James Hahn
Artificial Intelligence
Dr. Diane Litman

Programming Portion Report

Figure 1

Iteration	Number of positive samples in training	Number of negative samples in training	Number of positive samples in development	Number of negative samples in development
1	1450	2230	363	558
2	1450	2231	363	557
3	1450	2231	363	557
4	1451	2230	362	558
5	1451	2230	362	558

NOTE: The necessary data for iteration 1, 2, 3, …, is written to files 1_Train.txt, 1_Dev.txt, 2_Train.txt, … containing the training data and testing (development) data respectively.

NOTE: For iteration 1, the classifier utilizes fold #1 as the testing set and the remaining k-1 folds for training. This can be generalized to be: for iteration x, the classifier utilizes fold x as the testing set and the remaining k-1 folds for training.

NOTE: Positive samples are ones labelled as "spam" and negative samples are labelled as "not spam" in the dataset.

There should be a table here displaying the conditional probability of whether a feature is less than or greater than its mean given spam or not spam, but the table is rather large. The file features.xlsx contains all the necessary frequencies (5 iterations with 228 values each due to each of the 57 features possessing 4 conditional probabilities). However, importing this file to Microsoft Word would be a significant task in terms of space and ability to do so.

Figure 2

Iteration	False positives rate	False negatives ratio	Error rate
1	0.0556	0.1873	0.1075
2	0.0521	0.1956	0.1087
3	0.0539	0.1405	0.0880
4	0.0573	0.1326	0.0870
5	0.0573	0.1630	0.0989
Avg	0.0552	0.1638	0.0980

These results obviously reign superior over just choosing the majority class. From the spambase documentation, 39.4% of the emails were labelled spam, indicating not spam is the majority class with 60.6%. From figure 1, simple calculations in the development sets (training data) lead to 39.4% positive samples and 60.6% negative samples, matching exactly the demographics of the entire dataset.

If the classifier ran through each development set and predicted every sample as *not spam*, there would be a 60.6% accuracy (~40% error rate). In addition, there would be 0% chance of false positives, but 39.4% chance of false negatives. After analyzing the false positive and false negative ratios for each iteration's development set in figure 2, one can clearly observe the significant advantage to the naïve bayes classifier. Although the false positives are above 0%, they are relatively close at ~5.5%. The real competitive edge is the difference of ~16.4% average false negative ratio compared to the above 39.4% false negatives for majority class classification.

In real world training samples, there is no guarantee the development set's demographics will replicate that of the training set. As a result, majority class classification may provide better or worse results in terms of its original 60.4% accuracy. However, this is pure luck and there is no way to successfully predict the accuracy of this classification given an arbitrary set of training samples. On the other hand, naïve bayes provides ~90.2% accuracy based on actual conditional probabilities of features in our vast original training data. Given arbitrary testing data, naïve bayes should easily outperform majority class because:

1. Its original accuracy is higher.
2. It bases predictions off of previously seen data and analyzes conditional probabilities of the features that were provided, whereas majority class just utilizes the output class to predict future classes.

As a result, naïve bayes far surpasses majority class classification in terms of accuracy, the combination of the ratios of false positives and false negatives, and its potential to successfully predict future data.