Detecting Spam

Written by: Daniel Jouran, Ryan Smith, Morgan Wood, & Shiyunyang Zhao

For STOR 565 Final Project

06 December, 2023

Summary

This report compares the performance of various classification methods for detecting spam emails. Using data provided by the TidyTuesday project on the relative frequency of six different words or characters, we

1. provide predictive models that classify emails as either spam or non-spam with over 85% accuracy while only using six descriptive statistics of each email,
2. provide evidence that tree-based methods outperform many other machine learning models in prediction accuracy, and
3. provide insight into the importance of various email statistics in classification.

Statement on the use of AI tools

Our group did not use any AI tools during the creation of this report. This includes all categories listed in the "Project-23-updated.pdf" file.

Statement on Member Contribution

All group members contributed equally to the creation of this report. We will each fill out the Peer Evaluation Survey.

1 Overview

Within this report we aim to provide evidence that the classification of spam emails is possible with very few statistics on each email. To do this, we compare the performance of various machine learning models. Within this report we consider the following models,

- Naïve Bayes,
- Logistic Regression,
- Linear and Quadratic Determinate Analysis (LDA & QDA),
- K-Nearest Neighbors (KNN),
- Linear and Non-Linear Support Vector Machines (SVM), and
- Multiple Tree-Based Methods.

Regarding to the above machine learning algorithms, Naïve Bayes and Logistic Regression allows us to efficiently handle the large dataset and providing interpretability. LDA & QDA helps us understand the complex patterns in emails. KNN is used for its simplicity in similarity-based classification. SVMs are advantageous for achieving high accuracy with a clear margin of separation to categorize emails into "spam" or "non-spam". Finally, tree-based methods such as random forest models offer insights into variable importance.

To classify emails as either spam or non-spam, we consider a dataset provided by the TidyTuesday Project[1]. We will refer to this dataset as the Spam Dataset for the remainder of this report. The Spam Dataset contains the classification of 4601 emails as either spam or non-spam, and provides information on each email in the form of 6 variables which describe the relative frequency of certain words or characters. We describe these variables in detail Section 2. Of the 4601 emails, 1813 are spam observations and 2788 are non-spam observations resulting in an approximately 40:60 split.

The Spam Dataset is a subset of the Spam E-mail Database[2] distributed by R and collected by Hewlett-Packard Labs. In contrast to the Spam Dataset, the Spam E-mail Database contains 57 variables on the same 4601 emails. The Spam Dataset uses 6 variables on the total frequency of various words and characters along with 2 variables related to word and character length from the Spam E-mail Database to instead only display 6 variables on relative frequency.

To compare the performance of each model, our dataset is split approximately 1:1 into a training and test dataset. Each model is trained on 2300 observations, and then performance on the classification of the remaining 2301 test observations is reported.

Our code will be send along with this pdf document.

2 Variables

To distinguish between spam and non-spam emails, we utilized six key variables:

crl.tot (Total Capitalization)

Definition: The cumulative length of words in all capital letters within a given email.

Significance: Reflects the extent of capitalization, which is often a characteristic in spam emails.

dollar (Dollar Sign Occurrences)

Definition: The frequency of the "$" character expressed as a percentage of the total characters in a given email.

Significance: Indicates the prevalence of monetary symbols, commonly found in spam emails promoting financial scheme or other related content.

bang (Exclamation Mark Occurrences)

Definition: The frequency of the "!" character expressed as a percentage of the total characters in a given email.

Significance: Highlights the usage of exclamation marks, a common feature in attention-grabbing spam content.

money (Occurrences of the Word "Money")

Definition: The frequency of the word "money" expressed as a percentage of the total words in a given email.

Significance: Identifies instances where the word "money" is prominently featured, often associated with spam content.

n000 (Occurrences of the String "000")

Definition: The frequency of the string "000" expressed as a percentage of the total words in a given email.

Significance: Captures patterns related to numerical sequences, a characteristic often exploited in spam messages.

make (Occurrences of the Word "Make")

Definition: The frequency of the word "make" expressed as a percentage of the total words in a given email.

Significance: Highlights instances where the word "make" is overrepresented, a potential indicator of spam language.

Final Thoughts These variables collectively serve as effective predictors for discerning the likelihood of an email being spam. An examination of these features in multiple spam emails consistently reveals patterns reflected in the correlation matrix below in Table 1.

crl.tot	dollar	bang	money	n000	make
1.00	0.20	0.04	0.08	0.17	0.09
0.20	1.00	0.14	0.10	0.31	0.12
0.04	0.14	1.00	0.05	0.07	0.06

crl.tot	dollar	bang	money	n000	make
0.08	0.10	0.05	1.00	0.05	0.19
0.17	0.31	0.07	0.05	1.00	0.13
0.09	0.12	0.06	0.19	0.13	1.00

Table 1. Correlation matrix of all predictive variables used in our models.

3 Logistic Regression

We began our analysis by employing a logistic regression model to discern patterns indicative of spam, using all predictors available in our dataset. According to the summary output of this logistic model as shown in Table 2, it is evident that most of the predictor variables, including total length of words in capital letter, frequencies of dollar symbol `$` , bang symbol `!` , the word `money` , and string `000` , in the model are statistically significant, as indicated by their small p-values. This significance suggests that these variables play a crucial role in determining whether an email is spam. However, a notable exception is the `make` predictor which indicates the frequency of the word `make` . Its large p-value indicates that this predictor does not contribute significantly to the model in differentiating spam from non-spam emails.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-1.7347968	0.0772085	-22.4689847	0.0000000
crl.tot	0.0006256	0.0001406	4.4493738	0.0000086
dollar	9.0920525	0.9571517	9.4990711	0.0000000
bang	1.4929752	0.1716373	8.6984328	0.0000000
money	3.2710542	0.4438901	7.3690627	0.0000000
n000	4.2149209	0.6024123	6.9967380	0.0000000
make	-0.0094725	0.2071938	-0.0457183	0.9635348

Table 2. Table displaying the estimates in the logistic model using all predictors in the dataset.

In order to enhance our logistic model performance, we then employed a stepwise selection procedure. This method allowed us to identify and retain the most impactful features in the model. The stepwise selection process resulted in an optimal model, where the predictor `make` was excluded. As shown in Table 3, all remaining predictors were statistically significant.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-1.7354429	0.0759169	-22.859783	0.0e+00
crl.tot	0.0006257	0.0001406	4.450363	8.6e-06
dollar	9.0917402	0.9571411	9.498850	0.0e+00
bang	1.4928481	0.1716216	8.698487	0.0e+00
money	3.2692796	0.4421720	7.393683	0.0e+00
n000	4.2138049	0.6019007	7.000830	0.0e+00

Table 3. Table displaying the estimates in the optimized logistic model after a stepwise selection process.

To evaluate the performance of our optimized logistic model, we applied it to the test dataset. This involved generating predictions and comparing them against actual outcomes to assess the model's accuracy and error rates. As shown in Table 4, the optimized logistic model achieved an accuracy of 82.96%. The True Positive Rate reflects that the model correctly identify

65.78% spam emails, and the True Negative Rate of 94.06% is notably high, demonstrating the model's proficiency in correctly identifying non-spam emails.

Measure	Score
Test Accuracy	0.8296393
Correct Spam	0.6578073
Correct Non-Spam	0.9406295

Table 4. Table displaying the test accuracy, true positive rate and true negative rate of the optimized logistic model.

To further assess the performance of our optimized logistic regression model, we generated a ROC curve. The ROC curve for our optimized logistic model, as shown in Figure 1, is observed to be well above the diagonal line which represents the performance of a completely random classifier, characterized by an AUC of 0.5. Additionally, our model's ROC curve shows an AUC substantially higher than 0.5, which is an indicator of good predictive power.

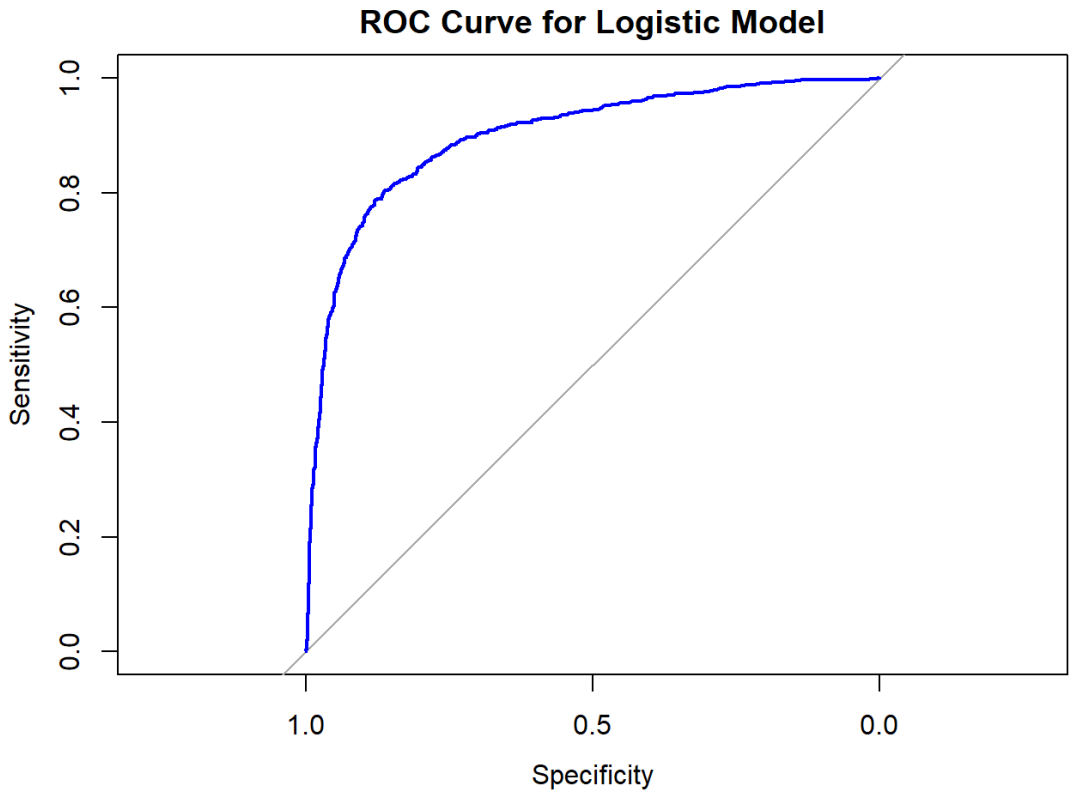


Figure 1. The ROC curve outperforms the diagonal representing a random classifier, indicating the logistic model's effective discrimination between spam and non-spam emails.

4 Naïve Bayes

In this section, we extend our analysis to include a Naïve Bayes classifier. According to the Naïve Bayes results as shown in Table 5, spam emails typically feature longer sequences of capital letters, more frequent mentions of money-related terms like dollar sign `$` and word `money`, and larger numerical figures, all consistent with common spam characteristics. The word `make` does not significantly differentiate between spam and non-spam, suggesting it's not a reliable feature for spam detection in this context, which is the same result as the logistic model suggested in the previous section.

Feature	Class	Mean	SD
crl.tot	n	163.6245000	353.9725000
crl.tot	y	492.6692000	903.5050000

Feature	Class	Mean	SD
dollar	n	0.0097777	0.0513450
dollar	y	0.1754527	0.3352918
bang	n	0.1023986	0.9358777
bang	y	0.4972802	0.7548440
money	n	0.0118058	0.1115096
money	y	0.2046813	0.4713259
n000	n	0.0075971	0.0780494
n000	y	0.2600220	0.5294182
make	n	0.0740432	0.3064346
make	y	0.1603187	0.2995090

Table 5. Table displaying the conditional probabilities of the Naïve Bayes model.

Based on the conclusion from the first Naïve Bayes regression, we decided to discard the predictor `make` which is not a impactful feature to detect spam emails. We then fit a new Naïve Bayes model excluding the `make` predictor and then test this new model on the test dataset. As shown in Table 6, our new Naïve Bayes demonstrated a test accuracy of 77.3% which is slightly lower than that of logistic model. However, its True Negative Rate of 95.4% highlights a strong ability to accurately recognize non-spam emails, while its True Positive Rate of 49.3% suggests moderate success in correctly identifying spam.

Measure	Score
Test Accuracy	0.7731421
Correct Spam	0.4928018
Correct Non-Spam	0.9542203

Table 6. Table displaying the test accuracy, true positive rate and true negative rate of the Naïve Bayes model.

We then generated the ROC curve for Naïve Bayes model, as shown in Figure 3. The curve is observed to be well above the diagonal line, and it shows an AUC higher than 0.5, which is an indicator of good predictive power.

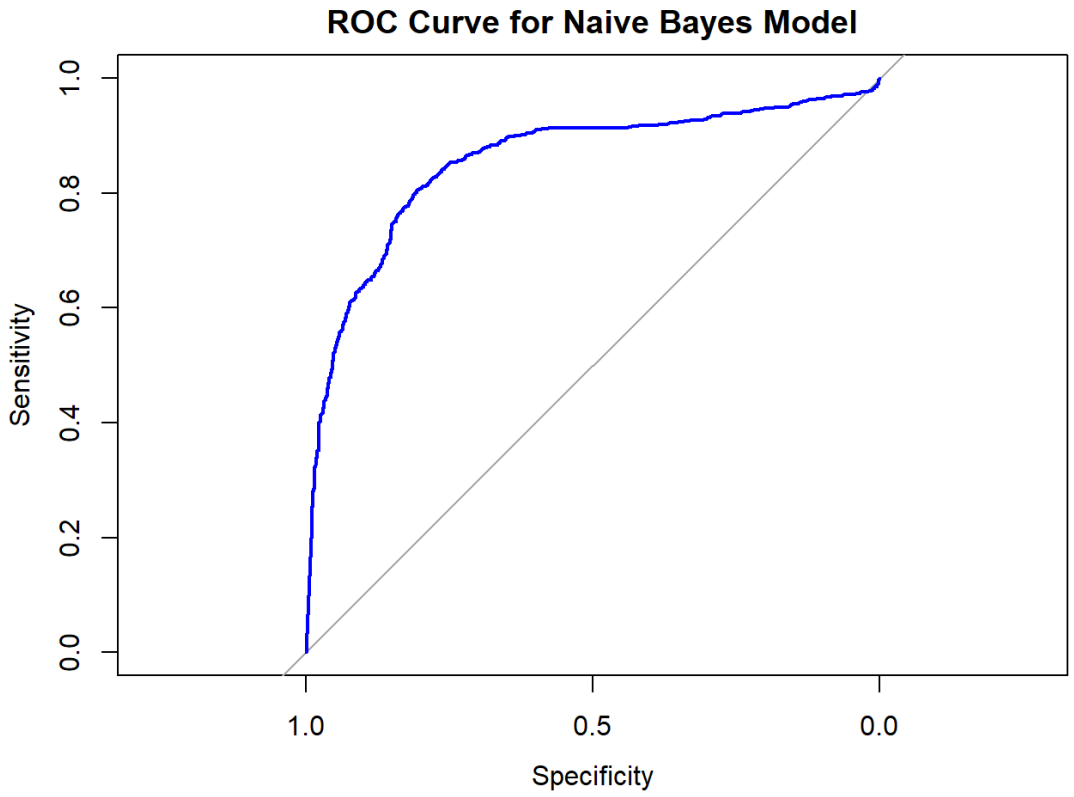


Figure 3. The ROC curve for Naïve Bayes model outperforms the diagonal representing a random classifier.

5 Linear Discriminant Analysis and Quadratic Discriminant Analysis

The next method we used was Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA). These processes, while being methods of classification for data analysis, also focus on dimensional reduction. LDA focuses more on the assumption of a common covariance matrix in the data whereas QDA can be more flexible and support a wider variety of datasets.

To ensure the assumptions of both LDA and QDA were met within the data, we scaled each predictor to have means of 0 and standard deviations of 1. Comparing the LDA's and QDA's of original and scaled data yielded the exact same results, thus we can assume that there is a common covariance matrix among the predictors. Below we performed both LDA and QDA and found Correct Spam, Correct Non-Spam, and Overall Accuracy Rates.

Measure	LDA_Score	QDA_Score
Test Accuracy	0.7570621	0.7753151
Correct Spam	0.9037559	0.8697318
Correct Non-Spam	0.7237333	0.7476110

Table 7. Table displaying the test accuracy, true positive rate and true negative rate of the LDA and QDA models.

Looking at the test dataset, QDA has the higher overall accuracy along with the higher rate of emails correctly identified as spam, but LDA had a higher rate of emails correctly identified as non-spam.

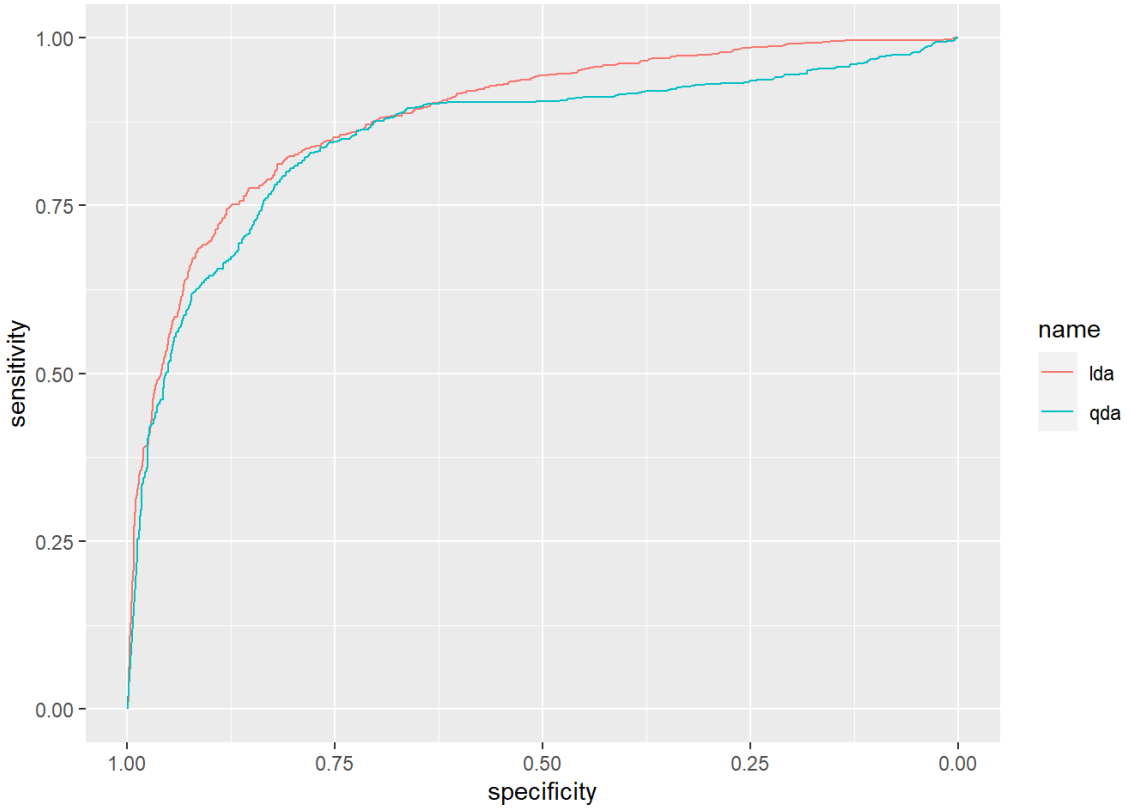


Figure 4. The ROC curve for LDA and QDA models are shown to both be curves above

```
##
## Area Under the Curve for LDA:  0.8853234
```

```
##
## Area Under the Curve for QDA:  0.8539869
```

Looking at the combined plot of ROC curves for the LDA and QDA methods, as shown in **Figure 4**, the curves are well above the diagonal random-classifier line. Below the graphic, we can see AUC values far above 0.5 showing that both curves are indicators of good predictive power. The LDA is shown in the graph to be better in predictions, but the QDA has higher accuracy in this specific dataset, specifically in the test data which shows conflicting results.

6 Support Vector Machines (Non-Linear and Linear)

We now consider the predictive performance of Support Vector Machines. The following section will explore two different Support Vector Machines:

- Linear Support Vector Machines and
- Non-Linear Support Vector Machines.

The models are presented in increasing predictive performance.

Linear Support Vector Machine Like other model techniques, we begin by using the trained data to build a linear support vector machine using all variables as predictors and the classification of the email being “yes” or “no” as the response. We first tune the linear support vector machine to prevent overfitting or creating too much variance. Then we employ the SVM on the test data to predict whether or not an email is spam. Finally, we represented the performance of our linear support vector machine in the contingency table below in Table 8.

	Truth_nonspam	Truth_spam
Classified_nonspam	1321	314
Classified_spam	77	589

Table 8. Contingency table displaying the predicted versus true classification of emails in the test set using our Linear Support Vector Machine.

Our linear support vector machine when predicting whether or not an email is spam generally performed well with an overall accuracy of 83%. Like our other models, the linear SVM performed very well in distinguishing emails that are not spam, producing a true-negativity rate of 94.49%. However, the linear SVM model struggled more with detecting spam emails, producing a true-positive rate of 65.23%. In most of our models mentioned in this paper, the distinguishing factor would be its ability to avoid type-1 errors. Although our true-positive rate is not up to par with our true-negativity rate, it is worth noting that it performed considerably well against other models mentioned in this paper.

Non-Linear Support Vector Machine Similar to our linear support vector machine, we begin with using the training data to build our non-linear support vector machine. We used the same predictors and response variables as the linear SVM. Identically, we first tune our Non-Linear SVM to avoid over-fitting or high variance. Then, we used our Non-Linear SVM model to predict whether or not a given email is spam. Finally, we represented the performance of our non-linear support vector machine in the contingency table below in Table 9.

	Truth_nonspam	Truth_spam
Classified_nonspam	1300	231
Classified_spam	98	672

Table 9. Contingency table displaying the predicted versus true classification of emails in the test set using our Non-Linear Support Vector Machine.

Our non-linear support vector machine performed better than our linear SVM in predicting whether or not an email is spam or not. Generally, the non-linear SVM had an accuracy of 85.7% as opposed to 83% of the linear SVM. An interesting observation was that the non-linear SVM particularly improved on the ability of the linear SVM to avoid type-1 error or in other words the true-positive rate. In fact, the non-linear SVM true-positive rate is 74.42% which is a 7.19% improvement of the linear SVM. However, the non-linear SVM declined in the true-negativity rate by 1.5% as opposed to the linear SVM.

Final Thoughts The non-linear SVM would be a more valuable predictive model than our linear SVM. Although the general accuracy improved by over 2.7%, ultimately the most valuable aspect of the non-linear SVM was its much better ability to predict email spam when it is spam. This metric is particularly important due to this being the most challenging part to predict across all of our models discussed so far.

7 K-Nearest Neighbors Method

Like other model techniques, we begin by using the trained data to build a K-Nearest Neighbors model using all variables as predictors and the classification of the email as “yes” or “no” as the response. We first chose a K value of 10 since this value performed the best under cross validation. Then we employ our KNN model on the test data to predict whether or not an email is spam. Finally, we represented the performance of our KNN model’s performance in the contingency table below in Table 10.

	Truth_nonspam	Truth_spam
Classified_nonspam	1179	405

	Truth_nonspam	Truth_spam
Classified_spam	219	498

Table 10. Contingency table displaying the predicted versus true classification of emails in the test set using our K-Nearest Neighbor Model.

Our K-Nearest Neighbor model did not perform as well as we had hoped. Other models discussed so far performed much better in almost all metrics including true-positive, true-negativity, and general accuracy rates. Our general accuracy of 72.75%. Our KNN model struggled to detect spam emails with a rate a true-positive rate of 55.15%. However, the KNN model did very poorly compared to all other models at predicting non-spam. Our KNN model performed the worst out of all models with a true-negativity rate of 84.12%. Our true-negativity performance was surprising since all models were efficient when predicting non-spam emails. Thus, we would not use our KNN model when predicting spam emails.

8 Tree-Based Methods

We now consider the predictive performance of tree-based methods. The following section will look at

- Classification Trees,
- Boosting Trees,
- Bagging Trees, and
- Random Forest Models.

The models are presented in increasing prediction performance.

Classification Trees We begin by training a pruned classification tree as displayed in Figure 4 where 'y' stands for a spam classification and 'n' stands for a non-spam classification. We find that the variables corresponding to the length of all-capital strings, the frequency of the exclamation mark "!", and the frequency of the dollar symbol "$" are important classifiers, with the frequency of the dollar symbol being the most important. Just as intuition would suggest, for each of these variables, larger values suggest that an email should be classified as spam.

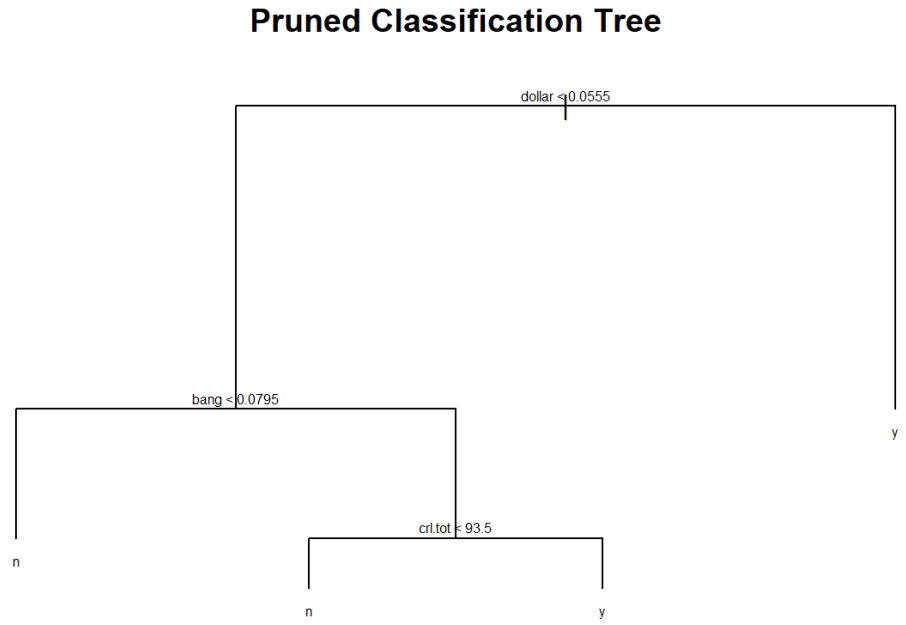


Figure 5. Pruned classification tree which uses the variables corresponding to the length of all-capital strings, the frequency of the exclamation mark "!", and the frequency of the dollar symbol "$". Here 'y' stands for classifying an email as spam and 'n' stands for classifying an email as non-spam.

For the classification tree displayed above, the tree was grown using deviance and then pruned using misclassification. The performance of the classification tree in Figure 5 is summarized in Table 11. In total, 84.2% of the test observations were correctly classified. Of the spam observations, 74.7% were correctly classified as spam. Of the non-spam observations, 90.4% were correctly classified as non-spam.

It is worth pointing out that the accuracy of this model is near 85% while also being interpretable and, arguably, very simple.

	Truth_nonspam	Truth_spam
Classified_nonspam	1264	229
Classified_spam	134	674

Table 11. Contingency table displaying the predicted verses true classification of emails in the test set using the pruned classification tree in Figure 1.

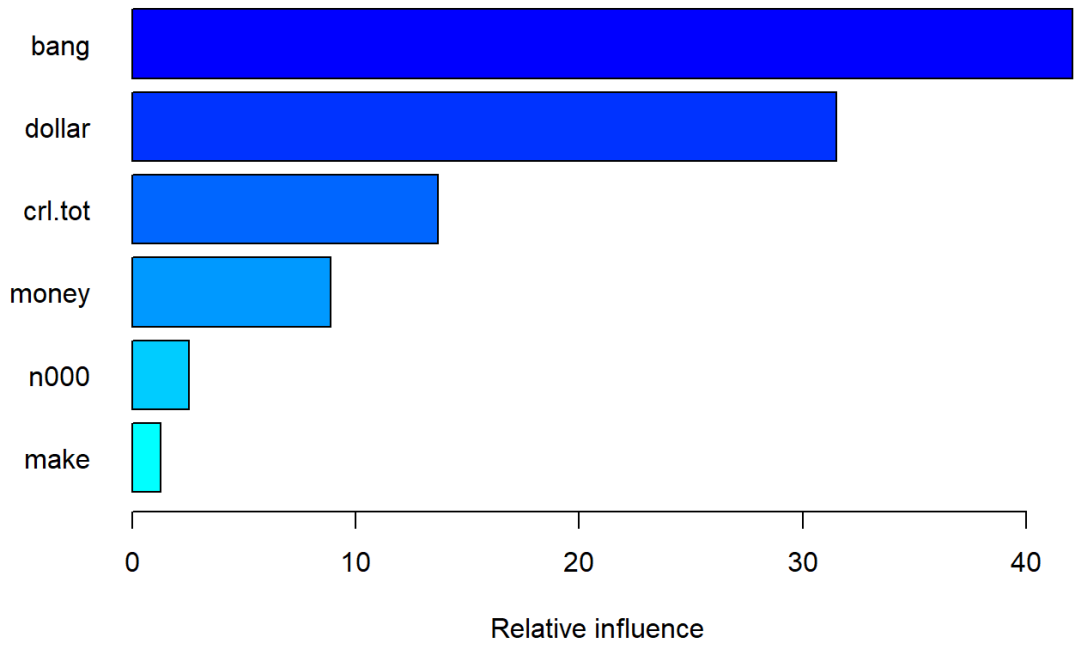
Boosting Trees We next consider boosting trees. Multiple shrinkage values were considered and in each model 1000 stumps were used (i.e., 1000 trees of interaction level 1). We present results for a shrinkage value of 0.06 which was chosen using cross validation.

For a boosted tree with shrinkage value 0.06, we first look at the performance of this model which is displayed in Table 12. In total, 86.4% of the test observations were correctly classified. Of the spam observations, 78.1% were correctly classified as spam. Of the non-spam observations, 91.8% were correctly classified as non-spam.

	Truth_nonspam	Truth_spam
Classified_nonspam	1278	193
Classified_spam	120	710

Table 12. Contingency table displaying the predicted verses true classification of emails in the test set using a boosting tree trained using shrinkage value 0.06 and 1000 stumps.

We next consider the relative influence of variables. This is displayed below in Figure 6. We find similarly to our pruned tree analysis that the variables corresponding to the length of all-capital strings, the frequency of the exclamation mark "!", and the frequency of the dollar symbol "$" are important classifiers. However, our boosting tree suggests that the frequency of the exclamation mark is the most important variable in classification.




```
##             var   rel.inf
## bang       bang 42.073856
## dollar   dollar 31.527925
## crl.tot crl.tot 13.677295
## money     money  8.883394
## n000       n000  2.546210
## make       make  1.291321
```

Figure 6. Figure displaying the relative influence of different predictive variables on classification listed from most (top) to least (bottom) influential.

Bagging Trees We next consider bagging trees. We turn to a more general version of random forest following this analysis.

To train our bagging tree model, we used 1000 trees. The performance is summarized below in Table 13. In total, 86.4% of the test observations were correctly classified. Of the spam observations, 81.2% were correctly classified as spam. Of the non-spam observations, 89.8% were correctly classified as non-spam.

	Truth_nonspam	Truth_spam
Classified_nonspam	1256	170
Classified_spam	142	733

Table 13. Contingency table displaying the predicted verses true classification of emails in the test set using a bagging tree trained using 1000 trees.

Using the bagging tree trained, we also looked at the influence of each of the respective variables. We suppress the numerics here as the results agree with the order of influence described in Figure 7.

Random Forest We next consider training random forest models with either two or three predictors in 1000 trees. Note that we have a total of six predictor variables. We use the rule of thumb that the square root of the number of predictor variables is a good number of predictors to use in each tree. The square root of six is approximately 2.5.

For this report, we give results based on the random forest model trained with two predictors as it slightly outperforms the model trained with three predictors which resulted in a total classification accuracy of 87.4%.

In Table 14, the performance of the random forest model trained with two predictors in each tree is displayed. In total, 88.0% of the test observations were correctly classified. Of the spam observations, 78.5% were correctly classified as spam. Of the non-spam observations, 94.1% were correctly classified as non-spam.

	Truth_nonspam	Truth_spam
Classified_nonspam	1315	194
Classified_spam	83	709

Table 14. Contingency table displaying the predicted verses true classification of emails in the test set using a random forest model trained using 1000 trees with two predictors each.

Using the random forest trained, we again looked at the influence of each of the respective variables. As before, we suppress the numerics as the results agree with the order of influence described in Figure 8.

9 Summary of Results

Method	Correct_Spam	Correct_Non_Spam	Overall_Correct
Random Forest	78.52	94.06	87.96
Bagging	81.17	89.84	86.44
Boosting	78.63	91.42	86.40
SVM (Non-Linear)	74.42	92.99	85.70
Classification Tree	74.64	90.41	84.22
SVM (Linear)	65.23	94.49	83.00
Logistic Regression	65.78	94.06	82.96
QDA	50.28	95.14	77.53
Naïve Bayes Regression	49.28	95.42	77.31
LDA	42.64	97.07	75.71
K-Nearest Neighbors	55.15	84.12	72.75

Figure 7. The table above shows Correct Spam (True Positive), Correct Non-Spam (True Negative), and Overall Accuracy rates for each method used

Shown in Figure 7, we can see each method's accuracy rates shown for predictions of spam vs non-spam emails. Overall, most methods yielded high accuracy as even the least accurate predictive method, K-Nearest Neighbors, still had 72.8% of emails correctly identified. The most accurate, based on these predictions, was Random Forest regression, with 88% correct, and was followed by Bagging and Boosting, each with 86.4% correct. This shows that tree-based methods generally performed the best in predicting email content.

We can also see that across each method, non-spam accuracy is far higher than spam accuracy which may represent places for future work. Non-spam accuracy is, on average, about 27% higher than spam accuracy meaning there is still exists a problem with Type I errors that could be improved in future work, as long as it does not compromise the low rate of Type II errors.

1. https://github.com/rfordatascience/tidytuesday/tree/979c7204bb80fd3a00ca1b622de7ebd0f49766bf/data/2023/2023-08-15 (https://github.com/rfordatascience/tidytuesday/tree/979c7204bb80fd3a00ca1b622de7ebd0f49766bf/data/2023/2023-08-15)↩
2. https://search.r-project.org/CRAN/refmans/kernlab/html/spam.html (https://search.r-project.org/CRAN/refmans/kernlab/html/spam.html)↩