TP2 : Decision Trees – Maxime TCHIBOZO

The main goal of this lab session is to handle decision trees, and illustrate the trade-off between training error and number of leaves.

Question 1:

Cf. notebook

To create the initial Decision Tree, we collect the training and target data and put them in pandas DataFrames.
We collect our labels from Skyserver.csv (which contains all the training data with the column names). This file also contains the columns : 'class' (whether the row corresponds to a STAR, GALAXY or QSO) and the 'objid' which are not present in the training data. We need to remove those two labels to ensure the number of labels is the same as the number of columns in the training data.

We create the decision tree with sklearn's DecisionTreeClassifier method and then save it to a PDF file.

```
In [63]: print "training errors:"+str((1-clf.score(sky_training_data,sky_training_target)))
         print "Number of nodes in the tree:" +str(clf.tree_.node_count)
         print "Generalized error: " + str(generalized_error(clf))

training errors:0.0
Number of nodes in the tree:203
Generalized error: 51.0
```

The resulting decision tree makes no mistakes on the training data but is considerably large. We will now see how we can improve it.

This tree is available in the pdf file : **skyDefault.pdf**

Question 2:

To improve the tree we introduce the notion of generalized error. To code this, we calculate the total number of leaves by identifying the number of instances of the item -1 in the array clf.tree_.children_left.

We have 102 leaves in the original tree!

This is problematic: we can say with confidence that the model has overfitted: it is too close to the training data, and will probably not work very well on any exterior test data that we have not yet analyzed. We want our model to be as general as possible, and to work well with unknown data. We must decrease the total number of leaves.

Question 3:

To improve this tree, we will add a constraint in the DecisionTreeClassifier method : we limit the number if keaves by adding a max_leaf_nodes amount.
We create a program which initially starts with a max_leaf_nodes of 102 (we know this number can only decrease). At each iteration, we reduce the max_leaf_nodes by 1 and save the resulting tree if the computed generalized error has been reduced.

Since the generalized error is proportional to the number of nodes, we can assume that it is also strongly correlated with the number of leaves. By constraining the number of nodes, we also limit the number of leaves, and thus, we forcibly decrease the error. This is true so long as the training error does not increase too much.

This program necessarily terminates, and will return the tree which yielded the minimal generalized error.

```
#graphv2 = graphviz.Source(dot_data)
clf2 =get_minimal_generalized_error_tree()
print "training errors:"+str((1-clf2.score(sky_training_data,sky_training_target)))
print "Number of nodes in the tree:" +str(clf2.tree_.node_count)
print "Generalized error: " + str(generalized_error(clf2))
```

```
training errors:0.011301130113011301126
Number of nodes in the tree:5
Generalized error: 1.5113011301130113
```

This new tree has only FIVE Nodes! While the training error is no longer equal to 0, we can consider it to be negligible. The trade-off between training accuracy and generality of the model is no longer a problem.

The obtained decision tree is available in the skyOptimal.pdf file.

Question 4 :

We can confidently say that the decision tree created at question 3 performs much better than the initial decision tree. It classifies celestial objects based on the value of the redshift with very high accuracy and is highly adaptable: it will perform well on test data that has not yet been seen in training.

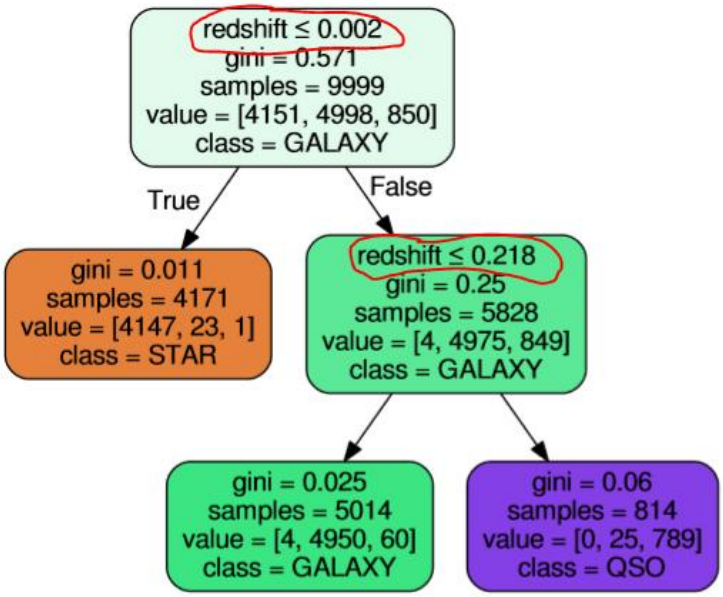
The initial decision tree, however would not perform very well on data it has not yet previously been trained on.

We should note however, that the original decision tree always performs better on data from training because it has a training error of 0, whereas the optimal tree has a training error of 0.011

We must also remember that the algorithm which determines the optimal tree is quite slow, and calculates about a hundred different decision trees.

Question 5 :

The optimal decision tree tells us that we can almost always identify an object by looking at the redshift.



Question 6:

The optimal decision tree has only three leaves, it is impossible to prune because we have exactly three target names: [STAR, GALAXY, QSO]. If we removed a leaf, we would no longer be able to distinguish two of the three object types.

Question 7:

We cannot post-prune our optimal tree as it has only three leaves!!