Abstract: Our goal is to find more ways of distinguishing between regular emails and spam emails. To accomplish this, we have taken a dataset of emails and plan to build off of previous work to not only develop models for determining whether an email is spam or not, but also to determine why the model makes said determination. By doing so, we hope to determine the common factors that can make spam filters more efficient. Our main methods are Classification and Trees, with Regression and Clustering being secondary methods.

Dataset: The dataset emails.csv is a table with 2 columns, one made up of the text of 5728 emails, including the subject, the other a variable that is either 1 if the described email is spam and 0 if the email is not. It was collected from Harshit Kumar's blog (Kumar, 2017) which in turn collected the data set from the academic paper *Spam filtering with naïve bayes-which naïve bayes?* (Metsis, et. al., 2006). It was previously one of the Enron-Spam datasets created for the purpose of testing a number of different Naïve Bayes spam filters, but as the links in the paper for said datasets has gone down in the interim, we cannot determine which of the six specific datasets it was before being renamed by Harshit to emails.csv. It is currently available for download on the aforementioned blog.

Related Work: The first group to analyze this dataset were the creators of said dataset, Metsis, et. al. They used the dataset, along with five datasets of similar type to test five different Naïve Bayes classifiers, coming to the conclusion that the Flexible Bayes model and the Multinomial Bayes model with Boolean attributes collectively performed the best of the five models, despite their relative lack of use in spam filters at the time (Metsis, et. al., 2006). The second to analyze the dataset was Harshit Kumar, who computed the number of times each individual word in the emails was used, filtered out those words which were least commonly used, and created variables for each of the remaining commonly used words, with a 1 if the word was found in the email and a 0 if it was not. Then, Harshit separated the generated dataset into training and test sets, then generated a number of models for determining whether or not a given email was spam or not. He created a logistic regression model, a CART model, and a random forest model with the training data, then used these models to predict the results of the test data and recorded the accuracy and AUC of each model. He came to the conclusion that the random forest model was the best performing in both measures of the generated models (Kumar, 2017).

Techniques: We used a number of techniques, to varying degrees of success as shown in the table below.

Table 1: Techniques and Results

Technique	Process	Results

Classification	Classes of authentic and spam with a vector generalized linear model	Success
Correlation	Correlation of spam variable and others to get the highest correlated variables	Inconclusive
Trees	Non pruned and pruned trees for predicting the results of the spam variable	Success
Clustering	Hierarchical clustering of the full dataset and the first five principal score vectors	Inconclusive
Regression	Classified into groups instead of classifying into the range from 0 to n	Success

The first technique used was Classification. The results of this technique are shown in the tables below.

Table 2: Classification Variables

	Estimate	Std. Error	z value	Pr(>|z|)
Intercept	1.3668669	0.1176881	11.614	< 2e-16
spellcheckErrorsSubject	0.5857643	0.0517860	11.311	< 2e-16
spellcheckErrorsContent	-0.0197768	0.0047329	-4.179	2.93e-05
containsNumber	0.5764582	0.1064801	5.414	6.17e-08
containsLink	-1.2882946	0.1342166	-9.599	< 2e-16
wordCount	0.0012379	0.0002601	4.759	1.95e-06
Count Common Word: free	-0.5009577	0.1223704	-4.094	4.24e-05

Contains Common Word: free	0.4470226	0.2168284	2.062	0.039242

Table 3: Classification Results

Classification	Authentic Emails	Spam Emails
Predicted Authentic	4159	554
Predicted Spam	201	814

As such, the accuracy of the Classification method was 86.8%, with a low false positive rate that may affect the suitability depending on the prevalence of spam in the broader environment. The second method was Correlation, where we got the following results with Pearson and Spearman correlation.

Table 4: Pearson Correlation Results

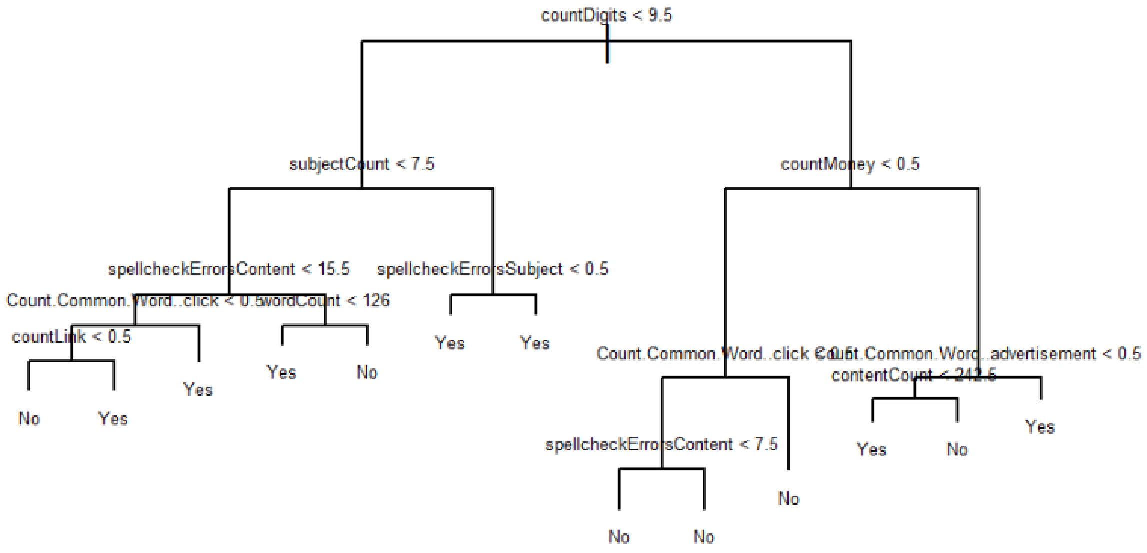
Contains Common Word: click	Subject Count	Count of Common Word: click	Contains Money	Content Count
0.3174	0.2610	0.2607	0.2576	-0.2287

Table 5: Spearman Correlation Results

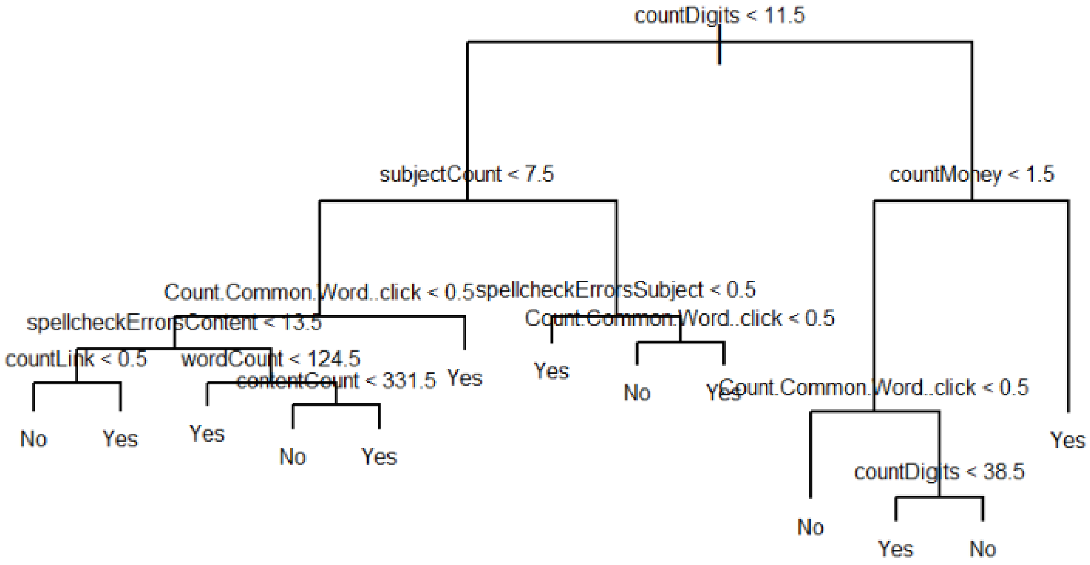
Count of Common Word: click	Contains Common Word: click	Count of Money	Count of Numbers	Contains Money
0.3185	0.3174	0.2636	-0.2581	0.2576

Given that none of the correlations was above .5, no strong correlations were found, rendering the results inconclusive. The next method was Trees, where pruned and non-pruned returned an accuracy of 84.6.

Graph 1: Non-Pruned Tree

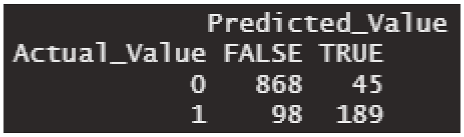

countDigits < 9.5
subjectCount < 7.5
countMoney < 0.5
spellcheckErrorsContent < 15.5
spellcheckErrorsSubject < 0.5
Count.Common.Word..click < 0.5
wordCount < 126
countLink < 0.5
Yes
Yes
Count.Common.Word..click < 0.5
Count.Common.Word..advertisement < 0.5
contentCount < 242.5
Yes
No
Yes
Yes
Yes
No
Yes
spellcheckErrorsContent < 7.5
No
Yes
Yes
No
No
No
No

Graph 2: Pruned Tree



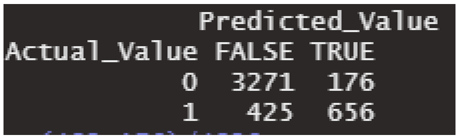
Clustering resulted in the vast majority of the data points, both spam and non-spam, being in cluster 1 for both the full dataset and the first five principal score vectors, making it inconclusive. Finally, Regression returned an accuracy of 0.88083 on test data, giving us our best accuracy.

Table 6: Regression Test Results



```
            Predicted_Value
Actual_Value FALSE TRUE
           0   868   45
           1    98  189
```

Table 7: Regression Train Results



```
            Predicted_Value
Actual_Value FALSE TRUE
           0  3271  176
           1   425  656
```

Project Management: We all worked on exploring the dataset and the existing code that existed already, and Jack, Harshil, and Connor all worked on deriving dataset parameters for each email, including whether the email contained a link, mentioned money, and the word count. Jack took on applying classification, Connor took on applying trees, clustering, and correlation, and Harshil took on applying regressions. Daniel wrote up the report, and Harshil made the poster with some help from Jack. Finally, everyone worked together to make the video presentation. We met up via Zoom every few days to check our progress, and we did not define specified team roles beyond who took on what tasks. We set deadlines for when we should have certain parts of our planned tasks completed by, such as creating dataset parameters, applying statistical techniques, or finishing the video, so that we would have everything done by the time the final project was due.

Conclusions: In conclusion, Regression gave the best overall accuracy, Classification gave the second-best accuracy with the best rate of false positives, and Trees gave the third best accuracy, while Correlation and Clustering were inconclusive. However, even the best overall accuracy was not very accurate, resulting in the conclusion that our derived variables of whether an email contained a link or a reference to money was less important than the language modeling Harshit did in his blog, as it produced a higher level of accuracy than any of our methods did.

References:

Kumar, H. (2017, August 25). Technical fridays. Email spam filtering: Text analysis in R. Retrieved December 13, 2022, from https://kharshit.github.io/blog/2017/08/25/email-spam-filtering-text-analysis-in-r

Metsis, V., Androutsopoulos, I., & Paliouras, G. (2006, July). Spam filtering with naive bayes-which naive bayes?. In *CEAS* (Vol. 17, pp. 28-69).

Rubin, K. (2022, May 6). *The ultimate list of 394 email spam trigger words to avoid in 2021*. HubSpot Blog. Retrieved December 15, 2022, from https://blog.hubspot.com/blog/tabid/6307/bid/30684/the-ultimate-list-of-email-spam-trigger-words.aspx