Data Mining

Assignment3

Spyridon Roumpis

181004877

Exercise 0: Compare the Number of Leaves and Size of Tree of both trees (i.e. with and without pruning) and explain any differences observed.

- With Pruning

Number of Leaves : 61

Size of the tree : 93

Correctly Classified Instances	210	90.5172 %
Incorrectly Classified Instances	22	9.4828 %
Kappa statistic	0.8954	
Mean absolute error	0.0155	
Root mean squared error	0.0929	
Relative absolute error	16.1763 %	
Root relative squared error	42.4372 %	
Total Number of Instances	232	

- Without Pruning

Number of Leaves : 121

Size of the tree : 175

Correctly Classified Instances	201	86.6379 %
Incorrectly Classified Instances	31	13.3621 %
Kappa statistic	0.8529	
Mean absolute error	0.0157	
Root mean squared error	0.1072	
Relative absolute error	16.2992 %	
Root relative squared error	48.9674 %	
Total Number of Instances	232	

It can be seen from the two above pictures that both of them, the number of leaves and the size of the tree is almost half when pruning is used, accounting for 61 and 93 respectively, compared to the number of leaves -121 and the size of the tree -175 without pruning.

This result is something we were expecting to happen, as we know from theory that Pruning reduces the size of decision trees by removing parts of the tree that provide little power to classify instances.

Exercise 1: Compare the Test Accuracy of both trees. Which tree shows a better performance? Explain your observation based on the notion of tree pruning.

The correctly classified instances for the pruned tree are 210, giving a percentage of 90.5172% while for the unpruned tree are 201 with 86.6379%. Again, this is something we were expecting to happen, the pruned tree to have better performance than the unpruned,

The idea behind pruning is that, apart from reducing the size of the tree and the complexity of the classifier, improving predictive accuracy by the reduction of overfitting. This can be seen also in our results.

Exercise 2: Which class is being heavily mis-classified? Why has this happened?

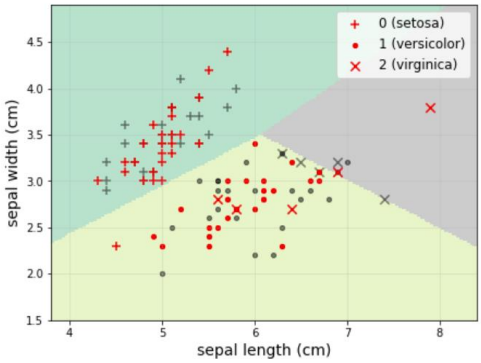
For this exercise we changed the number of instances for class 2, flower type 2 (virginica Iris), from 25 to 5. Below it can be seen the confusion matrixes for train, test, normalised train and normalised test instances.

```
Train accuracy = 0.909
Train confusion matrix:
 [[25  0  0]
 [ 0 23  2]
 [ 0  3  2]]

Test accuracy = 0.927
Test confusion matrix:
 [[24  1  0]
 [ 0 25  0]
 [ 0  3  2]]

Normalised train confusion matrix:
 [[1.   0.   0.  ]
 [0.   0.92 0.08]
 [0.   0.6  0.4 ]]

Normalised test confusion matrix:
 [[0.96 0.04 0.  ]
 [0.   1.   0.  ]
 [0.   0.6  0.4 ]]
```



The class which was heavily mis-classified was the class 2, flower type 2 (virginica Iris), the class we decided to reduced its instances. This happened as there were not enough data for the training set for this class.

Exercise 3: Obtain the accuracy for this class from the test dataset and identify the other class that it is being confused with.

It can be seen from the normalized test confusion matrix the accuracy for the mis-classified class, class 2, is only 0.4 while more than half of the instances of the test dataset of class 2, flower type 2 (virginica Iris), are confused and labelled as class 1, flower type 1 (versicolor Iris) (0.6 accuracy).

Exercise 4: What is the new accuracy for class 2 (virginica)? Compare this accuracy with the accuracy obtained in the previous section and explain any discrepancies.

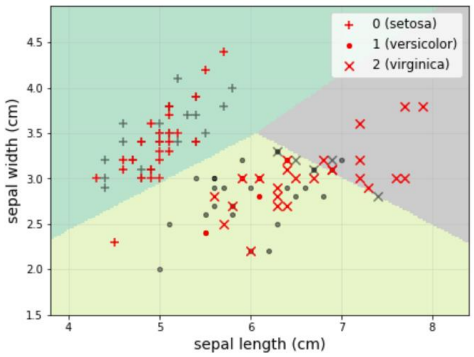
Now let's assume our classifier (which was trained using data of region A) is taken to another geographic region B, where the flower proportions are different again, specifically flower type 1 (versicolor Iris) is 5 times less common.

```
Train accuracy = 0.909
Train confusion matrix:
 [[25  0  0]
 [ 0 23  2]
 [ 0  3  2]]

Test accuracy = 0.727
Test confusion matrix:
 [[24  1  0]
 [ 0  5  0]
 [ 0 14 11]]

Normalised train confusion matrix:
 [[1.   0.   0.  ]
 [0.   0.92 0.08]
 [0.   0.6  0.4 ]]

Normalised test confusion matrix:
 [[0.96 0.04 0.  ]
 [0.   1.   0.  ]
 [0.   0.56 0.44]]
```



Now it can be seen that the accuracy has improved compared with the accuracy before. The new accuracy is 0.44. The reason that we see that is because the instances of the data set increased from 5 to 25 for class 2 (virginica) while the instances for class one (versicolor Iris) decreased 5 time less.

Exercise 5: What prior should you use to get maximum accuracy in region B? What accuracy do you get by using this value?

- prior = [1/3,1/3,1/3]

```
Train accuracy = 0.873
Train confusion matrix:
 [[25  0  0]
 [ 0 18  7]
 [ 0  0  5]]

Test accuracy = 0.836
Test confusion matrix:
 [[24  1  0]
 [ 0 21  4]
 [ 0  4  1]]

Normalised train confusion matrix:
 [[1.   0.   0.  ]
 [0.   0.72 0.28]
 [0.   0.   1.  ]]

Normalised test confusion matrix:
 [[0.96 0.04 0.  ]
 [0.   0.84 0.16]
 [0.   0.8  0.2 ]]
```

- prior = [0.05,0.9,0.05]

```
Train accuracy = 0.782
Train confusion matrix:
 [[18  7  0]
 [ 0 24  1]
 [ 0  4  1]]

Test accuracy = 0.727
Test confusion matrix:
 [[15 10  0]
 [ 0 25  0]
 [ 0  5  0]]

Normalised train confusion matrix:
 [[0.72 0.28 0.  ]
 [0.   0.96 0.04]
 [0.   0.8  0.2 ]]

Normalised test confusion matrix:
 [[0.6 0.4 0. ]
 [0.  1.  0. ]
 [0.  1.  0. ]]
```

To obtain the maximum accuracy we used a very good prior – 0.9 for the dataset of region B, although it improved the accuracy of that class it reduced both the accuracy of the training and the test. The class 2 is almost 100% misclassified and class 0 behaved significantly worse than before. Selecting high prior value for class 1 while really low for the other classes did not perform well for our classifier, yes it only performed well only for class 1. Balancing the values of the prior for the three classes in this example gives us a better accuracy for our classifier.

Exercise 6: Compare the performance of both classifiers in the 2-feature scenario with the performance in the 200-feature scenario and explain any differences you might observe.

- **2-feature scenario**

  Train accuracy (Logistic Regression): 0.78
  Test accuracy (Logistic Regression): 0.72

  Train accuracy (Naive Bayes): 0.78
  Test accuracy (Naive Bayes): 0.72

- **200-feature scenario**

  Train accuracy (Logistic Regression): 1.00
  Test accuracy (Logistic Regression): 0.78

  Train accuracy (Naive Bayes): 1.00
  Test accuracy (Naive Bayes): 1.00

New information is extracted in terms of the new features we are using. These features may have a higher ability to explain the variance in the training data. Thus, giving improved model accuracy. The more we know the better it performs, although there should always be a balance, as we don't want to overfit.