Q5

Understand the working of SLIQ and ARBC classifiers. Give a pseudocode and illustrate the same over a sample dataset of your choice.

1. SLIQ Classifier

Introduction:

Supervised Learning in Quest is a decision tree classifier that can handle both numeric and categorical attributes. SLIQ uses a pre-sorting technique in the tree-growth phase to reduce the cost of evaluating numeric attributes. This sorting procedure is integrated with a breadth-first tree growing strategy to enable SLIQ to classify disk-resident datasets. In addition, SLIQ uses a fast sub setting algorithm for determining splits for categorical attributes. SLIQ also uses a new tree-pruning algorithm based on the Minimum Description Length principle. This algorithm is inexpensive, and results in compact and accurate trees. The combination of these techniques enables SLIQ to scale for large data sets and classify data sets with a large number of classes, attributes, and examples.

Pseudo Code:

Key Features,

1. Tree Classifier, handling numeric and categoric attributes
2. Presorting numeric attributes before tree has been built
3. Breadth first growing strategy
4. Goodness test – Gini Index
5. Inexpensive tree pruning algorithm based on Minimum Description Length (MDL)

In Presorting,

1. Eliminate need for sorting data at each node
2. Create sorted list for each numeric attribute
3. Create class list

Split Evaluation,

EvaluateSplits()

For each attribute A do

Traverse attribute list of A

For each value v in attribute list do

Find the corresponding entry in the class list, and hence the corresponding class and the leaf node l

Update the class histogram in leaf l

If A is a numeric attribute then

Compute splitting index for test (A <= v) for l

If A is a categorical attribute then

For each leaf of the tree do

Find subset of A with best split

Update Class List,

UpdateLabels()

For each attribute A used in a split do

Traverse attribute list of A

For each value v in attribute list do

Find the corresponding entry in the class list e

Find the new class c to which v belongs by applying the splitting test at node referenced

from e

Update the class label for e to c

Update node referenced in e to the child corresponding to the class c

Trace: