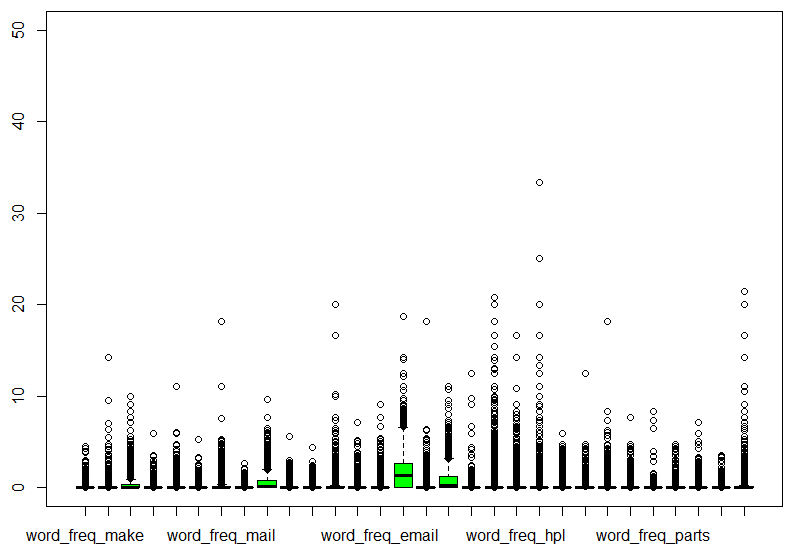
Author: Josephine Michalek 385809

## Data selection

Various attributes were removed from the data. Words with a small mean and high max were removed because that demonstrates there were lots of that particular word in one email, not in many emails so it may not necessarily be spam. All numbers were also removed as they were non-specific and if they were dollars in a spam email, any dollar value could be present in any spam email so these values would not assist with detecting spam. The six characters were also removed as they are generic characters that could be used in any spam or non-spam situation. Lastly, average, longest and total run length were discarded for simplicity. Again for simplicity, the data was not retrained to include more up to date spam terms.

## Data Preprocessing

A box plot was utilized to visualize outliers that may need to be removed for cleaning. Upon inspection, it can be determined that outliers are in an acceptable range to not need to be removed. Again, for simplicity and non-necessity, the outliers were kept and not cleaned. Please see the box plot as follows:



## Data Transformation

The dataset was normalized, not generalized. It was normalized through scaling in the numerical range (excluding the categorical spam attribute) and applied with the standard deviation with an interval of between -1 and 2 as the specified range. The data was normalized to aid in seeking attribute relations by allowing them to have a unit norm.

## Data Mining

J48 – This was used because it was a suitable one provided from the tutorials. It is the C4.5 Decision Tree algorithm implementation. It was selected because is a tree-based classification, which is needed for seeing variance in different types of techniques.

JRip – This was used because it was a suitable once provided from the tutorials. It is the Propositional Rule Learner algorithm implementation. It was selected because it is a rule-based classification, which is needed to compare to other techniques such as the tree-based J48 technique.

SVM – This was used because it was a suitable technique once again provided from the tutorials. It is the SVM implementation using Sequential Minimal Optimization. It was selected because it is a SVM-based classification technique, which would be interesting to compare to other types such as tree-based and rule-based previously listed.

IBk – IBk was used because it is a Lazy Learner for variance. It is also a k-nearest neighbor classifier and provides a fourth type of classifier to compare.

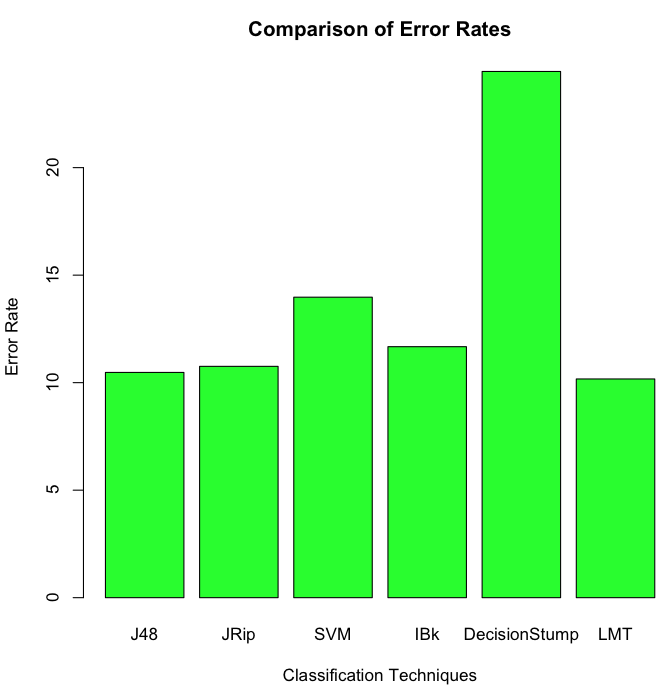
DecisionStump – DecisionStump was used because it seemed to be similar to J48 in that it was a type of tree, but different enough to want to use. Decision Stumps are trees with a single split only, and it would be interesting to compare due to it being different for typical trees and the other types of classification listed.

LMT – Logistic Model Trees are implemented in LMT. Originally this was going to be excluded because it takes a lot of CPU power and a relatively long compute time compared to the previous five classification techniques. However, it had a relatively low error rate so it was decided to keep it. While in some instances LMT provided the lowest error rate, it was still not as efficient due to processing time.

## Pattern Evaluation

As seen below, the comparison of the error rates of the six classification techniques are displayed. LMT has the best error rate, but because it has a very long compute time and uses a lot of CPU power, it is not the best one. The best technique from this data set is the J48 classifier. DecisionStump seemed to not be very effective. Additionally, all techniques were above 10% error rate, suggesting more fine-tuning on attribute selection may have made the techniques more effective.

10-fold cross validation was used instead of percentage split so we don’t have to split any training and testing datasets. Cross validation is an efficient method for data consistency, and 10-folds create a high accuracy.



## Function

The detect\_spam() function could be a lot more efficient, by passing the email in as a .txt file rather than a string, string splitting the file and using a for loop to go over each value in the row, which will automatically index features in the data set. However, a less compact method was chosen which include a lot of if else statements and factoring. While the number of lines is much greater, the function itself is still very efficient and operates as intended.