Classification

Shubham kumar
Dept. of CS&IT
MGCUB

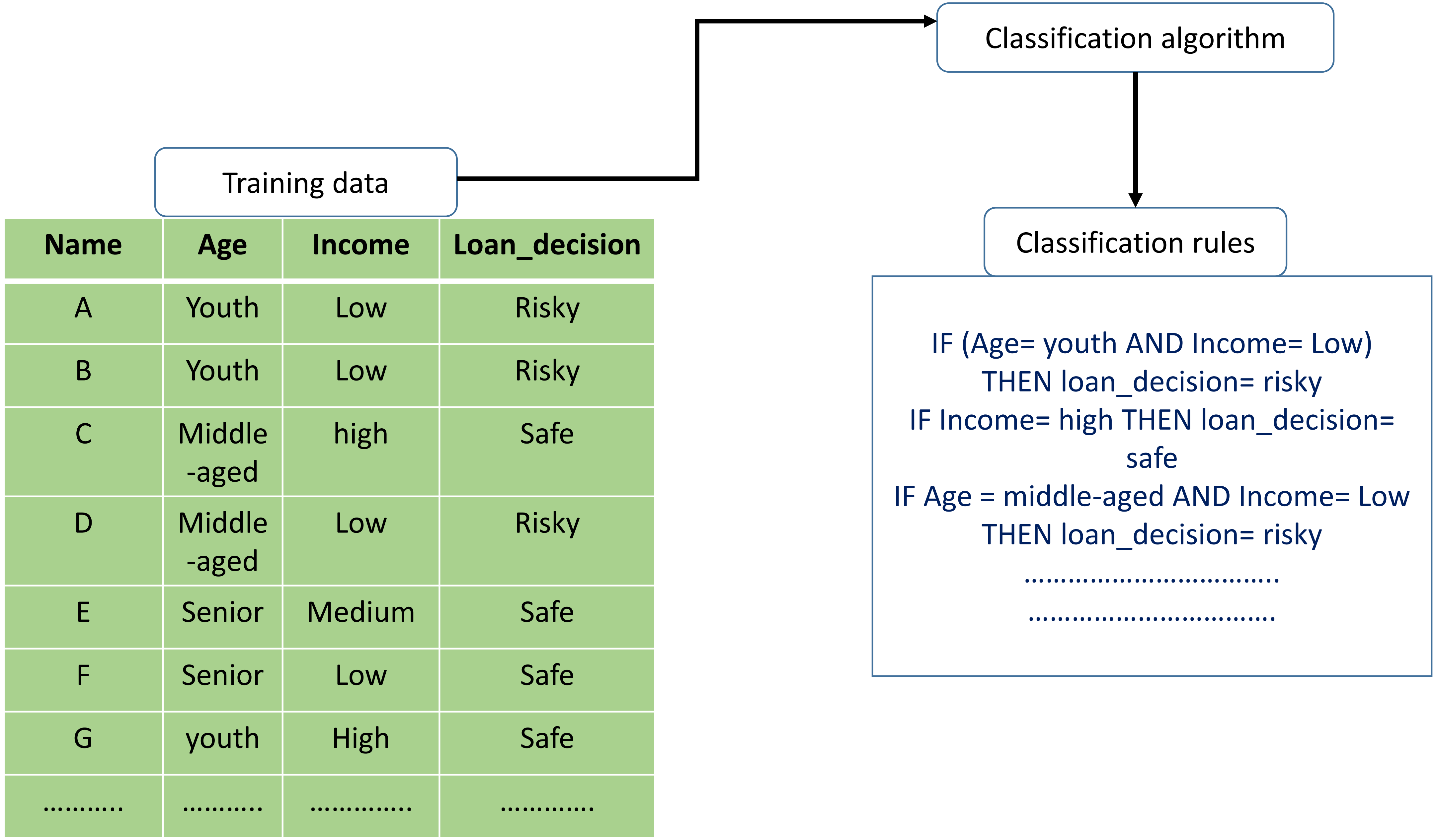
Classification

- It is a form of data analysis that extracts models describing important data classes.
- For example a classification model can be build to categorize bank loan applications as either safe or risky.
- Classification model has many applications like performance prediction, fraud detection, medical diagnosis and target marketing etc.
- Suppose a marketing manager at Allelectronics (a store in our example) needs data analysis to help guess whether a customer with a given profile will buy a new computer. Then the data analysis task is classification, where a **model or classifier** is designed to predict **class labels** “YES” or “NO”.
- These class labels are of discrete type and also the ordering among the values has no meaning.

- If the marketing manager wants to predict how much a given customer will spend during a sale at Allelectronic, then this data analysis task is called **numeric prediction**.
- In the numeric prediction, the model constructed is called **predictor** and it predicts a continuous-valued function, or ordered value.
- **Regression analysis** is used for numeric predictions.
- Hence we can say that there are two main types of prediction problems: **Classification** and **Numeric prediction**.

- Data Classification process can be mainly divided into two steps: **the learning step** and **the classification step.**
- In the first step **(learning step)**, a classification algorithm is used to build the classification model or classifier by analyzing or “learning from” a **training set** made up of database tuples and their associated class labels.

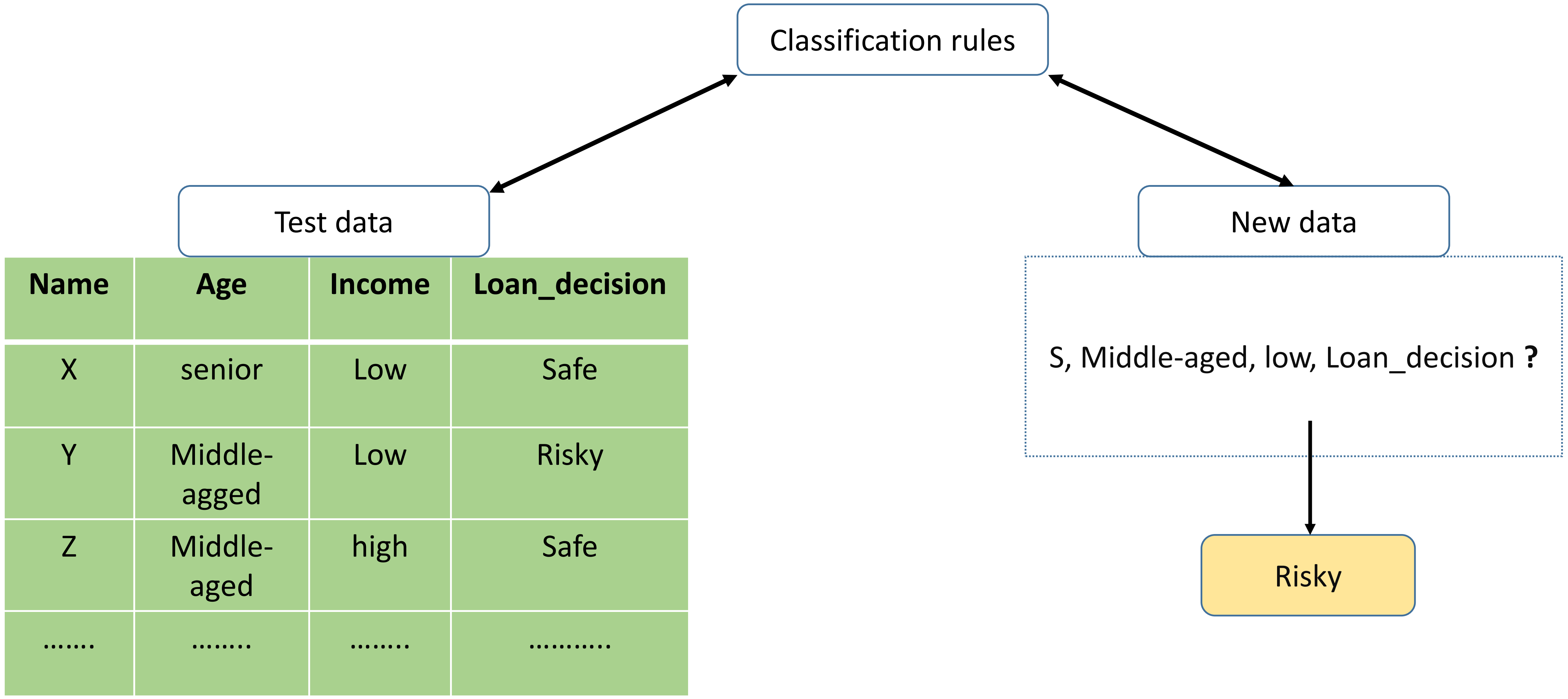
- The tuples of the training set are referred to an **training tuples** and are randomly sampled from the database under analysis.
- Since in the classification, the class label of each training tuple is provided, this step is also known as **Supervised learning**.
- The first step of the classification process can also be viewed as the learning of a mapping of function, y=f(X), that can predict associated class label y of a given tuple X.
- The second step (**classification step**) is used to predict the class labels for given data by using our constructed classifier.
- **Example:** let us take an example of loan application data where a classification model is to be constructed to categorize bank loan applications as either safe or risky.
- In the learning step (shown in next slide), training data are analyzed by a classification algorithm. Here the class label attribute is loan_decision. And the learned model is represented in the form of classification rules.



Name	Age	Income	Loan_decision
A	Youth	Low	Risky
B	Youth	Low	Risky
C	Middle -aged	high	Safe
D	Middle -aged	Low	Risky
E	Senior	Medium	Safe
F	Senior	Low	Safe
G	youth	High	Safe
………..	………..	…………..	………….

IF (Age= youth AND Income= Low)
THEN loan_decision= risky
IF Income= high THEN loan_decision=
safe
IF Age = middle-aged AND Income= Low
THEN loan_decision= risky
……………………………….
…………………………….

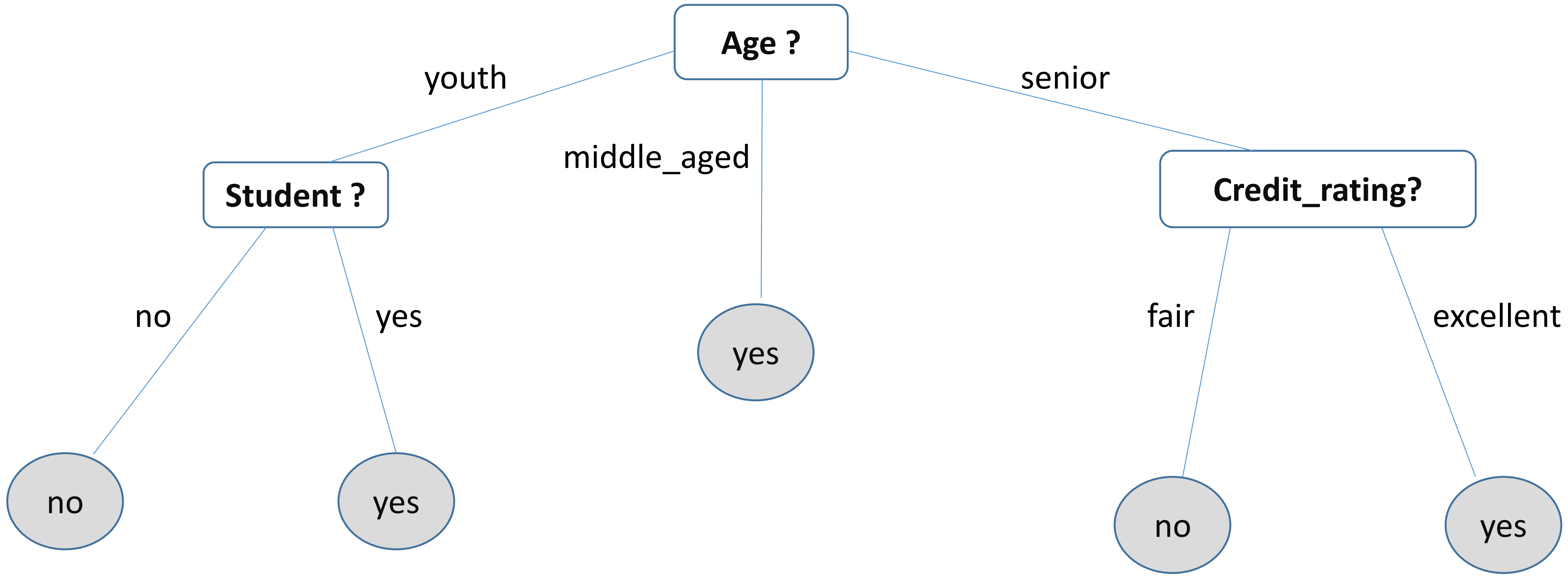
Now, in the second step, the test data are used to estimate the accuracy of the classification rules. If the accuracy satisfies the threshold, the rules can be applied to the classification of new data tuples.



Name	Age	Income	Loan_decision
X	senior	Low	Safe
Y	Middle-aged	Low	Risky
Z	Middle-aged	high	Safe
…….	……..	……..	……….

- In our example, we have seen that the classification rule learned can be used to approve or reject the new loan application.
- **Decision Tree Induction** : Decision tree induction is the learning of decision trees from the class-labeled training tuples.
- **Decision tree:** It is a flowchart like tree structure, where each internal node denotes a test, branches represents the outcome of the test and each leaf node denotes a class label. The top most node is called the root node.
- Example:

Let us take a decision tree to predict whether a customer at Allelectronics is likely to purchase a computer.



Internal nodes are represented by rectangles, and leaf nodes are denoted by ovals. Each internal node represents a test on an attribute. Each leaf node represents a class (either buys_computer = yes (or) buys_computer = no).

- The attribute values of the given tuple (whose associated class label is unknown) are tested against the decision tree. And the path is traced from the root to a leaf node, which holds the class prediction for that tuple.

- Decision trees can easily be converted to classification rules.

- Decision tree may be binary (each internal node has exactly two other nodes) or nonbinary.

- Some advantages of decision trees are:

  The construction of decision tree classifiers does not require any domain knowledge.

It can handle multidimensional data. Their representation of acquired knowledge in tree form is easy to assimilate by humans.

The learning and classification steps of decision tree induction are simple and fast.

It is able to handle both numeric and categorical attributes.

Types of decision trees:

- **Categorical Variable Decision Tree:** Decision Tree which has a categorical target variable then it called a Categorical variable decision tree**.**

- **Continuous Variable Decision Tree:** Decision Tree has a continuous target variable then it is called Continuous Variable Decision Tree.

Extracting classification rules from Decision tree:

1. Represent the knowledge in the form of IF-THEN rules.
2. One rule is created for each path from the root to a leaf.
3. Each attribute-value pair along a path forms a conjuction.
4. The leaf node holds the predicted class label.

In our example, the classification rules will be-

IF age= “youth” AND student= “no” **THEN** buys_computer= “no”

IF age= “youth” AND student= “yes” **THEN** buys_computer= “yes”

IF age = “middle_aged” **THEN** buys_computer= “yes”

IF age= “senior” AND credit_rating= “fair” **THEN** buys_computer= “no”

IF age= “senior” AND credit_rating= “excellent” **THEN** buys_computer= “yes”

Iterative Dichotomiser (ID3)

- It is a decision tree algorithm developed by J.Ross Quinlan in the early 1980s.
- It is a greedy approach in which decision trees are constructed in a top-down recursive divide-and- conquer manner.
- Top down approach starts with a training set of tuples and their associated class labels.
- The training set is recursively partitioned into smaller subsets as the tree is being built.
- The algorithm is called with 3 parameters: D (the data partition), Attribute_list, and Attribute_selection_method.
- Initially the D is the complete set of training tuples and their associated class labels.
- Attribute_list is a list of attributes describing the tuples.

- Attribute_selection_method is the procedure for selecting the attribute that best discriminates the given tuples according to class. This procedure employs an attribute selection measure such as information gain or the Gini index.

[Attribute selection measure will be discussed later]

The tree starts as a single node, N, representing the training tuples in D.

Assumption:

In the beginning, the whole training set is considered as the **root.**

Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.

Algorithm: (Generate decision tree)

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.

```
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 labelled 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 i.e. most common class in D.
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 // not restricted to binary trees
9.     THEN   attribute_list ← attribute_list − splitting attribute; // remove splitting attribute
10.   For each outcome j of splitting criterion, let $D_j$ be the set of data tuples in D satisfying outcome j
                  //partition the tuples and grow subtrees for each partition
11.   IF $D_j$ is empty then
12.     attach a leaf labeled with the majority class in D to node N; // majority class means most common class
13.   ELSE attach the node returned by Generate decision tree($D_j$ , attribute_list) to node N;
      EndFor
14.   return N;
```

Reference

- Jiawei Han, Micheline kamber and Jian pei. “DATA MINING concepts and Techniques” 3/e, Elsevier, 2012