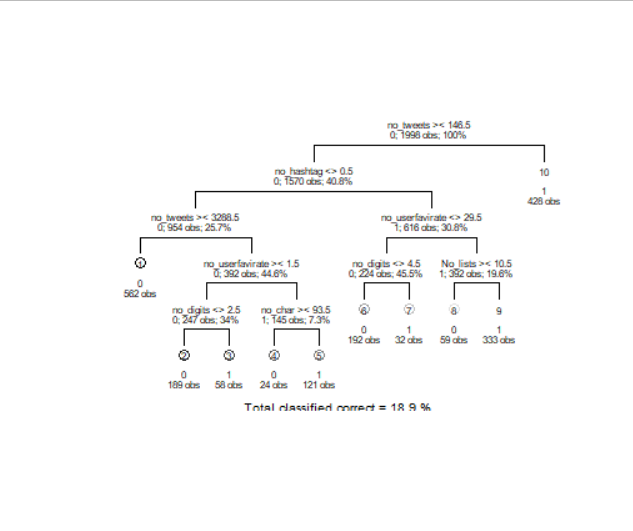
6 0.016016 6 0.45946 0.49650 0.019329

7 0.014014 8 0.42743 0.47147 0.018992

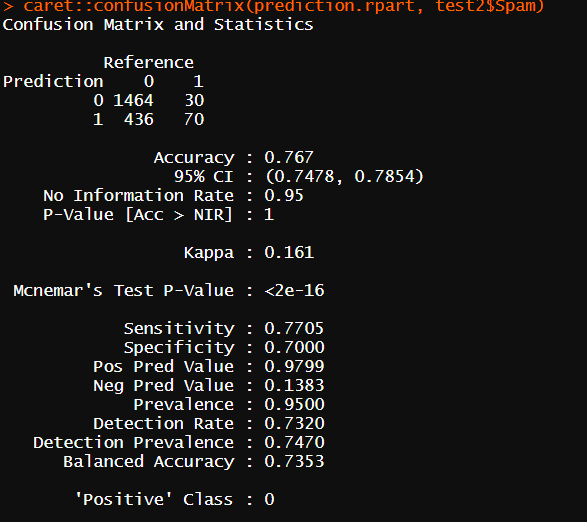
8 0.010000 9 0.41341 0.44144 0.018557

**Classification Tree:**





**Confusion matrix:**

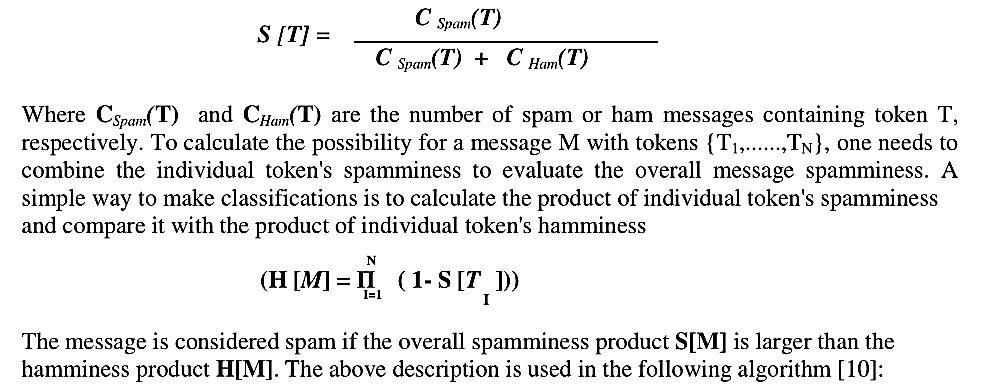


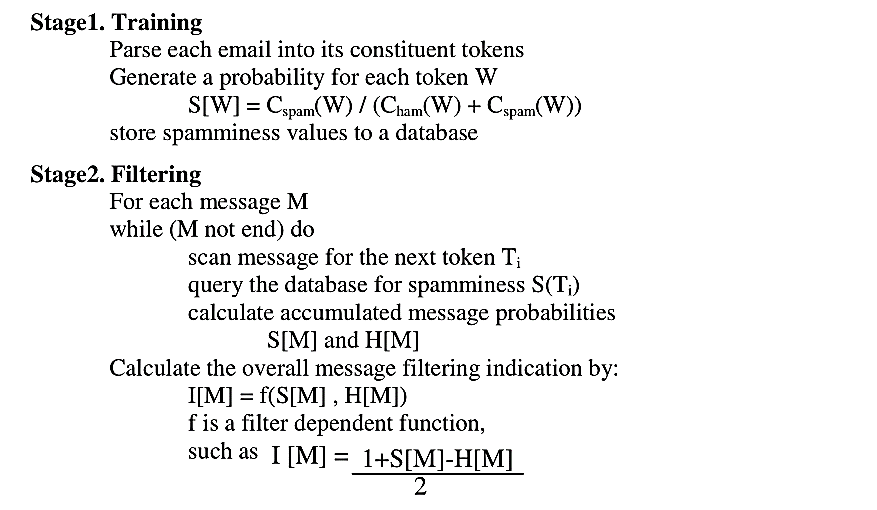
I have goted 76.70% accuracy.

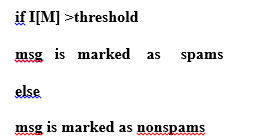
**Random-Forest Classifier:** **Random Forest** is built upon the CART decision tree model, which iteratively minimizing the impurity of the classification via a binary split of the optimal feature. Random Forest performs CART on an ensemble of many trees, each tree having a random subset of all instances and features of the data. This randomness helps to reduce overfitting, a common issue with CART. Both CART and Random Forest also contain a number of related parameters controlling the depth and size of the trees. The random forest method constructs an ensemble of independent decision trees, each using a sample of training observations through bootstrap aggregation, and selecting a feature for each node split using a random subsample of features. Once all trees are fitted, the algorithm averages the class prediction probabilities for each feature-value combination to deliver the final model predictions.

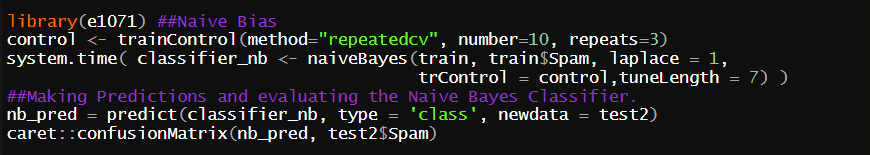
**Naïve-Bayes Machine-Learning classifier method:**

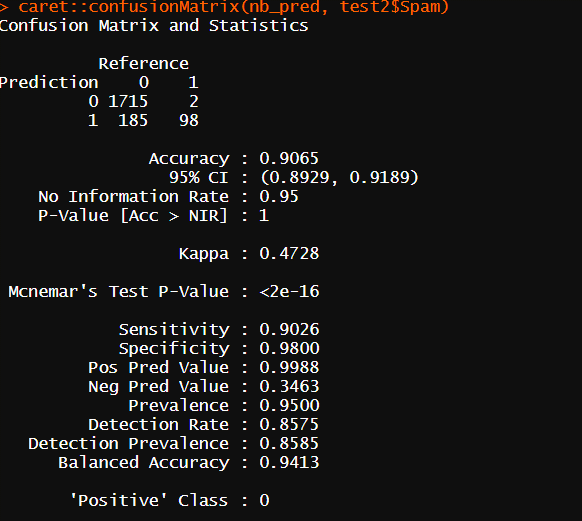
As a supervised classification problem, there are a variety of available models to structure my data. While I ultimately want to test against a variety of models, for the sake of efficiency, I will determine my modelling assumptions upon a select baseline, and later compare competing models against that baseline utilizing those same assumptions. For my baseline, I chose common models, Multinomial Naive Bayes Classifier. **Multinomial Naive Bayes Classifier** is a specific instance of the more general Naive Bayes Classifier, specifying a multinomial distribution of the output feature. The Naive Bayes Classifier more generally is built upon Bayes Theorem of conditional probabilities. It is advantageous in it is relatively simple and naturally suited for counts of text, but disadvantageous in that it relies on strong independence assumptions that are known to be false (it is thus 'naive').



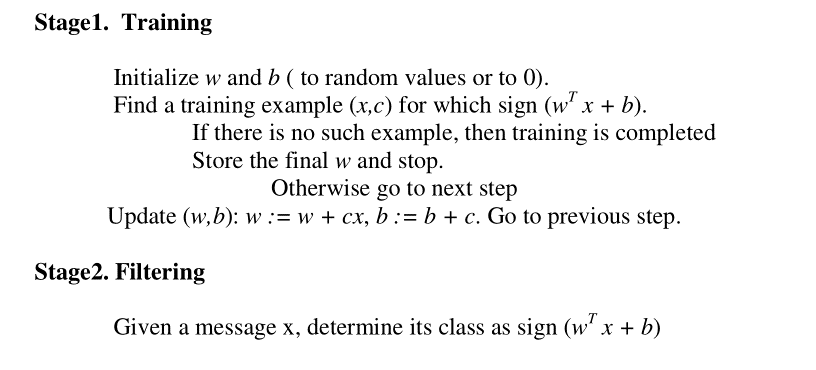




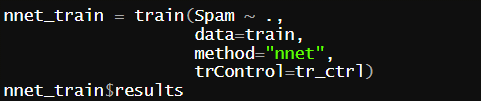




**Artificial-Neural Networks classifier method:** Two deep neural networks were created for classification: one fully connected network, and one convolutional network. All networks were constructed using the nnet package.

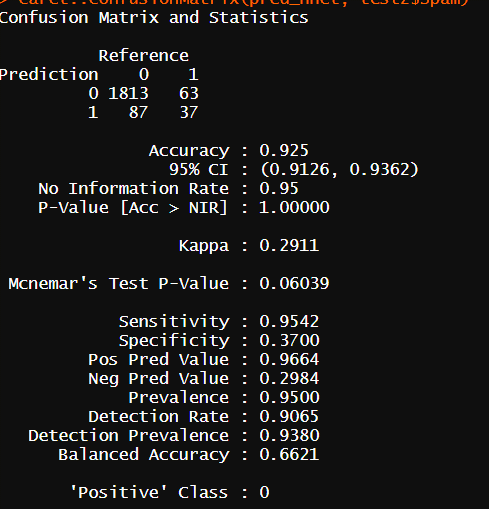


**Neural N/w MachineLearning Model:**



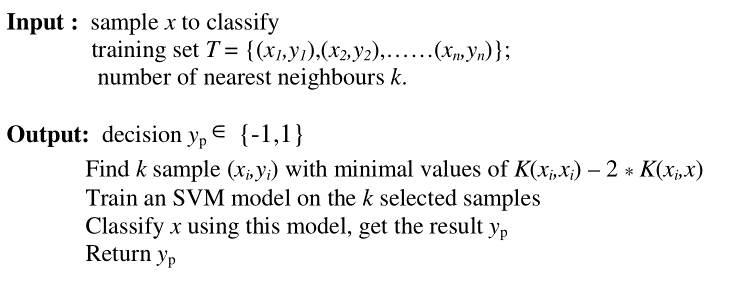
|  |
| --- |
| nnet\_train$results  size decay Accuracy Kappa AccuracySD KappaSD  1 1 0e+00 0.5775628 0.1558734 0.05329838 0.10602831  2 1 1e-04 0.6151307 0.2301791 0.04877220 0.09770025  3 1 1e-01 0.6161482 0.2322781 0.03662463 0.07330598  4 3 0e+00 0.6256256 0.2512348 0.03220681 0.06432754  5 3 1e-04 0.6371181 0.2741955 0.02731421 0.05426851  6 3 1e-01 0.6391508 0.2782536 0.03525573 0.07047460  7 5 0e+00 0.6366357 0.2732243 0.02530216 0.05064739  8 5 1e-04 0.6381407 0.2762456 0.03015160 0.06047954  9 5 1e-01 0.6291382 0.2582478 0.03407877 0.06809178 |
|  |
| |  | | --- | | > | |

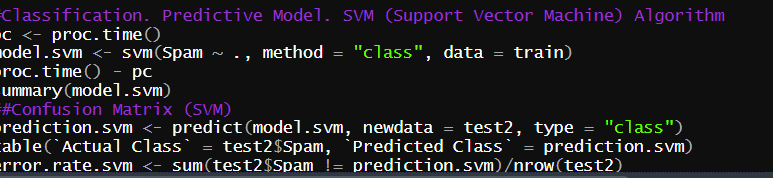
**Result: We have got 92.50% accuracy Neural N/w model:**

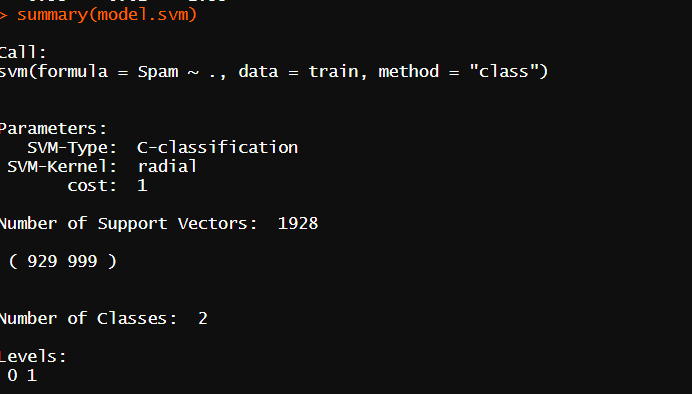


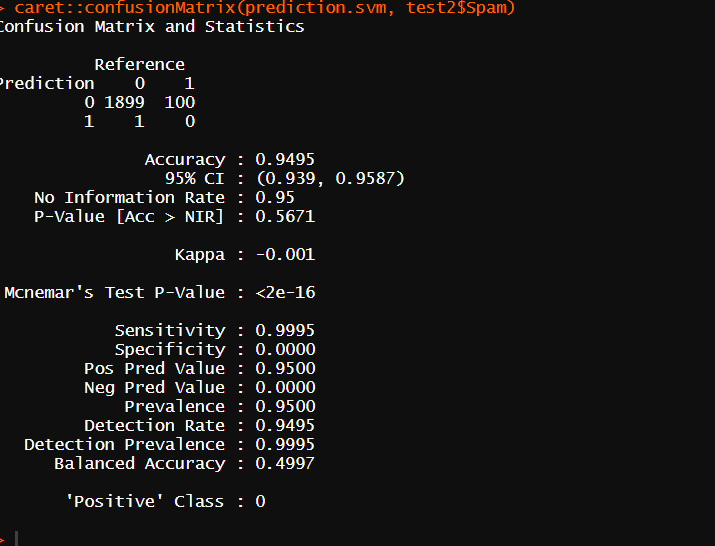
**Support-Vector Machines-classifier Model:**

**Support Vector Classifier** differs from my other models in that it does not define its class division using the entire dataset, but rather only the closest data points at the margins (the support vectors) between the two classes. The strictness of that margin is depends on the value of C hyper-parameter. Lowering the strictness helps to reduce overfitting.







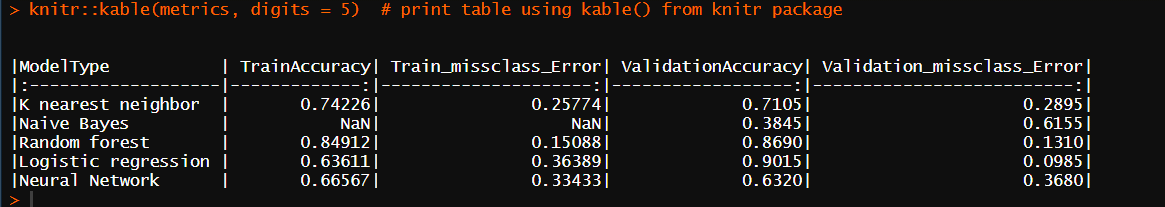


We have gotten 94.95% accuracy by SVM.

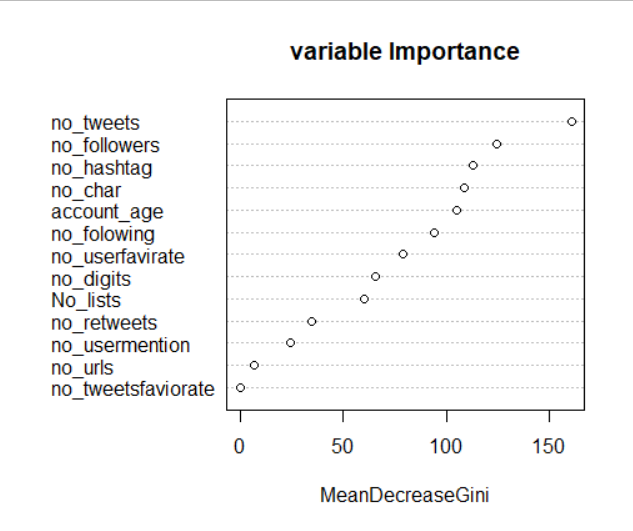
**Performance Comparison:** The algorithms used in the study consist of Logistic Regression, Radom Forest, Gradient Boosting, Support Vector Classifier and Neural Network. Without oversampling and parameter optimization, all algorithms show around 90% accuracy overall but about 20% sensitivity on fitting and prediction of tweets spam.  After applying cross validation, all the algorithms except the Support Vector Classifier showed improvement on balanced sensitivities and the top two improvements were from Gradient Boosting and Neural Network. The following outputs are then generated from Logistic Regression, Gradient Boosting and Neural Network. Here Logistic Regression is served as a baseline other algorithm to be compared with.

Compared with those true values, the prediction output on test set shows:  
1. The baseline model is accurate in making predictions in general: It is able to predict correctly over 90% of the time. However, it can only predict tweets spam correctly 23% of the time. So the model is not informative enough to our project goal.  
2. Random forest has been improved on predicting tweets spam correctly to 66% of the time and it is able to filter out those who will not tweets spam 84% of the time.  
3. Neural Network has been improved further on predicting tweets spam correctly to 84% of the time, however, its ability of filtering out those who will not spam was reduced to 43% of the time.  
4.  Compared with random forest, Neural Network prediction is more expensive because of its failure to identify those who will not spam.

**Model Comparison:**

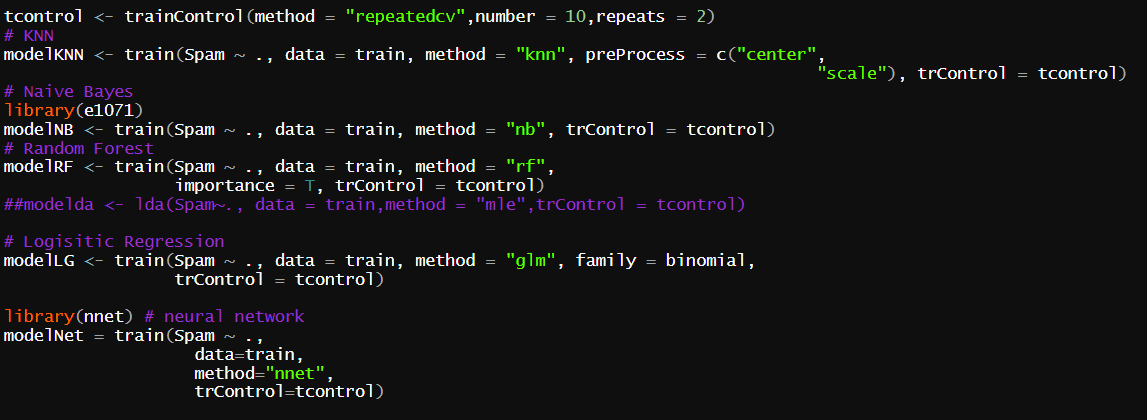


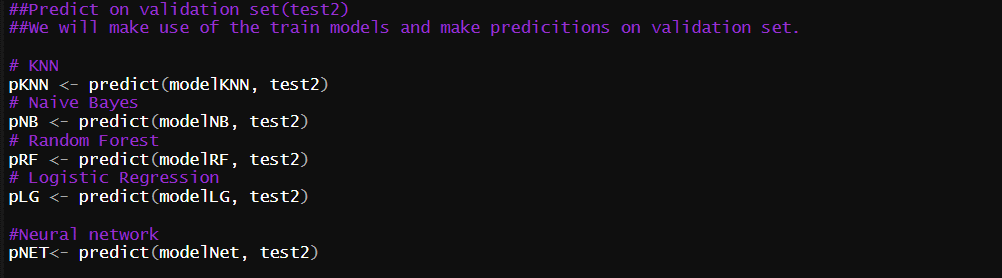
Finally, as shown below, Random Forest are able to identify those features which have important impacts on the response variable. These outputs supply very valuable insight to guide further cyber security.



**Logistic Regression**, in turn, is a Linear Regression transformed via a sigmoid function to produce a logit between 0 and 1 signifying the probability that each instance belongs to the positive class. As a linear function, it is advantageous in that it produces a coefficient of each term and that its log loss cost function is guaranteed to find a global minimum. It is disadvantageous in that it assumes the relationship of the data is inherently linear.

we make use of the train models and make predications on validation(test2) set:





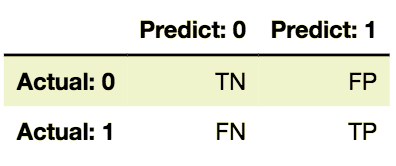
**Validity of the-model:**

## The essence of building the spams classifier is for the model to be able to effectively categories an incoming email as either spammer or non-spammer. The Random Forest and Naive Bayes performed exceptionally well in this project. Establishing Performance Measures

Evaluating a supervised classification model's effectiveness lies in comparing the model's predictions against the actual breakdown of positive and negative classes.

This breakdown is made most explicit via a **2x2 confusion matrix**, in which the columns indicate the true output values, and the rows indicate the model's output predictions (always pay attention as the rows and columns are sometimes switched!). Each cells represent a combination of the two: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN).

Here is a quick mock-up via a Pandas DataFrame:

[](https://nycdsa-blog-files.s3.us-east-2.amazonaws.com/2019/09/4ee8b947d567d0b409228b97a897665a/Screen-Shot-2018-08-03-at-7.21.58-PM.png)

Evaluative metrics, such as accuracy, precision, recall, and specificity, are composed of various sub-components of the confusion matrix. As my model's original output will show, evaluating these metrics in tandem, as opposed to relying on a single metric, is critical to interpreting the model's success.

**Accuracy**measures the correctness of **all*predictions***, regardless of class. It is measured as (TP + TN)/(TP + FP + TN + FN).

**Precision** measures the correctness of the ***positive predictions***. It is measured as the TP / (TP + FP)

**Recall** measures the correctness of the ***positive actual values***. It is measured as the TP / (TP + FN)

**CONCLUSION:** Cyber-security represent a common and dangerous threat for every individual who is connected to the internet and all enterprises. As individual using an Android device, cautious downloads of apps can prevent spam activities from occurring on those devices. Increased cyber security training for users is highly recommended to counteract the threat of phishing emails. Ransomware has become a new and lucrative trend among the hacker community, prompting increased investments in cyber security across all industries. Neural Network has the best ability to predict spam tweets but gives no insight on feature influence.Unsupervised machine learning can be used to identify factors that cluster organizations into high-risk or low-risk groups, which in turn can provide helpful information to implement security measures that can not only detect cyber threats faster but also prevent recurring security. In the future, Deep Learning will be another powerful tool to detect unknown network intrusions.

**References**

* D. Schuff, O. Turetke, D. Croson, F 2007, ‘Managing Email Overload: Solutions and Future Challenges’, IEEE Computer Society, vol. 40, No. 2, pp. 31-36
* N. Kushmerick, T. Lau, 2005, ‘Automated Email Activity Management: An Unsupervised learning Approach’, Proceedings of 10th International Conference on Intelligent User Interfaces, ACM Press, pp. 67-74
* L. Zhou, E Hovy, 2005, “On the Summarization of Dynamically Introduced Information: Online Discussions and Blogs”,`In Proceedings of AAAI-2006 Spring Symposium on Computational