Assignment 3: Spambayes

Ryan Bernstein

In order to create a Naive Bayesian classifier, we first split our training data into positive and negative examples. We also find the probability that an arbitrary instance in the training set is positive or negative.

Once this is done, we calculate means and standard deviations for each of the features in the dataset. We capture a separate mean and standard deviation array for the positive and negative instances.

Given some unknown feature vector, we can then calculate $P(x_i|+)$ and $P(x_i|-)$ for each feature in each class as follows:

$$N(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

If the standard deviation was 0, we replace it with 0.01 to avoid divide-by-zero errors.

Once this is done, we can find the class with the higher conditional probability by multiplying each class probability by the product of the conditional probabilities of each feature for that class. To make our numbers easier to work with, we take a logarithmic sum in lieu of a product.

This method assumes that all features are independent. In reality, this is probably not true for this particular dataset, since the word/character frequencies found in emails will likely depend on its content. However, even though our classifier is "naive", it still performs pretty well. It's reasonably well-known that naive Bayesian classifiers can still perform well if the dependencies themselves aren't biased toward a certain class (or if dependencies cancel each other out).

If my understanding of the problem is correct, this means that even if there are spam-biased features that influence each other, the presence of non-spam-biased features that are also conditionally dependent upon each other can cancel each other out. If this is true, my guess is that there may be conditional dependencies between the business-oriented features in the dataset (e.g. "meeting" may make "conference" more likely to appear) that balance out feature dependencies weighted toward the positive instances.

Confusion Matrix

865	437
42	957

Accuracy: 79.18%
Precision: 95.37%
Recall: 66.44%