Data Mining
Classification: Decision Trees

- **Classification**
- **Decision Trees: what they are and how they work**
- **Hunt's (TDIDT) algorithm**
- **How to select the best split**
- **How to handle**
  - Inconsistent data
  - Continuous attributes
  - Missing values
  - Overfitting

Sections 4.1-4.3, 4.4.1, 4.4.2, 4.4.5 of course book

- **ID3, C4.5, C5.0, CART**
- **Advantages and disadvantages of decision trees**
- **Extensions to predict continuous values**

Classification

- Given a collection of records
  - Each record contains a set of attributes, one of the attributes is the class.
- Find a model for class attribute as a function of the values of other attributes
- **Goals**
  - apply the model to previously unseen records to predict their class (class should be predicted as accurately as possible)
  - Carry out deployment based on the model (e.g. implement more profitable marketing strategies)
- The data set can be divided into
  - Training set used to build the model
  - Test set used to determine the accuracy of the model

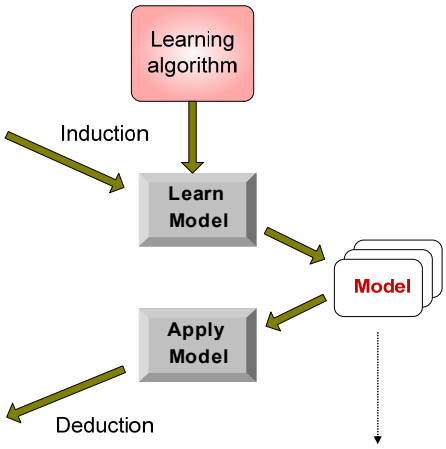
Illustrating Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



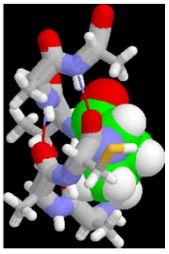
Attrib1 = yes → Class = No

Attrib1 = No ∧ Attrib3 < 95K → Class = Yes

Examples of Classification Task

- Predicting tumor cells as benign or malignant
- Classifying credit card transactions as legitimate or fraudulent

- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil
- Categorizing news stories as finance, weather, entertainment, sports, etc



- The UCI Repository of Datasets
  - http://archive.ics.uci.edu/ml/

Classification Techniques

- This lecture introduces

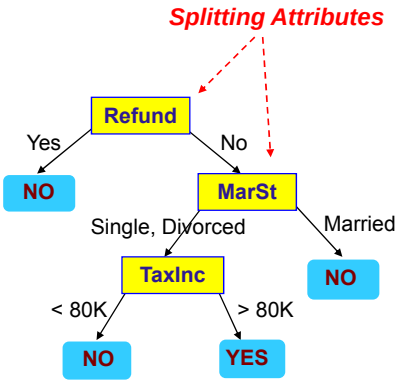
Decision Trees

- Other techniques will be presented in this course:
  - Rule-based classifiers
  - But, there are other methods
    - Nearest-neighbor classifiers
    - Naïve Bayes
    - Support-vector machines
    - Neural networks

Example of a Decision Tree

	categorical	categorical	continuous	class
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

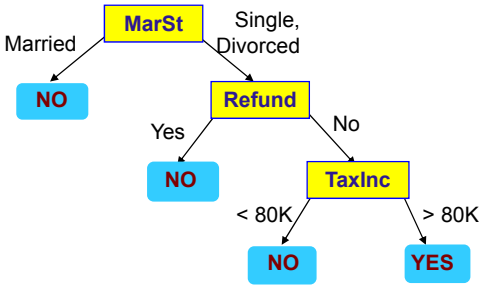
Training Data



Model: Decision Tree

Another Example of Decision Tree

Tid	Refund	Marital Status	Taxable Income	Cheat
		categorical	categorical	continuous	class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

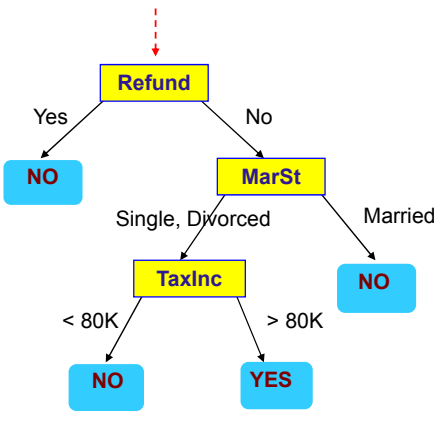


There could be more than one tree that fits the same data!

Search for the "best tree"

Apply Model to Test Data

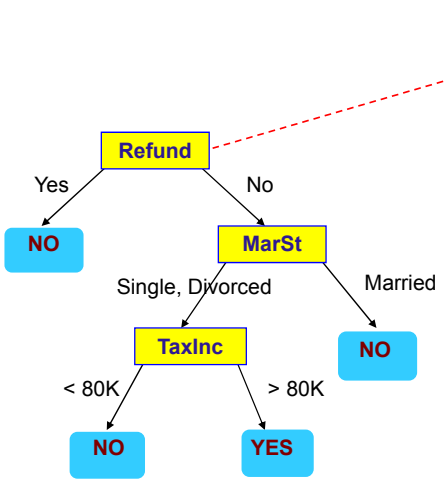
Start from the root of tree.



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

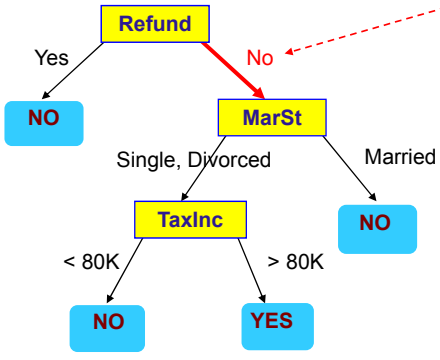
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

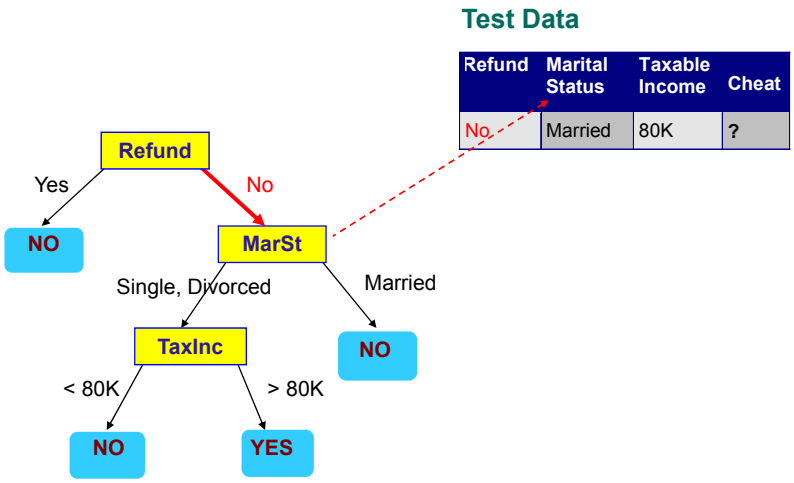
Apply Model to Test Data



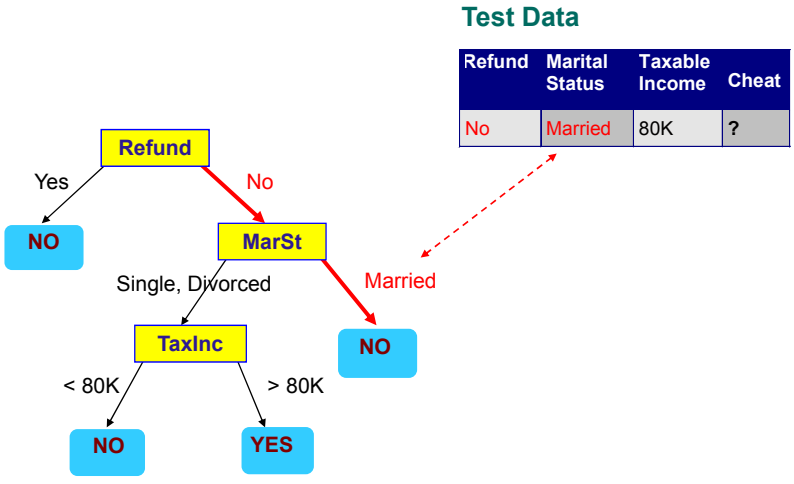
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

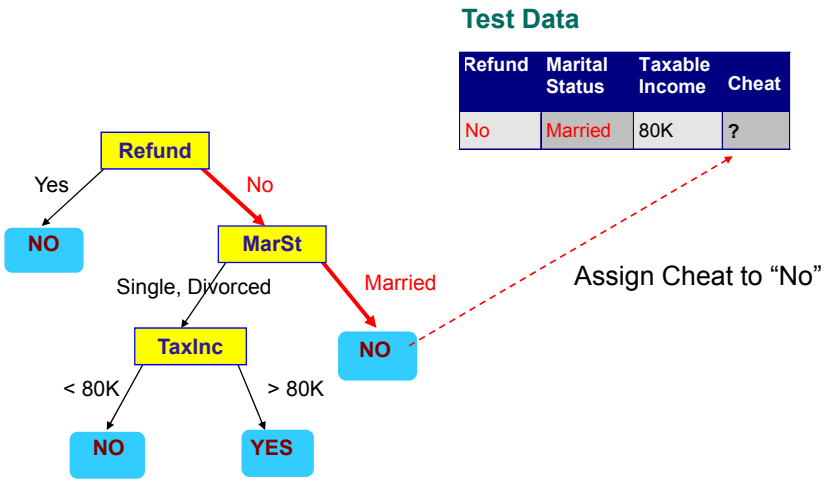
Apply Model to Test Data



Apply Model to Test Data



Apply Model to Test Data



Decision Tree Induction

- How to build a decision tree from a training set?
  - Many existing systems are based on

    Hunt's Algorithm

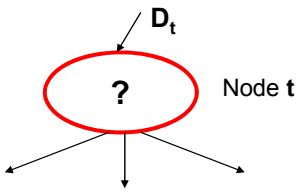
    Top-Down Induction of Decision Tree (TDIDT)

    - Employs a top-down search, greedy search through the space of possible decision trees

General Structure of Hunt's Algorithm

- Let D_t be the set of training records that reach a node t
- **General Procedure**:
  - If D_t contains records that belong the same class y_t, then t is a leaf node labeled as y_t
  - If D_t contains records that belong to more than one class, use an attribute test to split the data into smaller subsets. Recursively apply the procedure to each subset
    - Which attribute should be tested at each splitting node?
    - Use some heuristic

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

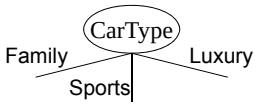


Tree Induction

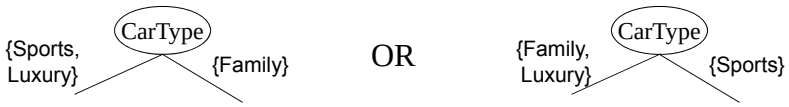
- Issues
  - Determine when to stop splitting
  - Determine how to split the records
    - Which attribute to use in a split node split?
      - How to determine the best split?
    - How to specify the attribute test condition?
      - E.g. `X < 1`? or `X+Y < 1`?
    - Shall we use 2-way split or multi-way split?

Splitting of Nominal Attributes

- Multi-way split: Use as many partitions as distinct values



- Binary split: Divides values into two subsets. Need to find optimal partitioning

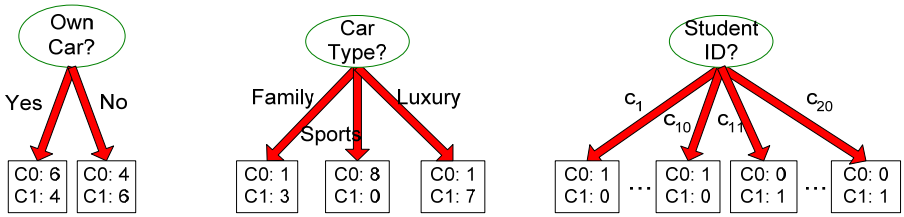


Stopping Criteria for Tree Induction

- Hunt's algorithm terminates when
  - All the records in a node belong to the same class
  - All records in a node have similar attribute values
    - Create a leaf node with the same class label as the majority of the training records reaching the node
  - A minimum pre-specified number of records belong to a node

Which Attribute Corresponds to the Best Split?

Before Splitting: *10 records of class 0,*
10 records of class 1



Which test condition is the best?

(assume the attribute is categorical)

How to determine the Best Split

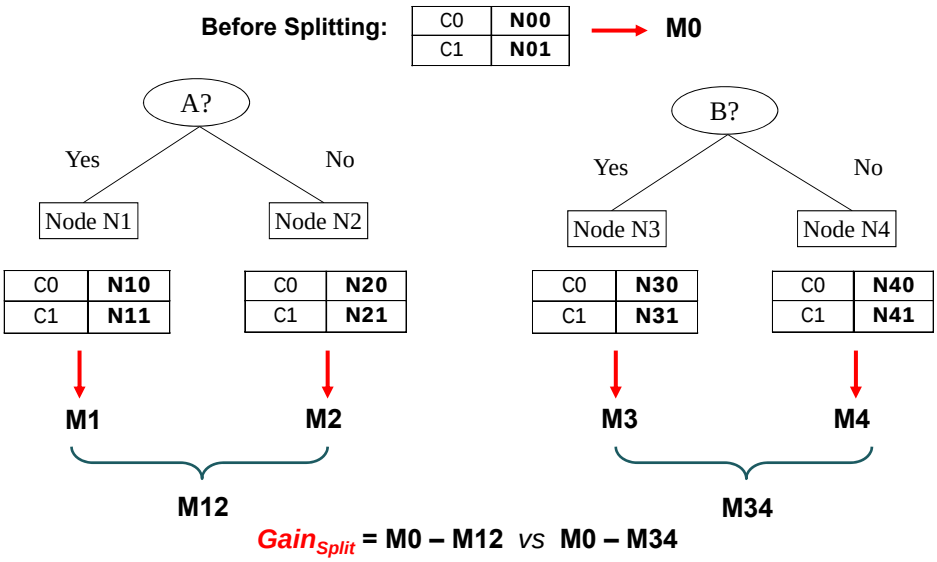
- Nodes with homogeneous class distribution are preferred
- Need a measure M of node impurity!!

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous, **High degree of impurity**	**Homogeneous,** **Low degree of impurity**

Measures of Node Impurity

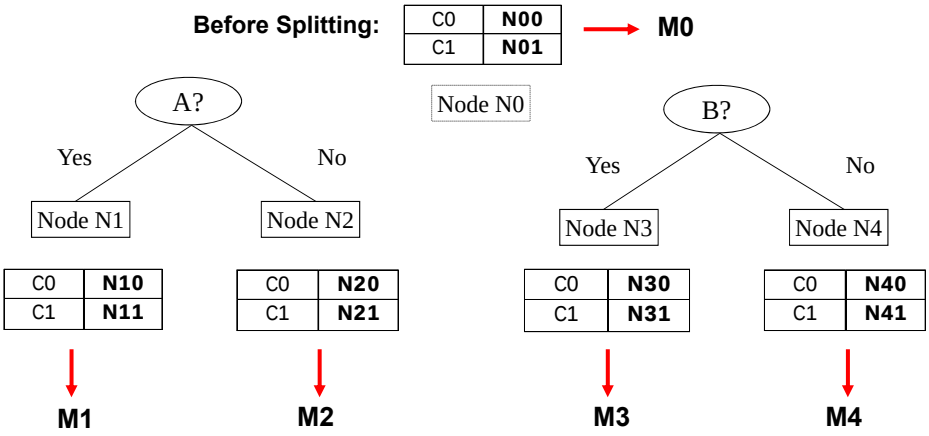
- Entropy
- Gini Index
- Misclassification error

How to Find the Best Split



$Gain_{Split}$ = M0 – M12 *vs* M0 – M34

How to Find the Best Split



$$M_i = Entropy(N_i) = -\sum_j p(j \mid N_i) \log p(j \mid N_i)$$

Splitting Criteria Based on Entropy

- Entropy at a given node t:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

(NOTE: *p(j | t)* is the relative frequency of class j at node t).

  - Measures homogeneity of a node
    - Maximum (**log n_c**) when records are **equally distributed** among all classes implying maximum impurity
      - **n_c** is the number of classes
    - Minimum (0.0) when all records belong to one class, implying least impurity

Examples for computing Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Entropy = – 0 $\log_2$ 0 – 1 $\log_2$ 1 = – 0 – 0 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (5/6) = 0.65

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

Splitting Based on Information Gain

- **Information Gain**:

$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

Parent node **p** with **n** records is split into **k** partitions;

$\mathbf{n_i}$ is number of records in partition (node) **i**

  - ***GAIN*$_{split}$** measures Reduction in Entropy achieved because of the split Choose the split that achieves most reduction (maximizes GAIN)
    - Used in ***ID3*** and ***C4.5***
  - **Disadvantage**: bias toward attributes with large number of values
    - Large trees with many branches are preferred
    - What happens if there is an ID attribute?

Splitting Based on GainRATIO

- Gain Ratio:

$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO} \qquad SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n}$$

Parent node **p** is split into **k** partitions

$\mathbf{n_i}$ is the number of records in partition **i**

- ***GAIN***$_{split}$ is penalized when large number of small partitions are produced by the split!
  - ***SplitINFO*** increases when a larger number of small partitions is produced
  - Used in ***C4.5*** ***(Ross Quinlan)***
- Designed to overcome the disadvantage of Information Gain.

Split Information

$$SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n}$$

A = 1	A= 2	A = 3	A = 4	*SplitINFO*
32	0	0	0	0
16	16	0	0	1
16	8	8	0	1.5
16	8	4	4	1.75
8	8	8	8	2

Other Measures of Impurity

- Gini Index for a given node t :

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

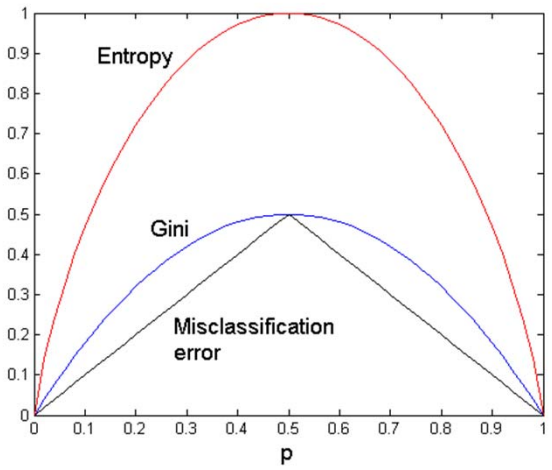
(NOTE: $p(j \mid t)$ is the relative frequency of class j at node t)

- Classification error at a node t:

$$Error(t) = 1 - \max_i P(i \mid t)$$

Comparison among Splitting Criteria

For a 2-class problem:



Practical Issues in Learning Decision Trees

- **Conclusion:** decision trees are built by greedy search algorithms, guided by some heuristic that measures "impurity"
- In real-world applications we need also to consider
  - Continuous attributes
  - Missing values
  - Improving computational efficiency
  - Overfitted trees

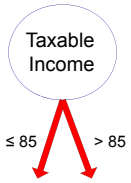
Splitting of Continuous Attributes

- Different ways of handling
  - Discretize once at the beginning
  - Binary Decision: ($A < v$) or ($A \geq v$)
    - A is a continuous attribute: consider all possible splits and find the best cut
    - can be more computational intensive

Continuous Attributes

- Several Choices for the splitting value
  - Number of possible splitting values = Number of distinct values n
- For each splitting value v
  1. Scan the data set and
  2. Compute class counts in each of the partitions, A < v and A ≥ v
  3. Compute the entropy/Gini index
- Choose the value v that gives lowest entropy/Gini index
- **Naïve algoritm**
  - Repetition of work **$O(n^2)$**
- **Efficient implementation** **$O(m \times n\log(n))$**
  - **m** is the nunber of attributes and **n** is the number of records

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

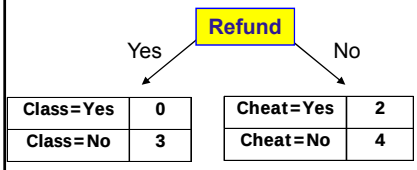


Handling Missing Attribute Values

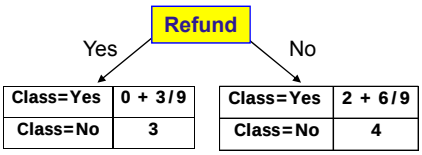
- Missing values affect decision tree construction in three different ways:
  - Affects how impurity measures are computed
  - Affects how to distribute instance with missing value to child nodes
    - How to build a decision tree when some records have missing values?
- Usually, missing values should be handled during the pre-processing phase

Distribute Training Instances with missing values

Tid	Refund	Marital Status	Taxable Income	Class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No



Tid	Refund	Marital Status	Taxable Income	Class
10	?	Married	90K	Yes

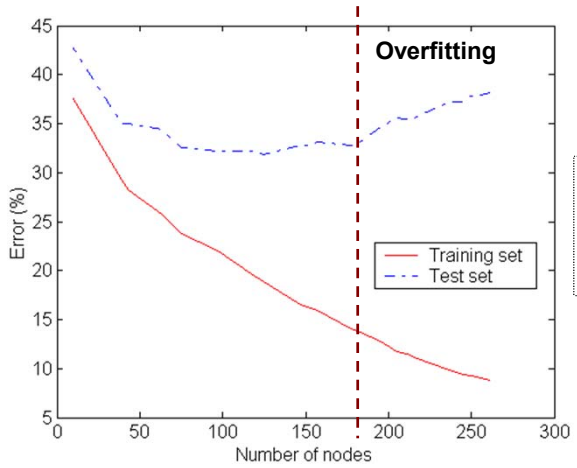


Send down record *Tid=10* to the left child with weight = 3/9 and to the right child with weight = 6/9

Overfitting

- A tree that fits the training data too well may not be a good classifier for new examples.
- **Overfitting** results in decision trees more complex than necessary
- Estimating error rates
  - Use statistical techniques
  - Re-substitution errors: error on training data set (training error)
  - Generalization errors: error on a testing data set (test error)
    - Typically, 2/3 of the data set is reserved to model building and 1/3 for error estimation
    - **Disadvantage**: less data is available for training
- Overfitted trees may have a low re-substitution error but a high generalization error.

Underfitting and Overfitting



When the tree becomes too large, its test error rate begins increasing while its training error rate continues too decrease.

What causes overfitting?

- Noise

Underfitting: when model is too simple, both training and test errors are large

How to Address Overfitting

- **Pre-Pruning** (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
    - Stop if all instances belong to the same class
    - Stop if all the attribute values are the same
  - Early stopping conditions:
    - Stop if number of instances is less than some user-specified threshold
      - e.g. 5 to 10 records per node
    - Stop if class distribution of instances are independent of the available attributes (e.g., using χ^2 test)
    - Stop if splitting the current node improves the impurity measure (e.g. Gini or information gain) below a given threshold

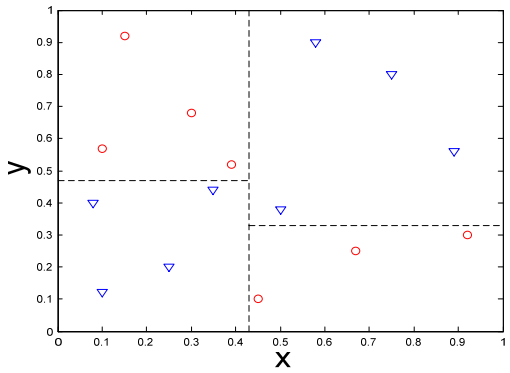
How to Address Overfitting…

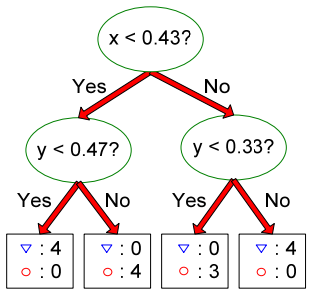
- **Post-pruning**
  - Grow decision tree to its entirety
  - Trim the nodes of the decision tree in a bottom-up fashion
  - Reduced-error pruning
    - Use a dataset not used in the training **pruning set**
      - Three data sets are needed: training set, test set, pruning set
    - If test error in the pruning set improves after trimming then prune the tree
  - Post-pruning can be achieved in two ways:
    - Sub-tree replacement
    - Sub-tree raising
    - See section 4.4.5 of course book

ID3, C4.5, C5.0, CART

- Ross Quinlan
  - **ID3** uses the Hunt's algorithm with information gain criterion and gain ratio
    - Available in WEKA (no discretization, no missing values)
  - **C4.5** improves ***ID3***
    - Needs entire data to fit in memory
    - Handles missing attributes and continuous attributes
    - Performs tree post-pruning
    - Available in WEKA as ***J48***
  - **C5.0** is the current commercial successor of ***C4.5***
- Breiman et al.
  - **CART** builds multivariate decision (binary) trees
    - Available in WEKA as ***SimpleCART***

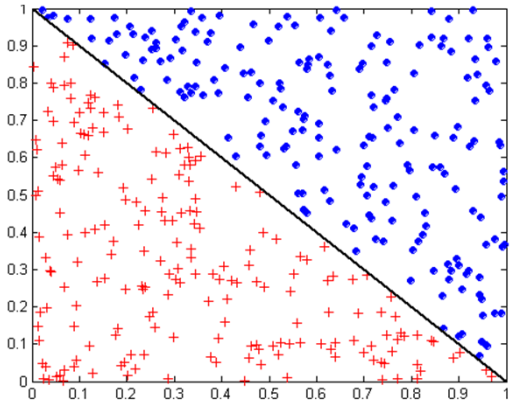
Decision Boundary

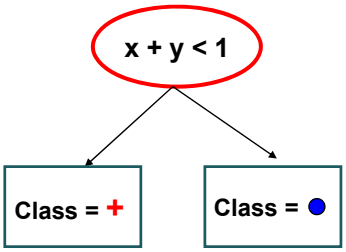




- Border line between two neighboring regions of different classes is known as decision boundary.
- Decision boundary is parallel to axes because test condition involves a single attribute at-a-time

Oblique Decision Trees





- Test condition may involve multiple attributes (e.g. ***CART***)
- More expressive representation
- Finding optimal test condition is computationally expensive

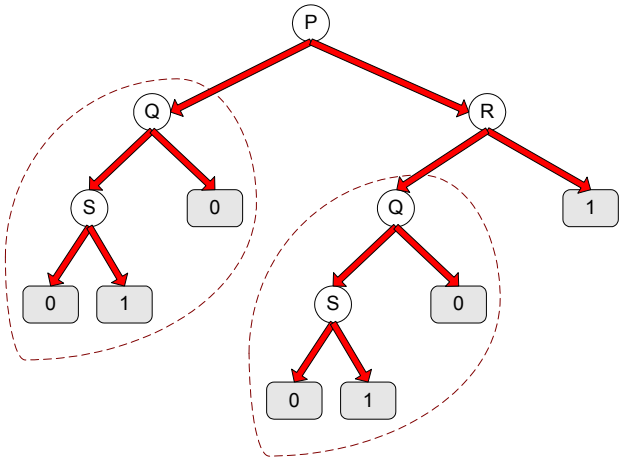
Advantages of Decision Trees

- Extremely fast at classifying unknown records
- Easy to interpret for small-sized trees
- Able to handle both continuous and discrete attributes
- Work well in the presence of redundant attributes
- If methods for avoiding overfitting are provided then decision trees are quite robust in the presence of noise
- Robust to the effect of outliers
- Provide a clear indication of which fields are most important for prediction

Disadvantages of Decision Trees

- Irrelevant attributes may affect badly the construction of a decision tree
  - E.g. ID numbers
- Decision boundaries are rectilinear
- Small variations in the data can imply that very different looking trees are generated
- A sub-tree can be replicated several times
- Error-prone with too many classes
- Not good for predicting the value of a continuous class attribute
  - This problem is addressed by regression trees and model trees

Tree Replication



- **Same subtree appears in multiple branches**