Ryan Filyas

Portland State UNIVERSITY Please scroll to bottom for report.

CS 445/545: Machine Learning, Spring 2022

Programming Assignment #2, A. Rhodes

Note: This assignment is **due by Sunday, 5/15 at 10pm**; you will turn in the assignment by email to our grader, as instructed below.

In this homework you will use Gaussian Naïve Bayes to classify the Spambase data from the **UCI ML repository**, which can be found here:
https://archive.ics.uci.edu/ml/datasets/spambase

Classification with Naïve Bayes

1. **Create training and test set:**

Split the data into a training and test set. Each of these should have about 2,300 instances, and each should have about 40% spam, 60% not-spam, to reflect the statistics of the full data set. Since you are assuming each feature is independent of all others, here it is not necessary to standardize the features.

2. **Create probabilistic model. (Write your own code to do this.)**

- Compute the prior probability for each class, 1 (spam) and 0 (not-spam) in the training data. As described in part 1, $P(1)$ should be about 0.4.

- For each of the 57 features, compute the mean and standard deviation in the training set of the values given each class. If any of the features has zero standard deviation, assign it a "minimal" standard deviation (e.g., 0.0001) to avoid a divide-by-zero error in Gaussian Naïve Bayes.

3. **Run Naïve Bayes on the test data. (Write your own code to do this.)**

- Use the Gaussian Naïve Bayes algorithm to classify the instances in your test set, using

$$P(x_i \mid c_j) = N(x_i; \mu_{i,c_j}, \sigma_{i,c_j}), \text{ where } N(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\left(\frac{(x-\mu)^2}{2\sigma^2}\right)}$$

Because a product of 58 probabilities will be very small, we will instead use the

log of the product. Recall that the classification method is:

$$class_{NB}(\mathbf{x}) = \underset{class}{\operatorname{argmax}} \left[P(class) \prod_i P(x_i \mid class) \right]$$

Since $\underset{z}{\operatorname{argmax}}\, f(z) = \underset{z}{\operatorname{argmax}} \log(z)$

we have:

$$class_{NB}(\mathbf{x}) = \underset{class}{\operatorname{argmax}} \log \left[P(class) \prod_i P(x_i \mid class) \right]$$
$$= \underset{class}{\operatorname{argmax}} \left[\log P(class) + \log P(x_1 \mid class) + \ldots + \log P(x_n \mid class) \right]$$

In your report, include a short description of what you did, and your results: the accuracy, precision, and recall on the test set, as well as a confusion matrix for the test set. Write a few sentences describing your results, and answer these questions: Do you think the attributes here are independent, as assumed by Naïve Bayes? Does Naïve Bayes do well on this problem in spite of the independence assumption? Speculate on other reasons Naïve Bayes might do well or poorly on this problem.

Here is what you need to turn in:

- Your report.
- Your well-commented code.

How to turn it in (read carefully!):

- Send these items in electronic format to our TA by the due date. No hard copy please!
- The report should be in pdf format and the code should be in plain-text format.
- Put "MACHINE LEARNING PROGRAMMING #2" in the subject line.

If there are any questions, don't hesitate to ask me or our TA.

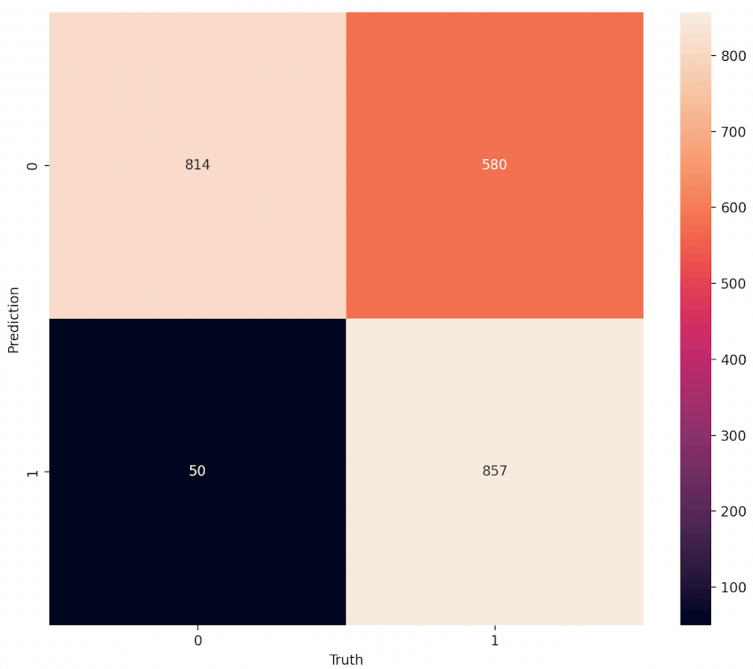
Summary: For part 1 to split the training and test set I first split into spam/not spam categories, permuted each section and divided it evenly into train and test sets resulting in 4 categories belonging to the two sets split 60/40 between not spam and spam. For part two I computed the mean and std in a function for each subset of data. For part 3 I wrote a function to compute naive bases for each set and followed the equations provided in my code.

For my results the recall was about 60% and the precision was about 94% with a total accuracy around 72% for the test data. This seems to make sense given probability.

Q1. The attributes do not seem independent as assumed by naive bayes. The elements found in spam and not spam can be incredibly similar (spam is trying to mimic real messages in many cases), so making the assumption that the classes are stricly different and using their probabilities to influence the calculations leads to a suboptimal result compared to other methods.

Q2: It is however remarkable that despite this built in assumption the model still has more accuracy than chance. The model seems to do fairly well, but again there's a lot of overlap between spam and not spam in similarity that can lead to downfalls in this model. On the other hand there are things found in spam uncommon to emails which helps a lot with accuracy giving this model some successes.

See confusion matrix and program output below.




```
Training Spam Prior:  0.3939
Training Not Spam Prior:  0.6061
Test Spam Prior:  0.3942
Test Not Spam Prior:  0.6058
Spam Accuracy:  0.9449
Not Spam Accuracy:  0.5839
Total Accuracy 0.7262
Precision:  0.9449
Recall:  0.5964
Number of test data points: 2301
```