Decision Tree

Ying Shen, SSE, Tongji University

Classification

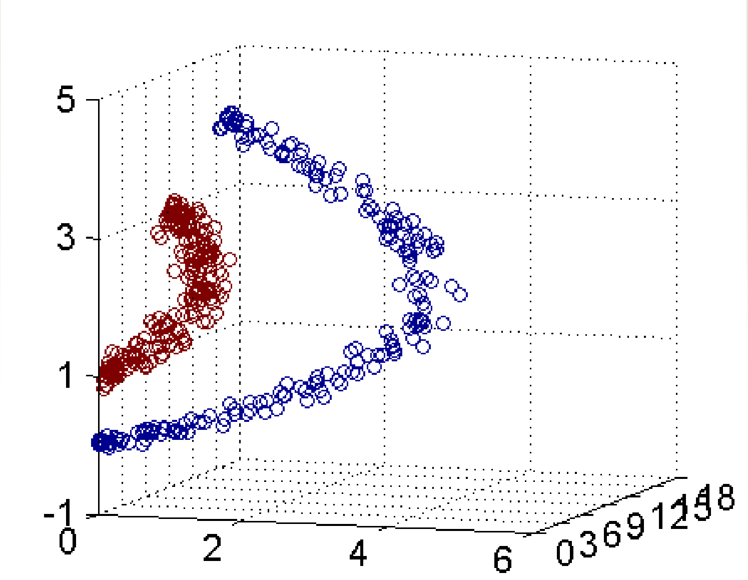
- Classification is the task of assigning objects to one of several predefined categories.
- It is an important problem in many applications
  - ❖ Detecting spam email messages based on the message header and content.
  - ❖ Categorizing cells as malignant or benign based on the results of MRI scans.
  - ❖ Classifying galaxies based on their shapes.

Classification

- The input data for a classification task is a collection of records.
- Each record, also known as an instance or example, is characterized by a tuple $(\mathbf{x}, y)$
- $\mathbf{x}$ is the attribute set
- y is the class label, also known as category or target attribute.
- The class label is a discrete attribute.

Classification

- Classification is the task of learning a target function f that maps each attribute set $\mathbf{x}$ to one of the predefined class labels y.
- The target function is also known informally as a classification model.



Classification

- A classification technique (or classifier) is a systematic approach to perform classification on an input data set.
- Examples include
  - Decision tree classifiers
  - Neural networks
  - Support vector machines

Classification

- A classification technique employs a learning algorithm to identify a model that best fits the relationship between the attribute set and the class label of the input data.
- The model generated by a learning algorithm should
  - Fit the input data well and
  - Correctly predict the class labels of records it has never seen before.
- A key objective of the learning algorithm is to build models with good generalization capability.

Classification

- First, a training set consisting of records whose class labels are known must be provided.
- The training set is used to build a classification model.
- This model is subsequently applied to the test set, which consists of records which are different from those in the training set.

Confusion matrix

- Evaluation of the performance of the model is based on the counts of correctly and incorrectly predicted test records.
- These counts are tabulated in a table known as a confusion matrix.
- Each entry f_{ij} in this table denotes the number of records from class i predicted to be of class j.

Confusion matrix

		Predicted Class	
		Class=1	Class=0
Actual Class	Class=1	f_{11}	f_{10}
	Class=0	f_{01}	f_{00}

Confusion matrix

- The total number of correct predictions made by the model is $f_{11}+f_{00}$.
- The total number of incorrect predictions is $f_{10}+f_{01}$.

Confusion matrix

- The information in a confusion matrix can be summarized with the following two measures
  - Accuracy

$$Accuracy = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

  - Error rate

$$Error\ rate = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

- Most classification algorithms aim at attaining the highest accuracy, or equivalently, the lowest error rate when applied to the test set.

Decision tree

- We can solve a classification problem by asking a series of carefully crafted questions about the attributes of the test record.
- Each time we receive an answer, a follow-up question is asked.
- This process is continued until we reach a conclusion about the class label of the record.

Decision tree

- The series of questions and answers can be organized in the form of a decision tree.
- It is a hierarchical structure consisting of nodes and directed edges.
- The tree has three types of nodes
  - A root node that has no incoming edges, and zero or more outgoing edges.
  - Internal nodes, each of which has exactly one incoming edge and two or more outgoing edges.
  - Leaf or terminal nodes, each of which has exactly one incoming edge and no outgoing edges.

Decision tree

- In a decision tree, each leaf node is assigned a class label.
- The non-terminal nodes, which include the root and other internal nodes, contain attribute test conditions to separate records that have different characteristics.

Decision tree

- Classifying a test record is straightforward once a decision tree has been constructed.
- Starting from the root node, we apply the test condition.
- We then follow the appropriate branch based on the outcome of the test.
- This will lead us either to
  - Another internal node, for which a new test condition is applied, or
  - A leaf node.
- The class label associated with the leaf node is then assigned to the record.

Decision tree construction

- Efficient algorithms have been developed to induce a reasonably accurate, although suboptimal, decision tree in a reasonable amount of time.
- These algorithms usually employ a greedy strategy that makes a series of locally optimal decisions about which attribute to use for partitioning the data.

Decision tree construction

- A decision tree is grown in a recursive fashion by partitioning the training records into successively purer subsets.
- We suppose
  - U_n is the set of training records that are associated with node n.
  - $C=\{c_1, c_2, \ldots, c_K\}$ is the set of class labels.

Decision tree construction

- If all the records in U_n belong to the same class c_k, then n is a leaf node labeled as c_k.
- If U_n contains records that belong to more than one class,
  - An attribute test condition is selected to partition the records into smaller subsets.
  - A child node is created for each outcome of the test condition.
  - The records in U_n are distributed to the children based on the outcomes.
- The algorithm is then recursively applied to each child node.

Decision tree construction

- For each node, let $p(c_k)$ denotes the fraction of training records from class k.
- In most cases, the leaf node is assigned to the class that has the majority number of training records.
- The fraction $p(c_k)$ for a node can also be used to estimate the probability that a record assigned to that node belongs to class k.

Decision tree construction

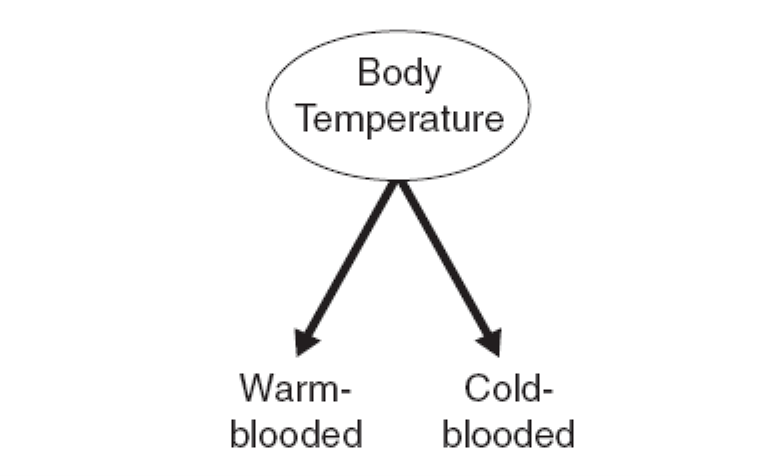
- Decision trees that are too large are susceptible to a phenomenon known as overfitting.
- A tree pruning step can be performed to reduce the size of the decision tree.
- Pruning helps by trimming the tree branches in a way that improves the generalization error.

Attribute test

- Each recursive step of the tree-growing process must select an attribute test condition to divide the records into smaller subsets.
- To implement this step, the algorithm must provide
  - A method for specifying the test condition for different attribute types and
  - An objective measure for evaluating the goodness of each test condition.

Attribute test

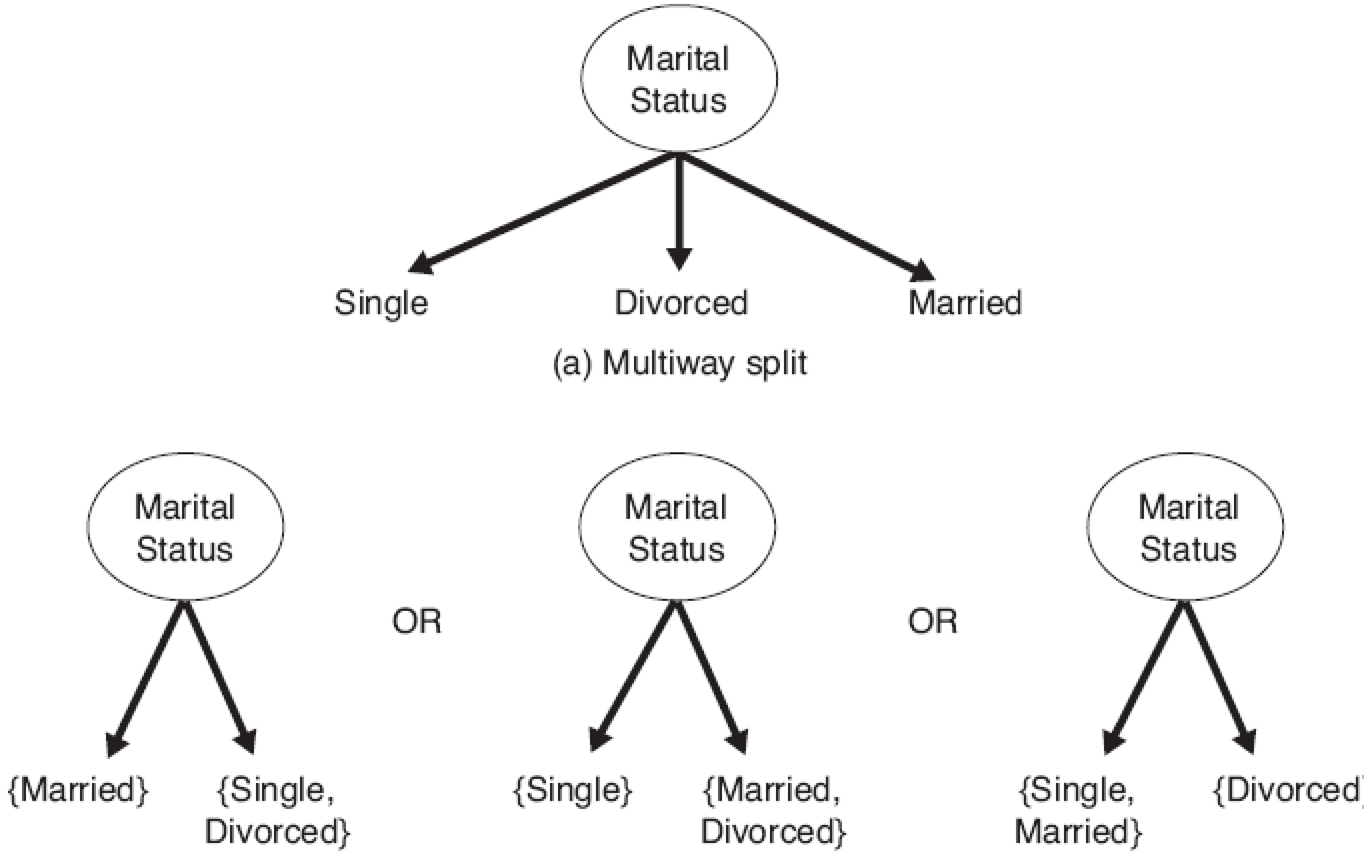
- Binary attributes
  - The test condition for a binary attribute generates two possible outcomes.



Attribute test

- Nominal attributes
  - A nominal attribute can produce binary or multiway splits.
  - There are $2^{N\text{-}1}$-1 ways of creating a binary partition of N attribute values.
  - For a multiway split, the number of outcomes depends on the number of distinct values for the corresponding attribute.

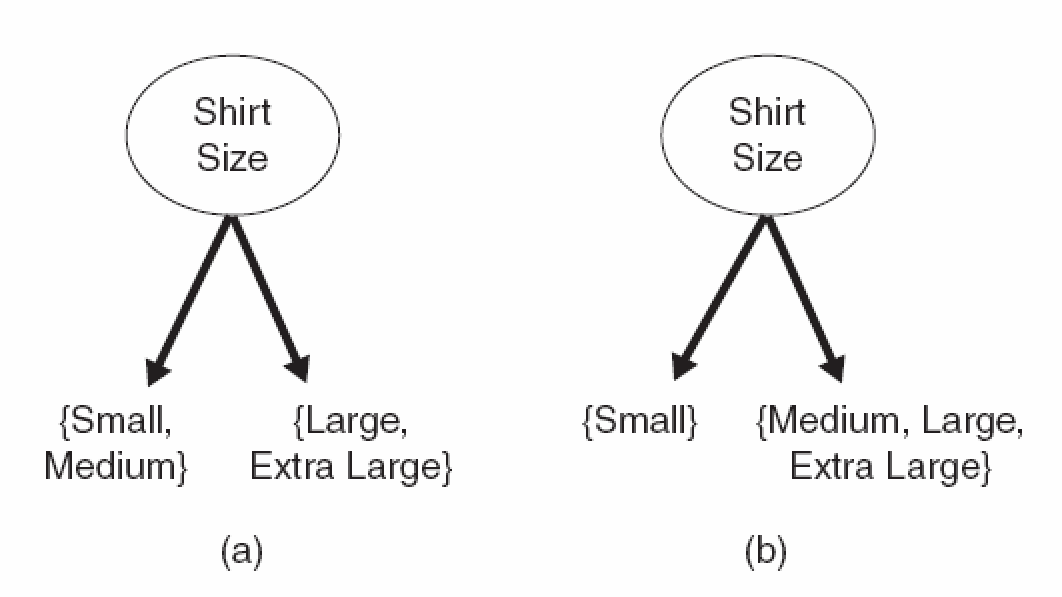
Attribute test



(b) Binary split {by grouping attribute values}

Attribute test

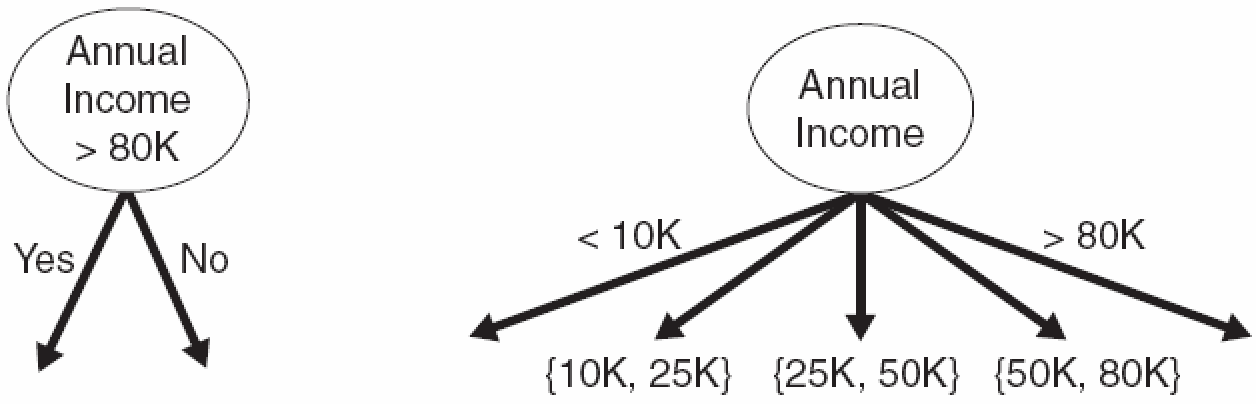
- Ordinal attributes
  - Ordinal attributes can also produce binary or multiway splits.
  - Ordinal attributes can be grouped as long as the grouping does not violate the order property of the attribute values.



Attribute test

- Continuous attributes
  - The test condition can be expressed as a comparison test $v \leq T$ or $v > T$ with binary outcomes, or
  - A range query with outcomes of the form $T_j \leq v < T_{j+1}$, for $j = 1, \ldots, J$
  - For the binary case
    - The decision tree algorithm must consider all possible split positions T, and
    - Select the one that produces the best partition.
  - For the multiway split,
    - The algorithm must consider multiple split positions.

Attribute test



(a)

(b)

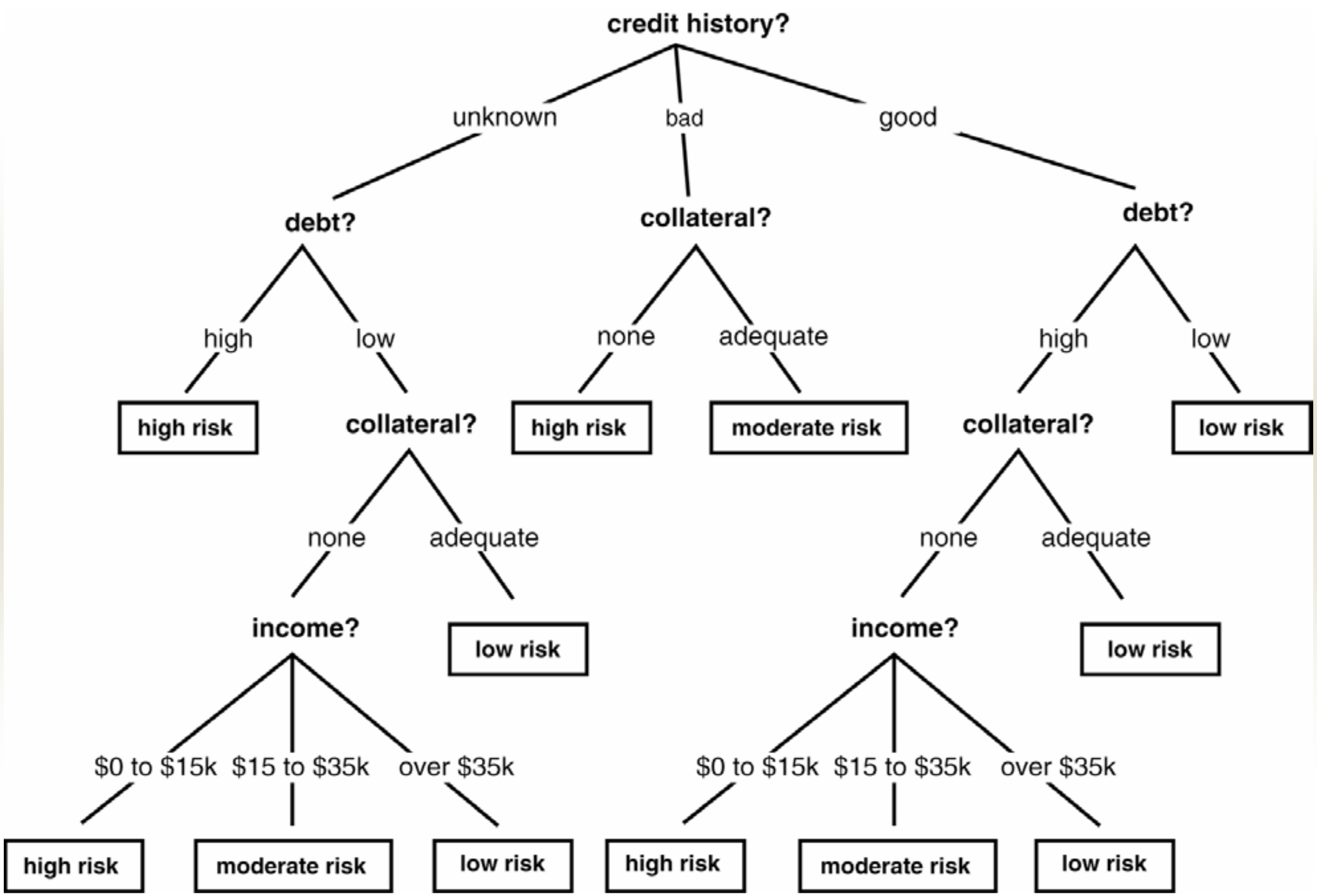
Decision tree construction: Example

- Example: credit risk estimation
- An individual's credit risk depends on such attributes as **credit history, current debt, collateral** and **income**.
- For this example, there exists a decision tree which can correctly classify all the objects.

Decision tree construction: Example

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	$0 to $15k
2.	high	unknown	high	none	$15 to $35k
3.	moderate	unknown	low	none	$15 to $35k
4.	high	unknown	low	none	$0 to $15k
5.	low	unknown	low	none	over $35k
6.	low	unknown	low	adequate	over $35k
7.	high	bad	low	none	$0 to $15k
8.	moderate	bad	low	adequate	over $35k
9.	low	good	low	none	over $35k
10.	low	good	high	adequate	over $35k
11.	high	good	high	none	$0 to $15k
12.	moderate	good	high	none	$15 to $35k
13.	low	good	high	none	over $35k
14.	high	bad	high	none	$15 to $35k

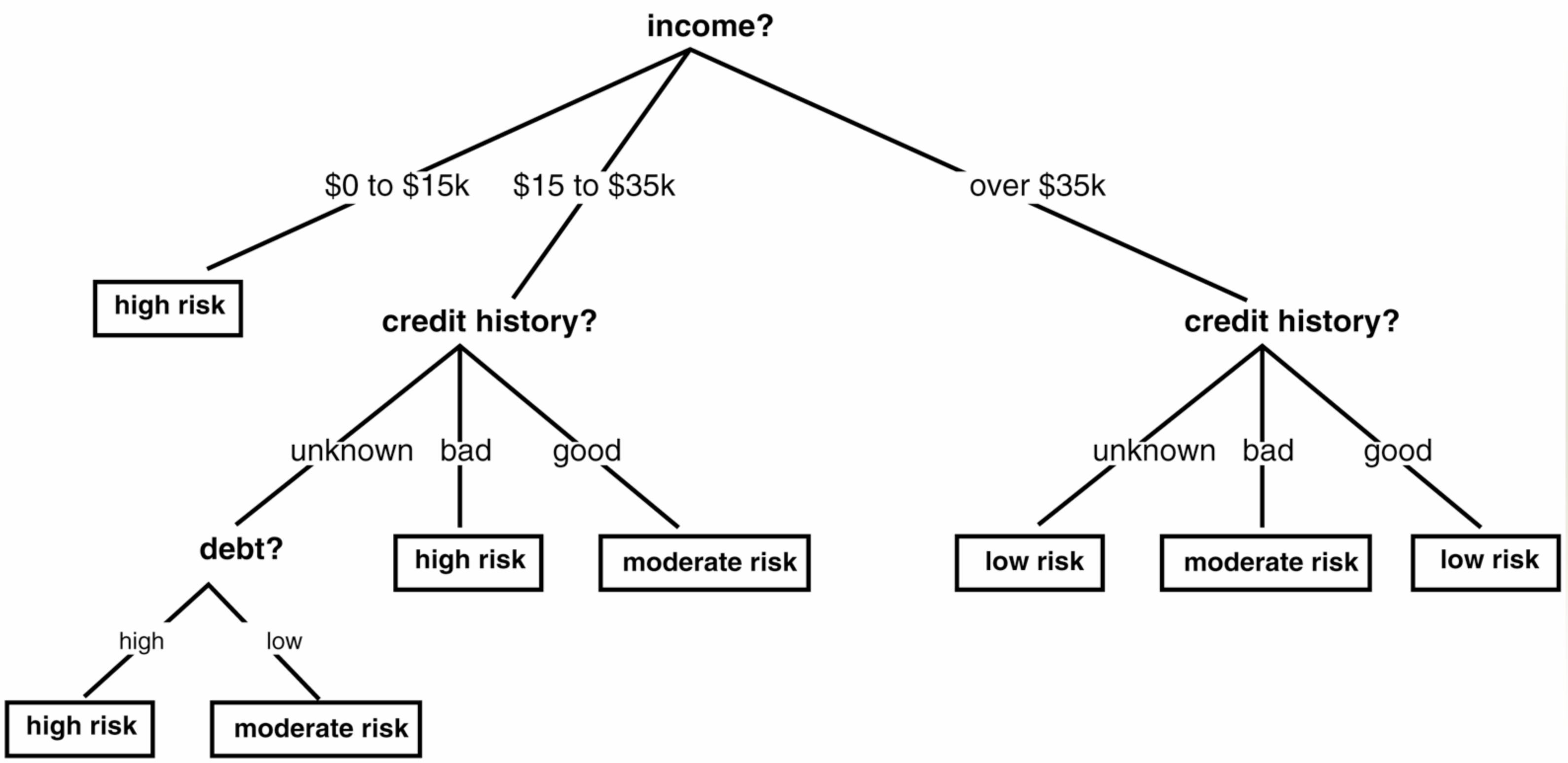
Decision tree construction: Example



Decision tree construction: Example

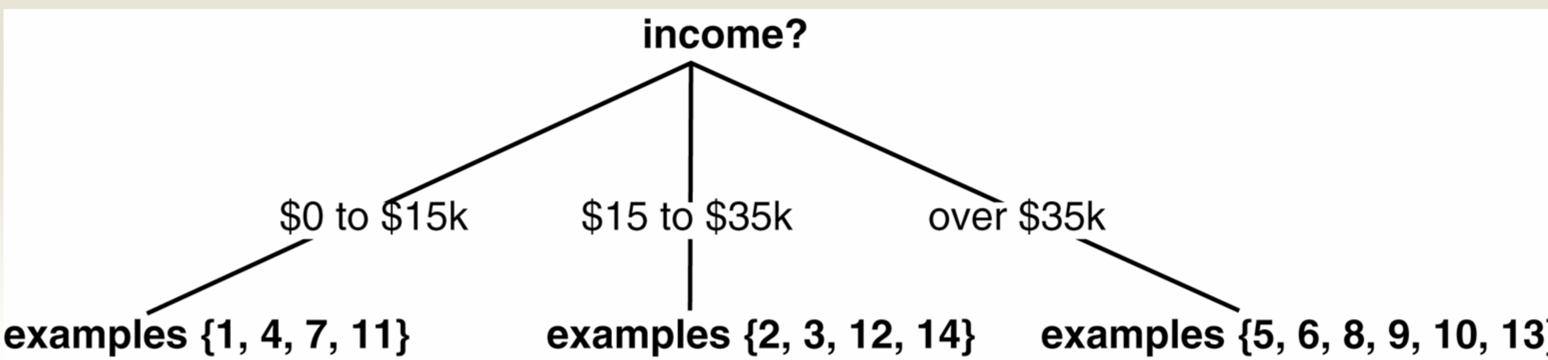
- In a decision tree, each internal node represents a test on some attribute, such as **credit history** or **debt**.
- Each possible value of that attribute corresponds to a branch of the tree.
- Leaf nodes represent classifications, such as **low** or **moderate risk**.

Decision tree construction: Example



Decision tree construction: Example

- Suppose income is selected as the root attribute to be tested.
- This partitions the example set as shown in the figure.

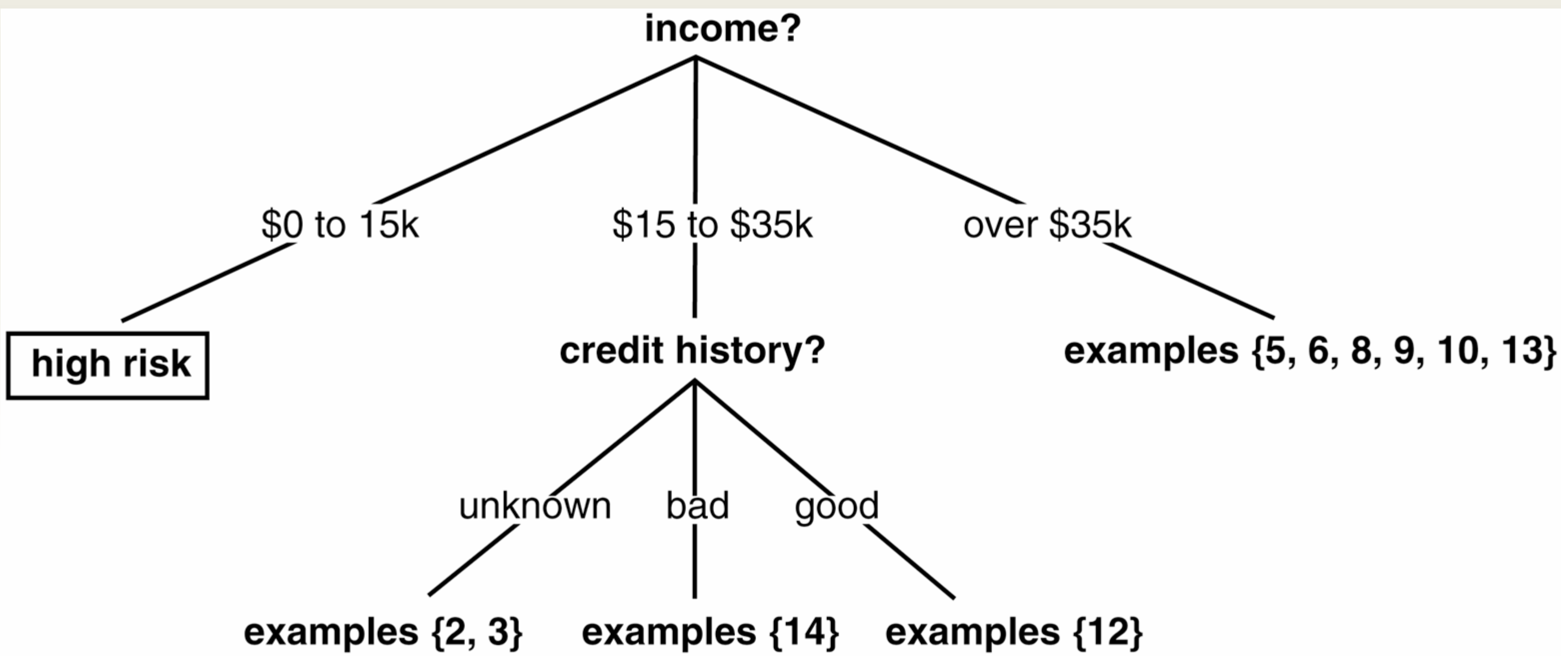


Decision tree construction: Example

- Since the partition {1,4,7,11} consists entirely of high-risk individuals, a leaf node is created.

Decision tree construction: Example

- For the partition {2,3,12,14}
  - **credit history** is selected as the attribute to be tested.
  - This further divides this partition into {2,3}, {14} and {12}.



Information theoretic test

- Each attribute of an instance contributes a certain amount of information to the classification process.
- We measure the amount of information gained by the selection of each attribute.
- We then select the attribute that provides the greatest information gain.

Information theoretic test

- Information theory provides a mathematical basis for measuring the information content of a message.
- We may think of a message as an instance in a collection of possible messages.
- The information content of a message depends on
  - The size of this collection
  - The frequency with which each possible message occurs.

Information theoretic test

- The amount of information in a message with occurrence probability p is defined as -$\log_2 p$.
- Suppose we are given
  - a collection of messages, $C=\{c_2, c_2,\ldots.., c_K\}$
  - the occurrence probability $p(c_k)$ for each c_k.
- We define the entropy I as the expected information content of a message in C:

$$I = -\sum_{k=1}^{K} p(c_k) \log_2 p(c_k)$$

- The information is measured in bits.

Information theoretic test

- We can measure the information content of a set of training instances from the probabilities of occurrences of the different classes.
- In our example
  - p(high risk)=6/14
  - p(moderate risk)=3/14
  - p(low risk)=5/14

Information theoretic test

- The set of training instances is denoted as U
- We can calculate the information content of the tree using the previous equation

$$
\begin{aligned}
I(U) &= -\frac{6}{14}\log_2\left(\frac{6}{14}\right) - \frac{3}{14}\log_2\left(\frac{3}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) \\
&= -\frac{6}{14}(-1.222) - \frac{3}{14}(-2.222) - \frac{5}{14}(-1.485) \\
&= 1.531 \text{ bits}
\end{aligned}
$$

Information theoretic test

- The information gain provided by making a test at a node is the difference between
  - The amount of information needed to complete the classification before performing the test.
  - The amount of information needed to complete the classification after performing the test.
- The latter is defined as the weighted average of the information in all its subtrees.

Information theoretic test

- If we select attribute P, with N values, this will partition U into subsets $\{U_1, U_2, \ldots, U_N\}$.
- The average information required to complete the classification after selecting P is

$$\bar{I}(P) = \sum_{n=1}^{N} \frac{|U_n|}{|U|} I(U_n)$$

Information theoretic test

- The information gain from attribute P is computed as follows.
  - $gain(P) = I(U) - \bar{I}(P)$
- If the attribute **income** is chosen, the examples are partitioned as follows:
  - {1,4,7,11}
  - {2,3,12,14}
  - {5,6,8,9,10,13}

Information theoretic test

- The expected information needed to complete the classification is

$$\bar{I}(income) = \frac{4}{14} I(U_1) + \frac{4}{14} I(U_2) + \frac{6}{14} I(U_3)$$
$$= \frac{4}{14}(0.0) + \frac{4}{14}(1.0) + \frac{6}{14}(0.650)$$
$$= 0.564 \text{ bits}$$

Information theoretic test

- The information gain can be computed as follows:

$$
\begin{aligned}
gain(income) &= I(U) - \bar{I}(income) \\
&= 1.531 - 0.564 \\
&= 0.967 \text{ bits}
\end{aligned}
$$

Information theoretic test

- Similarly, we can show that
  - gain(credit history)=0.266
  - gain(debt)=0.063
  - gain(collateral)=0.207
- The attribute **income** will be selected, since it provides the greatest information gain.

Continuous attributes

- If attribute P has continuous numeric values v, we can apply a binary test.
- The outcome of the test depends on a threshold value T.
- There are two possible outcomes:
  - $v \leq T$
  - $v > T$
- The training set is then partitioned into 2 subsets U_1 and U_2.

Continuous attributes

- We apply sorting to values of attribute P to result in the sequence $\{v_1, v_2, \ldots, v_R\}$.
- Any threshold between v_r and v_{r+1} will divide the set into two subsets
  - $\{v_1, v_2, \ldots, v_r\}$
  - $\{\mathrm{v}_{r+1}, v_{r+2}, \ldots, v_R\}$
- There are R-1 possible splits.

Continuous attributes

- For $r = 1,\ldots, R$-1, the corresponding threshold is chosen as $T_r = (v_r+v_{r+1})/2$.
- We can then evaluate the gain in information for each T_r

$$gain(P,T_r) = I(U) - \bar{I}(P,T_r)$$

where $\bar{I}(P,T_r)$ is a function of T_r.

- The threshold T_r which maximizes $gain(P,\ T_r)$ is then chosen.

Impurity measures

- The measures developed for selecting the best split are often based on the degree of impurity of the child nodes.
- Besides entropy, other examples of impurity measures include
  - Gini index

$$G = 1 - \sum_{k=1}^{K} p(c_k)^2$$

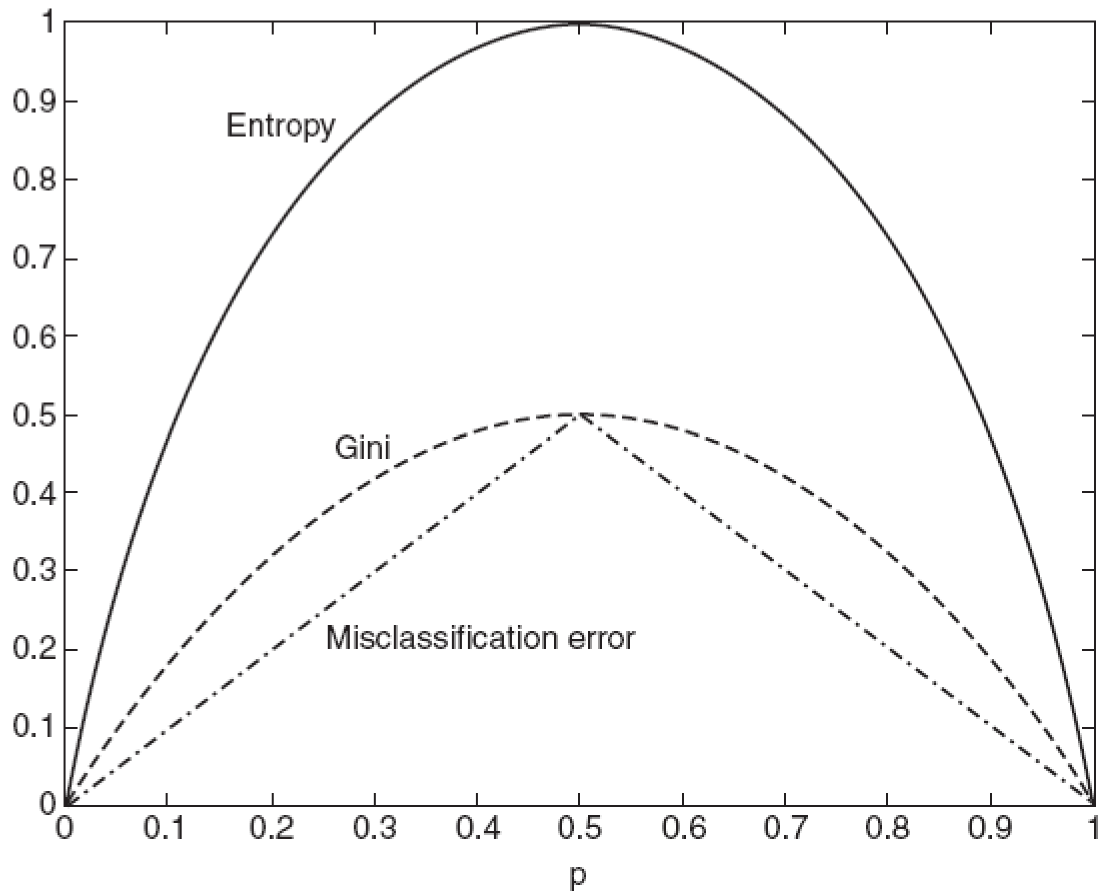
  - Classification error

$$E = 1 - \max_{k} p(c_k)$$

Impurity measures

- In the following figure, we compare the values of the impurity measures for binary classification problems.
- p refers to the fraction of records that belong to one of the two classes.
- All three measures attain their maximum value when $p=0.5$.
- The minimum values of the measures are attained when p equals 0 or 1.

Impurity measures



Gain ratio

- Impurity measures such as entropy and Gini index tend to favor attributes that have a large number of possible values.
- In many cases, a test condition that results in a large number of outcomes may not be desirable.
- This is because the number of records associated with each partition is too small to enable us to make any reliable predictions.

Gain ratio

- To solve this problem, we can modify the splitting criterion to take into account the number of possible attribute values.
- In the case of information gain, we can use the gain ratio which is defined as follows

$$Gain\ Ratio = \frac{Gain(P)}{Split\ Info}$$

where

$$Split\ Info = -\sum_{n=1}^{N} \frac{|U_n|}{|U|} \log_2 \frac{|U_n|}{|U|}$$

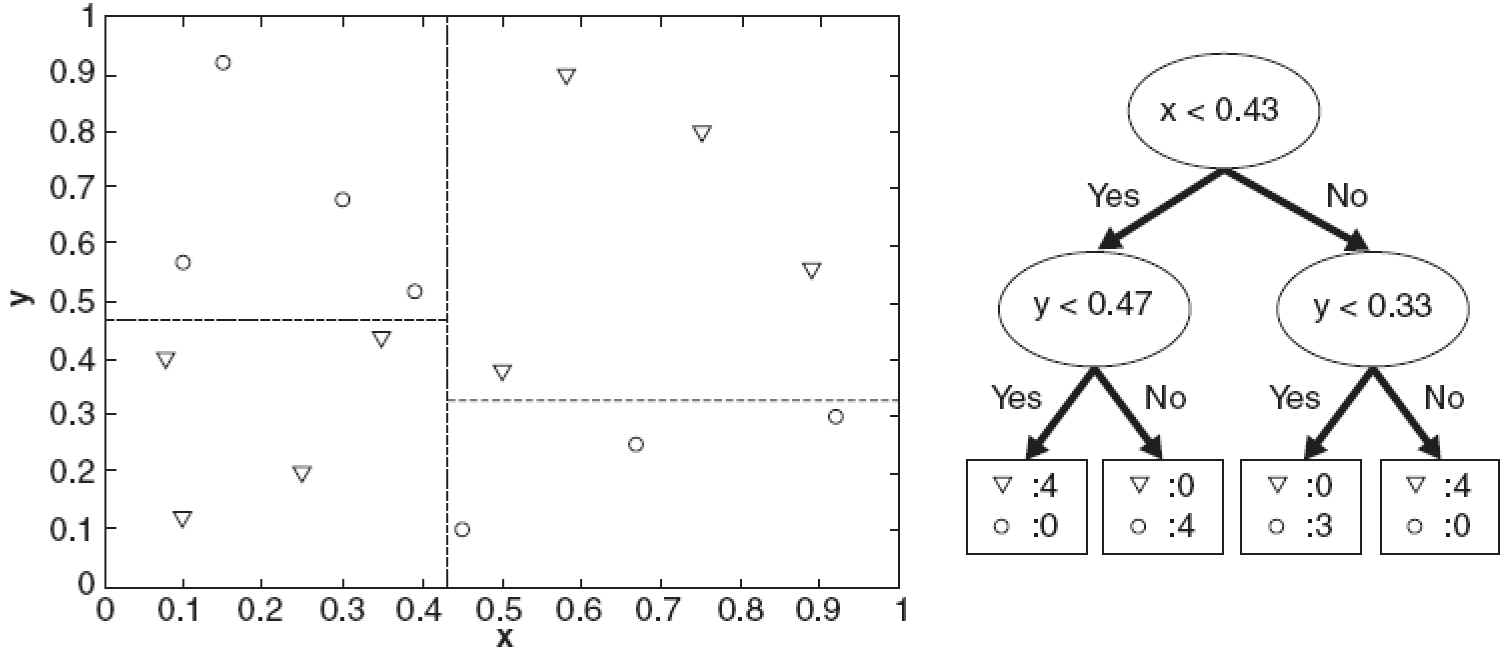
Oblique decision tree

- The test condition described so far involve using only a single attribute at a time.
- The tree-growing procedure can be viewed as the process of partitioning the attribute space into disjoint regions.
- The border between two neighboring regions of different classes is known as a decision boundary.

Oblique decision tree

- Since the test condition involves only a single attribute, the decision boundaries are rectilinear, i.e., parallel to the coordinate axes.
- This limits the expressiveness of the decision tree representation for modeling complex relationships among continuous attributes.

Oblique decision tree



Oblique decision tree

- An oblique decision tree allows test conditions that involve more than one attribute.
- The following figure illustrates a data set that cannot be classified effectively by a conventional decision tree.
- This data set can be easily represented by a single node of an oblique decision tree with the test condition $x+y<1$
- However, finding the optimal test condition for a given node can be computationally expensive.

Oblique decision tree