

Student: Suhani

Student Number: 119220491

Module: Data Mining (CS6405)

Course: MSc. Data Science and Analytics.

Lecturer: Dr. Alejandro Arbelaez

Declaration:

By submitting this assignment. I agree to the following:

“*I have read and understand the UCC academic policy on plagiarism and I agree to the* *requirements set out thereby in relation to plagiarism and referencing. I confirm that I have referenced and acknowledged properly all sources used in preparation of this assignment.*

*I declare that this assignment is entirely my own work based on my personal study. I further declare that I have not engaged the services of another to either assist me in, or complete this assignment*”

**Report: Text Mining and classification Newsgroup Dataset**

**Introduction:** This Project is entirely dedicated to 3 Parts. In the first part I have just explored dataset, data wrangling and corpus creation as per the requirement of project.

In the second part, Basic evaluation on three classifiers namely Naïve Bayes, Knn,Random Forest is done . Third part is Robust Evaluation, which include data cleaning, Holdout and cross validation, feature selection, Hyperparameter tuning for models: (knn, Random forest, SVM

And Decision tree. In the last section I have included conclusion on different classifiers on the basis of performance metrics I have reported as per results I have received in R.

**Part1: Exploration of the dataset**

**Obtaining Data:**

A vector with the names of the four newsgroups is created. I have stored the pathname in “newsPathNAme”

variable. I decided to create a single corpus for all 4 four folders and then stack labels (newsType) to it later.

The libraries I’m using for this project are as follows:

|  |
| --- |
| library(tm)  library(e1071)  library(tm)  library(plyr)  library(class)  library(caret)  library(wordcloud)  library(rpart)  library(tree)  library(caret)  library(mlr) |

Using the lapply() function, I was able to obtain the 4 relevant classes from the folders and scrape them into 4 lists.

**Data Wrangling and Corpus creation:**

Now that the data has been imported, I need to get it into a format that will be useful going forward. I took the 4 list elements and combined them into a data frame. For this, I have used TermDocumentMatrix and Corpus.

I have made two basic methods for ensuring this to happen “buildTDM” and “bindNewsgroupToTDM”.



Now, I have dataset with columns which have the words and rows having the frequency of them for all 400 documents present in the Newsgroup folder for all 4 classes “comp.sys.ibm.pc.hardware”,” sci.electronics”,

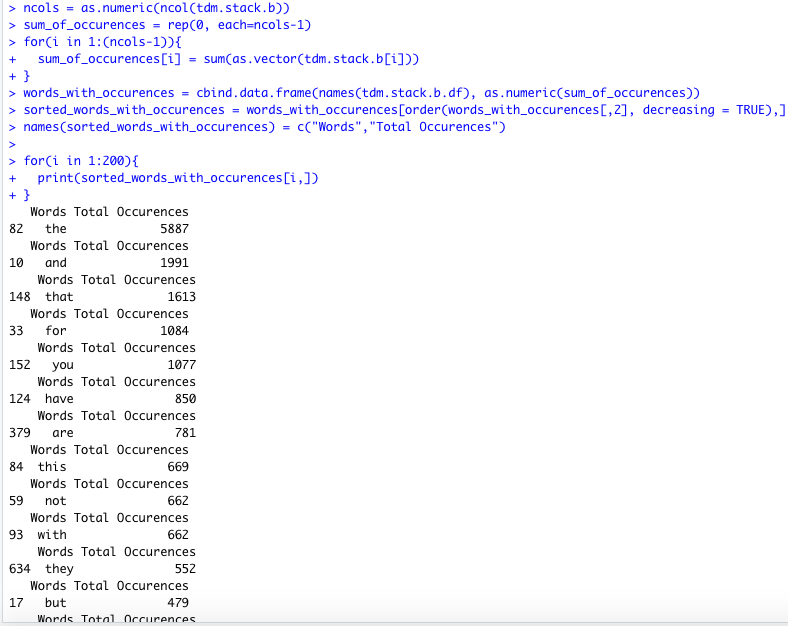
” talk.politics.guns”,” talk.politics.misc”.

I did check the dimension of data frame without cleaning of the data which is as follows:



**Computation of top 200 most popular words:**

For computing top 200 most common words i have written below mentioned code which helps to get words and their frequency in sorted order so that I can get it printed as per the requirement.



**Filtering token by length min. size=4, max size=20:**

A simple piece of code for extracting words with min. length 4 and max length 20 as shown below has been

written, which clearly shows that its returning words, no. of times it occurred(Occurences) and word length.

From the output it can be referred:

* “that” is the word with word length “4” and having number of occurrences as “1613”.
* Similarly, for “have”, “this”, “with”, “they” no. of occurrences is very high. But at the same time these

words are stop words’ tokens can be removed if they don’t add any new information to the Model

Classification problems normally don’t need stop words because it’s possible to talk about the

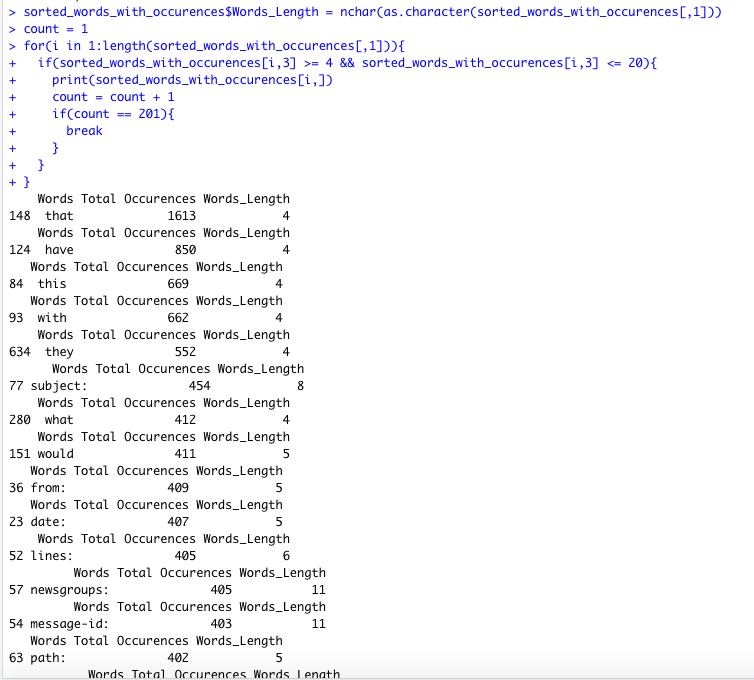
general idea of a text even if you remove stop words from it.

For Exploration part I am keeping these stop words but in robust evaluation, i will be removing these as

reducing the data set size is without any doubt a way of increasing performance. Training models takes time and if I have less tokens to be trained, the training time should decrease. It’s all matter of performance of the

Model being trained.

**Code and Output Reflected below:**



**Part2: Basic Evaluation**

**Building Different classification models**

**Naïve Bayes Model:**

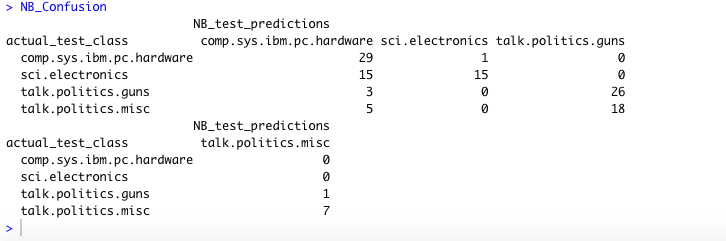
Building a Naive Bayesian Classifier for our 4 categories, comp.sys.ibm.pc.hardware, sci.electronics, talk.politics.guns, talk.politics.misc

I have built a classifier which uses term frequency as the single, relatively simple, feature for classification. Text input is tokenized (split up into individual words without punctuation) and then a frequency table constructed mapping each token to the number of times it’s used within the document.

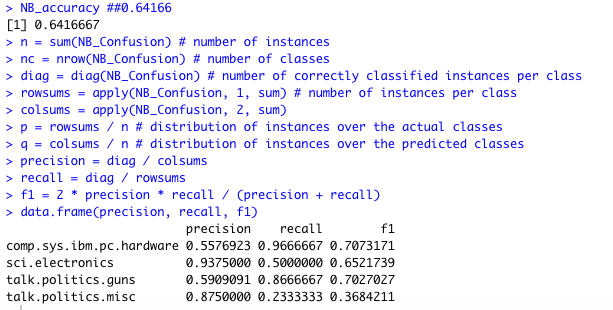
**Accuracy and confusion Matrix for Naïve Bayes:**

> NB\_accuracy

[1] 0.6416667



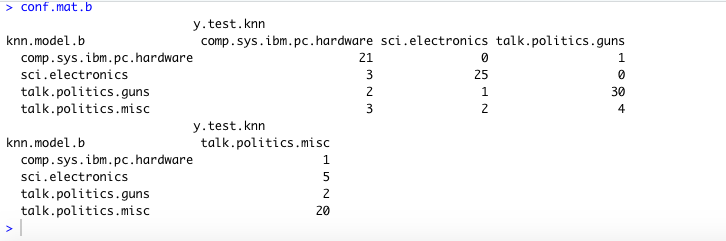
**Precision,Recall and F1 Score:**



**Knn Model:**

I have fit Knn model for basic evaluation without cleaning anything from my term document matrix:

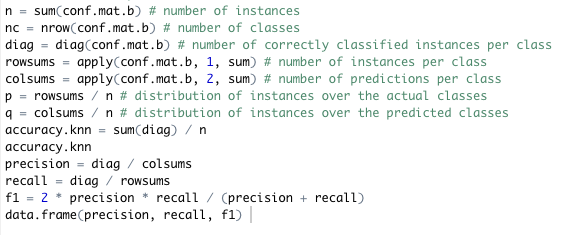
**Accurracy and confusion matrix for Knn:**

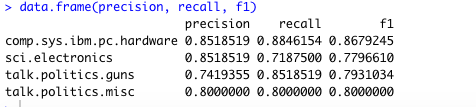




**Precision, Recall and F1 Score:**

**Method to calculate:**

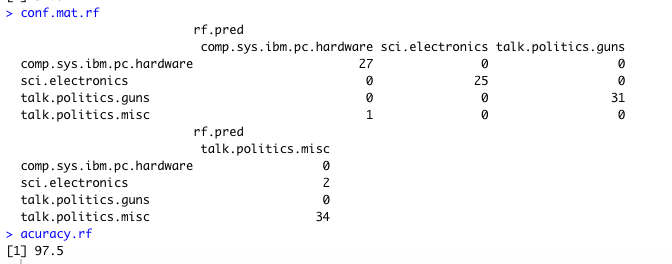




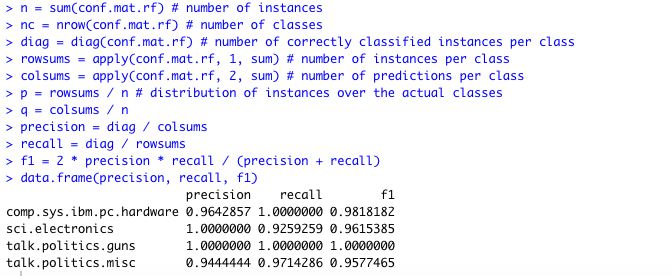
**Random Forest Modelling:**

we can fit the random forest classification to the training set, ensuring our data frame of predictors does not include the dependent variable.

**Confusion matrix and accuracy for random forest:**



**Precision, Recall and F1 Score:**

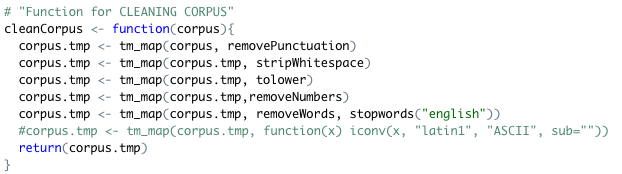


**Part3: Robust Evaluation**

1. **Cleaning the data:** we need to clean everything up.

* Removing stop words (and, the, etc.) to each corpus are both important steps.
* In addition to this, we need to stick everything in lower case.
* Removing all the numbers and punctuation, and make sure there are no tabs/unnecessary white space.

I have developed small method to implement cleaning:



1. **Feature Selection and engineering:** After Cleaning, I have selected important feature

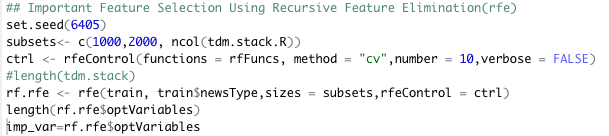
From my feature list to ensure better performance and classification of models I am

going to fit further. For Feature selection I have used “rfe” i.e. Recursive Feature

Elimination. It’s a feature selection method that fits a model and removes the weakest

feature (or Features) until the specified numbers of features is reached.

**Code:**



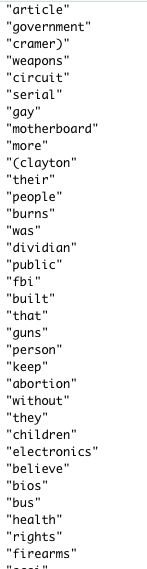


\* Important to note that now my feature list has been shortened to 1000 variable from 24789. I will use this feature list going ahead for further modelling. I have stored them in ‘imp\_var’(variable) I have made new dataset on which I will be running robust evaluation: newtrain=train[,imp\_var], newtest=test[,imp\_var]

**Holdout and CrossValidation:**

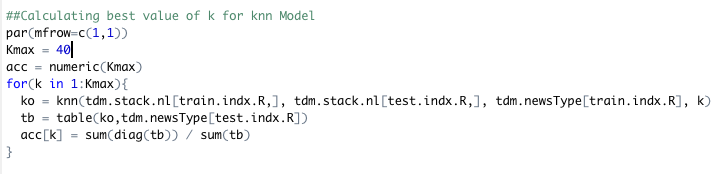
As seen above Holdout has been performed by selecting 70 :30 ratio of data set i.e. on 70 percent I will be training and on 30 % I will be testing. Cross-Validation is a very powerful tool. It helps us better use our data, and it gives us much more information about our algorithm performance. In complex machine learning models, it’s sometimes easy not pay enough attention and use the same data in different steps of the pipeline. This may lead to good but not real performance in most cases, or, introduce strange side effects in others. We have to pay attention that we’re confident in our models.

**Glimpse of features which, we can see in output now for example:**



**Knn Robust Evaluation**

1. **Selecting best value of “k” for knn Model:** I have used below loop to check the best value of k.



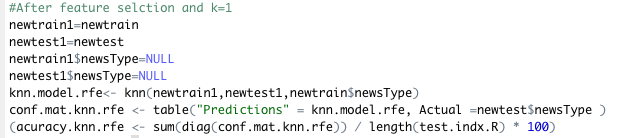
**Plot of error with respect to k value:** we can see from the plot that classification error is

minimum for (k=1).

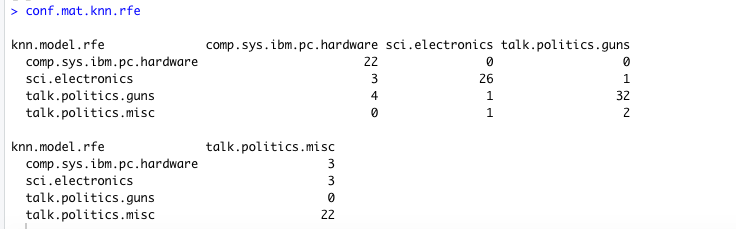


**Hyperparameter tuning for knn:**

With selected new features and k=1, I tried checking accuracy improvement for knn



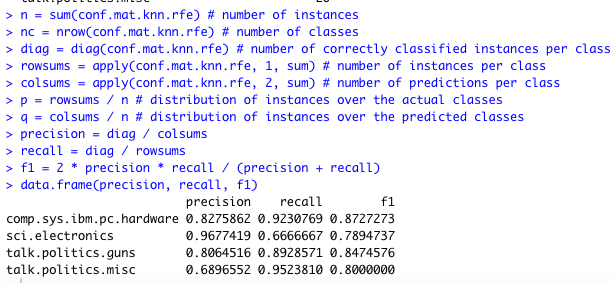
**Confusion matrix for knn:**



**Accuracy for knn:**



**Precision,Recall and F1 score:**

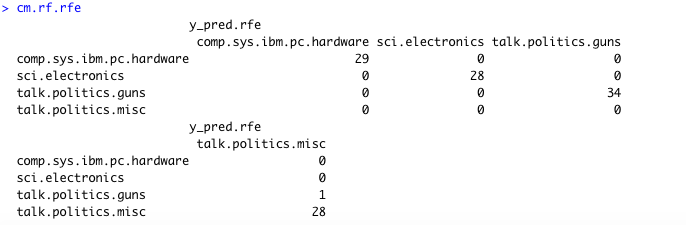


**Conclusion:**I found accuracy has been improved for knn from 80 to 85%

**Random Forest Robust Evaluation**

1. Selecting ntree =5000 and with selected feature I have fit random forest model:

**Confusion matrix for Random Forest:**



**Accuracy for random forest :**

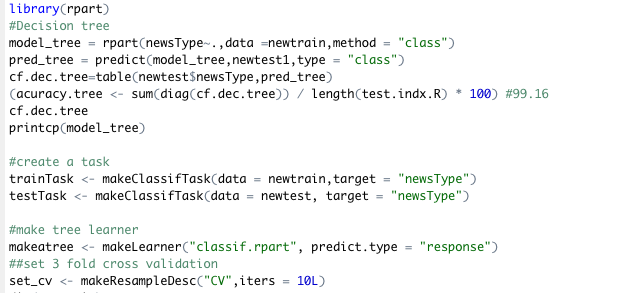


**Conclusion:** I did get improvement in random forest model accuracy from 97.5 to 99.16 after doing tuning.

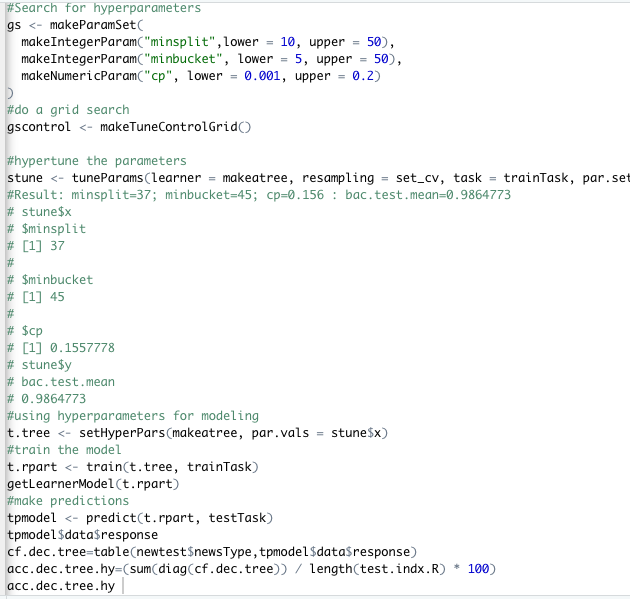
**Decision tree for Robust evaluation**

A decision tree captures non-linear relations better than a regression model. Let’s see if we can improve our model further. This time we’ll hyper tune the tree parameters to achieve optimal results. To get the list of parameters for any algorithm, I am using (rpart) here, snippet of code has been posted below:

Tuning the hyperparameter for decision tree:

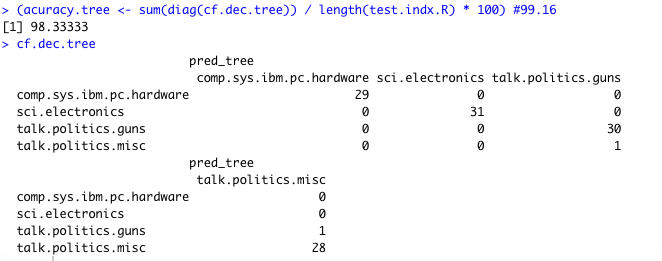


I’ve set 3 parameters. minsplit represents the minimum number of observations in a node for a split to take place. minbucket says the minimum number of observations I should keep in terminal nodes. cp is the complexity parameter. The lesser it is, the tree will learn more specific relations in the data which might result in overfitting.

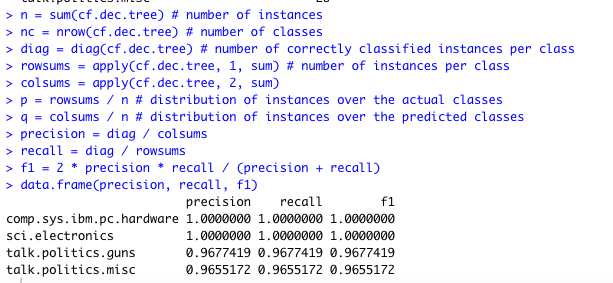


As can be seen in the code I have used tune parameters to fit new decision tree and confusion matrix and accuracy are as follows:

**Confusion matrix and accuracy for tuned decision tree:**



**Precision, Recall and F1 score:**

****

**Conclusion:**

Decision Tree is doing no better than random forest. This algorithm has returned the accuracy of 98.33% .

**Support Vector machine Robust evaluation**

Support Vector Machines (SVM) is also a supervised learning algorithm used for regression and classification problems. In general, it creates a hyperplane in n dimensional space to classify the data based on target class.

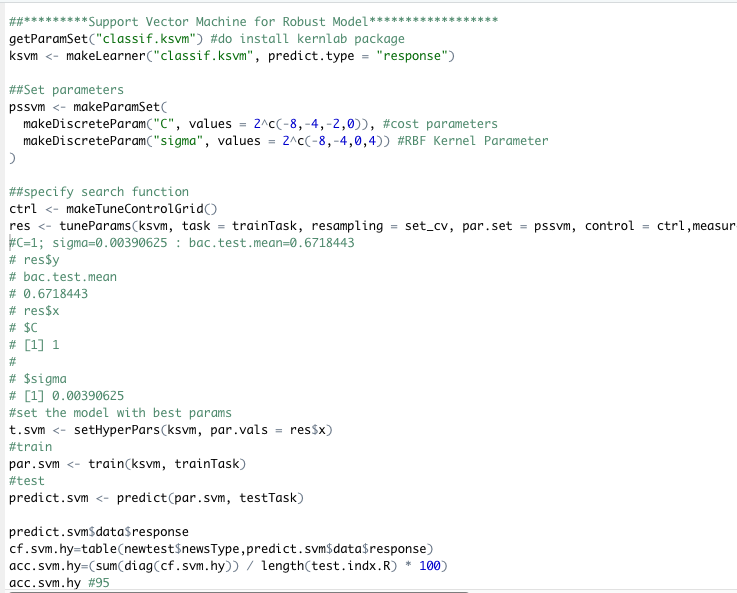
Using same method as for decision tree I have tried tuning parameter for SVM

And use them to fit another model to calculate best accuracy model:

**Best parameter selection by SVM:**

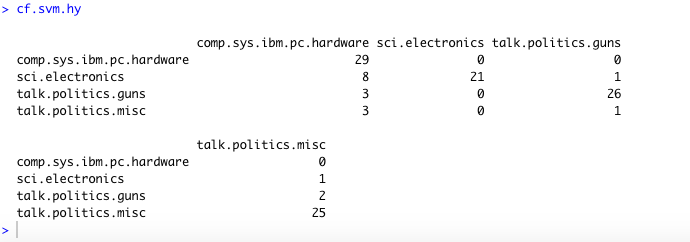


After tuning I have got C=1 and sigma =0.00390625 as my best parameter





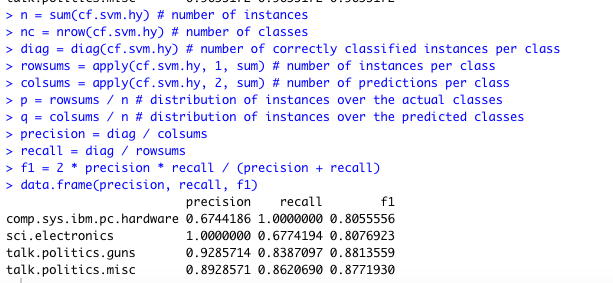
**Confusion matrix and accuracy for SVM:**

****

> acc.svm.hy #95

[1] 95

**Precision,Recall and F1 score:**

****

**Conclusion:**

This model returns an accuracy of 95 %. Which is very good,but lesser than highest score.

**Final Conclusion**

* During exploration of data set, it was noticed that there are lots of stop words and punctuation marks, numbers etc. were present in data set we need to remove them so that the machine learning classifiers can predict the classification more accurately.
* **For Basic evaluation** I got accuracy’s as below:

Naïve Baye’s : 64.16%

Knn :80%

Random forest: 97.5%

Hence, I am concluding that Random forest is the best classifier for my classification

Problem.

* **For robust evaluation:** After cleaning the data and feature selection, holdout and cross validation on random forest model and hyper parameter tuning accuracies are:

Knn. :85%

Random forest:99.167%

Decision Tree:98.3%

Support Vector Machine:95%

It can be noticed that after cleaning and Feature selection, hyperparameter tuning the parameters there is an increase in accuracies. If I have to select any one classifier I would say random forest is best classifier for my problem set.

**References:**

1.class notes of Module CS6405

2. Medium. 2020. *Random Forest Text Classification: Trump V. Obama*. [online] Available at: <https://towardsdatascience.com/random-forest-text-classification-trump-v-obama-c09f947173dc> [Accessed 3 April 2020].

3. Package, P., 2020. *Practicing Machine Learning Techniques In R With MLR Package*. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2016/08/practicing-machine-learning-techniques-in-r-with-mlr-package/> [Accessed 3 April 2020].

4. Martinschweinberger.de. 2020. [online] Available at: <http://martinschweinberger.de/docs/articles/TextClass.pdf> [Accessed 3 April 2020].