BASIC DECISION TREE ALGORITHM

BIJALI JAYALAKSHMI JAYAN

Bijali

- The DT approach is most useful in classification problems.
- With this technique, a tree is constructed to model the classification process.
- Once the tree is built, it is applied to each tuple in the db and results in a classification for that tuple.
- There are 2 basic steps in technique:
  Building the tree and
  Applying tree to db

  DT induction is the learning of decision from class labelled training tuples.
  A DT is a structure that includes a root node, branches, and leaf nodes. Each internal node denotes a test on an attribute, each branch denotes the outcome of a test, and each leaf node holds a class label. The topmost node in the tree is the root node.
  Node is represented as feature(attribute).
  Each link(branch) is a decision(rule)
  And each leaf is an outcome.

  A DT is a flowchart tree like structure that is made from training set tuples. The dataset is broken down into smaller subsets and is present in the form of nodes of a tree. The tree structure has a root node, internal node or decision nodes, leaf node and branches.
  The root node is the topmost node. It represents the best attribute selected for classification. Internal nodes (non leaf node) of the decision nodes represent a test of an attribute of the dataset leaf node or terminal node which represents the classification or decision class label. The branches show the outcomes of the test performed.
  Some DT only have binary nodes, that means exactly 2 branches of a node, while some DT are non binary.
  Internal node – rectangle
  Leaf node-oval

Advantages

- DTs certainly are easy to use and efficient.
- Rules can be generated that are easy to interpret and understand.
- They scale well for large db bcz the tree size is independent of db size.

Disadvantages

- Do not easily handle continuous data.
- Handling missing data is difficult bcz correct branches in tree could not be taken.
- Since DT is constructed from training data, overfitting may occur.
- This can be overcome via tree pruning.

Benefits

- It does not require any domain knowledge.

- It is easy to comprehend.
- The learning and classification steps of a DT are simple and fast.

HOW DT USED FOR CLASSIFICATION

- Given a tuple X for which the associated class label is unknown, the attribute value of the tuple are tested against the DT
- A path is traced from the root to a leaf node, which holds the class prediction for that tuple.
- DT can easily be converted to classification rules.
- The DT induction is so popular bcz the construction of DT classifiers does not require any domain knowledge or parameter setting.

  DT generation consists of 2 phases:
  Tree construction : at start all the training example are at the root. Partition examples recursively based on selected attributes.
  Tree pruning: identify and remove branches that reflect noise or outliers.

Use of DT: classifying an unknown sample. Also test the attribute values of the sample against the DT.

the following DT is for the concept buys_computer that indicates whether a customer is likely to buy a computer or not. Each internal nodes represents a test on an attribute. Each leaf node represents a class.

A ML researcher named J.Ross Quinlan in 1980 developed a DT algorithm known as ID3(Iterative Dichotomiser). Later, he presented C4.5, which was the successor of ID3. ID3 and C4.5 adopt a greedy approach. In this algorithm, there is no backtracking. The tree constructed in a top down recursive D&C manner.

How to select Attributes for Creating Tree ?

The most popular methods of selecting the attributes are information gain (Entropy), Gini index, and gain ratio.

Attribute selection measures are also called setting rules to decide how the tuples are going to split.

The splitting criteria are used to best partition the dataset.

This measure provides a ranking to the attributes for partitioning the training tuples.

Basic algm (Greedy algorithm)

- Tree is constructed in a top down (from general to specific) recursive D&C manner.
- At start, all the training examples are at the root.
- Attributes are categorical (if continuous values, discretized in advance).
- Examples are partitioned recursively based on selected attributes.
- Attributes are selected based on heuristic or statistical measure (information gain).

When to stop

- All examples for a given node belomg to same class (pure) or
- No remaining attributes to select from or
- Majority voting to determine class label for node.
- No examples left.

Algorithm: Generate_decision_tree. Generate a decision tree from the training tuples of data partition, *D*.

Input:

- Data partition, *D*, which is a set of training tuples and their associated class labels;
- *attribute_list*, the set of candidate attributes;
- *Attribute_selection_method*, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting_attribute* and, possibly, either a *split-point* or *splitting subset*.

Output: A decision tree.

Method:

```
(1)  create a node N;
(2)  if tuples in D are all of the same class, C, then
(3)       return N as a leaf node labeled with the class C;
(4)  if attribute_list is empty then
(5)       return N as a leaf node labeled with the majority class in D; // majority voting
(6)  apply Attribute_selection_method(D, attribute_list) to find the "best" splitting_criterion;
(7)  label node N with splitting_criterion;
(8)  if splitting_attribute is discrete-valued and
          multiway splits allowed then // not restricted to binary trees
(9)       attribute_list ← attribute_list — splitting_attribute; // remove splitting_attribute
(10) for each outcome j of splitting_criterion
     // partition the tuples and grow subtrees for each partition
(11)      let Dj be the set of data tuples in D satisfying outcome j; // a partition
(12)      if Dj is empty then
(13)           attach a leaf labeled with the majority class in D to node N;
(14)      else attach the node returned by Generate_decision_tree(Dj, attribute_list) to node N;
     endfor
(15) return N;
```

ID 3

- The ID3 technique to buildings a DT based on information theory and attempts to minimize the expected number of comparisons.
- The basic strategy used by ID3 is to choose splitting attributes with the highest information gain first.
- The concept used to quantify information is called Entropy.
- Entropy is used to measure the amount of uncertainty or surprise or randomness in a set of data.
- Certainly, when all data in a set belong to a single class, there is no uncertainty. In this case entropy is zero.
- The objective of DT classification is to iteratively partition the given data set into subsets where all elements in each final subset belong to same class.

C4.5

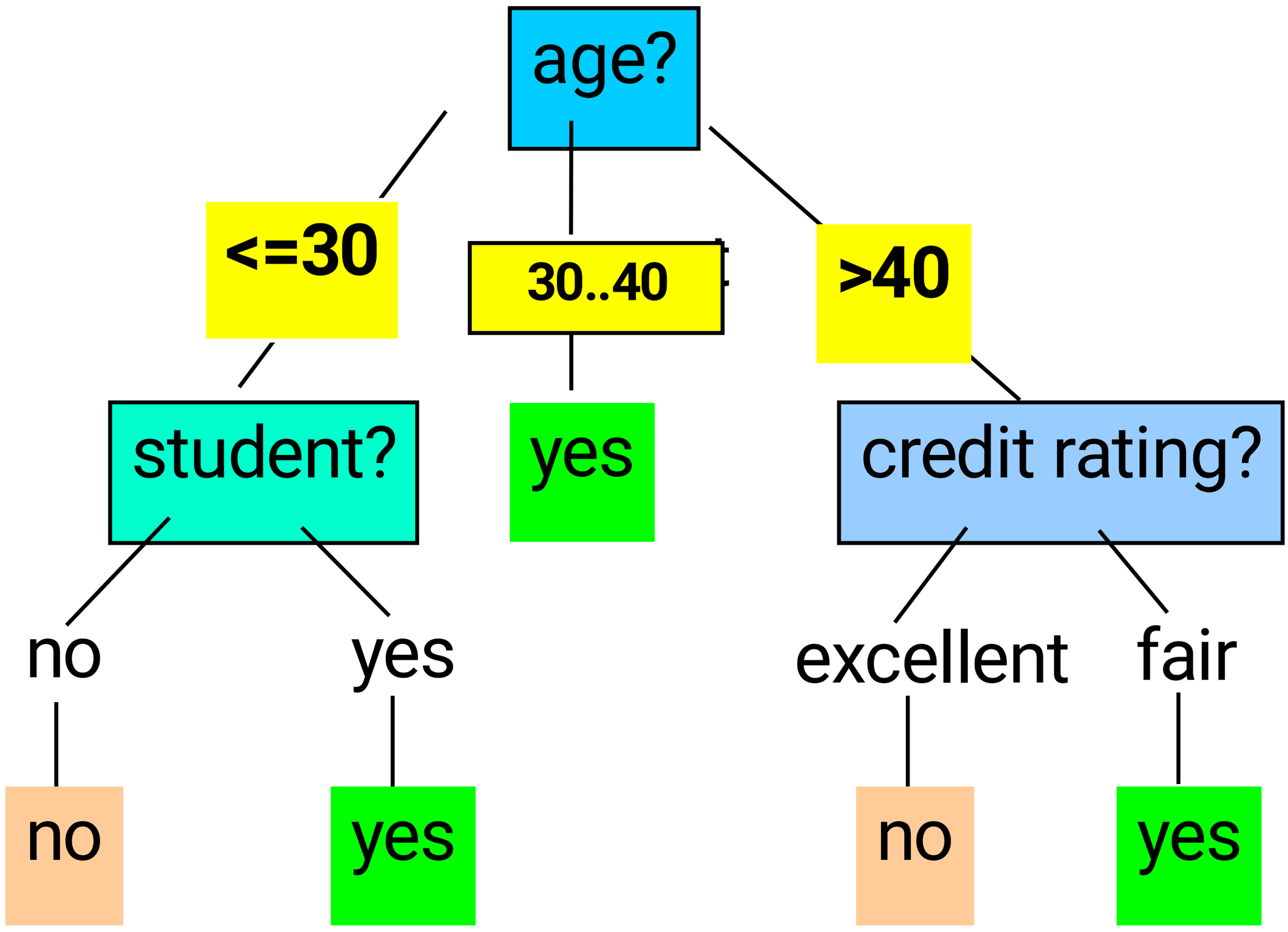
DT algm C4.5 improves ID3:

Missing data

Continuous data

Pruning

age	income	student	credit_rating	buys_computer
youth	high	no	fair	no
youth	high	no	excellent	no
middle-aged	high	no	fair	yes
senior	medium	no	fair	yes
senior	low	yes	fair	yes
senior	low	yes	excellent	no
middle-aged	low	yes	excellent	yes
youth	medium	no	fair	no
youth	low	yes	fair	yes
senior	medium	yes	fair	yes
youth	medium	yes	excellent	yes
middle-aged	medium	no	excellent	yes
middle-aged	high	yes	fair	yes
senior	medium	no	excellent	no


age?
<=30
30..40
>40
student?
yes
credit rating?
no
yes
excellent
fair
no
yes
no
yes