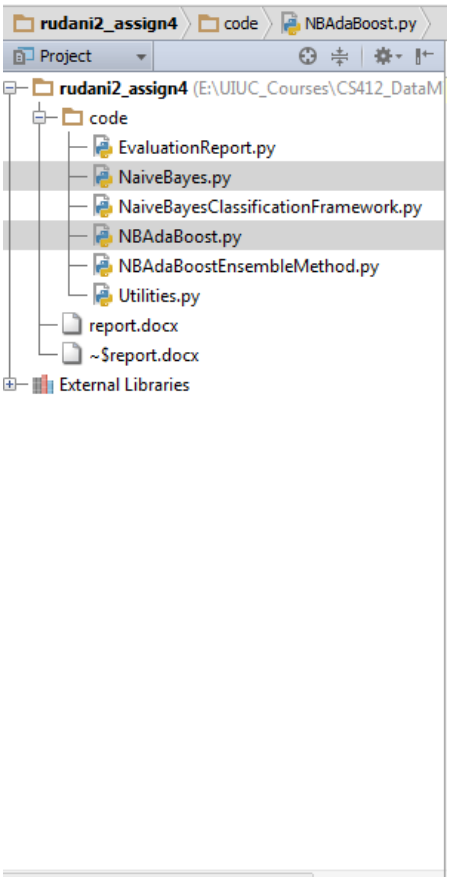
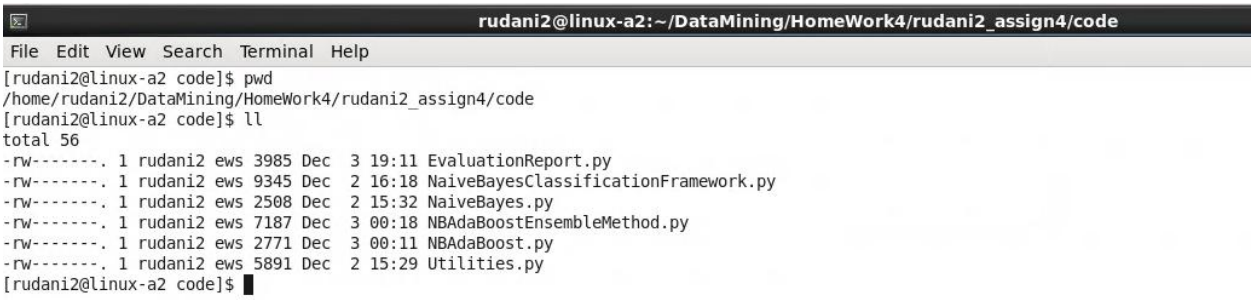
Machine Problem 4 Naïve Bayes and AdaBoost (Fall 2014)

Organization





rudani_assign4/

|----------report.pdf

|----------code/

|----------NaiveBayes.py # Main NaiveBayes code file

|----------NBAdaBoost.py # Main NBAdaBoost code file

 # Contains functions which are used while classification and prediction
|----------NaiveBayesClassificationFramework.py
 # Contains functions which are used while classification and prediction
|----------NBAdaBoostEnsembleMethod.py
 # Utilities contains common methods to both classifier
|----------Utilities.py
 # Evaluation contains methods which is used to calculate important metrics like TP/FN/FP/TP
|----------EvaluationReport.py

Classification method:

Primary Classification Method Employed: Naïve Bayesian Classification
Ensemble Method: Boosting via Adaptive Boosting

Naïve Bayesian classification is a weak learner. The AdaBoost algorithm iteratively works on the Naïve Bayesian classifier with normalized weights and it classifies the given input into different classes with some attributes. Because we train multiple Bayesian classifiers, each time hoping to improve the predictive ability of the current classifier, we hope to do better in each iteration

Algorithm

Step 1: Read in training dataset and test dataset, and store them in memory.

I have created in memory data structures like List and Dictionary to store attribute and its unique value. Entire input training and test file is modified into list of list like ***[[-1,0,0,1…],[+1,1,1,0,0,…]].*** Attribute value pair is like ***attribute : [{uniq1:label},{uniq2:label}...,{uniq n:label}}]***

Step 2: Implement Naïve Bayes

Naïve Bayes Algorithm

1. After parsing input we have all attributes along with their values in the memory.
2. I have calculated the probabilistic model for each attributes along with class label.
   Formula used: -
   Class label → ***_total_minus_one/_total_observation***
   _total_plus_one/_total_observation

Attributes → ***_total_count_of attribute_uniq_value/_total_count_of_class_label Where class_label is +1 and -1. Also applied wherever Laplacian required***

Functions → ***naivebayes_classifier() → to train the model***
_get_laplacian_flag() → to check whether to perform Laplacian
_get_probablity() → calculate the probability

3. For prediction I have used the probabilities calculated for each attribute from classifier phase.
   Functions → ***1. predict_label() → to predict class label for test tuple***
4. Finally based on predicted value True Positive, False Positive, True Negative, False Negative are calculated.
   Functions → ***1. count_metrics() → to calculate TP, FP, TN, FN***

NAÏVE BAYES OUTPUT

Dataset

adult.train/adult.test
python E:/UIUC_Courses/CS412_DataMining/rudani2_assign4/NaiveBayes.py
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\adult.train
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\adult.test

313 82 220 990
5814 1632 4221 19289

breast_cancer.train/breast_cancer.test
python E:/UIUC_Courses/CS412_DataMining/rudani2_assign4/NaiveBayes.py
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\breast_cancer.train
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\breast_cancer.test

27 29 20 104
13 16 12 65

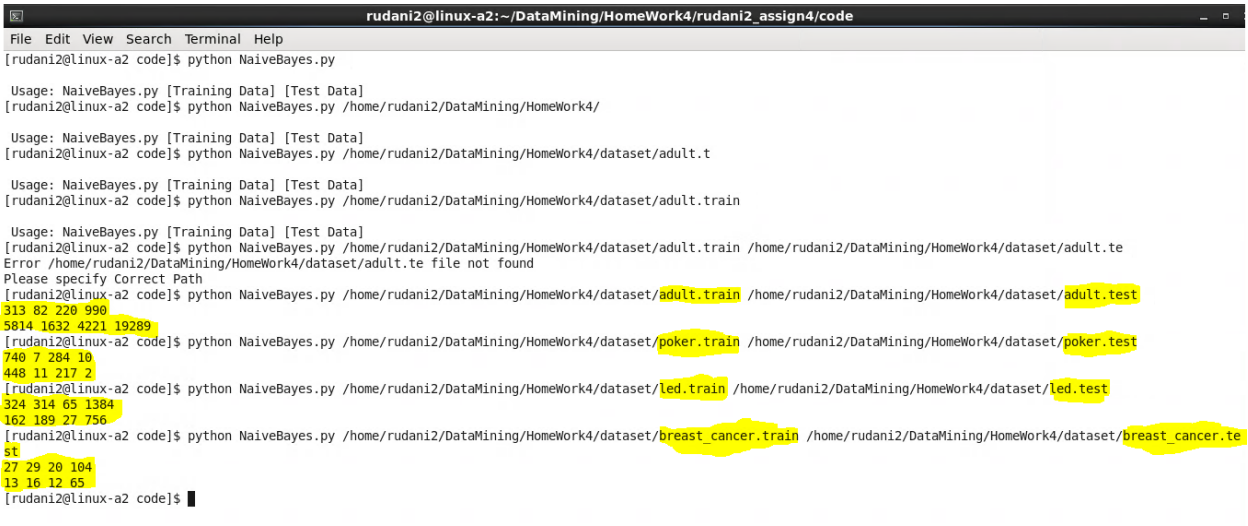
led.train/led.test
python E:/UIUC_Courses/CS412_DataMining/rudani2_assign4/NaiveBayes.py
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\led.train
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\led.test

324 314 65 1384
162 189 27 756

poker.train/poker.test
python E:/UIUC_Courses/CS412_DataMining/rudani2_assign4/NaiveBayes.py
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\poker.train
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\poker.test

740 7 284 10

448 11 217 2



Step 3: Implement AdaBoost

AdaBoost Algorithm

1. After parsing input we have all attributes along with their values in the memory.
2. Ada Boost is an Ensemble method which helps to boost the classifier.
3. It uses the technique of weights. It assigns increasing weights to wrong predicted tuples and decreases the weight of correctly predicted tuple.
4. Initially weights of all tuple is initialized as 1/D where D is size of training set
   Functions → _initial_weight() → to calculate initial weights
5. In order to train accurately random sample of tuples are selected. Weighted sample of tuple is used to randomly select the tuple.
   Functions → _get_random_sample() → to get random sample
6. Loop through k times to prepare k model.
7. **Choosing k →**
   k is the number of model created in classifier phase. Choose k such that model is not over fitted and also it gives accurate results. I have tried various k values like 8, 9, 10 and 16. I observed that the performance was degrading as number of iterations increased. The highest accuracy and performance was observed with k as 5.
8. If the error rate > 50% model is discarded. New model is created
9. If the error rate < 50% weight of corrected tuple is decreased
   Weight[i] = Weight[i] * (err_rate / 1 - err_rate)
10. Save the model which is then used in prediction phase
11. Finally normalize the weight so that sum of new weight is equal to 1
    Weight[i] = _weight_of_each_tuple[each_tuple]/_sum_of_weights
12. Prediction phase is used to predict the tuple with the help of model selected in classifier phase.

13. For each test tuple, each k model is used to predict the tuple value and weight of each model is calculated by

weight (α) → (1 – err_model)/(err_model)

14. Predict the tuple class label based on the sum of weight of each k model's prediction. If sum of weight of model for class label 1 > sum of weight of model for class label 2 predict class label 1 else class label 2

Function → nb_Adaboost_Predict() → to predict class label for test tuple

15. Finally based on predicted value True Positive, False Positive, True Negative, False Negative are calculated.

Functions → count_metrics() → to calculate TP, FP, TN, FN

AdaBoost OUTPUT

Dataset

adult.train/adult.test

python E:/UIUC_Courses/CS412_DataMining/rudani2_assign4/NBAdaBoost.py
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\adult.train
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\adult.test

295 100 206 1004
5592 1854 3850 19660

breast_cancer.train/breast_cancer.test

python E:/UIUC_Courses/CS412_DataMining/rudani2_assign4/NBAdaBoost.py
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\breast_cancer.train
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\breast_cancer.test

24 32 16 108
15 14 13 64

led.train/led.test

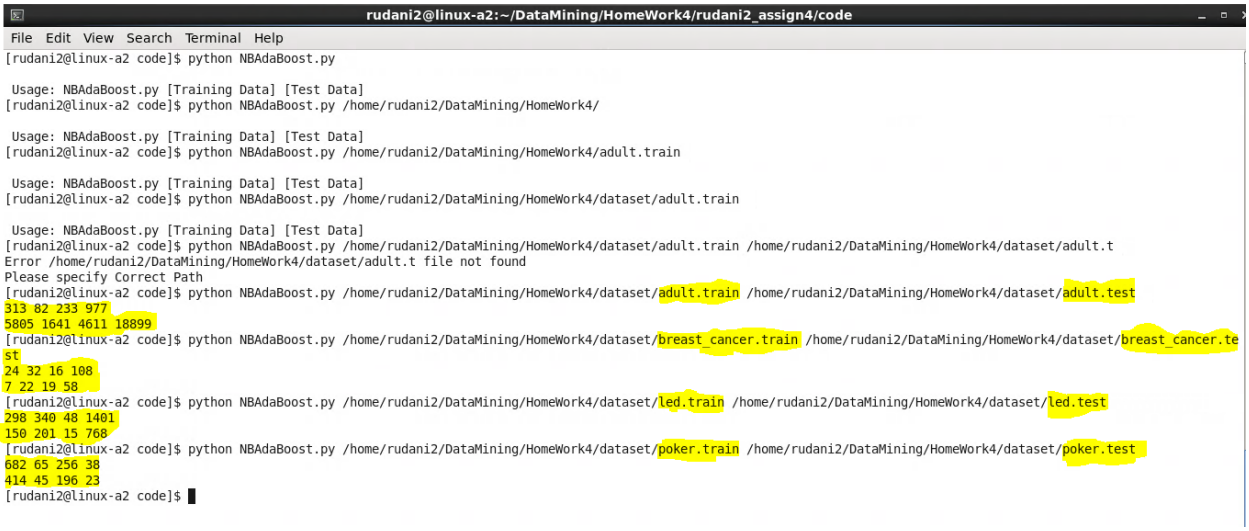
python E:/UIUC_Courses/CS412_DataMining/rudani2_assign4/NBAdaBoost.py
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\led.train
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\led.test

405 233 97 1352
212 139 42 741

poker.train/poker.test

python E:/UIUC_Courses/CS412_DataMining/rudani2_assign4/NBAdaBoost.py
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\poker.train
E:\UIUC_Courses\CS412_DataMining\MachineProblem4\dataset\poker.test

719 28 270 24
435 24 207 12



Step 4: Model Evaluation and Report

Following are the metrics for different data set

Naïve Bayes →

Data Set →

adult.train

Accuracy	0.811838006231
Error Rate	0.188161993769
Sensitivity	0.792405063291
Specificity	0.818181818182
Precision	0.587242026266
F1Score	0.674568965517
Fbeta (0.5)	0.619311436486
Fbeta (2)	0.740653099858

adult.test

Accuracy	0.810925184132
Error Rate	0.189074815868
Sensitivity	0.780821917808
Specificity	0.820459378988
Precision	0.579372197309

F1Score	0.665179337567
Fbeta (0.5)	0.610893960409
Fbeta (2)	0.730053492052

breast_cancer.train

Accuracy	0.727777777778
Error Rate	0.272222222222
Sensitivity	0.482142857143
Specificity	0.838709677419
Precision	0.574468085106
F1Score	0.52427184466
Fbeta (0.5)	0.553278688525
Fbeta (0.2)	0.49815498155

breast_cancer.test

Accuracy	0.735849056604
Error Rate	0.264150943396
Sensitivity	0.448275862069
Specificity	0.844155844156
Precision	0.52
F1Score	0.481481481481
Fbeta (0.5)	0.503875968992
Fbeta (0.2)	0.460992907801

led.train

Accuracy	0.818399616675
Error Rate	0.181600383325
Sensitivity	0.507836990596
Specificity	0.955141476881
Precision	0.832904884319
F1Score	0.630963972736
Fbeta (0.5)	0.73837739289
Fbeta (0.2)	0.550833049983

led.test

Accuracy	0.809523809524
Error Rate	0.190476190476
Sensitivity	0.461538461538
Specificity	0.965517241379
Precision	0.857142857143
F1Score	0.6

Fbeta (0.5)	0.731707317073
Fbeta (0.2)	0.508474576271

poker.train

Accuracy	0.720461095101
Error Rate	0.279538904899
Sensitivity	0.9906291834
Specificity	0.0340136054422
Precision	0.72265625
F1Score	0.835686053077
Fbeta (0.5)	0.763989262854
Fbeta (0.2)	0.9222333001

poker.test

Accuracy	0.663716814159
Error Rate	0.336283185841
Sensitivity	0.976034858388
Specificity	0.00913242009132
Precision	0.673684210526
F1Score	0.797153024911
Fbeta (0.5)	0.718178903495
Fbeta (0.2)	0.895641743303

AdaBoost →

Data Set →

adult.train

Accuracy	0.809345794393
Error Rate	0.190654205607
Sensitivity	0.746835443038
Specificity	0.829752066116
Precision	0.588822355289
F1Score	0.658482142857
Fbeta (0.5)	0.614839516465
Fbeta (0.2)	0.708793849111

adult.test

Accuracy	0.815738467502
Error Rate	0.184261532498
Sensitivity	0.751007252216
Specificity	0.836239897916
Precision	0.592247405211
F1Score	0.662245381336

Fbeta (0.5)	0.618392533286
Fbeta (0.2)	0.712792535563

breast_cancer.train

Accuracy	0.733333333333
Error Rate	0.266666666667
Sensitivity	0.428571428571
Specificity	0.870967741935
Precision	0.6
F1Score	0.5
Fbeta (0.5)	0.555555555556
Fbeta (0.2)	0.454545454545

breast_cancer.test

Accuracy	0.745283018868
Error Rate	0.254716981132
Sensitivity	0.51724137931
Specificity	0.831168831169
Precision	0.535714285714
F1Score	0.526315789474
Fbeta (0.5)	0.531914893617
Fbeta (0.2)	0.520833333333

led.train

Accuracy	0.841878294202
Error Rate	0.158121705798
Sensitivity	0.634796238245
Specificity	0.933057280883
Precision	0.806772908367
F1Score	0.710526315789
Fbeta (0.5)	0.765306122449
Fbeta (0.2)	0.663064833006

led.test

Accuracy	0.840388007055
Error Rate	0.159611992945
Sensitivity	0.603988603989
Specificity	0.946360153257
Precision	0.834645669291
F1Score	0.700826446281
Fbeta (0.5)	0.775420629115
Fbeta (0.2)	0.639324487334

poker.train

Accuracy	0.713736791547
Error Rate	0.286263208453
Sensitivity	0.962516733601
Specificity	0.0816326530612
Precision	0.726996966633
F1Score	0.828341013825
Fbeta (0.5)	0.76440569849
Fbeta (0.2)	0.903947699271

poker.test

Accuracy	0.659292035398
Error Rate	0.340707964602
Sensitivity	0.947712418301
Specificity	0.0547945205479
Precision	0.677570093458
F1Score	0.790190735695
Fbeta (0.5)	0.718533201189
Fbeta (0.2)	0.877723970944

Observation

The Ensemble method Ada Boosting produced mixed results with different datasets. In most however the performance deprecated slightly in cohort of the ensemble method. Above result certainly makes it clear that Naïve Bayesian classification and Ada boosting are not Ensemble Compatible. Decision Trees based classification with Ada boosting are Ensemble compatible.