Everett Hu
ehu6

CS 412 Assignment 4 Report

Classification Method Descriptions

Naïve Bayes

The Naïve Bayes classifier used in my classification framework is based on Bayes' Theorem which states:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}.$$

We apply this theorem by letting A represent the hypothesis that a data tuple belongs to a certain class and B represents the data tuple. The goal of the Naïve Bayes classifier is to classify a data tuple by maximizing the posteriori probability P(A|B). Since P(B) is constant for all classes, substituting X in for B and C in for A yields the equation:

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

for the posteriori probability this classifier seeks to maximize across different classes. This classifier runs under the assumption that the attributes of any data tuple to be classified are conditionally independent. Because of this assumption, we use this equation to calculate the total likelihood probability of a data tuple:

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \ldots \times P(x_n|C_i)$$

In addition, laplacian correction is applied to the data to help solve the zero-probability problem by adding 1 to each possible case count and updating the total counts accordingly to maintain the probability estimates.

In general, the training and test files are read into data structures in the program and the total number of attributes is determined. A train function is then called to calculate the likelihood probabilities of each case as well as the probabilities for each class. Finally, both training and test data are classified using a classify function which calculates the posteriori probabilities for each possible class. Since this assignment only deals with binary classification, there are only two possible classes. The posteriori probabilities are compared and a classification is made based on this comparison. The Naïve Bayes classification is then compared with the actual classification and metrics are generated.

NBAdaBoost

In addition to the Naïve Bayes algorithm, a boosting algorithm is used in conjunction to optimize accuracy. First, a weight is assigned to each data tuple in the training dataset. The intial weights are set using the init_weights method, which initializes each weight to be 1/d, where d is the size of the training data set. Then, a generate_classifiers method is called, which runs k times, creating k instances of the Naïve Bayes classifier.

During each round of generate_classifiers, a sample data set is sampled (with replacement) from the original training set based on the weights of each tuple (higher weights had a higher chance of being pulled). A classifier is then initialized and trained using this sample data set. Then the weighted error rate of the current classifier model is calculated by classifying the original training set using the model. This error rate is calculated using the following equation:

$$\epsilon_i = \sum_{j=1}^{d} w_i(j)\, 1\left(M_i(x_j) \neq y_j\right)$$

1 if misclassified
0 otherwise

Then, using this error rate, all tuples which were correctly classified have their weight adjusted according to this equation:

$$w_{i+1}(j) = w_i(j) * \epsilon_i/(1-\epsilon_i)$$

This weight adjustment reduces the weight of data tuples correctly classified, providing more attention to tuples that were misclassified in subsequent rounds. After all data tuple weights have been adjusted, the weights are normalized so that their sum is 1. In addition, the weight for each classifier is calculated using the error rate as well using the following equation:

$$\alpha_i = \log\left(\frac{\epsilon_i}{1-\epsilon_i}\right)$$

This continues, which each round using the weights of the last round to generate a new sample set and classifier.

Finally, the training and test data sets are run against each classifier. The final classification is based on this equation:

$$\sum_{i=1}^{k} \alpha_i M_i(x)$$

where alpha is the weight of the classifier and M(x) is the classification by that classifier. The resulting value from this equation yields the final classification. Since this is binary classification, positive and negative values correspond the separate classifications.

Model Evaluation Measures

NaiveBayes: adult.train

True Positive: 203	False Negative: 192
False Positive: 84	True Negative: 1128

Accuracy: 0.828037	Precision: 0.707317
Error Rate: 0.171963	F-1 Score: 0.595308
Sensitivity: 0.513924	F-0.5 Score: 0.657809
Specificity: 0.930579	F-2 Score: 0.543653

NBAdaBoost: adult.train

True Positive: 234	False Negative: 161
False Positive: 92	True Negative: 1118

Accuracy: 0.842368	Precision: 0.717791
Error Rate: 0.157632	F-1 Score: 0.649098
Sensitivity: 0.592405	F-0.5 Score: 0.688640
Specificity: 0.923967	F-2 Score: 0.613851

NaiveBayes: adult.test

True Positive: 3844	False Negative: 3602
False Positive: 1456	True Negative: 22054

Accuracy: 0.836607	Precision: 0.725283
Error Rate: 0.163393	F-1 Score: 0.603170
Sensitivity: 0.516250	F-0.5 Score: 0.670949
Specificity: 0.930579	F-2 Score: 0.547828

NBAdaBoost: adult.test

True Positive: 4266	False Negative: 3180
False Positive: 1816	True Negative: 21694

Accuracy: 0.838610	Precision: 0.701414
Error Rate: 0.161390	F-1 Score: 0.630692
Sensitivity: 0.572925	F-0.5 Score: 0.671304
Specificity: 0.922756	F-2 Score: 0.594714

NaiveBayes: breast_cancer.train

True Positive: 25	False Negative: 31
False Positive: 16	True Negative: 108

Accuracy: 0.738889	Precision: 0.609756
Error Rate: 0.261111	F-1 Score: 0.515464
Sensitivity: 0.446429	F-0.5 Score: 0.568182
Specificity: 0.870968	F-2 Score: 0.471698

NBAdaBoost: breast_cancer.train

True Positive: 22	False Negative: 34
False Positive: 14	True Negative: 110

Accuracy: 0.733333	Precision: 0.611111
Error Rate: 0.266667	F-1 Score: 0.478261
Sensitivity: 0.392857	F-0.5 Score: 0.550000
Specificity: 0.887097	F-2 Score: 0.423077

NaiveBayes: breast_cancer.test

True Positive: 11	False Negative: 18
False Positive: 10	True Negative: 67

Accuracy: 0.735849	Precision: 0.523810
Error Rate: 0.264151	F-1 Score: 0.440000
Sensitivity: 0.379310	F-0.5 Score: 0.786726
Specificity: 0.870130	F-2 Score: 0.401460

NBAdaBoost: breast_cancer.test

True Positive: 10	False Negative: 19
False Positive: 8	True Negative: 69

Accuracy: 0.745283	Precision: 0.555556
Error Rate: 0.254717	F-1 Score: 0.425532
Sensitivity: 0.344828	F-0.5 Score: 0.495050
Specificity: 0.896104	F-2 Score: 0.373134

NaiveBayes: led.train

True Positive: 324	False Negative: 314
False Positive: 65	True Negative: 1384

Accuracy: 0.818400	Precision: 0.832905
Error Rate: 0.181600	F-1 Score: 0.630964
Sensitivity: 0.507837	F-0.5 Score: 0.738377
Specificity: 0.955141	F-2 Score: 0.550833

NBAdaBoost: led.train

True Positive: 466	False Negative: 172
False Positive: 153	True Negative: 1296

Accuracy: 0.844274	Precision: 0.752827
Error Rate: 0.155726	F-1 Score: 0.741448
Sensitivity: 0.730408	F-0.5 Score: 0.748234
Specificity: 0.894410	F-2 Score: 0.734784

NaiveBayes: led.test

True Positive: 162	False Negative: 189
False Positive: 27	True Negative: 756

Accuracy: 0.809524	Precision: 0.857143
Error Rate: 0.190476	F-1 Score: 0.600000
Sensitivity: 0.461538	F-0.5 Score: 0.731707
Specificity: 0.965517	F-2 Score: 0.508475

NBAdaBoost: led.test

True Positive: 266	False Negative: 85
False Positive: 72	True Negative: 711

Accuracy: 0.861552	Precision: 0.786982
Error Rate: 0.138448	F-1 Score: 0.772134
Sensitivity: 0.757835	F-0.5 Score: 0.780975
Specificity: 0.908046	F-2 Score: 0.763490

NaiveBayes: poker.train

True Positive: 743	False Negative: 5
False Positive: 285	True Negative: 9

Accuracy: 0.721689	Precision: 0.722763
Error Rate: 0.278311	F-1 Score: 0.836712
Sensitivity: 0.993316	F-0.5 Score: 0.764403
Specificity: 0.030612	F-2 Score: 0.924129

NBAdaBoost: poker.train

True Positive: 700	False Negative: 47
False Positive: 263	True Negative: 32

Accuracy: 0.702495	Precision: 0.726895
Error Rate: 0.297505	F-1 Score: 0.818713
Sensitivity: 0.937082	F-0.5 Score: 0.761035
Specificity: 0.108475	F-2 Score: 0.885852

NaiveBayes: poker.test

True Positive: 448	False Negative: 11
False Positive: 217	True Negative: 2

Accuracy: 0.663717	Precision: 0.673684
Error Rate: 0.336283	F-1 Score: 0.797153
Sensitivity: 0.976035	F-0.5 Score: 0.718179
Specificity: 0.009132	F-2 Score: 0.895642

NBAdaBoost: poker.test

True Positive: 426	False Negative: 33
False Positive: 196	True Negative: 23

Accuracy: 0.662242	Precision: 0.684887
Error Rate: 0.337758	F-1 Score: 0.788159
Sensitivity: 0.928105	F-0.5 Score: 0.722769
Specificity: 0.105023	F-2 Score: 0.866558

Parameter Explanation

Since the classifier is a binary classifier, the number of possible classes that a data tuple can be classified into is 2. I chose the value k = 7 for the NBAdaBoost algorithm because testing showed that generating 7 classifiers seemed to be the most optimal accuracy/runtime ratio. 7 classifiers produced a significant improvement in performance over 6, whereas adding more classifiers beyond 7 produced either nonexistent or negligible improvements.

Conclusion

It seems that the ensemble method does have the potential to improve the performance of the basic classification method. However, the performance is volatile and also circumstantial. For some training/test datasets, I had to run the algorithm multiple times to produce results with a significant accuracy improvement. In addition, for the poker datasets, I found that the NBAdaBoost algorithm had very little improvement over only the Naïve Bayes, and in most cases, the accuracy was actually lower. On the other hand, for the other datasets, the NBAdaBoost algorithm produced more accurate results for the test datasets, especially in the led datasets, where the accuracy increased by more than 5% in the test dataset. Therefore, overall it seems that the ensemble method does improve performance in most cases.