Business Understanding:

Email is a service that has all but replaced traditional mail in the 21st century. Unfortunately, junk mail, called spam mail in the context of email, has filled inboxes as email becomes more ubiquitous. Spam mail is significantly cheaper to send than junk mail, so there is very little stopping advertisers from sending spam emails to as many people as possible. Spam mail can range from mostly harmless advertising to phishing or malware links. The range of outcomes from clicking a spam link requires a system to properly identify spam. A human educated about spam mail can often spot spam emails at a glance, but what about a less tech savvy operator? An automated system to detect spam mail can be used to unclutter inboxes and help protect those who have trouble identifying spam mail.

In this analysis, we will create an automated system to identify spam emails using machine learning. More specifically, we will use information about the email such as the number of characters in the body, whether the email a reply email, if the email body has images, among other characteristics. Using the extracted information, we create a tree based model in order to detect if an email is spam. Since we were given the data and did not perform the data extraction ourselves, we are assuming that the data extraction was done in a robust manner. The following analysis is only as robust as the data that we were provided.

Modeling Preparations:

As stated in the Business Understanding, we will be using a tree based algorithm in order to determine whether an email is spam or not. Decision trees are fairly useful for binary classification problems. The direction an observation flows through the tree structure is determined by decision nodes. If the decision node is based on a binary variable, the observation flows right or left based on the value of the variable for the observation. If the decision node is based on a continuous variable, the observation flows right or left based on if the observation’s value is above or below a certain threshold. The tree model has a binary structure and is a non-parametric model, so the variables used in the model don’t need to follow any distribution. This makes a tree model extremely useful for binary classification problems such as this one.

In order to find the “best” decision tree, you need to test every possible tree for the model. In order to avoid testing every possible tree, we decided to leverage the random forest model. The random forest takes a number of decision trees, which each make a decision if the email is “spam” or “ham”. If enough of trees classify the email as “spam”, then the email is classified as spam. If the number of trees classifying the email as spam is below a certain threshold, then we classify the email as ham. Utilizing random forest allows us to test fewer trees in the model creation step and allows us to have a more robust model.

To build the most accurate model, we have made the decision to create a random forest. This model is made up of many decision trees, who each have a single vote as to what the data point should be classified as. In a two class problem, the data point is classified by whichever class has the majority vote. What makes these forests “random” is how variables are chosen. In our model call, we specify how many variables are randomly chosen for each tree. This allows for many different trees to be created, instead of many similar trees with the same variables.

To determine the best model, we are using a few evaluation metrics. Accuracy, which is the number of correct predictions divided by the number of total observations, is important, but it doesn’t necessarily tell the entire story. It is important to minimize the number of false positives, but a large number of false negatives is more problematic. It is a bigger problem to classify a real email as spam, rather than classifying a spam email as not spam. If a spam email gets through our filter, then a human will waste a couple seconds deleting the email from their inbox. If an email gets incorrectly classified as spam, a potentially important email may not be seen. Therefore, we are also interested in looking at sensitivity and specificity. Sensitivity answers the question “how many of the positives classified by the model are actually positive,” while specificity answers the question “how many of the negatives in this model are actually negative.” “Ham” is considered to be our positive class, so more weight will be given to sensitivity, but specificity will also be considered.

We will consider ROC-Curves and AUC to determine our best models as well. The ROC Curve plots false positive rate on the x-axis and true positive rate on the y-axis. AUC is the “area under the curve” of this ROC plot. An AUC close to 1 indicates a model with a high true positive rate with a low false positive rate, which is a desired model. F1 score will be considered as well, which is the recripocal of the mean of precision and sensitivity, where precision is number of true positives divided by the number of true positives plus the number of false positives. It is important to note that we consider “spam” to be the positive class.

If you are interested in reading more about the algorithms and accuracy measures used in this analysis, here are a few useful links:

Decision Tree Learning: https://en.wikipedia.org/wiki/Decision\_tree\_learning

Accuracy Measures: <https://en.wikipedia.org/wiki/Sensitivity_and_specificity>

Random Forest: <https://en.wikipedia.org/wiki/Random_forest>