CS 412 Introduction to Data Mining

Assignment 4 Report

Haoran Yu - hyu34

1. brief introduction of the classification methods in your classification framework;

i. Naive Bayes

Naïve Bayes Classifier is a widely used classification scheme, which is based on Bayes' Theorem. Basically Bay's theorem states

$$P(H|X) = \frac{P(X|H)\times P(H)}{P(X)}$$

where *X* is a data tuple and H is the hypothesis that X belongs to a specified class C_i. And then we compare the posteriori probability of each hypothesis *H*. By derivation and simplification, we compare train and classify data tuple based on the magnitude of $P(X|C_i)P(C_i)$ for various C_i, $i = 1\ldots k$. Information about $P(C_i)$ is learned from the labels of training set. The class-conditional independence is assumed, which greatly simplify the calculation of $P(X|C_i)$:

$$P(X|C_i) = \prod_{k=1}^{n} P(X_k \vee C_i) \ldots (1)$$

To predict the class label of X, $P(X|C_i)\ P(C_i)$ is evaluated for each class C_i. The classifier predicts that the class label of tuple *X* is the class, if and only if

$$P(X|C_i)P(C_i) > P(X|C_i)P(C_i), for\ 1 \le j \le m, j \ne i \quad \ldots (2)$$

In this project, we only focus on binary class classification.

In addition, a very simple smoothing method is hired here. When we come across some value a feature never have in testing data, we set the likelihood to a very small number. Here, 0.0000001 is small enough for the use of smoothing.

Basically, first the filename is read and the fixed number of features is set. Then I used a Train function to train the model based on Naive Bayes Method and a 3D Ptable is generated as the main part of the model. Then a test function is used to feed all the tuple for test with a loop. In the test function, there is a judge function called by it, testing on each tuple it received as a parameter.

ii. NBAdaBoost

In addition to the Naive Bayes method. Three new functions called adaBoostTrain/ adaBoostTest/ adaBoostJudge.

In adaBoostTrain I have a loop to iterate in order to build k = 7 weak classifiers. In each I did sampling first according to the weight(at first the weight is equal and sum to one) of tuples in the input training dataset and then use the train function to train a weak classification model. Also I calculated the error here according to

$$Error(M_j) = \sum_{j=1}^{d} w_j \times err(X_j)$$

in which sums the weight of misclassified X_j within the d tuples of the sample. If the error rate is too high(> 0.5) , this weak classifier would be abandoned and this iteration will happen again and regenerate another weak classifier(until error <= 0.5). Then the weights of tuples correctly predicted are all calculated before normalized with all other weights of tuples together as

$$New\ weight = Old\ weight \times \frac{Error(M_j)}{1-Error(M_j)}$$

“adaBoostTest” is similar to test function, in which there is a loop, sending each tuple in the input testing dataset to adaBoostJudge and determine a class-label according to the model of nbAdaBoost.

In adaBoostJudge, first I initialized the class-weight as 0 for both positive and negative class. Then there is a loop for k = 7 weak classifiers. Within each loop, I calculate the weight of classifier’s vote according to

$$W_i = log\frac{1-error(M_i)}{error(M_i)}$$

and then add it to the predicted class label for the input tuple according to the k^{th} weak classifier. Finally I will reture the class label with the largest class-weight.

2. all model evaluation measures you calculated above (8 metrics * 2 methods * 4 datasets * 2 (training and test));
    i. NaiveBayes a1a.train a1a.train

	Test Negative	Test Positive
Real Negative	TN: 965	FP: 245
Real Positive	FN: 69	TP: 326

Accuracy: 0.804361 Error Rate: 0.195639
Sensitivity: 0.825316 Specificity: 0.797521
Precision: 0.570928 F-1 Score: 0.674948
F-0.5 Score: 0.608436 F-2 Score: 0.757787

    ii. NaiveBayes a1a.train a1a.test

	Test Negative	Test Positive
Real Negative	TN: 18670	FP: 484-
Real Positive	FN: 1581	TP:5865

Accuracy: 0.792577 Error Rate: 0.207423
Sensitivity: 0.787671 Specificity: 0.79413
Precision: 0.547875 F-1 Score: 0.646245
F-0.5 Score: 0.583396 F-2 Score: 0.724271

    iii. NBAdaBoost a1a.train a1a.train

	Test Negative	Test Positive
Real Negative	TN: 1012	FP: 198
Real Positive	FN: 101	TP: 294

Accuracy: 0.813707 Error Rate: 0.186293
Sensitivity: 0.744304 Specificity: 0.836364
Precision: 0.597561 F-1 Score: 0.662909
F-0.5 Score: 0.622091 F-2 Score: 0.709459

    iv. NBAdaBoost a1a.train a1a.test

	Test Negative	Test Positive
Real Negative	TN: 19650	FP: 3860
Real Positive	FN: 2014	TP: 5432

Accuracy: 0.810247 Error Rate: 0.189753
Sensitivity: 0.729519 Specificity: 0.835815
Precision: 0.584589 F-1 Score: 0.649062
F-0.5 Score: 0.608778 F-2 Score: 0.695056

v. NaiveBayes breast_cancer.train breast_cancer.train

	Test Negative	Test Positive
Real Negative	TN: 105	FP: 19
Real Positive	FN:26	TP: 30

Accuracy: 0.75 Error Rate: 0.25
Sensitivity: 0.535714 Specificity: 0.846774
Precision: 0.612245 F-1 Score: 0.571429
F-0.5 Score: 0.595238 F-2 Score: 0.549451

vi. NaiveBayes breast_cancer.train breast_cancer.test

	Test Negative	Test Positive
Real Negative	TN: 63	FP: 14
Real Positive	FN:14	TP: 15

Accuracy: 0.735849 Error Rate: 0.264151
Sensitivity: 0.517241 Specificity: 0.818182
Precision: 0.517241 F-1 Score: 0.517241
F-0.5 Score: 0.517241 F-2 Score: 0.517241

vii. NBAdaBoost breast_cancer.train breast_cancer.train

	Test Negative	Test Positive
Real Negative	TN: 104	FP: 20
Real Positive	FN:27	TP: 29

Accuracy: 0.738889 Error Rate: 0.261111
Sensitivity: 0.517857 Specificity: 0.83871
Precision: 0.591837 F-1 Score: 0.552381
F-0.5 Score: 0.575397 F-2 Score: 0.531136

viii. NBAdaBoost breast_cancer.train breast_cancer.test

	Test Negative	Test Positive
Real Negative	TN: 63	FP: 14
Real Positive	FN:14	TP: 15

Accuracy: 0.735849 Error Rate: 0.264151
Sensitivity: 0.517241 Specificity: 0.818182
Precision: 0.517241 F-1 Score: 0.517241
F-0.5 Score: 0.517241 F-2 Score: 0.517241

ix. NaiveBayes led.train led.train

	Test Negative	Test Positive
Real Negative	TN: 1357	FP: 92
Real Positive	FN: 239	TP: 399

Accuracy: 0.841399 Error Rate: 0.158601
Sensitivity: 0.625392 Specificity: 0.936508
Precision: 0.812627 F-1 Score: 0.70682
F-0.5 Score: 0.766718 F-2 Score: 0.655603

x. NaiveBayes led.train led.test

	Test Negative	Test Positive
Real Negative	TN:742	FP: 41
Real Positive	FN: 144	TP: 207

Accuracy: 0.836861 Error Rate: 0.163139
Sensitivity: 0.589744 Specificity: 0.947637
Precision: 0.834677 F-1 Score: 0.691152
F-0.5 Score: 0.770663 F-2 Score: 0.626513

xi. NBAdaBoost led.train led.train

	Test Negative	Test Positive
Real Negative	TN: 1351	FP: 98
Real Positive	FN: 225	TP: 413

Accuracy: 0.845232 Error Rate: 0.154768
Sensitivity: 0.647335 Specificity: 0.932367
Precision: 0.808219 F-1 Score: 0.718886
F-0.5 Score: 0.769948 F-2 Score: 0.674176

xii. NBAdaBoost led.train led.test

	Test Negative	Test Positive
Real Negative	TN:740	FP: 43
Real Positive	FN: 134	TP: 217

Accuracy: 0.843915 Error Rate: 0.156085
Sensitivity: 0.618234 Specificity: 0.945083
Precision: 0.834615 F-1 Score: 0.710311
F-0.5 Score: 0.780014 F-2 Score: 0.652043

xiii. NaiveBayes poker.train poker.train

	Test Negative	Test Positive
Real Negative	TN: 10	FP: 284
Real Positive	FN: 7	TP: 740

Accuracy: 0.720461 Error Rate: 0.279539
Sensitivity: 0.990629 Specificity: 0.0340136
Precision: 0.722656 F-1 Score: 0.835686
F-0.5 Score: 0.763989 F-2 Score: 0.922233

xiv. NaiveBayes poker.train poker.test

	Test Negative	Test Positive
Real Negative	TN: 2	FP: 217
Real Positive	FN: 11	TP: 448

Accuracy: 0.663717 Error Rate: 0.336283
Sensitivity: 0.976035 Specificity: 0.00913242
Precision: 0.673684 F-1 Score: 0.797153
F-0.5 Score: 0.718179 F-2 Score: 0.895642

xv. NBAdaBoost poker.train poker.train

	Test Negative	Test Positive
Real Negative	TN: 30	FP: 264
Real Positive	FN: 37	TP: 710

Accuracy: 0.710855 Error Rate: 0.289145
Sensitivity: 0.950469 Specificity: 0.102041
Precision: 0.728953 F-1 Score: 0.825102
F-0.5 Score: 0.764592 F-2 Score: 0.896012

xvi. NBAdaBoost poker.train poker.test

	Test Negative	Test Positive
Real Negative	TN: 18	FP: 201
Real Positive	FN: 38	TP: 421

Accuracy: 0.647493 Error Rate: 0.352507
Sensitivity: 0.917211 Specificity: 0.0821918
Precision: 0.676849 F-1 Score: 0.778908
F-0.5 Score: 0.714286 F-2 Score: 0.856387

3. Does your framework perform equally good on training and test datasets? Why or why not?

   No, both NaiveBayes and NBAdaBoost perform better on training set than test set. Since the model come from the training data, the framework will obviously fit the training data better.

4. parameters you chose during implementation and why you chose these parameters;

   - Number of class = 2 : all dataset are binary and only have +1 and -1 as class labels.
   - Smoothing factor = 0.0000001: according to the size of all the dataset, 0.0000001 is a number that is small enough. The idea here is similar to laplacian correction but may not fit in to some dataset with much much more features. But the choice here won't have big influence on the performance of smoothing on these four dataset.
   - Number of weak classifiers k = 7 for NBAdaBoost. k = 7 already result in a better outcome comparing to Naive Bayes and increase k to be greater than 7 don't have much influence.result .

5. and also your conclusion on whether the ensemble method improves the performance of the basic classification method you chose, why or why not;

   For most of the case in these four dataset. NBAdaBoost proform better than pure Naive Bayes. Because NBAdaBoost did sampling and voting, making the weight of each tuples different which make it possible for the classifier to pay more attention to those misclassified tuples.

6. verify your guess on whether your basic classification method is ensemble compatible.

   According to the result answered in the second question. The performance of basic classification method(Naive Bayes) is ensemble compatible since the result of NBAdaBoost is only slightly better than Naive Bayes in general.