Spam Analysis

Sami Dia

<u>Objective:</u> Classification of spam emails.

<u>Introduction</u>

This report is about the effectiveness of a number of different methodologies applied to the classification of emails. The objective is to classify each email as either spam or not spam. In most email services there is a hidden filter that receives any incoming email and either puts it into your inbox or into your spam box. The aim was to find the best method to determine the relevance of the email is coming in with the smallest misclassification rate. There should be a minimum amount of non-spam emails being sent to the spam box and vice versa with spam email being sent to the inbox.

<u>Data</u>

The data set was found on the UCI Machine Learning Repository and it contains word and character frequencies from actual emails. In the data set there are 3 attributes that are non-frequency measurements. These 3 attributes measure the length of sequences of consecutive capital letters. The last column of the data set gives the classification of the email, whether its spam or non-spam email. The UCI Machine Learning Repository has really good descriptions for the attributes. Here is a summary of the data set's attributes and details about how the values were calculated (all of the following was taken from the data set's website, which is referenced):

"48 continuous real [0,100] attributes of type word_freq_WORD = percentage of words in the email that match WORD, i.e. 100 * (number of times the WORD appears in the email) / total number of words in email. A "word" in this case is any string of alphanumeric characters bounded by non-alphanumeric characters or end-of-string.

6 continuous real [0,100] attributes of type char_freq_CHAR] = percentage of characters in the email that match CHAR, i.e. 100 * (number of CHAR occurrences) / total characters in email

1 continuous real [1,...] attribute of type capital_run_length_average = average length of uninterrupted sequences of capital letters

1 continuous integer [1,...] attribute of type capital_run_length_longest = length of longest uninterrupted sequence of capital letters

1 continuous integer [1,...] attribute of type capital_run_length_total = sum of length of uninterrupted sequences of capital letters = total number of capital letters in the email

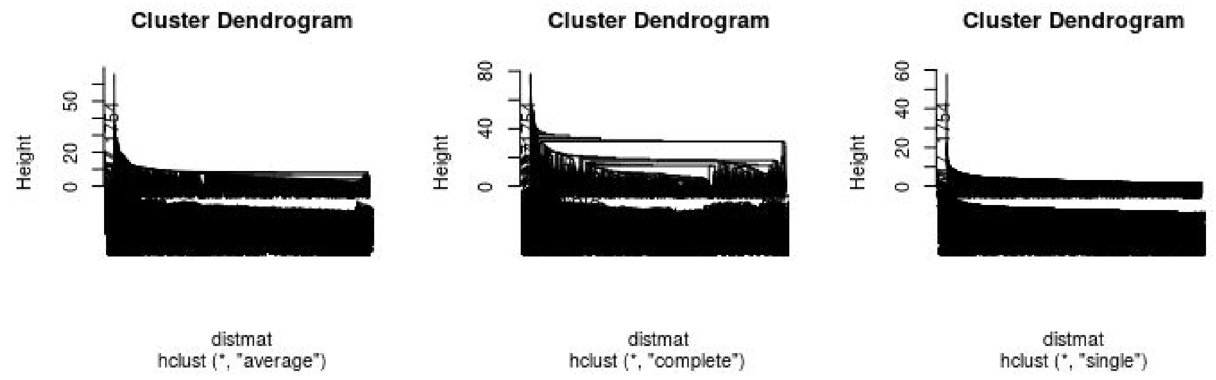
1 nominal {0,1} class attribute of type spam = denotes whether the email was considered spam (1) or not (0), i.e. unsolicited commercial email. “

It is worth mentioning that the actual emails are internal technical reports from Hewlett-Packard.

Methodology

Hierarchical clustering (single, complete and average linkage)

Hierarchical clustering was performed on the spam data set with the hopes of isolating two clear groups, thus classifying the emails in spam and non-spam groups. The data set was scaled, most of the variables are frequencies, but there are a few attributes that are not frequencies (e.g capital_run_length_longest). A distance matrix was created, which was used to generate three different clusters, with single, average, and complete linkage used as the method of linkage. The dendrogram plots of the clusters are shown below (order: average, complete and single linkage):



Based on the dendrograms one can see that hierarchical clustering is not a useful statistical analysis for the spam/non-spam email classification, since clearly there are no 2 clusters that would classify the emails properly. One can note that the observations seem to chain to each other on the dendrograms, even though the data set was scaled. 4600 observations are in one cluster and 1 in the second cluster, when the cluster trees are cut at 2 groups. Out of the 4600 observations only 2788 observations should be clustered into one group, thus the result of this statistical analysis is a 39.38 % misclassifications rate.

knn Classification

K-Nearest Neighbours was used to try and classify the dataset. The principle behind the analysis is to predict the point using the predictor variables then based on the k-nearest neighbours it classifies that point into the category that has the highest percentage of a particular group in those k neighbors. For the results k = 5 was found to the best k through trial and error of different k values; k =5 yielded the smallest misclassification rate. This was first trained on the train set and then predicted using the testing set of values for all the variables in the data set. The results were then compared to the actual results to see how close the model was at predicting the results. The misclassification rate was 18.6% which is large enough to determine that this method is not good enough to classify the spam email data set.

K-means clustering

K-means clustering was used on the dataset to see how useful this method would be. This is where you select a random k points in the data set and it slowly converts the points around them to be a part of the k groups. In the case of this data, set k = 2 since the mail can only be spam or not spam. It was first used on the training set then used to predict the results using the test data and all variables. Comparing with the actual classification yielded a misclassification of 36.4% which was one of the worst results obtained. K-means should not be used for the type of analysis.

Linear Discriminant Analysis

A linear discriminant analysis model was built with a model formula created by the stepAIC function in R, which used backward selection to find a suitable model formula. The linear discriminant analysis was performed on the training data set of 3601 observations and the response variables were predicted for a 1000 observation testing data set. The linear discriminant analysis calculated the probabilities that an observation would be in a specific group by parameter estimation. For linear discriminant analysis all the covariance matrices are the same, therefore there are not as many parameters to estimate as a quadratic discriminant analysis would require. After all the estimations are done for the parameters that needed to be estimated, the linear discriminant analysis model classifies observations to the group that they most likely would belong to based on maximum likelihood estimation. The results of the linear discriminant analysis performed on the spam data set were 11.0 % misclassification on the testing set. The response variable, in this case the group that an observation belongs to was given by the class attribute of the linear discriminant analysis model.

Linear Discriminant Analysis with cross validation

Leave-one-out cross-validation was used with a linear discriminant analysis model to get more accurate results by validating the predictions on the testing data set.The predictions were cross validated by leaving one observation out and creating 1000 validations sets, then averaging out the results to come up with a more accurate predictions for each observation’s response variable. The result obtained on this particular statistical analysis was 11.8 % misclassification on testing data set.

Quadratic Discriminant Analysis

A quadratic discriminant analysis (QDA) model was built with a model formula created by the stepAIC function in R, which performed backward selection on the entire data set to find a suitable model formula. The quadratic discriminant analysis was performed on the entire data set of 4601 observations and the response variables were predicted for the entire data set as well. The quadratic discriminant analysis calculated the probabilities that an observation would be in a specific group by parameter estimation. Unlike in the linear discriminant analysis (LDA), all the covariance matrices vary in quadratic discriminant analysis; therefore there are a lot more parameters to be estimated. Since there are more parameters to be estimated, the training set did not have enough observations to estimate all the parameters, thus one had to use the entire data set to perform this specific statistical analysis. After all the estimations are done, the quadratic discriminant analysis model classifies observations to the group that they most likely would belong to based on maximum likelihood estimation, but QDA has different decision boundaries than LDA. The result of the quadratic discriminant analysis was 17.12671 % misclassification on the entire data set. The response variable, in this case the group that an observation belongs to was given by the class attribute of the linear discriminant analysis model.

Quadratic Discriminant Analysis with cross validation

Leave-one-out cross-validation was used with a quadratic discriminant analysis model to get more accurate results by validating the predictions on the entire data set.The predictions were cross validated by leaving one observation out and creating 4601 validations sets, then averaging out the results to come up with a more accurate prediction for each observation's response variable. The result obtained on this particular statistical analysis was 16.90937 % misclassification on the entire data set.

Logistic regression

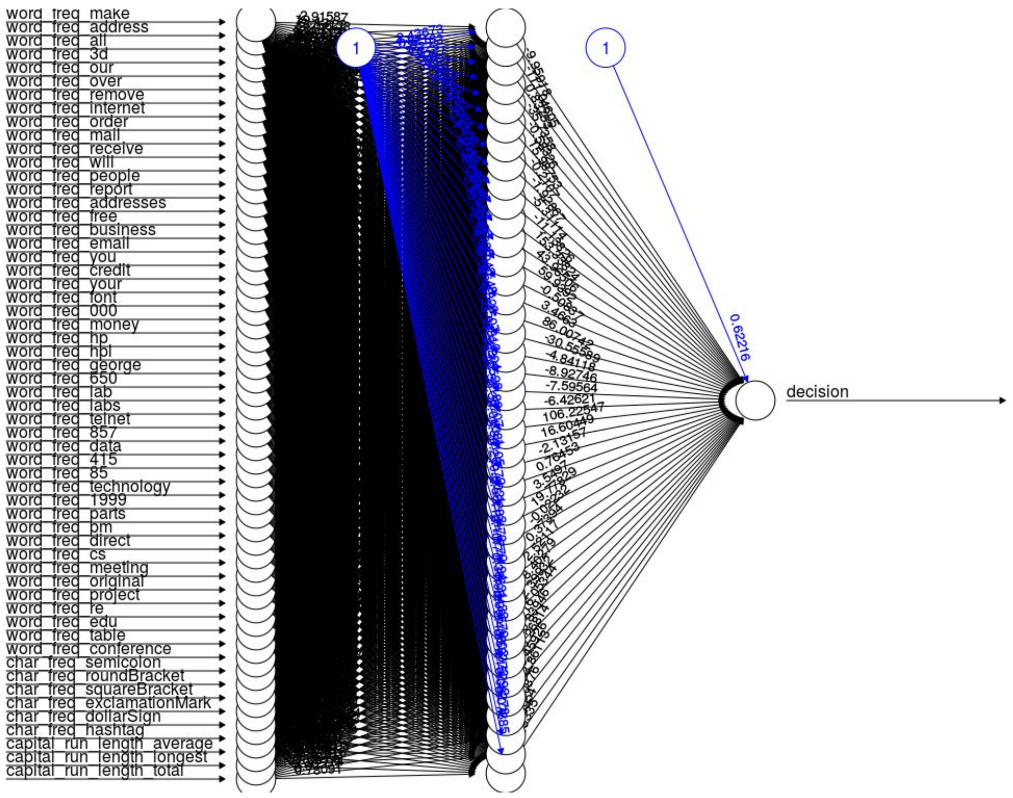
Logistic regression is similar to the normal linear model however this model assumes the output to be a classification into groups as opposed to being on a continuous scale from 0-N. This regression had also uniquely used all the variables with no form of selection to lower the variables to the most significant. After creating the logistic model on the training set it was predicted to the test set and then compared to the decision column to decide how bad the classification of spam email was to the actual determination of emails. The results for this model were actually very good and it performed above the expectation with a misclassification rate of 6.8% which is within the threshold of acceptable misclassification. The summary of the model is too long to bring into the report.

Artificial Neural Network

Several artificial neural networks models were built, which used all the attributes of the spam data set. The neural networks were trained on the training set consisting of 3601 observations and the response variable was predicted using the testing set containing 1000 observations. Deciding on the number of hidden layers in each neural network, also on the number of hidden variable in a layer is a difficult task. Michy Alice's article about 'Fitting a neural network in R' says that "one hidden layer is enough for a vast numbers of applications. As far as the number of neurons is concerned, it should be between the input layer size and the output layer size, usually 2/3 of the input size". In this case the input layer size is 57 attributes and the output layer size is 1 response variable, also 2/3 of the input size is 38; therefore around 38 hidden variables should be optimal. Just for the sake of getting as good results as possible, numerous neural network models have been built. The misclassification percentages with different combinations of hidden layers and variable are shown below:

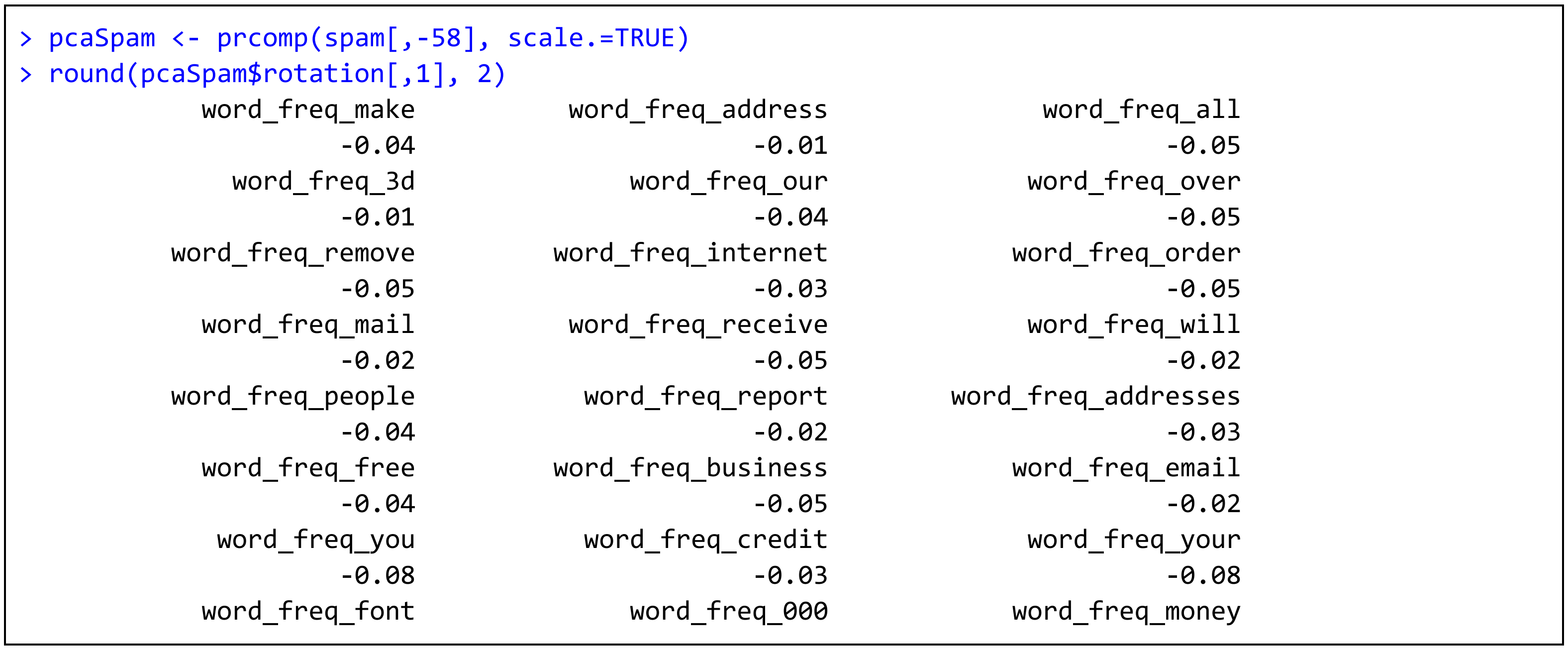
Misclassification %	**Hidden = ...**	**Misclassification %**	**Hidden = ...**
7.7	5	6.7	c(12,11)
6.6	10	8.9	c(18,17)
7	20	8.7	c(20,10)
7.6	30	7.6	c(20,15)
7	35	7.5	c(20,18)
6.9	36	6.9	c(25,13)
6.4	37	7.6	c(25,15)
8.1	38	8.3	c(20,20)
6.7	39	8.5	c(25,20)
6.4	40	8.2	c(10,10,5)
7.1	45	8.1	c(10,10,10)
6.7	50	8.5	c(15,10,5)
6.8	c(7,7)	7.9	c(15,10,10)
7.3	c(10,10)	7.7	c(15,15,10)
7.4	c(15,10)	7	c(15,15,15)
6.7	c(15,15)	9	c(20,10,10)
7.8	c(13,13)	7.6	c(20,15,10)
7.4	c(11,11)	9.9	c(20,15,15)

As you can see the best performance of artificial neural networks is 6.4 % misclassification with a neural network model with only one hidden layer and 37 or 40 hidden variables on the hidden layer. The two models have similar performances. A plot of this model (40 hidden variables) can be seen below:



Principal Component Analysis

Principal component analysis was performed on the scaled 4601 observation spam data set. A screenshot with the loadings of the first principal component rounded to 2 decimal places can be seen below:



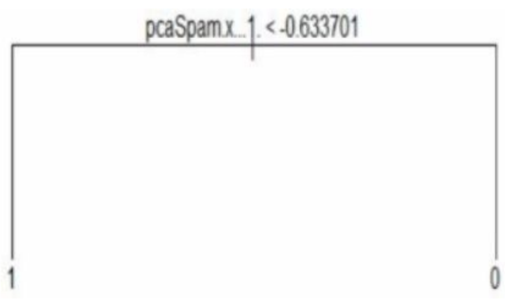
```
> pcaSpam <- prcomp(spam[,-58], scale.=TRUE)
> round(pcaSpam$rotation[,1], 2)
          word_freq_make        word_freq_address            word_freq_all
                   -0.04                    -0.01                    -0.05
            word_freq_3d            word_freq_our           word_freq_over
                   -0.01                    -0.04                    -0.05
        word_freq_remove       word_freq_internet          word_freq_order
                   -0.05                    -0.03                    -0.05
          word_freq_mail        word_freq_receive           word_freq_will
                   -0.02                    -0.05                    -0.02
        word_freq_people         word_freq_report      word_freq_addresses
                   -0.04                    -0.02                    -0.03
          word_freq_free       word_freq_business          word_freq_email
                   -0.04                    -0.05                    -0.02
           word_freq_you         word_freq_credit           word_freq_your
                   -0.08                    -0.03                    -0.08
          word_freq_font            word_freq_000          word_freq_money
```

```
                   -0.01                       -0.05                        -0.04
            word_freq_hp                word_freq_hpl              word_freq_george
                    0.21                        0.21                         0.04
           word_freq_650                word_freq_lab                word_freq_labs
                    0.28                        0.22                         0.30
        word_freq_telnet                word_freq_857                word_freq_data
                    0.31                        0.35                         0.01
           word_freq_415                 word_freq_85          word_freq_technology
                    0.35                        0.27                         0.32
          word_freq_1999              word_freq_parts                  word_freq_pm
                    0.05                        0.00                         0.04
        word_freq_direct                 word_freq_cs             word_freq_meeting
                    0.32                        0.01                         0.02
      word_freq_original            word_freq_project                  word_freq_re
                    0.07                        0.01                         0.01
           word_freq_edu              word_freq_table          word_freq_conference
                    0.00                        0.00                         0.00
     char_freq_semicolon       char_freq_roundBracket      char_freq_squareBracket
                    0.00                        0.14                         0.02
char_freq_exclamationMark         char_freq_dollarSign            char_freq_hashtag
                   -0.04                       -0.05                         0.00
capital_run_length_average capital_run_length_longest   capital_run_length_total
                   -0.02                       -0.03                        -0.04
```

What seems to be interesting about the first principle component values is that the attributes ‘word_freq_hp’, ‘word_freq_650’ , ‘word_freq_technology’, ‘word_freq_telnet’, ‘word_freq_415’, ‘word_freq_direct’, ‘word_freq_857’, ‘word_freq_85’, ‘word_freq_lab’, ‘word_freq_labs’and ‘word_freq_hpl’ have larger scores for the first principle component. In order to interpret these attributes and their loading scores, one was able to take the rotated data and put it in a classification tree. A summary of the classification tree, and its splits are included below:

```
> dataFromPca <- data.frame(pcaSpam$x[,1])
> trainResponse <- factor(spamDecisions)
> pcaTree <- tree(trainResponse~., data = dataFromPca)
> summary(pcaTree)

Classification tree:
tree(formula = trainResponse ~ ., data = dataFromPca)
Number of terminal nodes:  4
Residual mean deviance:  0.7599335 = 3493.414 / 4597
Misclassification error rate: 0.149098 = 686 / 4601
```



This classification tree also classifies the emails as spam or non spam, and its misclassification percentage is 14.91 %. There is only one split in the classification tree, which classifies emails as spam (response = 1) when the first principal component's rotated data is less than 0.633701 and it classifies emails as non-spam (response = 0) otherwise. The first principal component's large loading scores can be interpreted as if the words 'hp', '650', 'technology', 'telnet', '415', 'direct','857', '85', 'lab', 'labs' and 'hpl' occur more often in an email, chances are that the email is not spam and this can be confirmed with the classification tree's results, but also the data set repository mentions some background information about what kind of emails are in the data set: "Hewlett-Packard Internal-only Technical Report. External forthcoming".

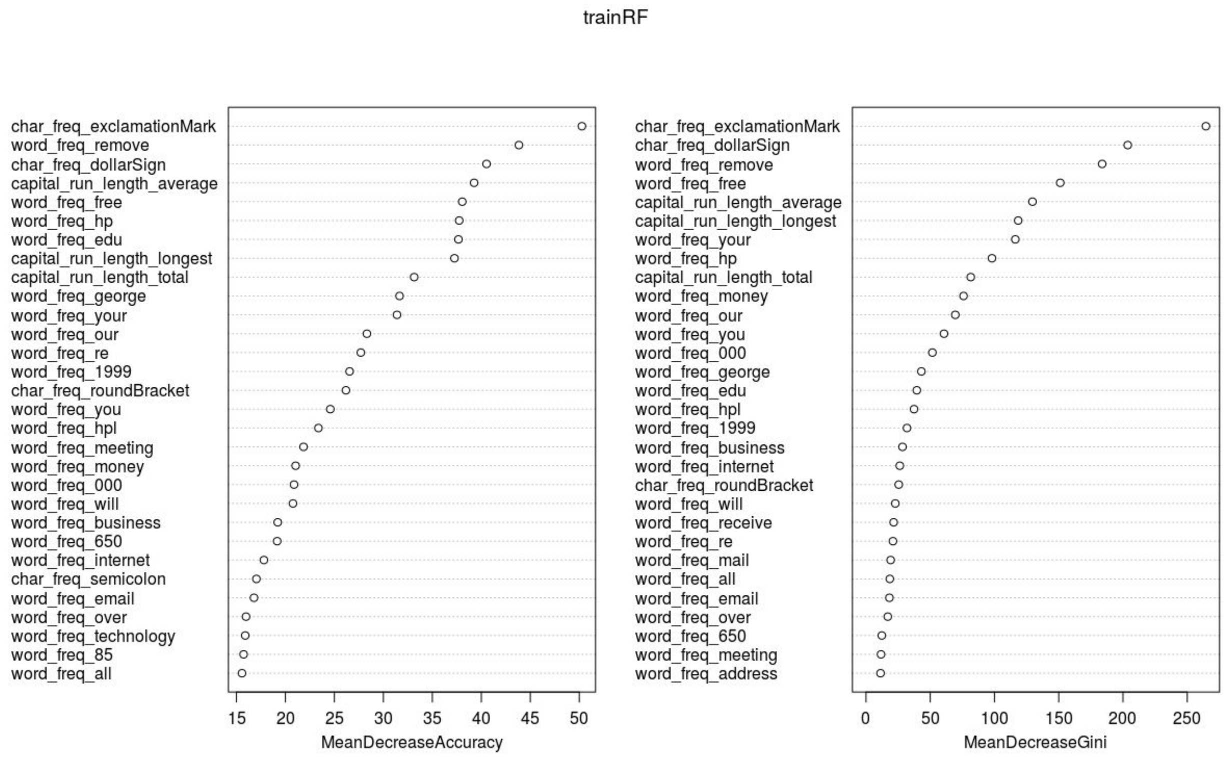
Bagging

Bagging was first conducted using the whole data set in order to identify subsets with repetitions. Bagging (bootstrap aggregation) is the method of applying bootstrap methodology to the entire model fitting process instead of just generating standard errors. Bootstrap samples usually leave out ⅓ of the observations. Therefore, cross-validation is essentially built into the model. By using the training and testing set, we attempt to estimate the repeated observations and calculate the misclassification rate. We have used the mtry as 57 because we have 57 prediction variables. This is used to generate a model that again classifies spam vs non-spam as factor variables. The number of trees in the model are 500 and generates a misclassification rate of 5.24%. Since bagging is an average of models, it loses its interpretability, and just has the end classification.

Random forest

Random forests use the identical procedure as that in bagging, except the mtry is set to default i.e. the square root of the number of variables in the dataset. At each split, (the square root of p) variables are removed. This decreases variance and therefore, increases stability in the model. The model still loses its interpretability but we are able to identify important variables. The model generates a misclassification rate of 4.75%. With reference to the plot, we believe that exclamation marks, "remove", dollar signs and the length of character letter inclusion are the variables that impact the classification the most.

Below one can see the variable importance of the data set based on the random forest model's mean squared error value.

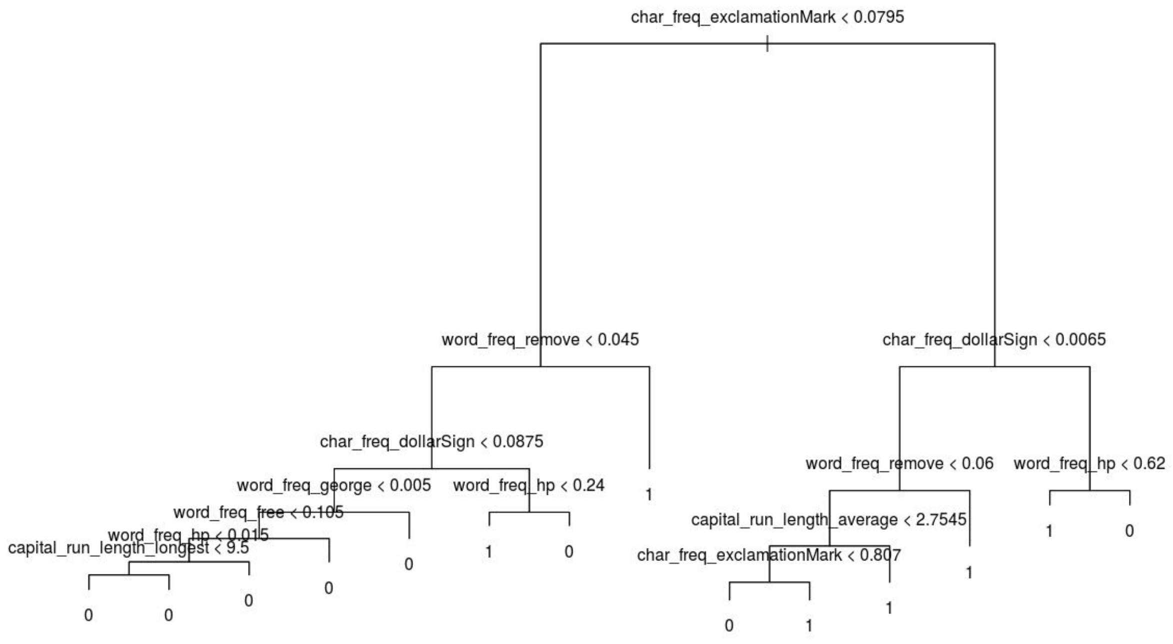


Classification Trees

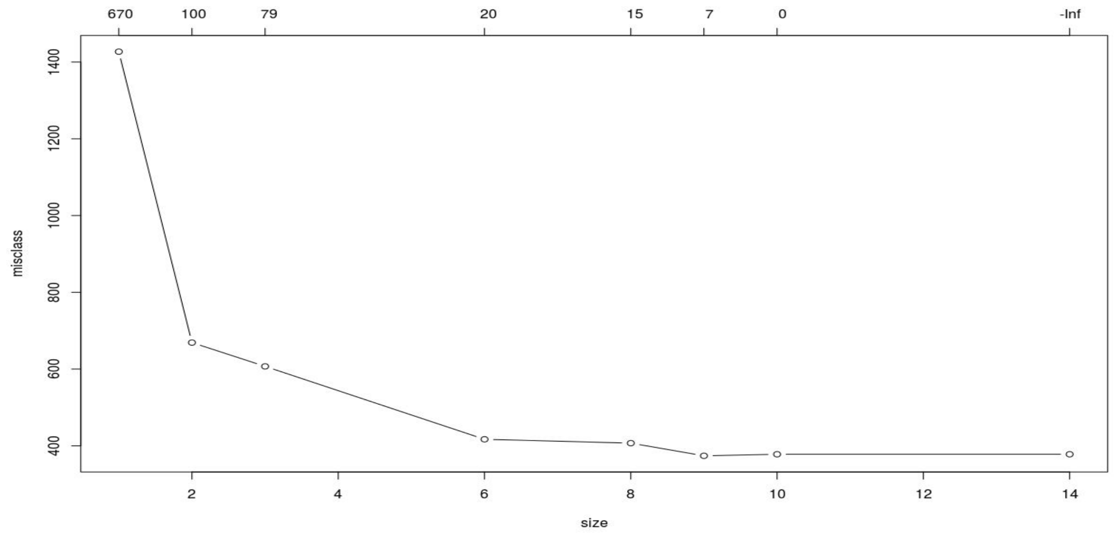
Classification trees were formed using the training data set and was then applied to the testing dataset. The decision variable was converted to a factor variable to easily generate classification. This generated a misclassification rate of 9.85% with 14 terminal nodes. Cross-validation was then conducted in order to prune the tree and make it more interpretable. We decided to prune at size = 9 because the misclassification curve evens out after that. Having a bigger tree than that would not give much benefit towards classification. This generates a cleaner classification tree as shown below with a misclassification rate of 10% in the training set.

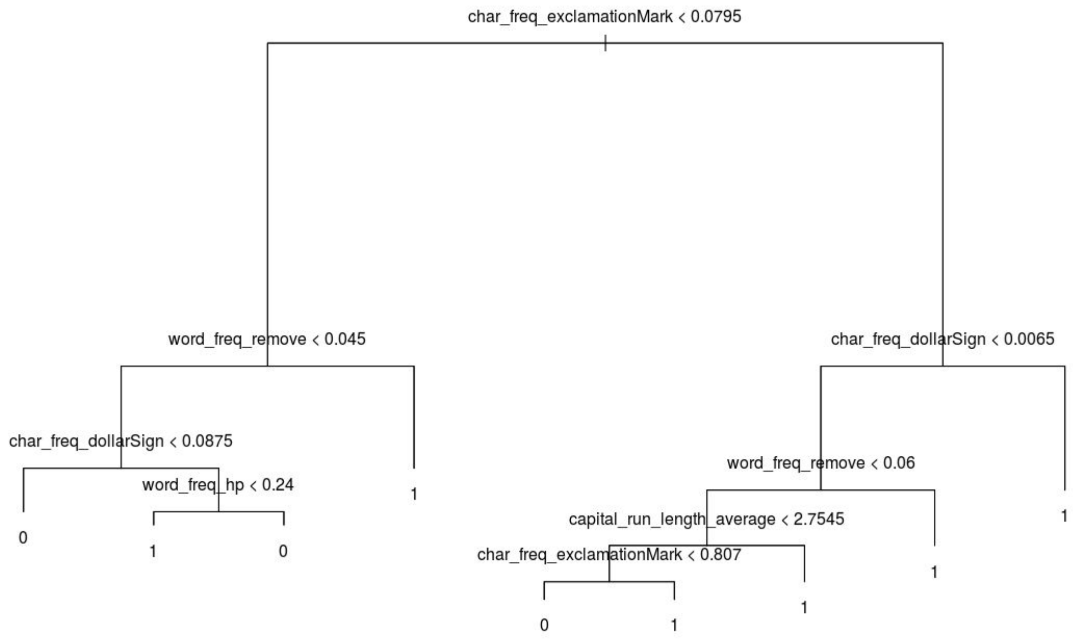
We then use the pruned tree to perform classification in the testing set and observe a misclassification rate of 9.9%.

Below there is an image of the classification tree's splits before the tree was pruned. A classification of 1 would make the email spam and a 0 would consider the email to be non spam.


char_freq_exclamationMark < 0.0795
word_freq_remove < 0.045
char_freq_dollarSign < 0.0065
char_freq_dollarSign < 0.0875
word_freq_george < 0.005
word_freq_hp < 0.24
word_freq_free < 0.105
word_freq_hp < 0.015
capital_run_length_longest < 9.5
word_freq_remove < 0.06
word_freq_hp < 0.62
capital_run_length_average < 2.7545
char_freq_exclamationMark < 0.807
0
0
0
0
0
1
0
1
0
1
1
1
1
1
0

Also one can notice the next plot, which shows the relationship between cross validation and pruning misclassification. A decision was made to prune the tree at size = 9 because the misclassification curve evens out after size equals 9.

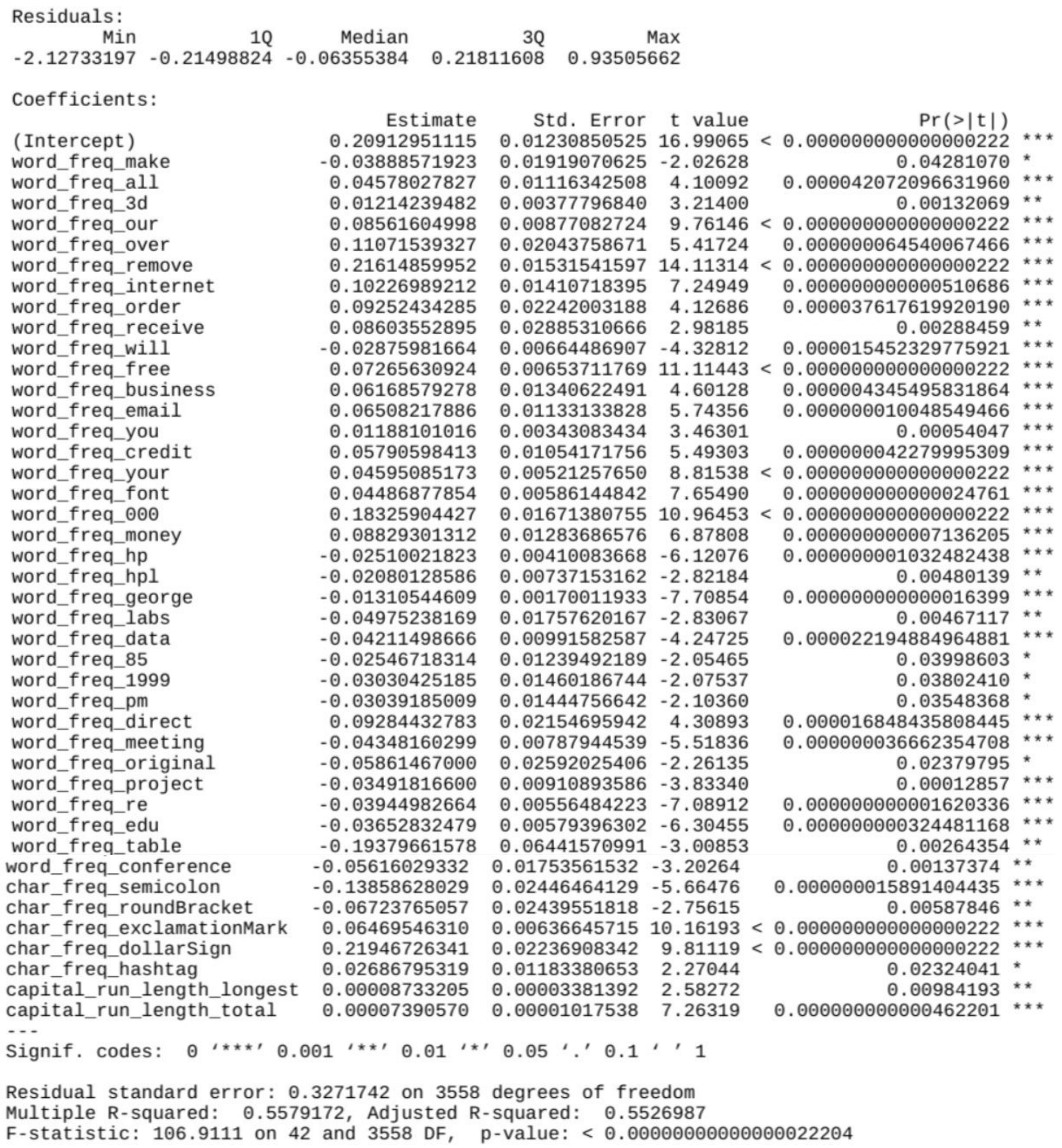



char_freq_exclamationMark < 0.0795
word_freq_remove < 0.045
char_freq_dollarSign < 0.0065
char_freq_dollarSign < 0.0875
word_freq_hp < 0.24
word_freq_remove < 0.06
capital_run_length_average < 2.7545
char_freq_exclamationMark < 0.807
0
1
0
1
0
1
1
1
1

Above there is an image of the pruned classification tree's splits and 'decisions'.

Multiple Linear Regression without interactions

Multiple linear regression was performed on the spam email dataset. The model was first found using backward selection on the training set and starting with all the variables predicting the spam column. After the backward selection completed using p-values was finished the model was then applied to the test set to get a prediction line. The predictions were then measured against the decision column to see how close the prediction came. It was found that the results were not simply 0 and 1 meaning that linear regression was not a suitable analysis for the spam classification. To try and get some of the results it was decided that rounding the answers would do even though we broke assumptions to model since a continuous number was not needed. With rounding it was found that there was a 12.1% misclassification rate which with all the faults considered was not a bad result. Below is the summary of the model used to get the results stated.



```
Residuals:
       Min          1Q      Median          3Q         Max 
-2.12733197 -0.21498824 -0.06355384  0.21811608  0.93505662 

Coefficients:
                                Estimate     Std. Error  t value               Pr(>|t|)    
(Intercept)                0.20912951115  0.01230850525 16.99065 < 0.000000000000000222 ***
word_freq_make            -0.03888571923  0.01919070625 -2.02628             0.04281070 *  
word_freq_all              0.04578027827  0.01116342508  4.10092   0.000042072096631960 ***
word_freq_3d               0.01214239482  0.00377796840  3.21400             0.00132069 ** 
word_freq_our              0.08561604998  0.00877082724  9.76146 < 0.000000000000000222 ***
word_freq_over             0.11071539327  0.02043758671  5.41724   0.000000064540067466 ***
word_freq_remove           0.21614859952  0.01531541597 14.11314 < 0.000000000000000222 ***
word_freq_internet         0.10226989212  0.01410718395  7.24949   0.000000000000510686 ***
word_freq_order            0.09252434285  0.02242003188  4.12686   0.000037617619920190 ***
word_freq_receive          0.08603552895  0.02885310666  2.98185             0.00288459 ** 
word_freq_will            -0.02875981664  0.00664486907 -4.32812   0.000015452329775921 ***
word_freq_free             0.07265630924  0.00653711769 11.11443 < 0.000000000000000222 ***
word_freq_business         0.06168579278  0.01340622491  4.60128   0.000004345495831864 ***
word_freq_email            0.06508217886  0.01133133828  5.74356   0.000000010048549466 ***
word_freq_you              0.01188101016  0.00343083434  3.46301             0.00054047 ***
word_freq_credit           0.05790598413  0.01054171756  5.49303   0.000000042279995309 ***
word_freq_your             0.04595085173  0.00521257650  8.81538 < 0.000000000000000222 ***
word_freq_font             0.04486877854  0.00586144842  7.65490   0.000000000000024761 ***
word_freq_000              0.18325904427  0.01671380755 10.96453 < 0.000000000000000222 ***
word_freq_money            0.08829301312  0.01283686576  6.87808   0.000000000007136205 ***
word_freq_hp              -0.02510021823  0.00410083668 -6.12076   0.000000001032482438 ***
word_freq_hpl             -0.02080128586  0.00737153162 -2.82184             0.00480139 ** 
word_freq_george          -0.01310544609  0.00170011933 -7.70854   0.000000000000016399 ***
word_freq_labs            -0.04975238169  0.01757620167 -2.83067             0.00467117 ** 
word_freq_data            -0.04211498666  0.00991582587 -4.24725   0.000022194884964881 ***
word_freq_85              -0.02546718314  0.01239492189 -2.05465             0.03998603 *  
word_freq_1999            -0.03030425185  0.01460186744 -2.07537             0.03802410 *  
word_freq_pm              -0.03039185009  0.01444756642 -2.10360             0.03548368 *  
word_freq_direct           0.09284432783  0.02154695942  4.30893   0.000016848435808445 ***
word_freq_meeting         -0.04348160299  0.00787944539 -5.51836   0.000000036662354708 ***
word_freq_original        -0.05861467000  0.02592025406 -2.26135             0.02379795 *  
word_freq_project         -0.03491816600  0.00910893586 -3.83340             0.00012857 ***
word_freq_re              -0.03944982664  0.00556484223 -7.08912   0.000000000001620336 ***
word_freq_edu             -0.03652832479  0.00579396302 -6.30455   0.000000000324481168 ***
word_freq_table           -0.19379661578  0.06441570991 -3.00853             0.00264354 ** 
word_freq_conference      -0.05616029332  0.01753561532 -3.20264             0.00137374 ** 
char_freq_semicolon       -0.13858628029  0.02446464129 -5.66476   0.000000015891404435 ***
char_freq_roundBracket    -0.06723765057  0.02439551818 -2.75615             0.00587846 ** 
char_freq_exclamationMark  0.06469546310  0.00636645715 10.16193 < 0.000000000000000222 ***
char_freq_dollarSign       0.21946726341  0.02236908342  9.81119 < 0.000000000000000222 ***
char_freq_hashtag          0.02686795319  0.01183380653  2.27044             0.02324041 *  
capital_run_length_longest 0.00008733205  0.00003381392  2.58272             0.00984193 ** 
capital_run_length_total   0.00007390570  0.00001017538  7.26319   0.000000000000462201 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3271742 on 3558 degrees of freedom
Multiple R-squared:  0.5579172, Adjusted R-squared:  0.5526987 
F-statistic: 106.9111 on 42 and 3558 DF,  p-value: < 0.000000000000000222204
```

Multiple Linear Regression with interactions

Multiple linear regression was performed on the spam email dataset. The model was first found using backward selection on the training set and starting with all the variables predicting the spam column along with all the interactions between the variables. After the backward selection completed using AIC was finished the model was then applied to the test set to get a prediction line. The AIC selection took a very long time run making this method inefficient since it has to process so many variables. The predictions were then measured against the decision column to see how close the prediction came. It was found that the results were not simply 0 and 1 meaning that linear regression was not a suitable analysis for the spam classification. To try and get some of the results it was decided that rounding the answers would do even though we broke assumptions to model since a continuous number was not needed. With rounding it was found that there was a 23% misclassification rate which with all the faults considered was not a bad result. The results showed a lot more values besides 0 and 1 meaning a lot of the results were completely garbage and nowhere close to the spam classification. It was found that linear regression in any sense is not good for what we were doing. The summary was too long for the R output to generate so a picture is unavailable.

Conclusion

After conducting the above classification methods, the lowest misclassification rate was attributed to Random Forests and Bagging, Artificial Neural Networks and Logistic regressions. Bagging methodology is just a variation of Random Forest methodology and therefore, close results in the two methods can be anticipated. Artificial neural networks surprisingly does not outperform Random Forests but such exceptions do occur sometimes. On average, Random Forests has shown that exclamation marks, “random”, long frequency of capitalised letters and dollar signs are the best indicators of spam emails.

References

"UCI Machine Learning Repository: Spambase Data Set." UCI Machine Learning Repository: Spambase Data Set. Web. 06 Apr. 2016. <http://archive.ics.uci.edu/ml/datasets/Spambase>.