Project 3
CSL 302 (Artificial Intelligence)
Team:

Prateek Singh (2011CS1028)
Rajeev Kumar (2011CS1031)

Part 1: ID3 Decision Tree Done by: Rajeev Kumar

How to run the code

1) Compile the c++ code using g++.
2) data.txt file, which contains the dataset should the present in the same directory as the code.
3) Now run the exectuable file.

Output format

The code randomly splits the dataset into 5 mutually exclusive stratified sets.

a) Now the code uses one of these sets as test data and other 4 are used for training and calculates and prints the accuracy on test set.

b) Step a) is performed 5 times each time using a different set as test data set.

c) Finally the code prints the average of the five accuracies calculated.

Now the code code asks for the percentage of noise to be added to the dataset. Enter the appropriate value.

The code will give the Accuracy after adding the noise.

Results:

Accuracy on five fold varification without adding noise = 100%
Accuracy on five fold varification with 1% noise = 98.47%
Accuracy on five fold varification with 5% noise = 91.65%
Accuracy on five fold varification with 10% noise = 87.06%
Accuracy on five fold varification with 15% noise = 81.24%

*Note that the accuracies can vary a bit in different run of the code. This is because on each run of the algorithms training dataset and test dataset are chosen randomly.

Observations:

1) If the training dataset is partitioned into 5 mutually exlusive and stratified sets. Then 4 of the are used for training and 1 for test validation, then accuracy comes out to be 100% most of the time.
2) As the percentage of the noise increases, the size of the tree increases and the accuracy on test data set decreases.

How missing attribute values have been handled in the code

1) During the training if the missing attribute value is observed , then the most common value of that attribute among other examples of the same target value has been assigned.
2) During parsing if a missing value is observed, then the parser checks for all the values of

the attribute ie checking all the children of the node and returns the most common value predicted by the children.