Spam Message Detection

Stuart Miller, Justin Howard, and Paul Adams

October 6, 2020

1 Introduction

2 Methods

2.1 Data

- SpamAssassin
- Categories

2.2 Modeling

- NB
- Decision Trees (rpart)
- pick another one

2.3 Results

2.3.1 An Analysis of Odds

We used log-odds to assess the fit of a probability-based model, where the probability of a message being spam is equal to the count of spam words divided by the count of spam messages plus 0.5 and the probability of a message being not spam is equal to the count of non-spam words divided by the count of non-spam emails plus 0.5. This enabled us to gauge how apt the assumptions of the model are, with respect to spam and not-spam.

With respect to the following analysis, the null hypothesis states that a given message is not spam:

Based on the distribution of Type I and Type II errors of log likelihood values in fig 1. (line plots) - where Type I error is rejecting the null hypothesis when the null hypothesis should not be rejected and Type II error is failing to reject the null when there is statistically significant evidence the null should be rejected - the threshold cutoff for determining spam is a log-likelihood value of -48. Translating this to a threshold value meaning, a message with a log-likelihood value of greater than -48 is spam and a message with a log-likelihood value of less than -48 is not spam.

The distributions of log-likelihoods for each class can be inspected in fig. 2 (box-plots). Based on distribution spread of log likelihood ratios for each class, the probability approach appears better at identifying spam than messages that are not spam. This is because the spam class has an overall distribution - including the mean - closer to 0 than the non-spam class, which is indicative of a better fit. However, there is more repeatability of results within the non-spam class words as the interquartile range is much narrower than that of the spam class. This ultimately means words used in non-spam messages are not as unique to non-spam messages as those in spam messages. Intuitively, this makes sense because many of the words used in spam messages are often intentional misspellings - such as words that substitute "3" in place of "e" so spam filters cannot catch them as easily - that do not commonly appear in non-spam messages. Conversely, most words used in non-spam messages also appear in spam messages.

Because of this, in addition to the Naive Bayes approach, we constructed additional models to classify spam vs. not spam, including logistic regression and a decision tree classifier.

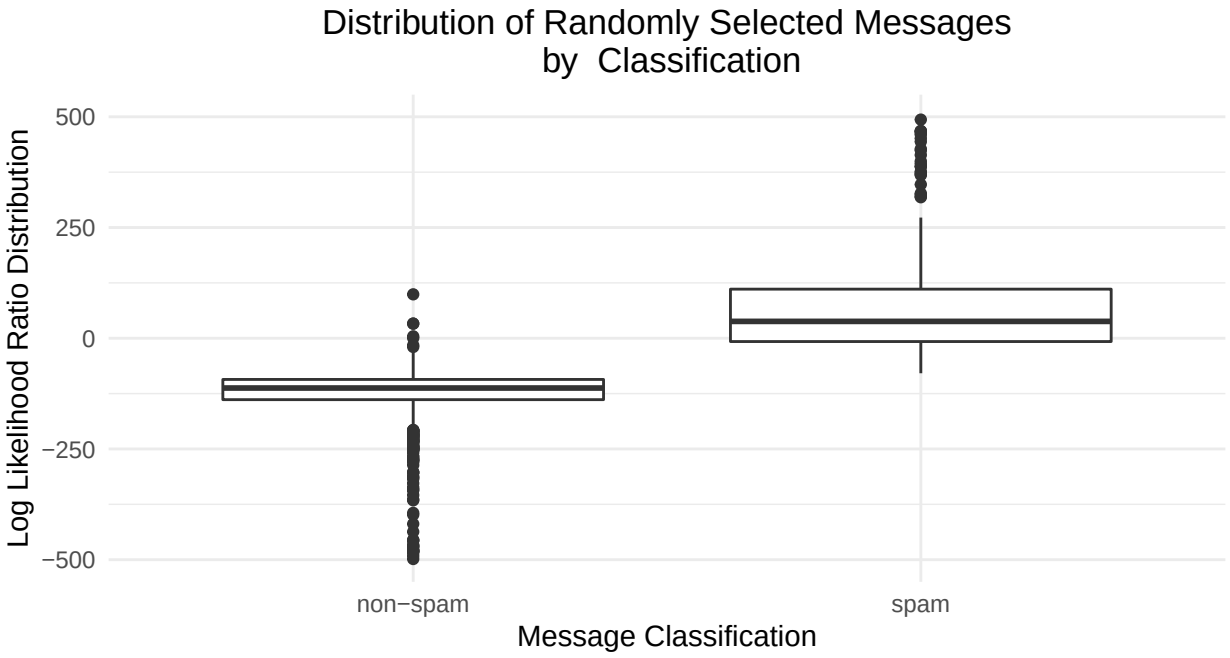

Figure 1: (#fig:spam_box_plot)Figure 1: NB Log Likelihood Ratios Distributions

2.3.2 Model Results

Using F1 as the baseline metric for error assessment, the Partition Tree outperformed Naive Bayes and Logistic Regression during both internal and external cross-validation. On internal 5-fold cross-validation, the Partition Tree scored an F1 of 0.957, while the Naive Bayes - outperforming the other probability-based model (Logistic Regression) by 0.042 - scored 0.945.

After applying the models to the external test set, error further diverged between the tree-based and probability-based models. Contrary to the internal cross-validation results, Logistic Regression also appears to have generalized better than Naive Bayes, marginally outperforming the Bayesian model by 0.012.

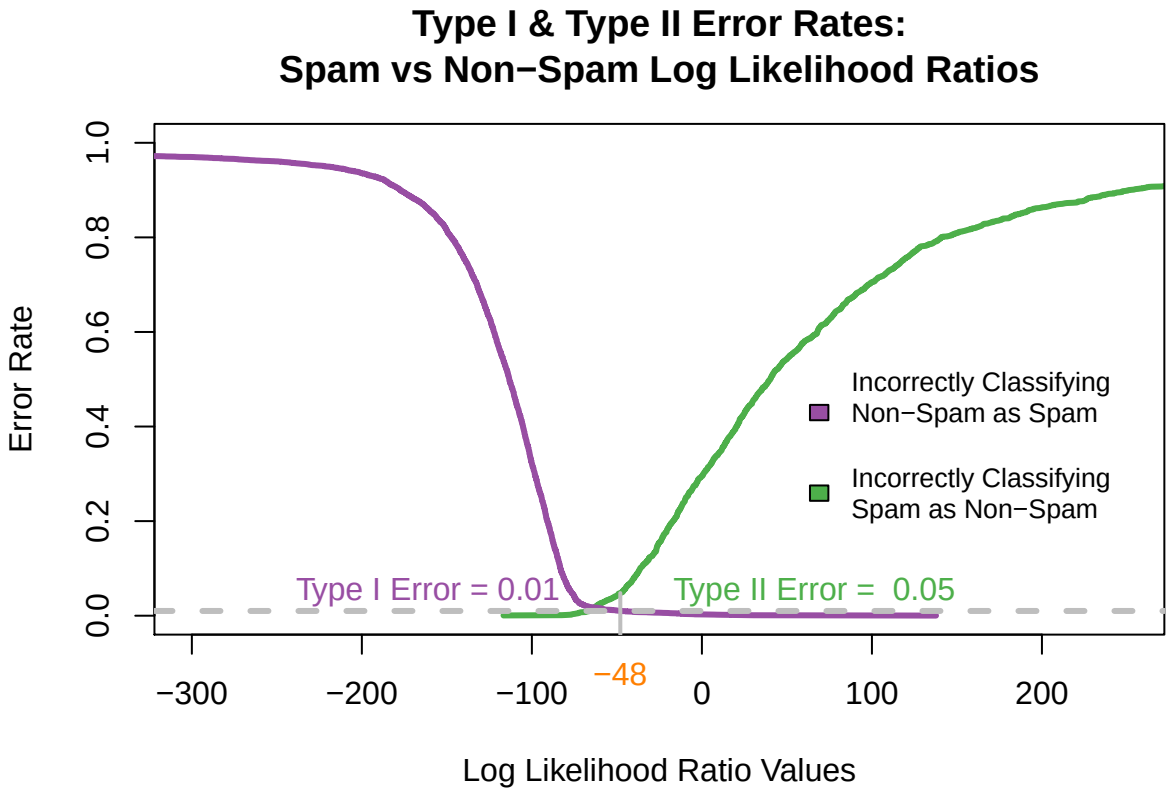

Figure 2: (#fig:Error Types)Figure 2: NB Type I and Type II Error

However, the tree-based Partition Tree model maintained superiority over the probability-based models; Logistic Regression scored and F1 of 0.911 and the Partition Tree scored an F1 of 0.957.

Da-dum.

Table 1. Hyperparameter Tuning Results

Model	Parameter	Value
Naive Bayes	laplace	0
Naive Bayes	usekernal	`True`
Naive Bayes	adjust	`True`
Partition Tree	cp	0.0015

Table 2. Cross Validation Results

Model	F1 Score
Naive Bayes	0.903
Partition Tree	0.957
Logistic Regression	0.945

Table 3. Test Results

Model	F1 Score
Naive Bayes	0.899
Partition Tree	0.953
Logistic Regression	0.911

3 Conclusion

A Code

```
Code goes here
```