**MIST7770 Final Project "Spambase"**
**Group 3: Ryan Cullen, Judd Douglas, Matthew Karnatz, Collin Ladina, Shaan Patel**

## Context and Question

Spam emails have become a consistent burden on the inboxes of most users. In 2023, some estimates reported that 160 billion up to 320 billion spam emails were sent each day, and "94% of malware is delivered via this medium" (Forbes, EmailTester). While many spam emails do not contain malicious content or even have malicious intent, their overwhelming volume creates clutter, reduces productivity, and increases the risk of falling victim to phishing or other cyberattacks. The rapid proliferation of these spam emails calls for effective and efficient solutions to improve email security, enhance user experiences, and increase workplace productivity.

Spam emails affect a wide range of industries, presenting a significant challenge for effective detection. Key industries impacted include technology, telecommunications, and virtually any enterprise, regardless of size. The technology sector includes companies like Google, Microsoft, and Yahoo, which manage billions of email accounts and rely on advanced, constantly updated spam filters to protect their users. The telecom industry has major companies like AT&T, Verizon, T-Mobile, etc. that manage the network infrastructure for the emails that are being sent. Enterprises of all sizes are vulnerable—smaller companies may lack the robust detection systems of their larger counterparts, while larger organizations often face a higher volume of spam with potentially malicious intent. From banks and energy providers to government agencies, any industry plays a role in each person's daily life so any interruption has the chance to disrupt the processes of the world.

Spam emails affect a variety of actors as well. End users like ourselves, face the impact of inbox clutter and security risks, while IT professionals work to implement and maintain effective spam detection systems so spam can be minimized. Data scientists and engineers help develop the algorithms and models to improve detection accuracy. Cybersecurity professionals design strategies to mitigate risks such as phishing and malware attacks that could be targeted through spam emails. Governmental bodies enact policies/legislation that help regulate and control spam content.

The primary objective of this project is to accurately predict and classify incoming emails as 'spam' or 'not spam' while limiting the prevalence of both Type I and Type II errors. Our problem statement is: *Can we accurately classify an email as spam or non-spam using the features provided in the dataset?* This is crucial for improving email security and reducing the clutter in inboxes. A successful model would reduce the clutter in inboxes, improve email organization, and mitigate the delivery of malware and phishing attempts.

## Data and Variables

Our data source, Spambase, comes from the UC Irvine Machine Learning Repository. It consists of 4601 observations (emails) with 57 numeric features that capture the word, character, symbol, and case frequencies. The categorical, target variable is binary, where 0 is non-spam and 1 is spam. This dataset will adequately address the problem of spam since each observation has numerous features that help classify spam or not spam. There were no missing

values in the dataset, but if there were, the missing numeric data would have been replaced with the median value of the feature involved (next paragraph explains why the median).

The R code focuses on handling outliers in addition to providing the descriptive statistics for the numeric features (all X's) including the mean, standard deviation, median, minimum, maximum, and range (max - min); it also provides the relative frequency table for the categorical target variable (is_spam). Outliers were handled with the Interquartile Range (IQR) method. The IQR is the middle 50% of the data (between 25th percentile and 75th percentile). The lower bound is the 25th percentile - (1.5 * IQR). The upper bound is the 75th percentile + (1.5 * IQR). If a value is below the lower bound or above the upper bound, then this value is considered an outlier. Any outlier was replaced with the median of the column involved. We do this because the median value is particularly robust to outliers and is more representative of the data. This also helps preserve the distribution of our data, whereas using the mean could skew the distribution. This will help ensure that all the data is handled, remains consistent, and does not have extreme values that could skew results while still maximizing the prediction coverage.

## Descriptive Visualizations

The purpose of using descriptive visualizations is to help illustrate the distribution of the variables, as well as the associations they have with each other. All input features were included when running the model to help maximize prediction coverage, but BigML only included features that were important and useful in predictions. Descriptive statistics of the input features (X's) including, but not limited to, the mean, standard deviation, median, minimum, maximum, and range (max - min) can be seen in the image below (screenshotted from the knitted file from R in HTML format). These statistics were calculated using the describe method from the psych package in R.

```
##                               mean     sd median min      max    range
## word_freq_all                 0.28   0.50   0.00   0     5.10     5.10
## word_freq_our                 0.31   0.67   0.00   0    10.00    10.00
## word_freq_will                0.54   0.86   0.10   0     9.67     9.67
## word_freq_free                0.25   0.83   0.00   0    20.00    20.00
## word_freq_you                 1.66   1.78   1.31   0    18.75    18.75
## word_freq_your                0.81   1.20   0.22   0    11.11    11.11
## word_freq_re                  0.30   1.01   0.00   0    21.42    21.42
## char_freq_(                   0.14   0.27   0.06   0     9.75     9.75
## char_freq_!                   0.27   0.82   0.00   0    32.48    32.48
## char_freq_$                   0.08   0.25   0.00   0     6.00     6.00
## capital_run_length_average    5.19  31.73   2.28   1  1102.50  1101.50
## capital_run_length_longest   52.17 194.89  15.00   1  9989.00  9988.00
## capital_run_length_total    283.29 606.35  95.00   1 15841.00 15840.00
```

```
        0         1
0.6059552 0.3940448
```

Above is a relative frequency table of the categorical (specifically binary) target variable, is_spam. The table illustrates that about 60% of the dataset is not spam, and about 40% of the dataset is spam, signifying a relatively balanced dataset (not too many spams or too many non-spams).

Since certain words, phrases, symbols, or mannerisms (case) are often used together, we can visualize their relationships to help understand the model prediction power. For example, an all caps message designed to grab
users' attention may be convoluted with dollar signs, exclamation marks, or the word “free”. This relationship is visualized by the association webs above, further clarified when removing some extraneous variables by increasing the leverage in the BigML software.

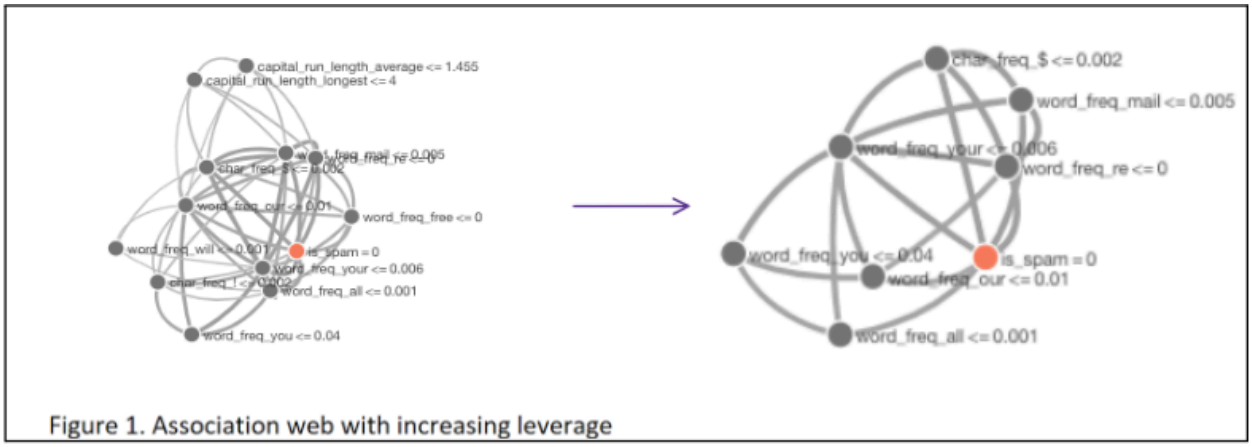


Figure 1. Association web with increasing leverage

## Method Selection

When exploring our data and deciding on what type of data analysis technique we should use to build our model, a few factors were taken into account. There are plenty of feasible methods to pursue that would be helpful in predicting the classification of an observation, given the numeric form of our data.

First, due to the binary classification goal of our project, we contemplated using logistic regression. It is a straightforward method that gives us an easily interpretable output where we can see the individual contribution of each feature. It also works well with text data.

The second method we debated using was a random forest. This ensemble approach is a powerful predictor when analyzing non-linear relationships like frequencies. A random forest would help incorporate patterns found within these spam emails to correctly identify spam.

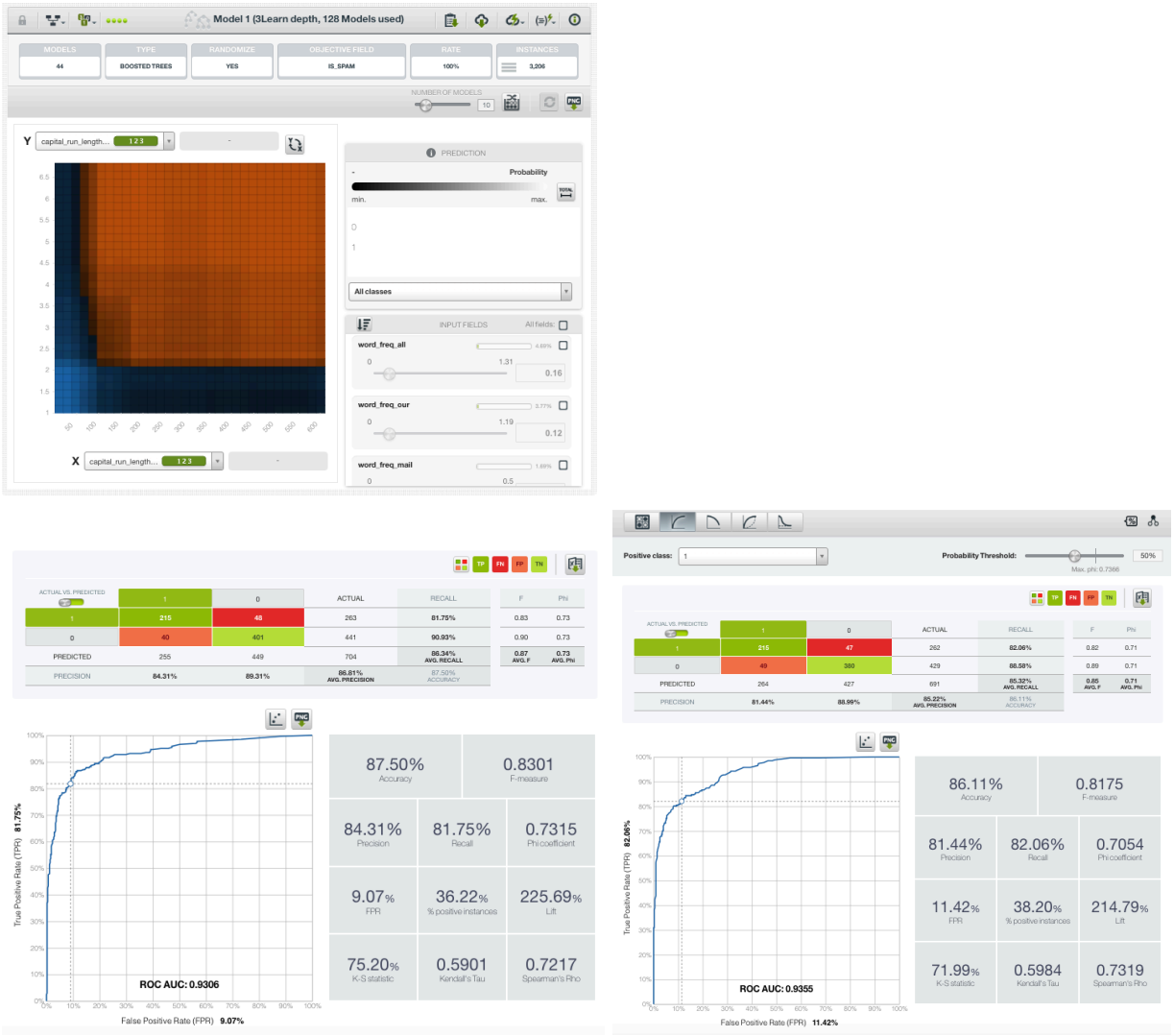
Finally, we looked at using a gradient boosted tree model. Boosting is a viable approach to answering our problem statement since the iterative approach it takes translates to higher performance by implementing error correction. But most importantly, it also serves to limit the Type I errors (false positives), which we emphasize as an important function of the model. This is because when weighing what would be worse in a business situation, dealing with some spam emails that evaded capture versus missing an important email that was wrongly convicted as spam, it would be much more detrimental to miss a real one. When placing significance on reducing type I errors, we are focusing primarily on the model’s performance in terms of

precision. Although our approach emphasizes precision, we still want to balance the model's accuracy and recall in order for it to be well-rounded. For these reasons, we ultimately decided that the boosted model was the best fit in this scenario. This choice was further reinforced as it outperformed its competitor models after running each technique in BigML and making adjustments.
To begin the Gradient boosting method, we split the Spambase data into a training, testing, and valuation set with the proportions of 70:15:15 respectively. This is very useful to measure the performance of the model in an unbiased manner. Other tweaks and adjustments, like altering the learning rate and number of nodes used, were performed and described later. When trying to avoid leakage, we were vigilant in making alterations to only our training data to keep the sets exclusive to each other so the performance is not inflated. Additionally, we made changes to our model to prevent overfitting by reducing the number of boosting iterations, since too many trees may lead to overfitting.

## Results

After our initial trial of different models and discussing the results, it was determined that a 70-15-15 split between training, validation, and testing datasets would suffice for our models. Using this new split we remade our models from the first iteration and still found that a gradient boosted model was the best model for predicting spam based on our data. The decision forest model maxed out at an accuracy of around 82% on the validation data and 83% on the test data. The least accurate boosted tree model that we made bottomed out at 84% and the most accurate model had an 86% accuracy on the validation data and an 87% on the test data.
We were also able to balance out our recall and precision to a more reasonable degree and had both of them over 80%. This positive response to both the validation data for tuning and the unseen test data for experimentation is an encouraging outcome for our model and indicates that we have improved our accuracy since our first attempt without overfitting the model to our data.

Using the BigML software and the automatic optimization function, we tested multiple ensemble models with different combinations of training duration and ensemble candidates. The software automatically tests the different hyperparameters and outputs the best model found given the training time provided and the number of ensemble candidates to choose from. This means that a longer training time and greater number of ensemble candidates will provide a better model to predict the training data, but may overfit the data. We found that a training time of three and setting the number of ensemble candidates at 128 provided the best accuracy between both the test and validation sets.




## Limitations

Although our model is robust and performs well, it is not without limitations. One significant challenge is that our model's predictive power may be limited when handling emails in languages other than English, as the dataset and feature engineering primarily focus on English-language patterns and keywords. Additionally, the model currently treats each email as originating from a unique sender without considering the sender's history or reputation. Incorporating a database of known spam offenders or suspicious IP addresses could enhance our ability to identify spam from repeat sources.

Additionally, our model is designed as a general-purpose solution rather than being tailored to the specific needs of individual businesses or industries. Creating specialized models for different organizations, with features adjusted to capture their unique spam patterns, could enhance accuracy. Another limitation is that our model does not analyze text within images,

making it vulnerable to spam messages that embed text in images or attachments to evade detection. As spamming techniques evolve, our model may also require periodic retraining and updates to stay effective. Lastly, our dataset includes 4,601 samples, which, while adequate for initial analysis, may not be large enough to capture the full variety of spam behaviors. Expanding the dataset with more recent and diverse examples would likely enhance the model's robustness and predictive capability.

## Recommendations

Along with an employee awareness program to train employees to be cognizant of potential spam threats, this model should be continuously updated with more observations. One way to do this is through the use of  certain libraries that are designed specifically for gradient boosted trees, like eXtreme Gradient Boosting (XGBoost). This optimized method uses L1 and L2 regularization to prevent overfitting and has become a popular algorithm for boosted trees with tabular data. It is also particularly good at working with large amounts of data, so we recommend the implementation of XGBoost in the future once the scale of the data set has increased.

Additionally, we recommend some basic feature engineering by integrating semantic features like Natural Language Processing (NLP) based sentiment analysis or topic modeling. Using sentiment analysis in conjunction with other tools will help further the certainty that an email is spam due to certain phrases with persuasive or negative connotations embedded in them.

## References

1. Spambase Dataset
2. Forbes: https://www.forbes.com/sites/daveywinder/2020/05/03/this-surprisingly-simple-email-trick-will-stop-spam-with-one-click/
3. Email tester: https://www.emailtooltester.com/en/blog/spam-statistics/