DATA MINING & MACHINE LEARNING

Lecture 3

Data Mining Algorithms for Classification

Prediction Problems

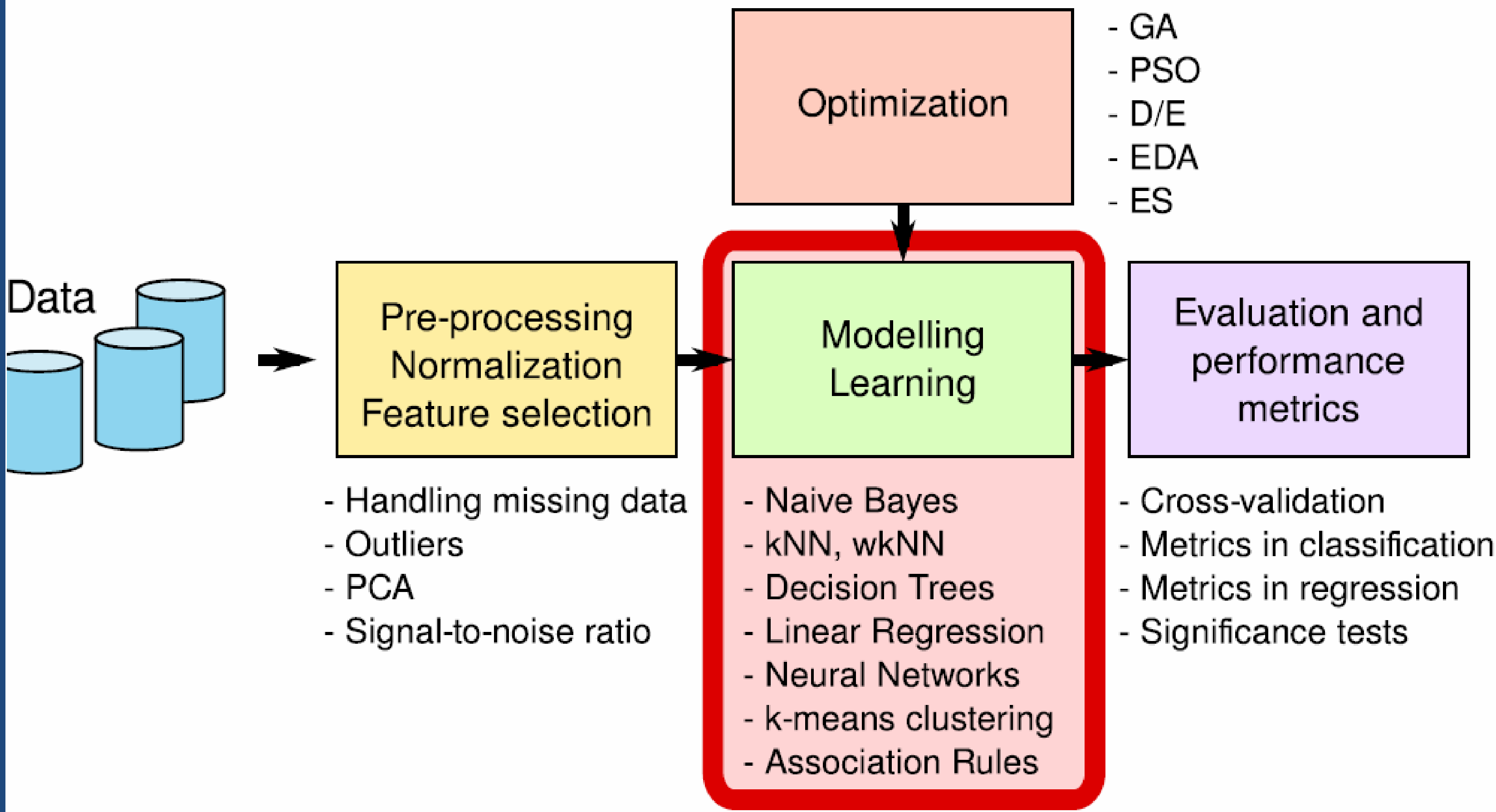
- Classification
  - *predicts categorical class labels (discrete or nominal)*
  - *classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data*

- Numeric Prediction
  - *models continuous-valued functions, i.e., predicts unknown or missing values*

Supervised vs. Unsupervised Learning

- Supervised learning (classification)
  - *Supervision: The training data (observations, measurements, etc.) are accompanied by **labels** indicating the class of the observations*
  - *New data is classified based on the training set*
- Unsupervised learning (clustering)
  - *The class labels of training data is unknown*
  - *Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data*

Course Outline



Classification: Definition

- Given a collection of records (*training set*)
  - *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.
- Goal: <u>previously unseen</u> records should be assigned a class as accurately as possible.
  - *A test set is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.*

Learning Outcomes

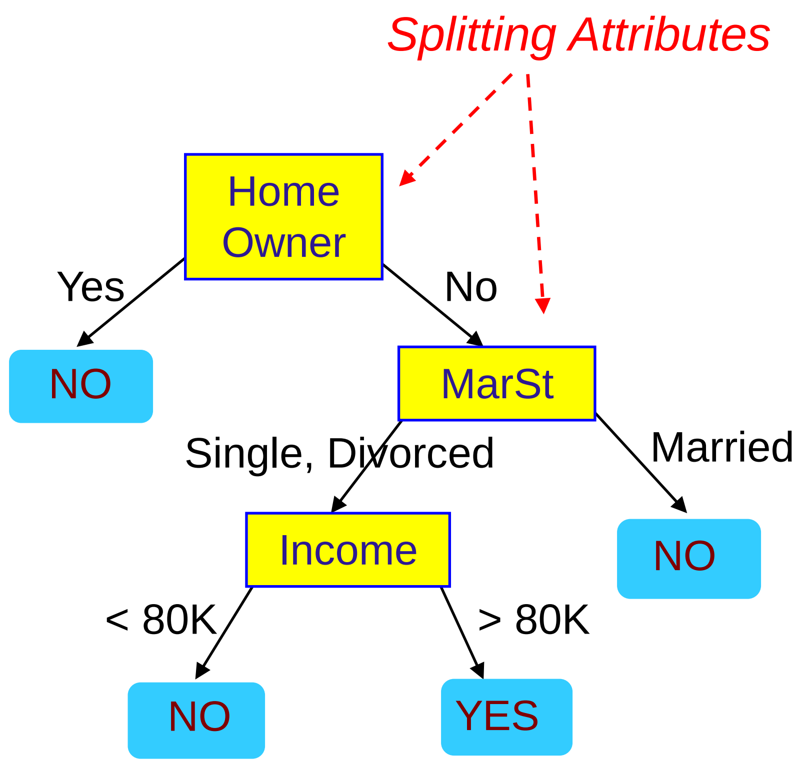
- Examine the algorithms used in popular classification schemes
  - *Decision Trees*
  - *Naïve Bayes*
  - *Nearest Neighbour*
  - *Neural Networks*

Example of a Decision Tree

categorical categorical continuous class

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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



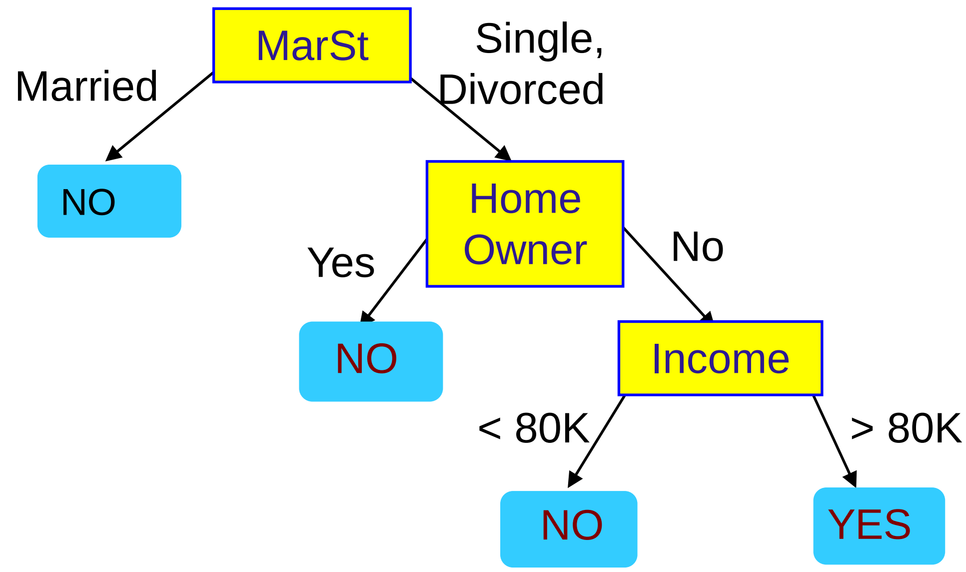
Model: Decision Tree

Example of a Decision Tree

categorical categorical continuous class

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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

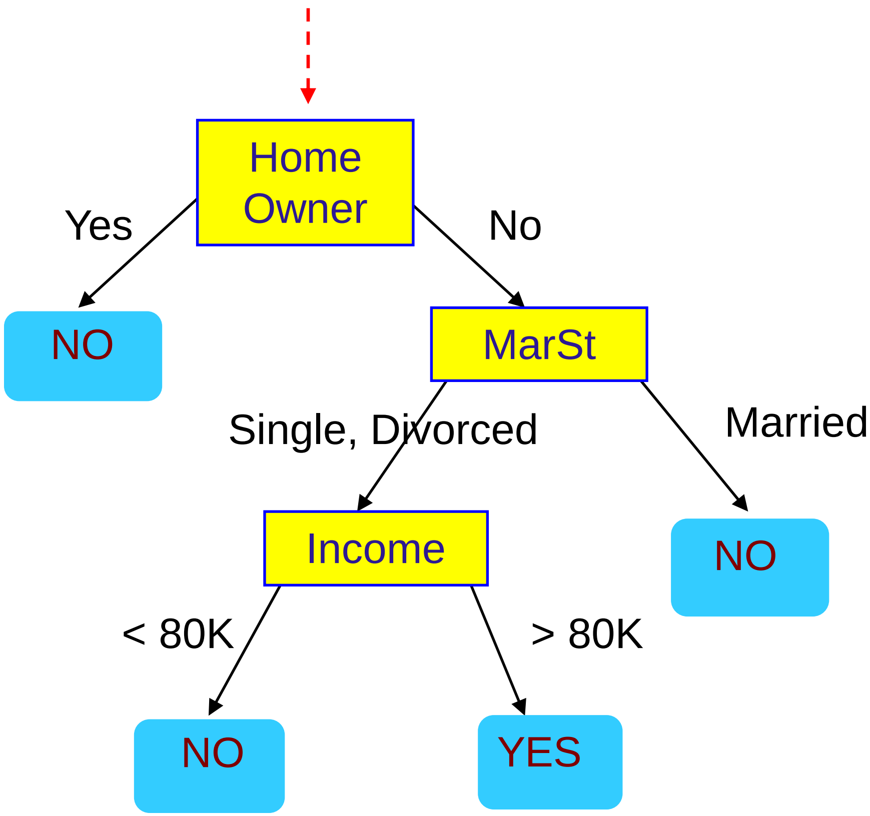
Training Data



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

Apply Model to Test Data

Start from the root of tree.

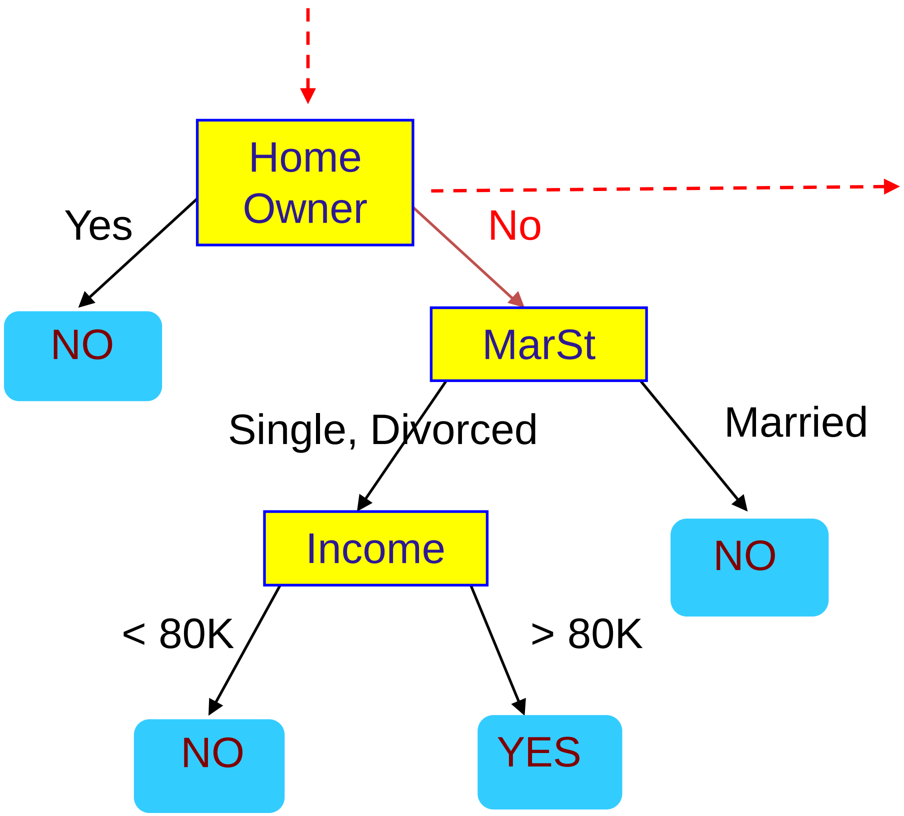


Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

Apply Model to Test Data

Start from the root of tree.



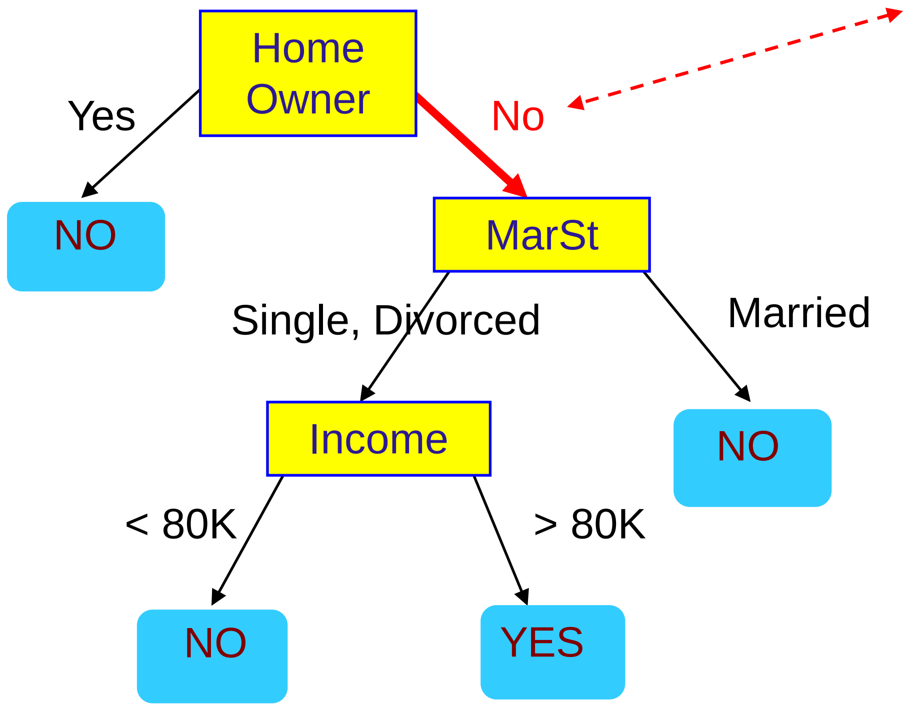
Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

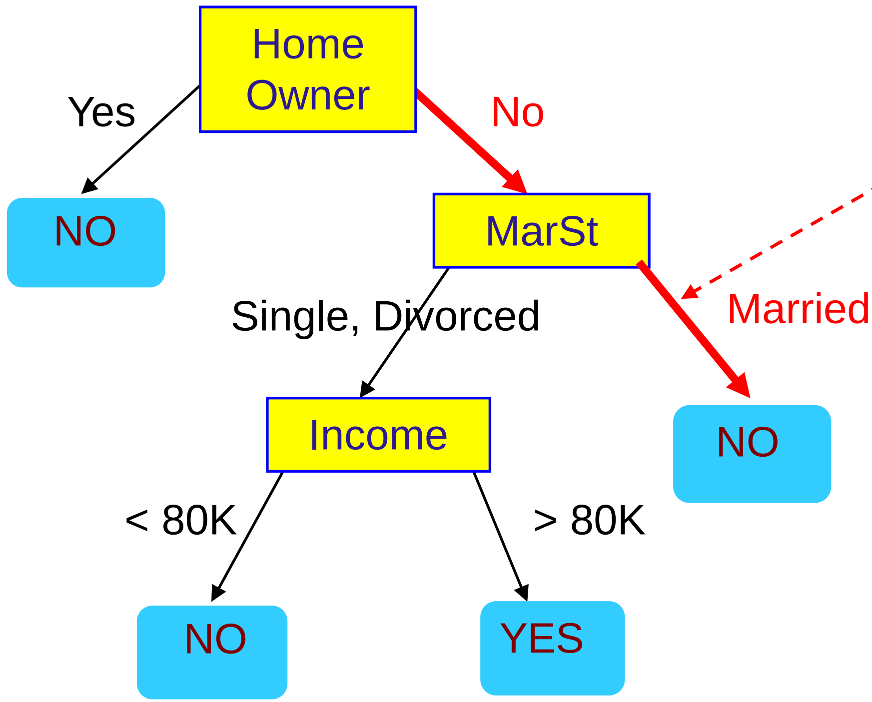
Apply Model to Test Data

Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



Apply Model to Test Data



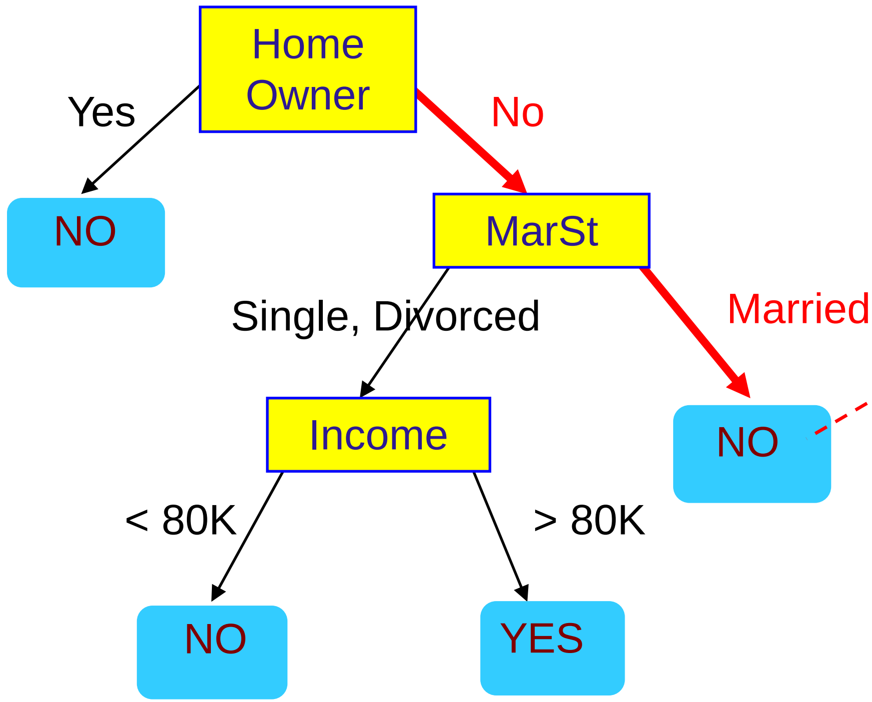
Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

Apply Model to Test Data

Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

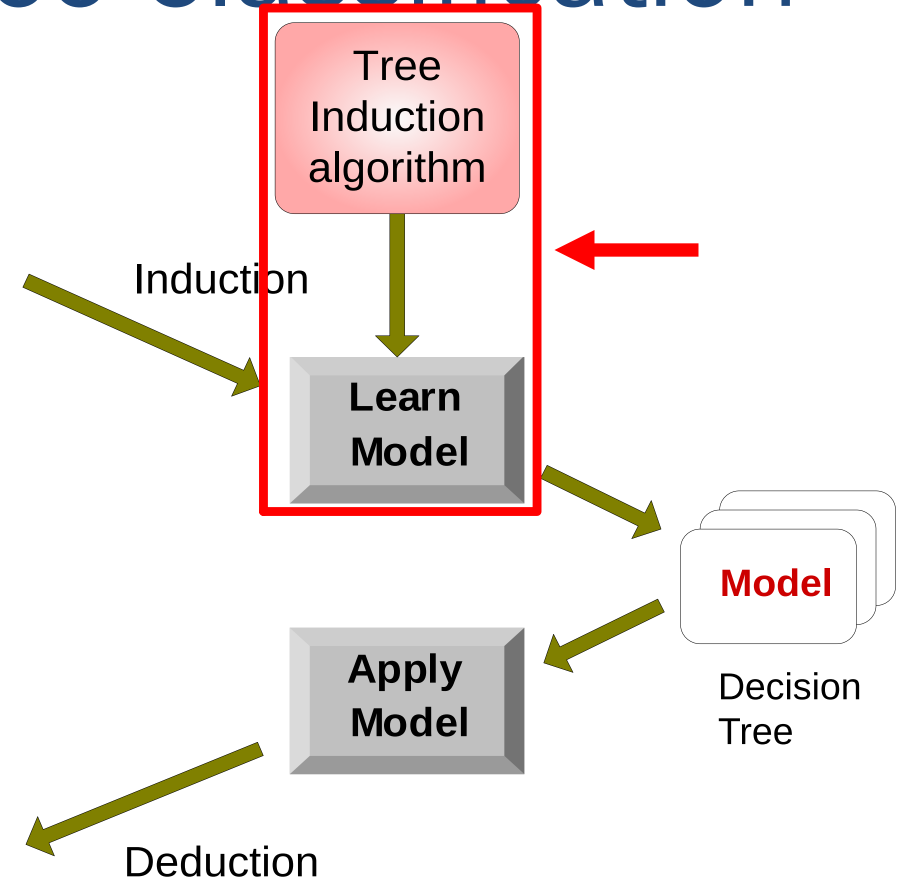


Assign Defaulted to "No"

Decision Tree 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

Decision Tree Induction

- Many Algorithms:
  - *Hunt's Algorithm (one of the earliest)*
  - *CART*
  - *ID3, C4.5*
  - *SLIQ,SPRINT*

Predict if John will play tennis

- Hard to guess
- Try to understand when John plays
- Divide & conquer:
  - *split into subsets*
  - *are they pure? (all yes or all no)*
  - *if yes: stop*
  - *if not: repeat*
- See which subset
  - *New data falls into*

Training examples: 9 yes/ 5 no

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	Overcast	High	Weak	Yes
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Weak	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Strong	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No

New data:

D15	Rain	High	Weak	?

Predict if John will play tennis

- Hard to guess
- Try to understand when John plays
- Divide & conquer:
  - split into subsets
  - are they pure? (all yes or all no)
  - if yes: stop
  - if not: repeat
- See which subset
  - New data falls into

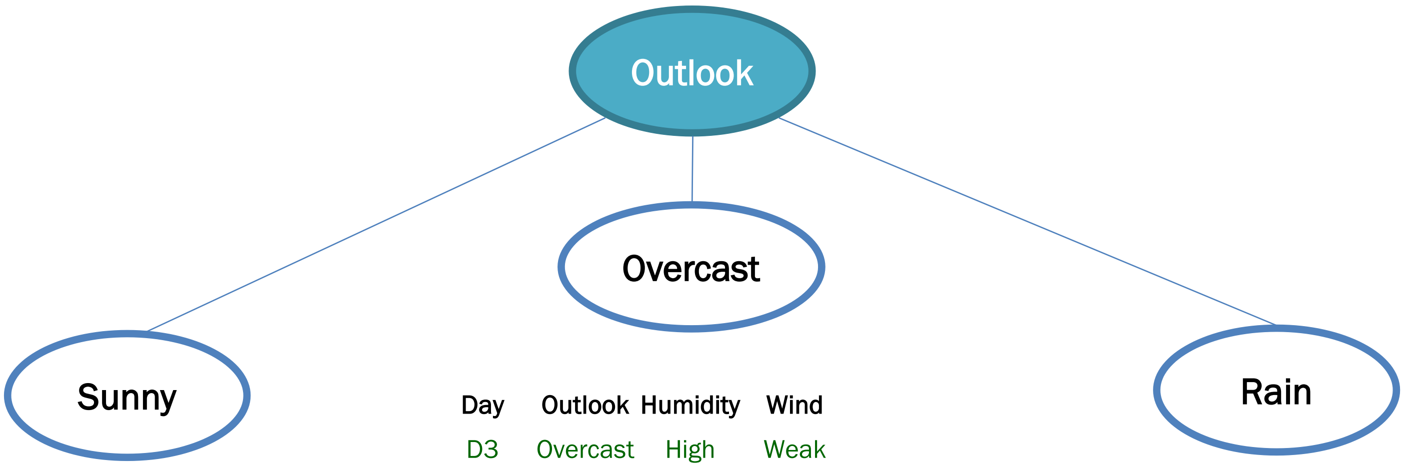
Training examples: 9 yes/ 5 no

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	Overcast	High	Weak	Yes
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Weak	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Strong	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No

New data:

Day	Outlook	Humidity	Wind	Play
D15	Rain	High	Weak	?

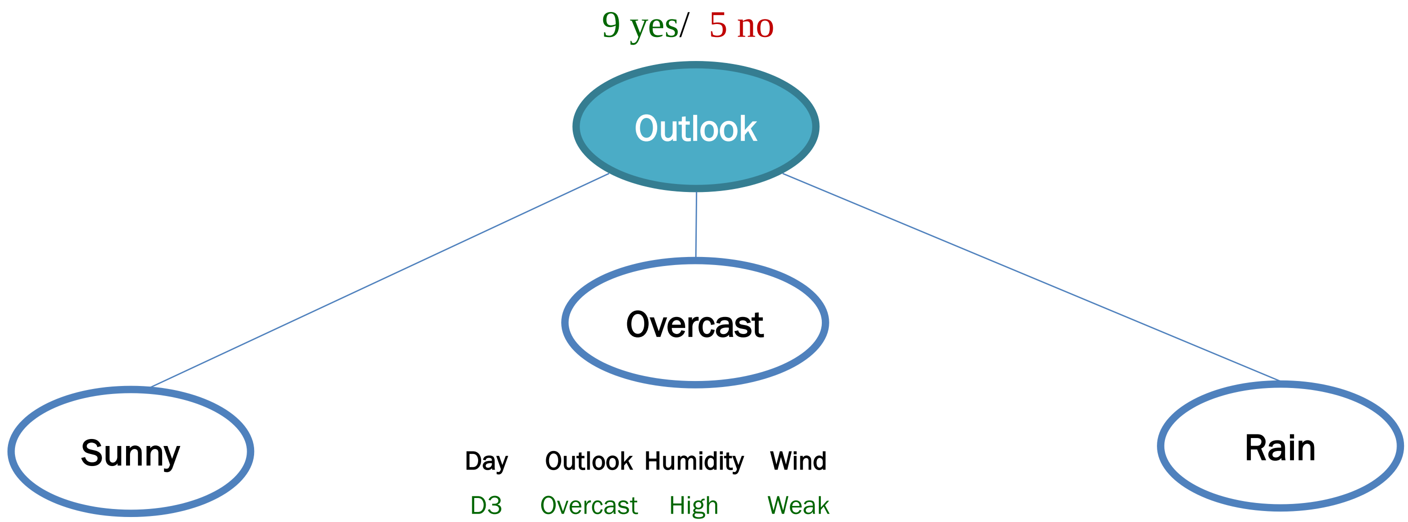
9 yes/ 5 no



Day	Outlook	Humidity	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

Day	Outlook	Humidity	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Weak
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

Day	Outlook	Humidity	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D10	Rain	Normal	Strong
D14	Rain	High	Strong



Sunny

Day	Outlook	Humidity	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

2 yes/ 3 no
Split further

Overcast

Day	Outlook	Humidity	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Weak
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

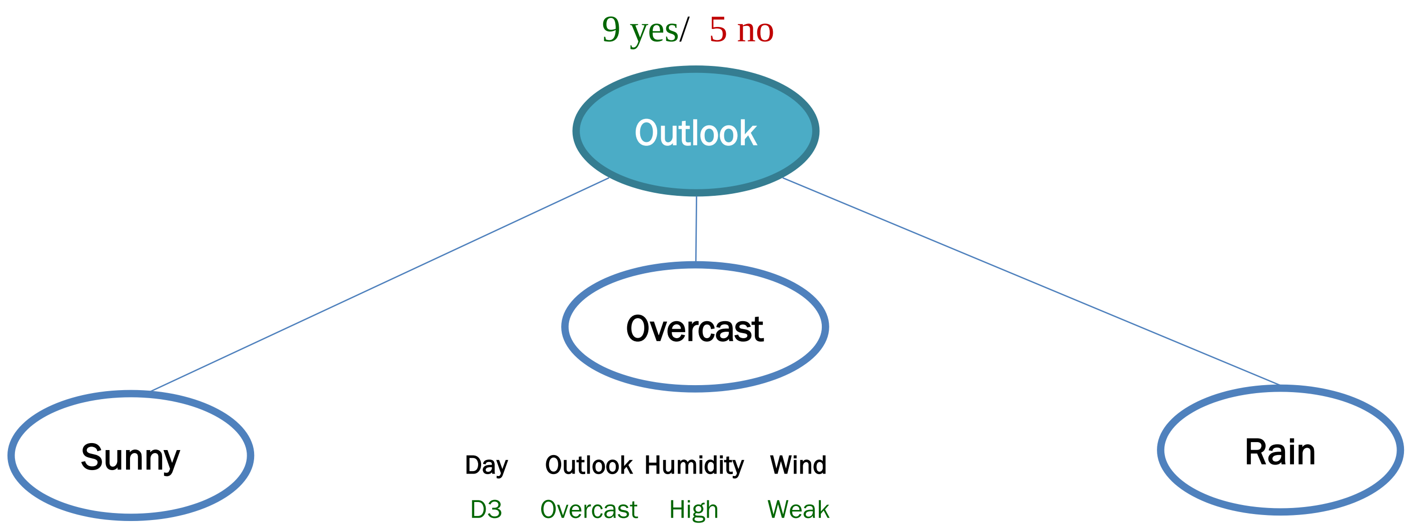
4 yes/ 0 no
pure subset

Rain

Day	Outlook	Humidity	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D10	Rain	Normal	Strong
D14	Rain	High	Strong

4 yes/ 1 no

Split further



Day	Outlook	Humidity	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

2 yes/ 3 no
Split further

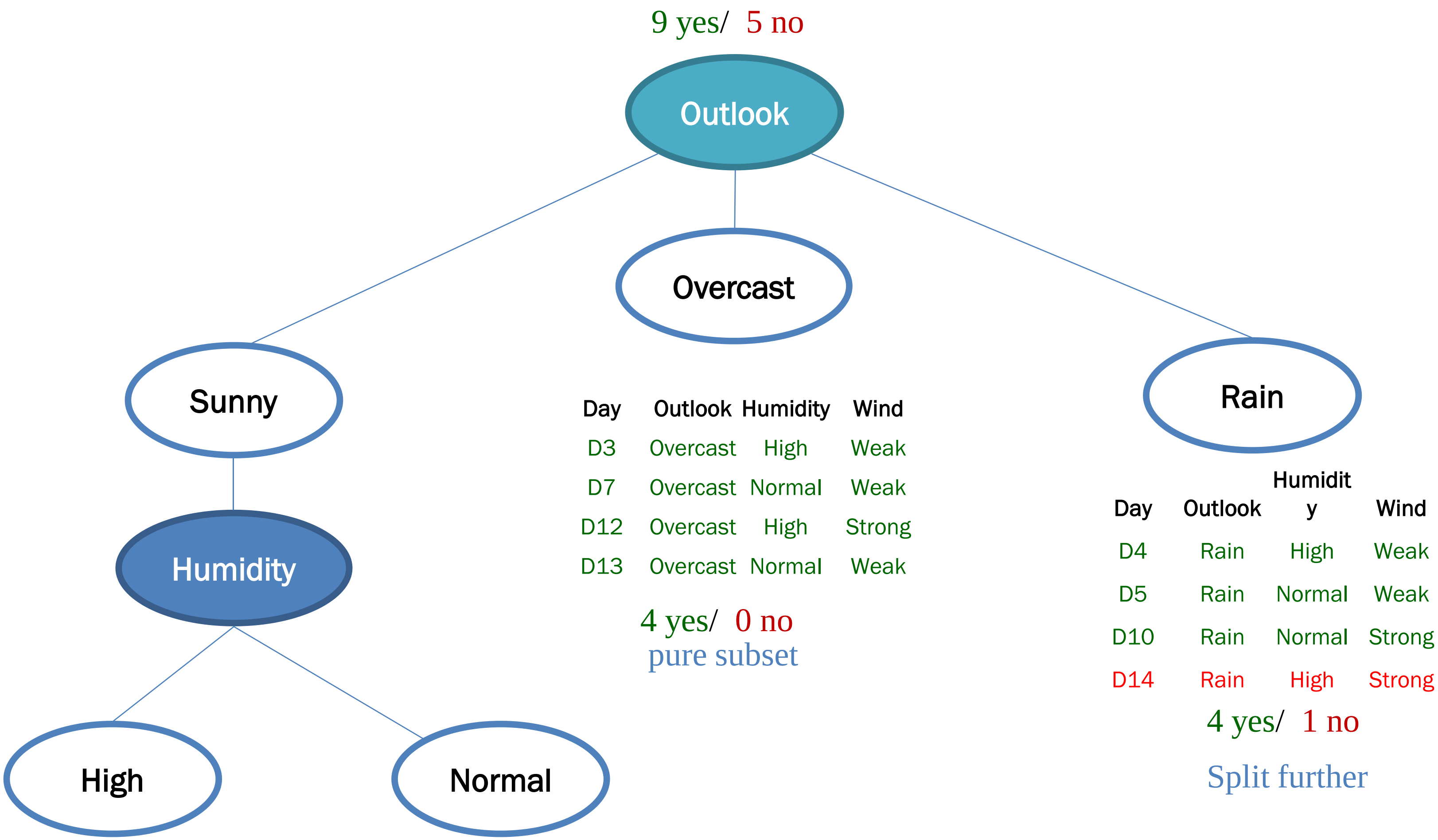
Day	Outlook	Humidity	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Weak
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

4 yes/ 0 no
pure subset

Day	Outlook	Humidity	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D10	Rain	Normal	Strong
D14	Rain	High	Strong

4 yes/ 1 no

Split further

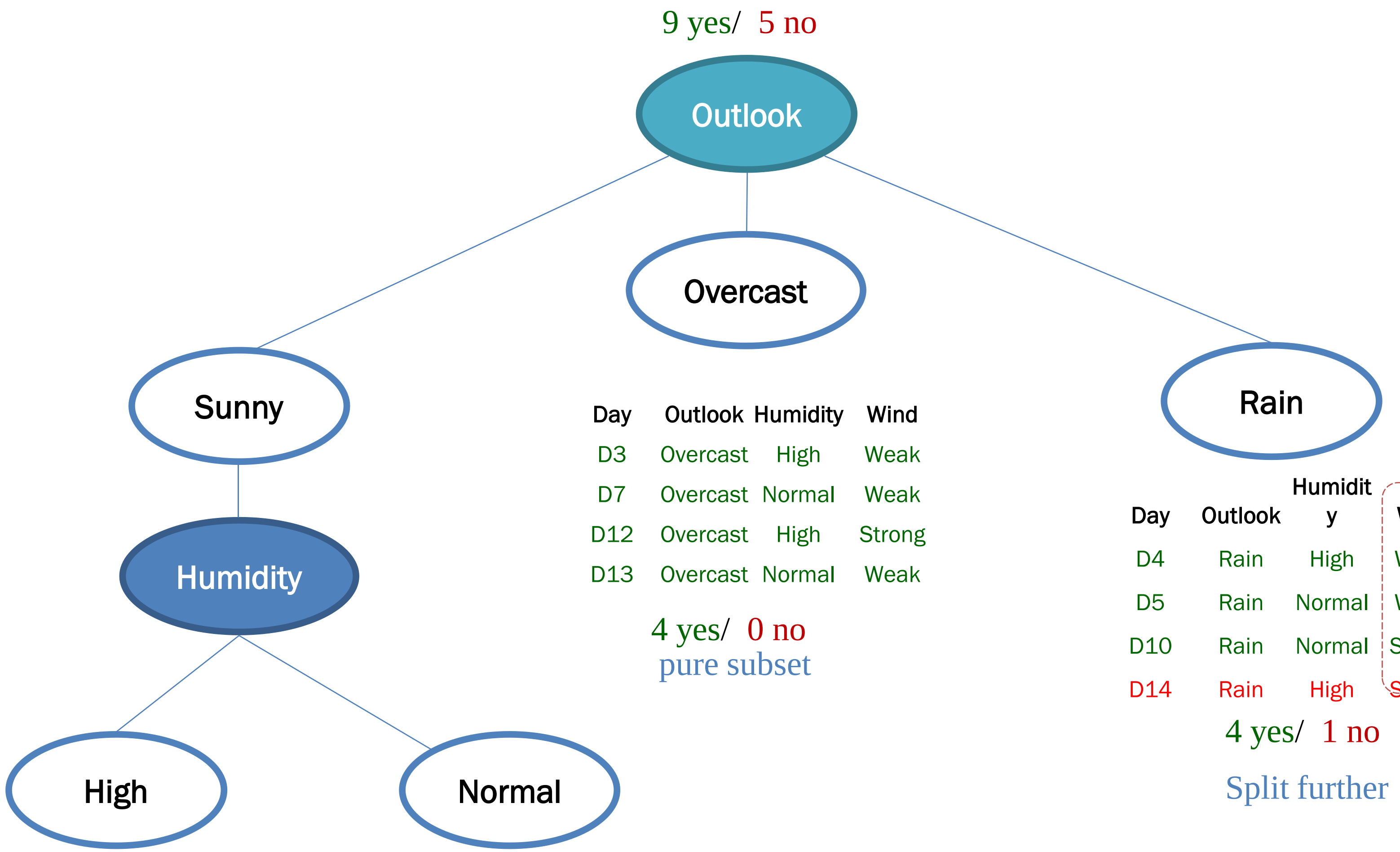


Day	Outlook	Humidity	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Weak
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

Day	Outlook	Humidit y	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D10	Rain	Normal	Strong
D14	Rain	High	Strong

Day	Humidit y	Wind
D1	High	Weak
D2	High	Strong
D8	High	Weak

Day	Humidity	Wind
D9	Normal	Weak
D11	Normal	Strong



Day	Outlook	Humidity	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Weak
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

4 yes/ 0 no
pure subset

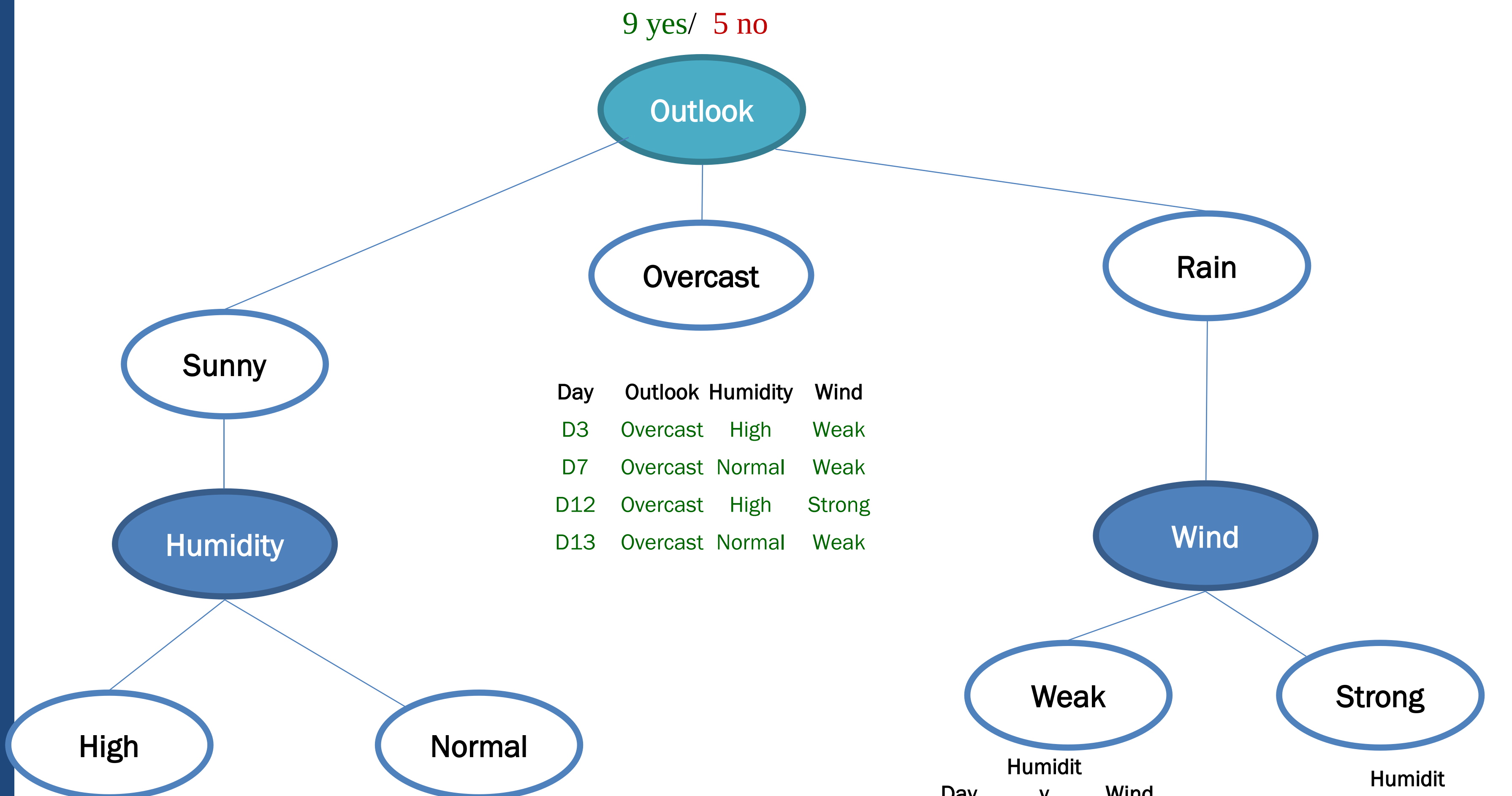
Day	Outlook	Humidity	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D10	Rain	Normal	Strong
D14	Rain	High	~~Strong~~

4 yes/ 1 no

Split further

Day	Humidity	Wind
D1	High	Weak
D2	High	Strong
D8	High	Weak

Day	Humidity	Wind
D9	Normal	Weak
D11	Normal	Strong



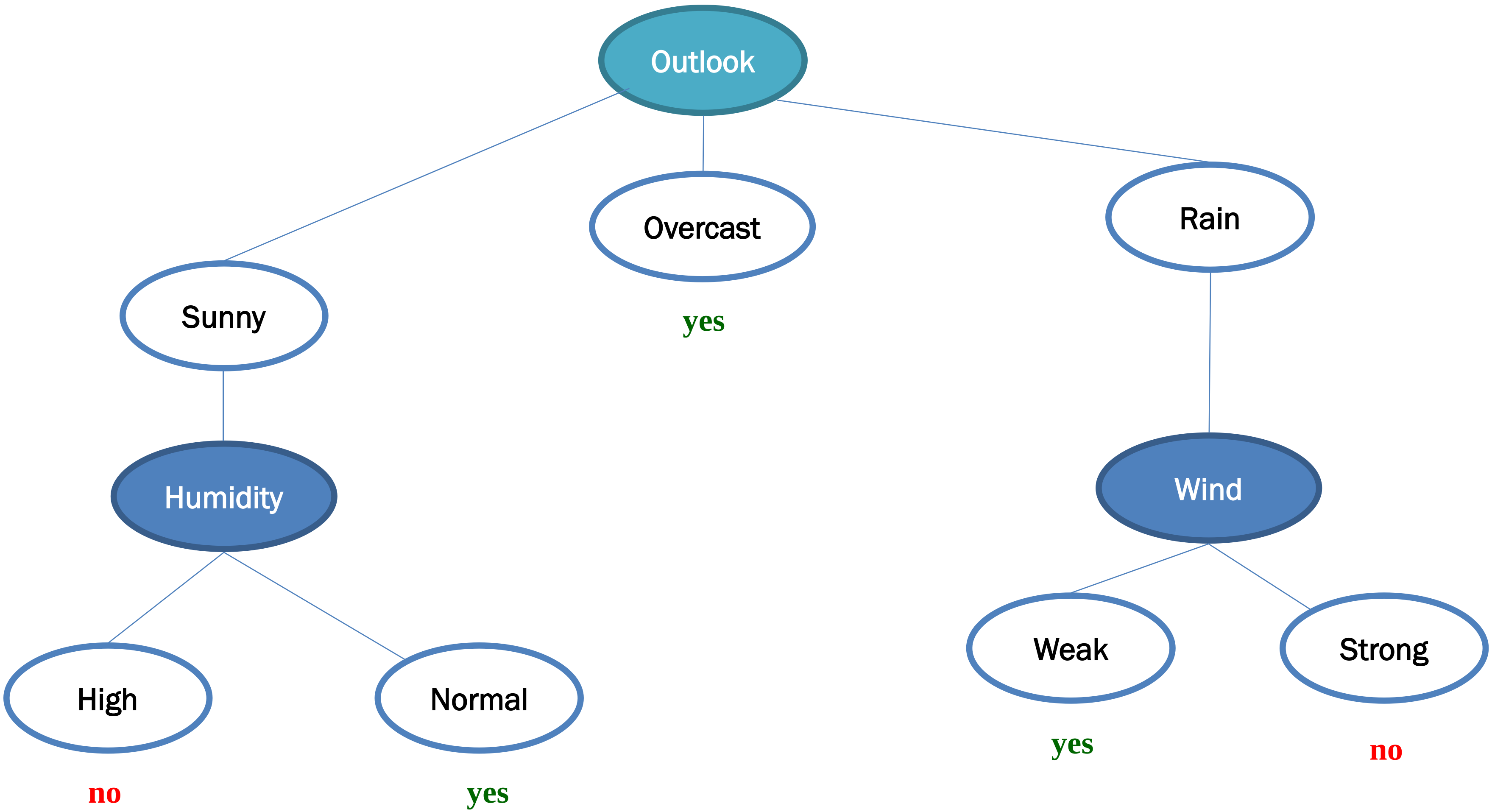
Day	Outlook	Humidity	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Weak
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

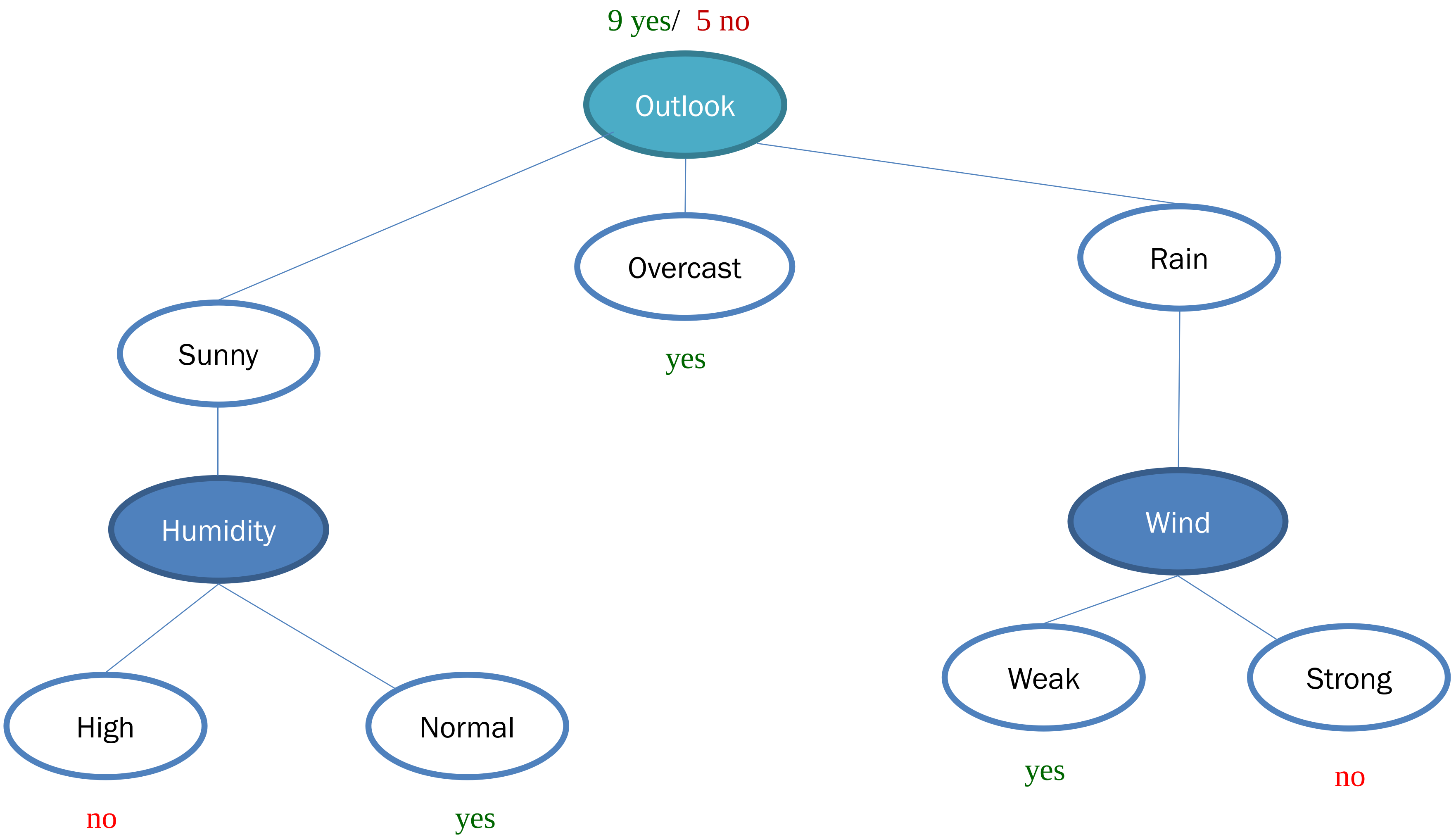
Day	Humidity	Wind
D1	High	Weak
D2	High	Strong
D8	High	Weak

Day	Humidity	Wind
D9	Normal	Weak
D11	Normal	Strong

Day	Humidity	Wind
D4	High	Weak
D5	Normal	Weak
D10	Normal	Weak

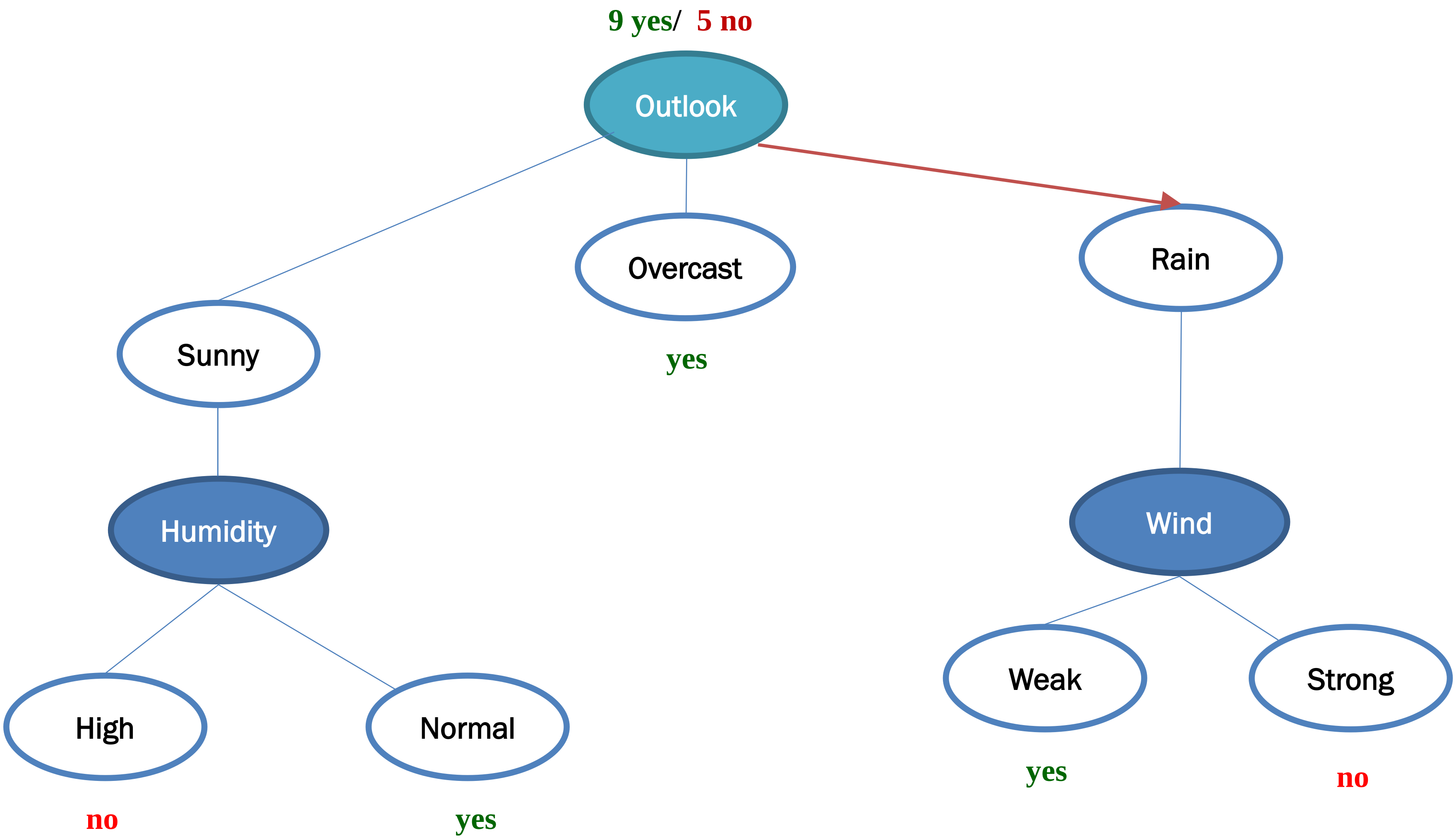
Day	Humidity	Wind
D6	Normal	Strong
D14	High	Strong


Outlook
Sunny
Overcast
Rain
yes
Humidity
Wind
High
Normal
Weak
Strong
no
yes
yes
no



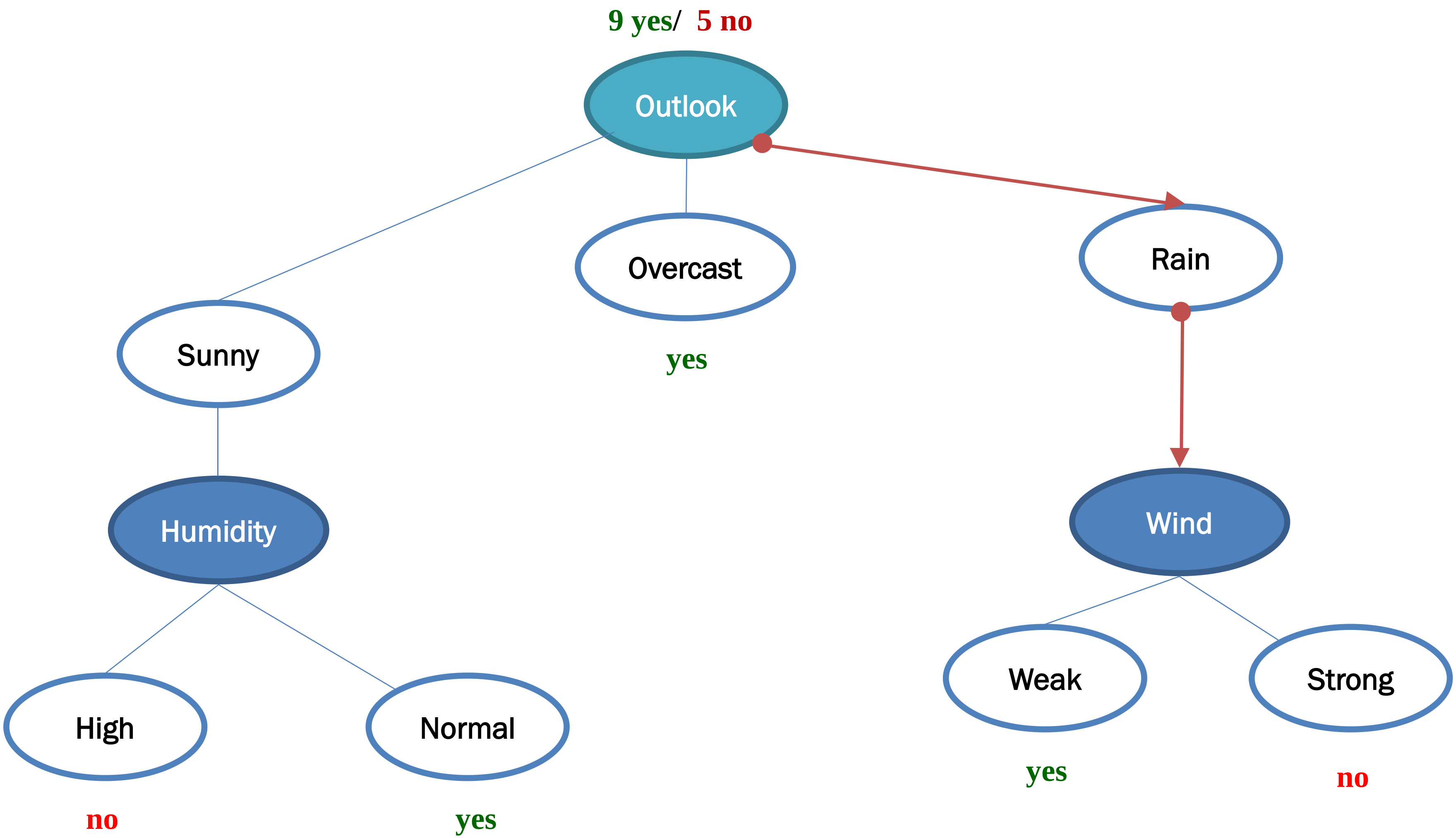
New data:

Day	Outlook	Humidity	Wind	Play
D15	Rain	High	Weak	?



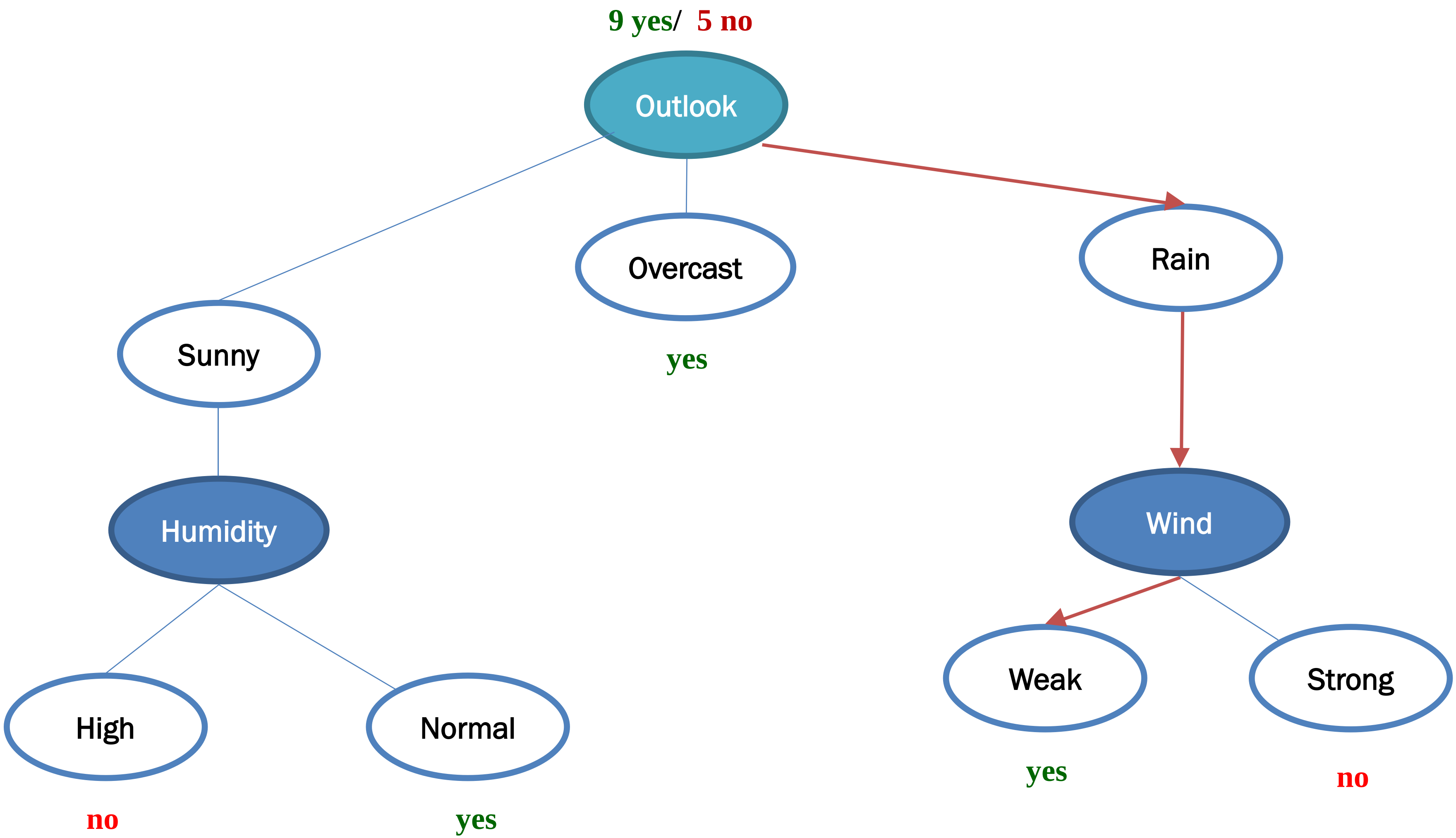
New data:

Day	Outlook	Humidity	Wind	**Play**
D15	Rain	High	Weak	?



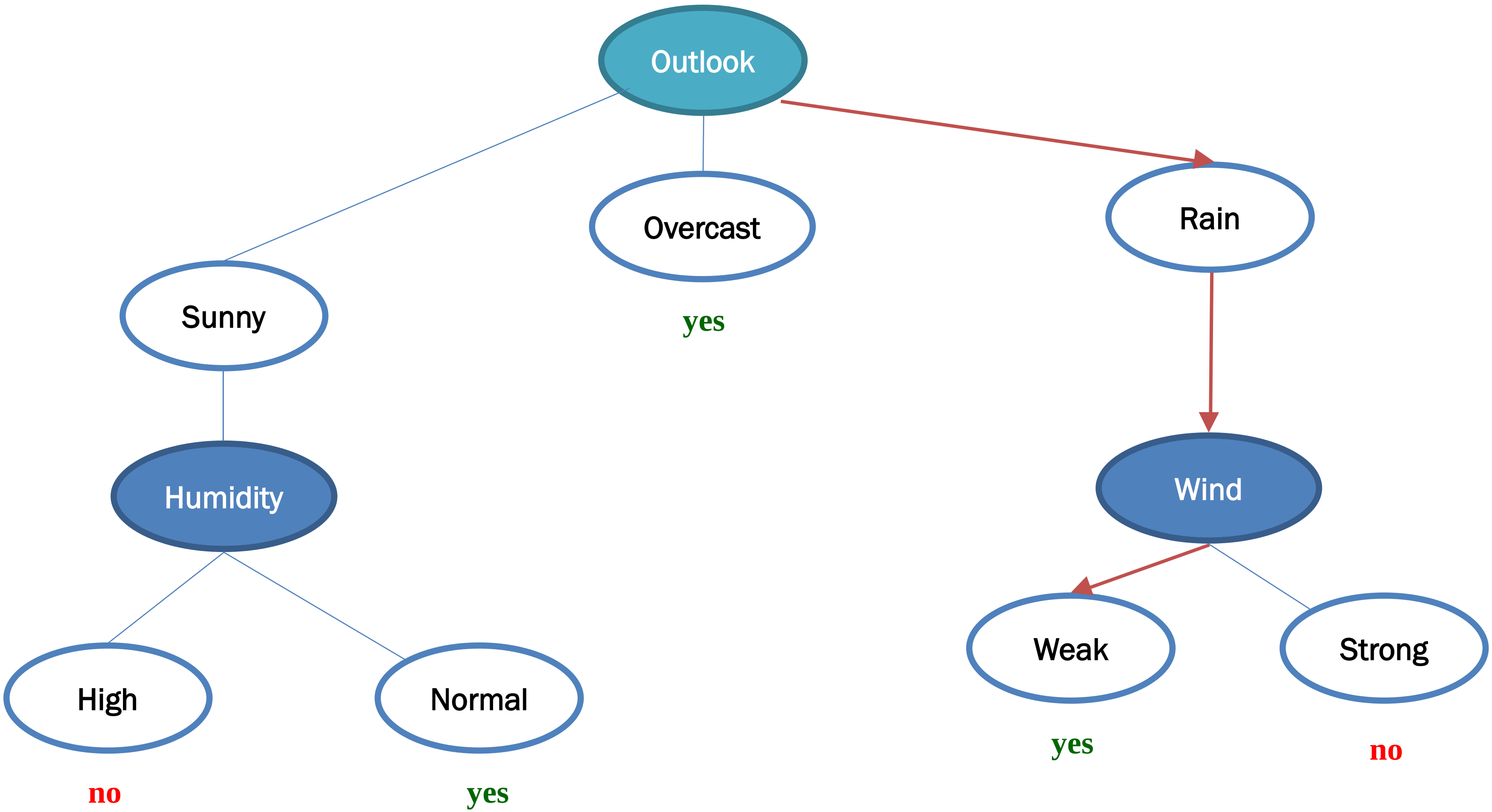
New data:

Day	Outlook	Humidity	Wind	Play
D15	Rain	High	Weak	?



New data:

Day	Outlook	Humidity	Wind	Play
D15	Rain	High	Weak	yes



New data:

Day	Outlook	Humidity	Wind	Play
D15	Rain	High	Weak	yes

Tree Induction

- Greedy strategy.
  - *Split the records based on an attribute test that optimizes certain criterion.*

- Issues
  - *Determine how to split the records*
    - How to specify the attribute test condition?
    - How to determine the best split?
  - *Determine when to stop splitting*

Tree Induction

- Greedy strategy.
  - *Split the records based on an attribute test that optimizes certain criterion.*

- Issues
  - *Determine how to split the records*
    - How to specify the attribute test condition?
    - How to determine the best split?
  - *Determine when to stop splitting*

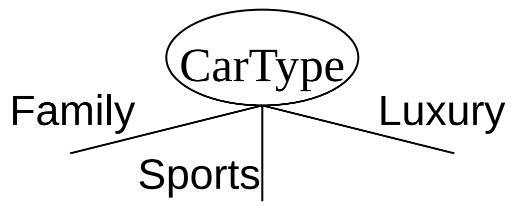
How to Specify Test Condition?

- Depends on attribute types
  - *Nominal*
  - *Ordinal*
  - *Continuous*

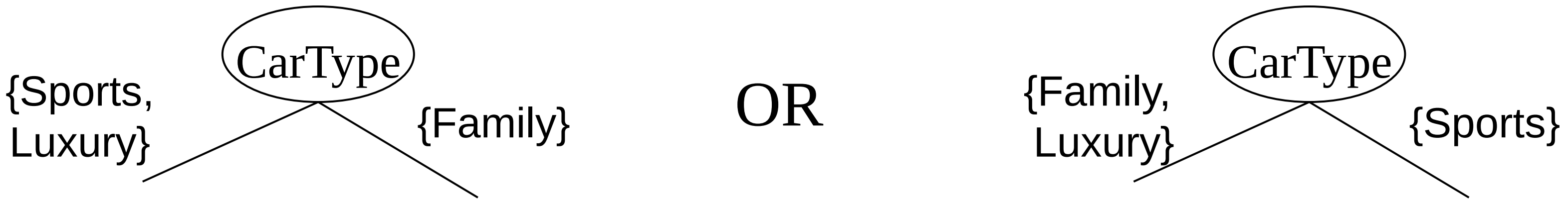
- Depends on number of ways to split
  - *2-way split*
  - *Multi-way split*

Splitting Based on Nominal Attributes

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

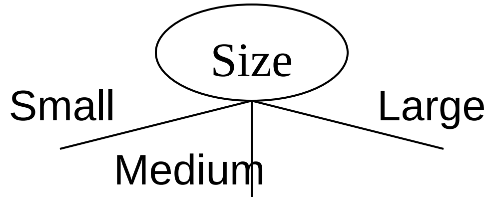


- **Binary split:** Divides values into two subsets.
  Need to find optimal partitioning.

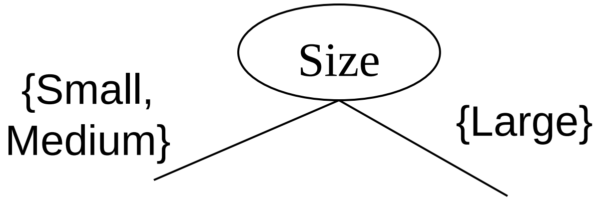


Splitting Based on Ordinal Attributes

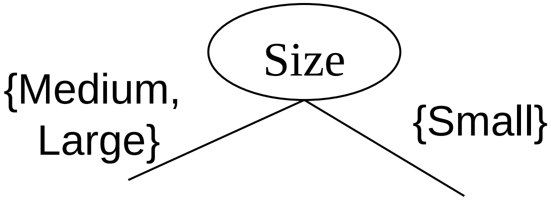
- Multi-way split: Use as many partitions as distinct values.



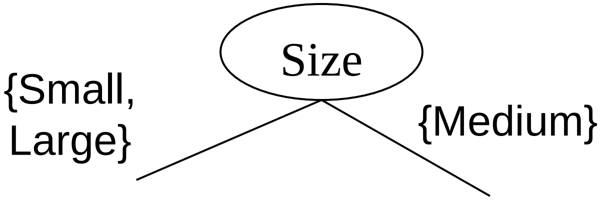
- Binary split: Divides values into two subsets.
  Need to find optimal partitioning.



OR



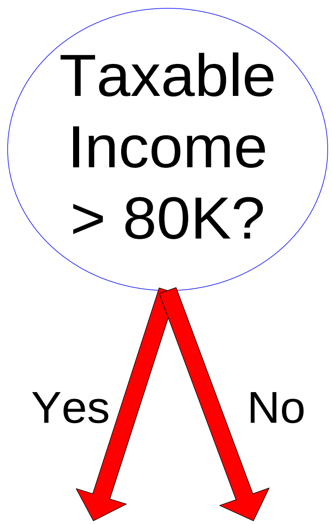
- What about this split?



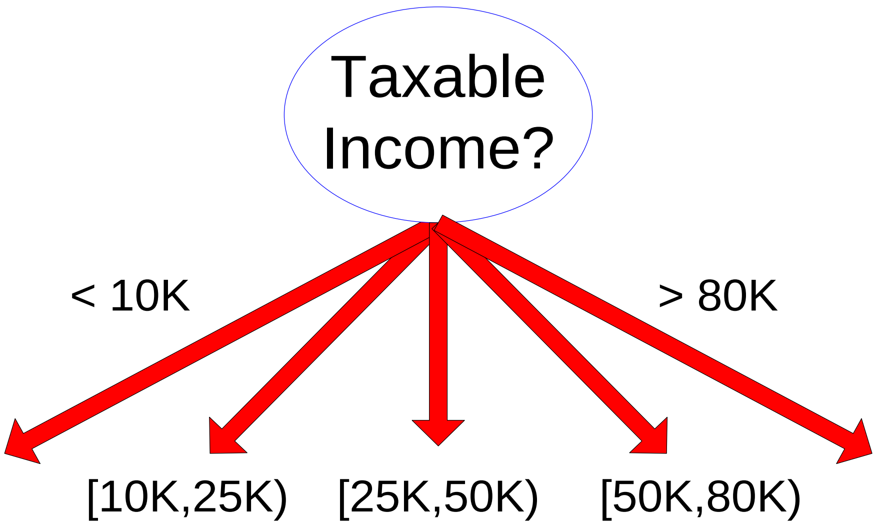
Splitting Based on Continuous Attributes

- Different ways of handling
  - *Discretization to form an ordinal categorical attribute*
    - Static – discretize once at the beginning
    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - *Binary Decision: $(A < v)$ or $(A \geq v)$*
    - consider all possible splits and finds the best cut
    - can be more computationally intensive

Splitting Based on Continuous Attributes



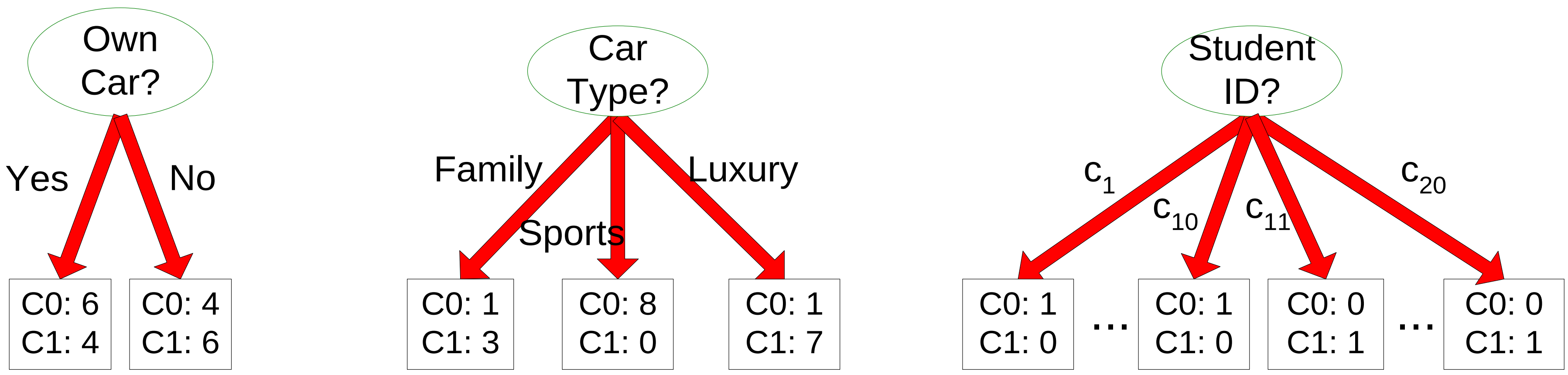
(i) Binary split



(ii) Multi-way split

How to determine the Best Split?

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



Which test condition is the best?

How to determine the Best Split?

- Greedy approach:
  – *Nodes with homogeneous class distribution are preferred*
- Need a measure of node impurity:

C0: 5 C1: 5	C0: 9 C1: 1

Non-homogeneous,

High degree of impurity

Homogeneous,

Low degree of impurity

- Ideas?

Measures of Node Impurity

p(i|t): fraction of records associated with node **t** belonging to class **i**

- Entropy

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

- Misclassification Error

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

Splitting Criteria based on Classification Error

- Classification error at a node t :

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

- Measures misclassification error made by a node.
  - Maximum $(1 - 1/n_c)$ when records are equally distributed among all classes, implying least interesting information
  - Minimum (0.0) when all records belong to one class, implying most interesting information

Computing Error of a Single Node

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

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

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

Error = 1 – max (0, 1) = 1 – 1 = 0

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

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

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

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

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

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

Example

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

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

Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

Error = 1 – max (0, 1) = 1 – 1 = 0

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

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

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

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

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

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

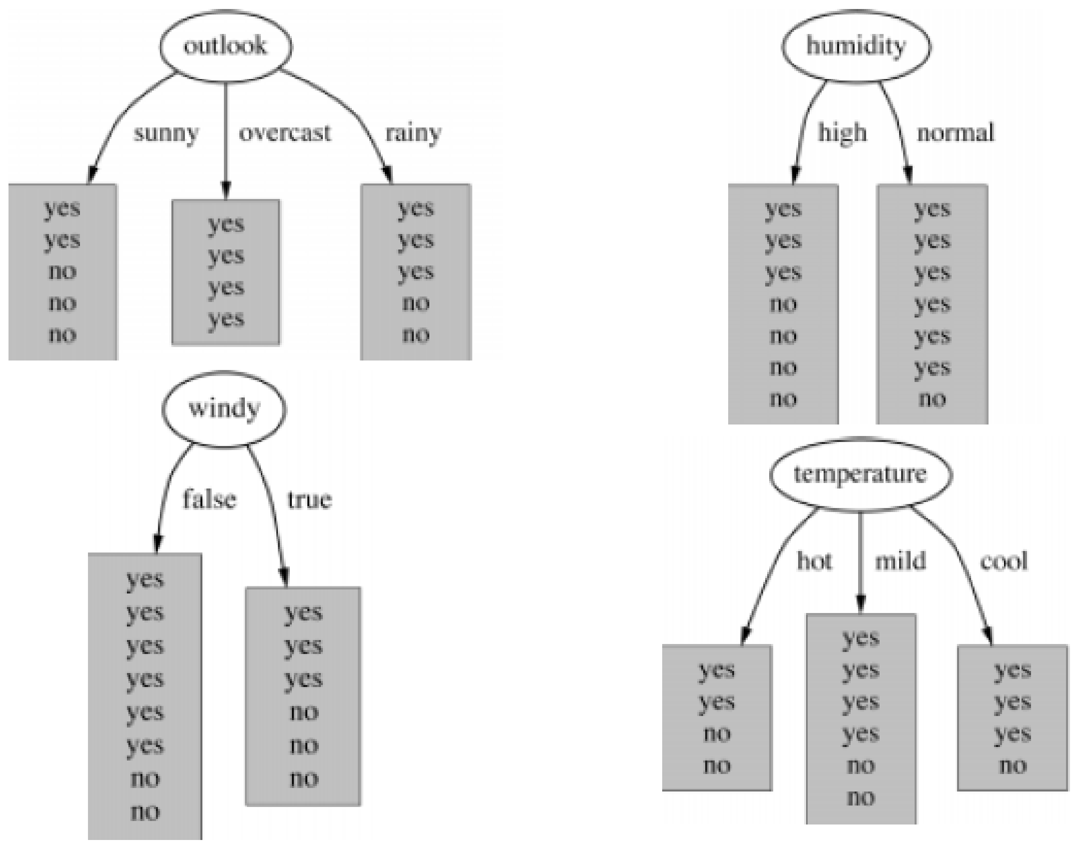
Impurity measures

- All of the impurity measures take value zero (minimum) for the case of a pure node where a single value has probability 1
- All of the impurity measures take maximum value when the class distribution in a node is uniform.

Decision Trees Classifiers

- Normal procedure: top down in recursive divide and conquer fashion
- First: attribute is selected for root node and branch is created for each possible attribute value
- Then: the instances are split into subsets (one for each branch extending from the node)
- Finally: procedure is repeated recursively for each branch, using only instances that reach the branch
- Process stops if all instances have the same class

Decision Trees: which attribute to select?



Criterion for attribute selection

- Which is the best attribute?
  - *The one which will result in the smallest tree*
  - *Heuristic: choose the attribute that produces the "purest" nodes*
- Popular purity criterion: information gain
  - *Information gain increases with the average purity of the subsets that an attribute produces*
- Strategy: choose attribute that results in greatest information gain

Computing Information Gain

- Information is measured in bits
  - *Given a probability distribution, the info required to predict an event is the distribution's entropy*
  - *Entropy gives the information required in bits (this can involve fractions of bits!)*
- Formula for computing the entropy:

$$\text{entropy}(p_1, p_2, \ldots, p_n) = -p_1 \log p_1 - p_2 \log p_2 \ldots - p_n \log p_n$$

Where p_1, p_2, …., p_n denote the probability of occurrence of outcomes for class 1, 2, …, n respectively

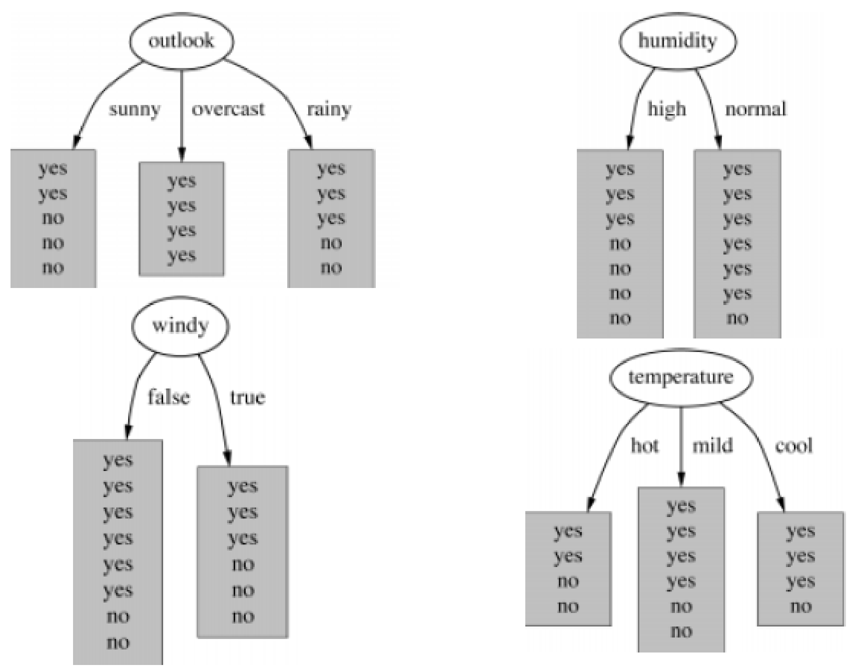
Example: Attribute Outlook

- "Outlook" = "Sunny":

$$\text{info}([2,3]) = \text{entropy}(2/5,3/5) = -2/5\log(2/5) - 3/5\log(3/5) = 0.971 \text{ bits}$$

- "Outlook" = "Overcast":

$$\text{info}([4,0]) = \text{entropy}(1,0) = -1\log(1) - 0\log(0) = 0 \text{ bits}$$



Example: Attribute Outlook

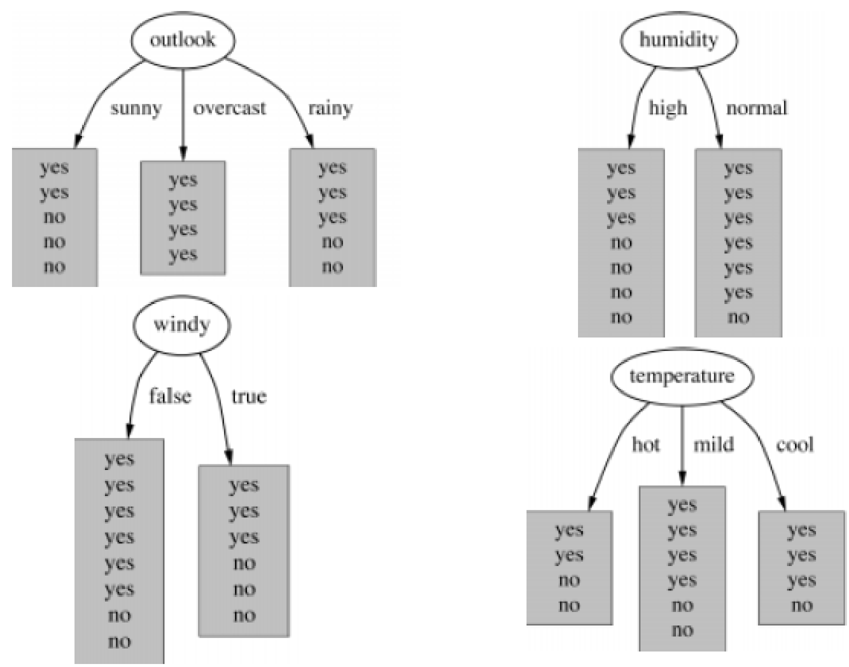
- "Outlook" = "Rainy":

$$\text{info}([3,2]) = \text{entropy}(3/5, 2/5) = -3/5\log(3/5) - 2/5\log(2/5) = 0.971\text{ bits}$$

- Expected information for attribute:

$$\text{info}([3,2],[4,0],[3,2]) = (5/14)\times 0.971 + (4/14)\times 0 + (5/14)\times 0.971$$

$$= 0.693\text{ bits}$$



Computing Gain

- Information gain: information before splitting – information after splitting

gain(“Outlook”) = info([9,5]) – info([2,3],[4,0],[3,2]) = 0.940-0.693

- Information gain for attributes from weather data:

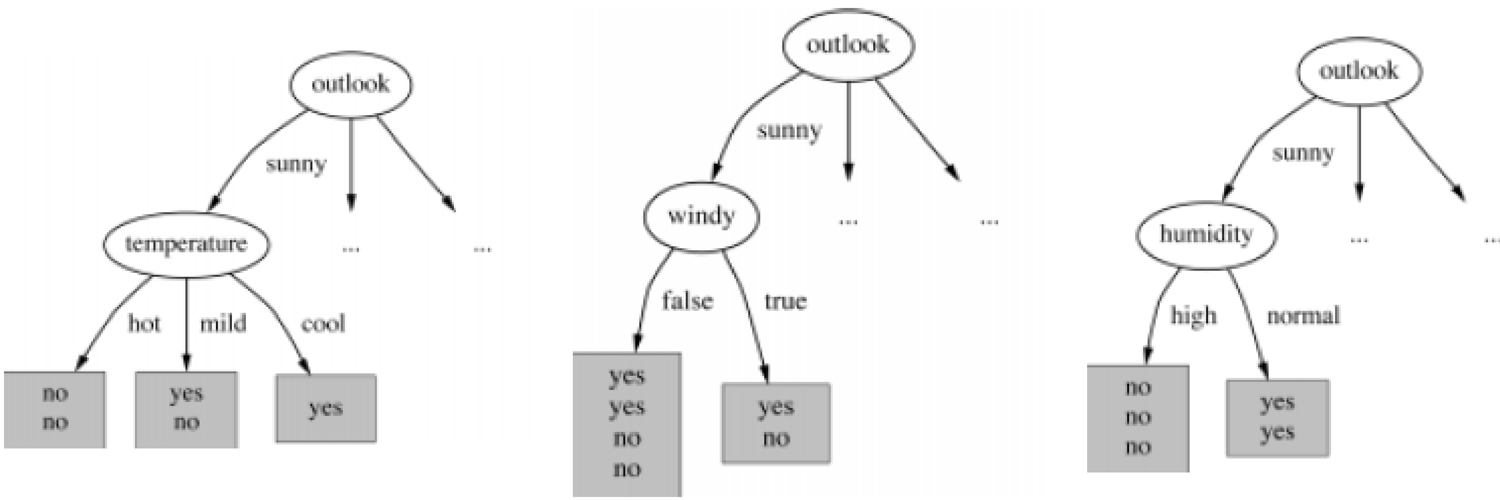
$$\text{gain}(\text{"Outlook"}) = 0.247 \text{ bits}$$

$$\text{gain}(\text{"Temperature"}) = 0.029 \text{ bits}$$

$$\text{gain}(\text{"Humidity"}) = 0.152 \text{ bits}$$

$$\text{gain}(\text{"Windy"}) = 0.048 \text{ bits}$$

Continuing the split



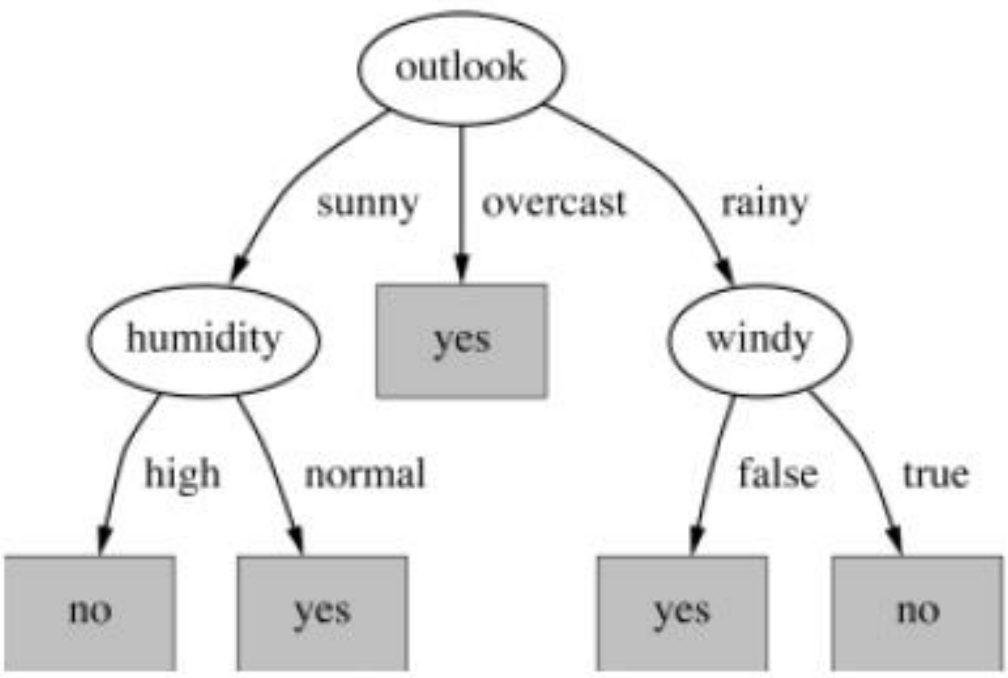
$$\text{gain}(\text{"Temperature"}) = 0.571 \text{ bits}$$

$$\text{gain}(\text{"Humidity"}) = 0.971 \text{ bits}$$

$$\text{gain}(\text{"Windy"}) = 0.020 \text{ bits}$$

The final tree

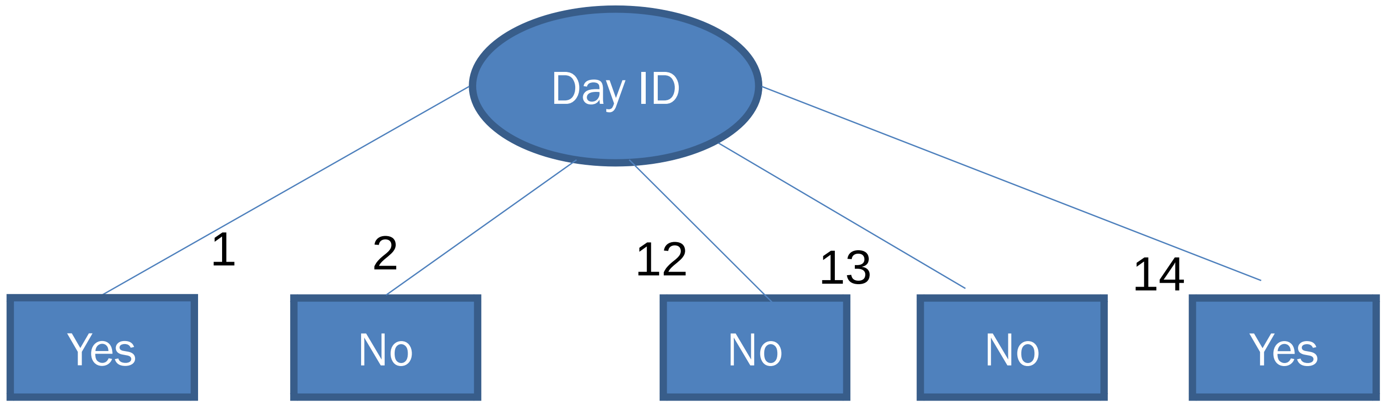
- Note: not all leaves need to be pure
- Some leaf nodes will have instances belonging to more than one class – in this case the leaf node is labelled with the *majority* class
- Splitting stops when data can't be split any further – when we run out of attributes to split on or the number of instances at the node to be split is less than a specified minimum number (MinNumObj in Weka)



Getting rid of bias introduced by Information Gain (IG)

- IG is a good method of identifying splits in the tree but it can cause problems for features with high cardinality
- In the weather example the Day ID perfectly correlates to the right play decision in *historical data*
- This means that Day ID has the highest IG value
- However, it has no power whatsoever to predict decisions, hence no generalization capability!
- The solution in this case is to use Gain Ratio

Problem with highly branching features



- Remember there were 9 playing days and 5 non playing days and so:

  info(9,11)=1/14(entropy(0,1)+entropy(0,1)+...........+entropy(0,1))
  =0

- Thus it follows that Day ID has the maximum information gain (=0.94)

Intrinsic Information

- The problem occurs due to the high intrinsic information held by highly branching features
- Intrinsic Information contained by a split S created by a feature F is given by:

I(F,S)$=-\sum_i \frac{|B_i|}{|B|} \log\left(\frac{|B_i|}{B}\right)$ where $|B_i$ is the number of data instances on branch i and $|B|$ is the total number across all branches (size of training dataset)

- For day ID, I$=14\left(-\frac{1}{14} log\left(\frac{1}{14}\right)\right)=3.807$ which is the highest out of all the features

Gain Ratio

- We are now in a better position to get rid of troublesome features
- We define Gain Ratio (GR) as the ratio of Information Gain (IG) to Intrinsic Information (IS)

$$GR(F,S) = \frac{IG(F,S)}{I(F,S)}$$ where F is the feature and S is the split

Feature\Measure	Information Gain	Gain Ratio
Outlook	0.247	0.157
Humidity	0.152	0.152
Windy	0.029	0.049
Temperature	0.048	0.019
Day ID	**0.940**	0.246

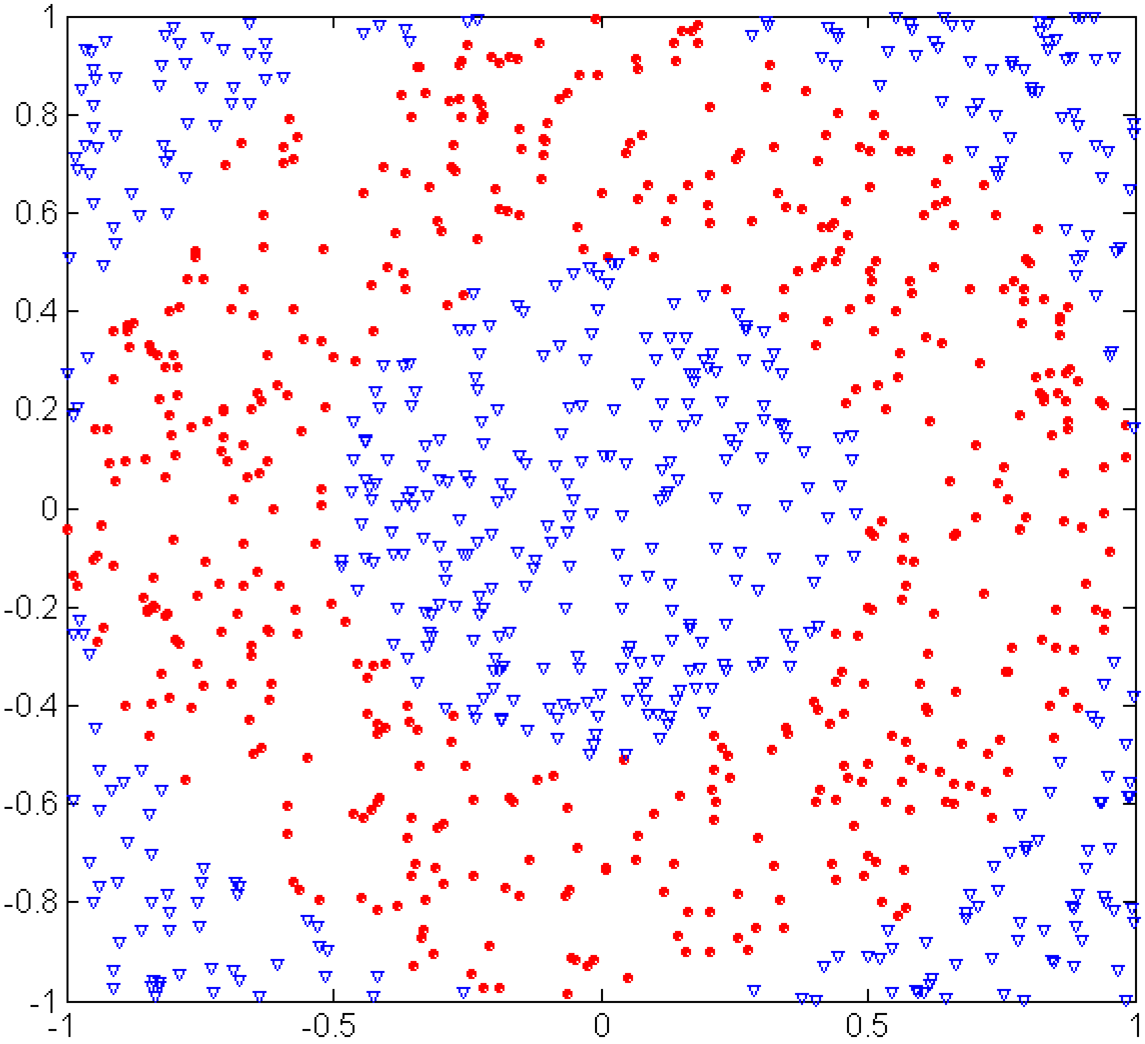
Gain Ratio vs Information Gain

- Thus it can be seen that GR has narrowed the gap between Day ID and Outlook
- However, Day ID *still wins* as it has the highest Gain Ratio
- The lesson here is that the data miner *has to use his/her judgement and not rely totally on automated tools*
- It is easy for the human data miner to identify that Day ID is not suitable as it has no predictive power (generalizability).

Practical Issues of Classification

- Underfitting and Overfitting
- Missing Values
- Costs of Classification

Under-fitting and Over-fitting (Example)



500 circular and 500 triangular data points.

Circular points:

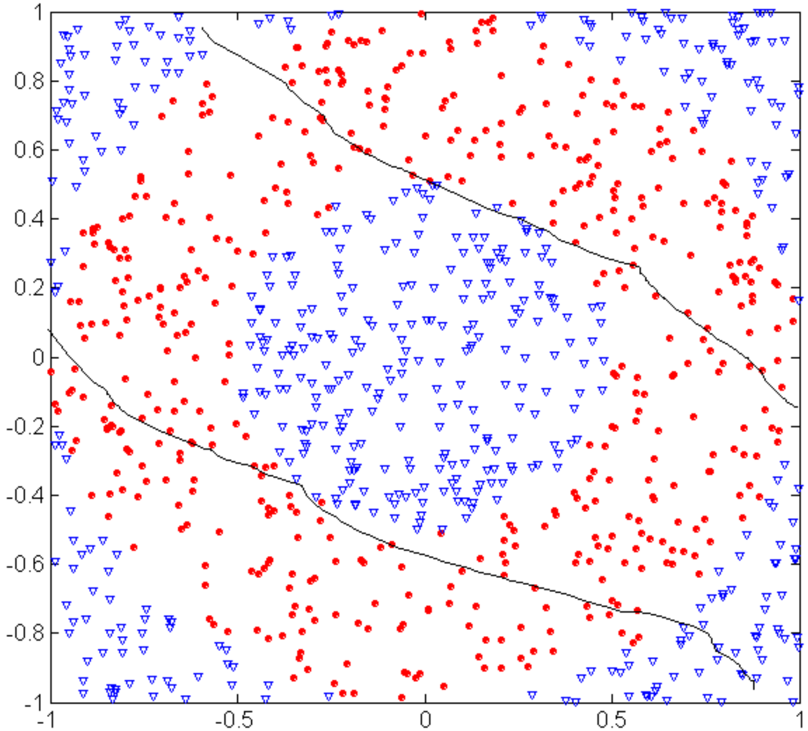
$0.5 \leq \mathrm{sqrt}(x_1^2+x_2^2) \leq 1$

Triangular points:

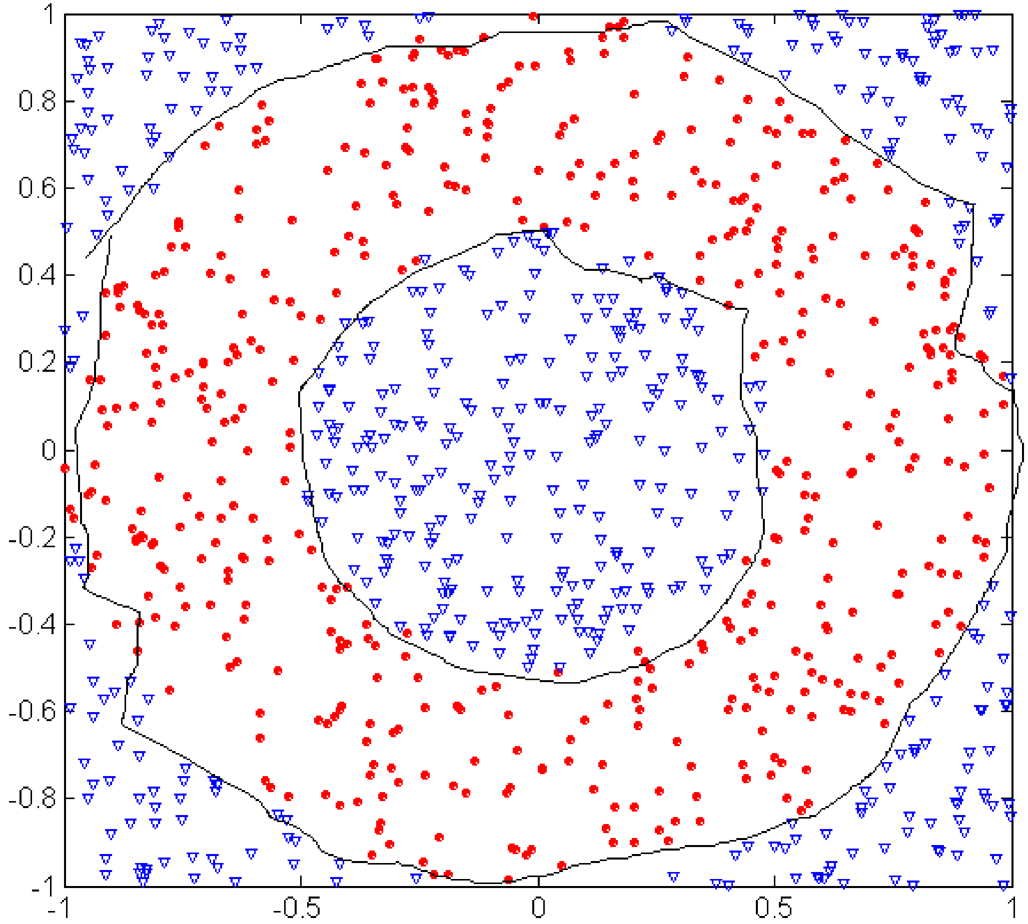
$\mathrm{sqrt}(x_1^2+x_2^2) < 0.5$ or

$\mathrm{sqrt}(x_1^2+x_2^2) > 1$

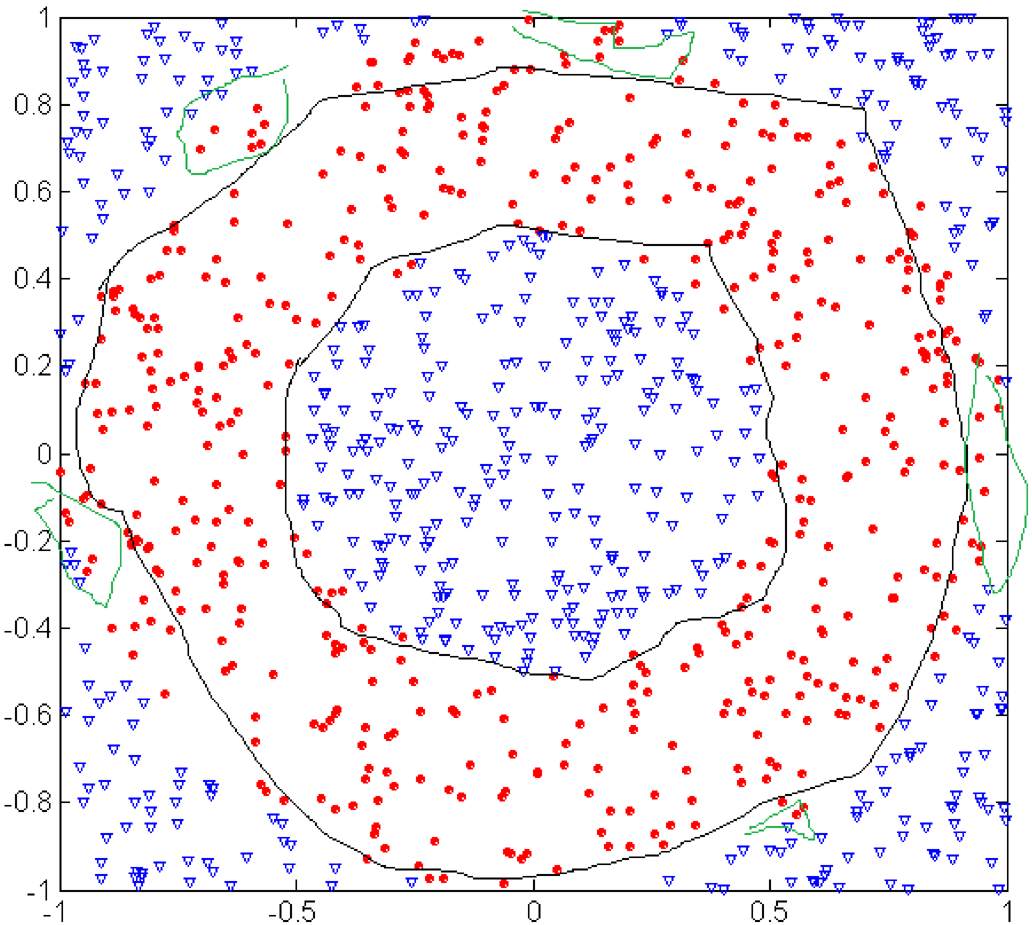
Is this Model Good?



What about this?



And This?



Underfitting and Overfitting

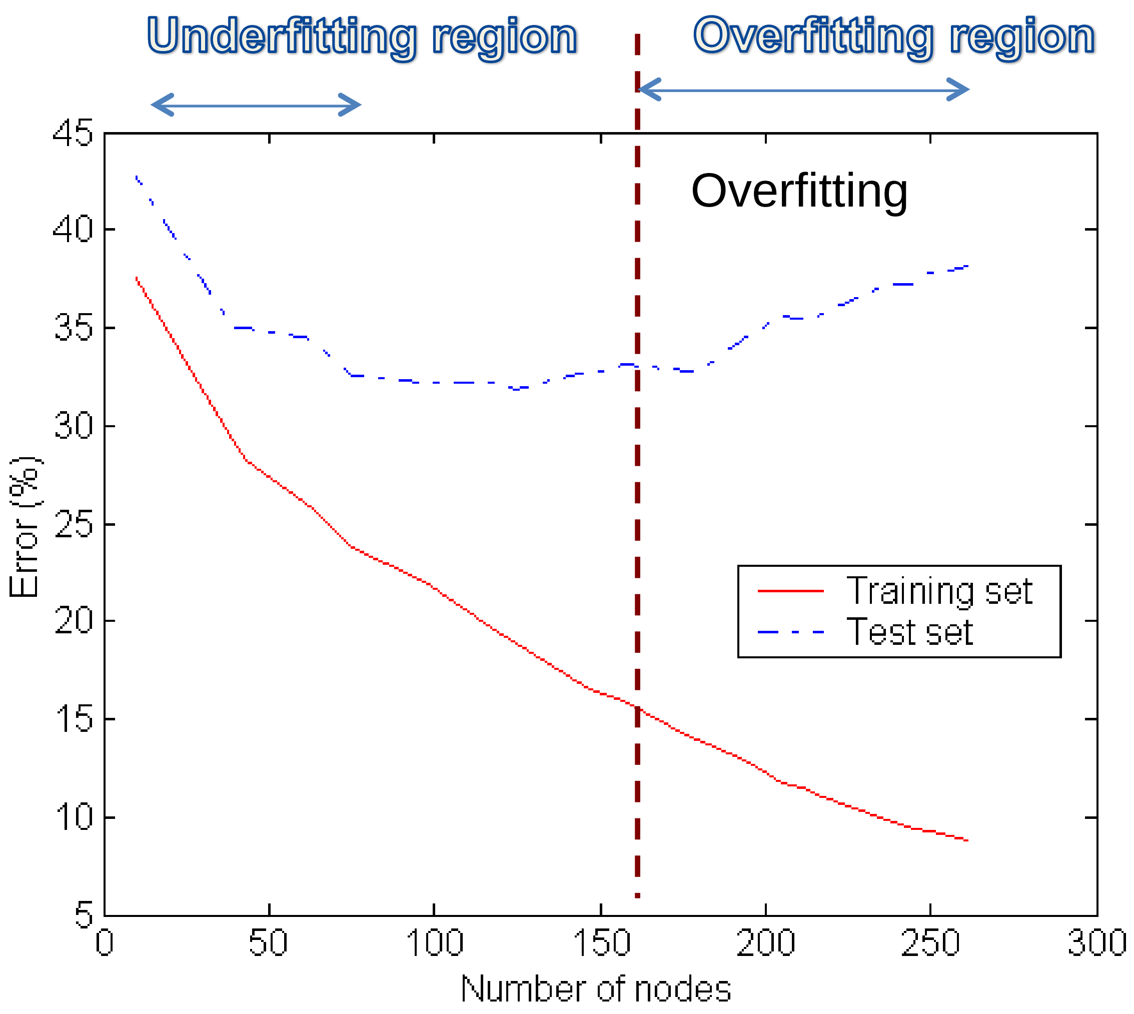
- Two problems that can arise with models developed with Data Mining are: Overfitting and Underfitting
- Underfitting occurs when the model has not fully learned all the patterns in the data, resulting in poor prediction accuracy (test accuracy).
- Underfitting is generally caused by *the inability of the algorithm to find all patterns in the training dataset.*
- In the case of a Decision Tree method the tree developed is not of sufficient depth and size to learn all the patterns present in the data

Overfitting

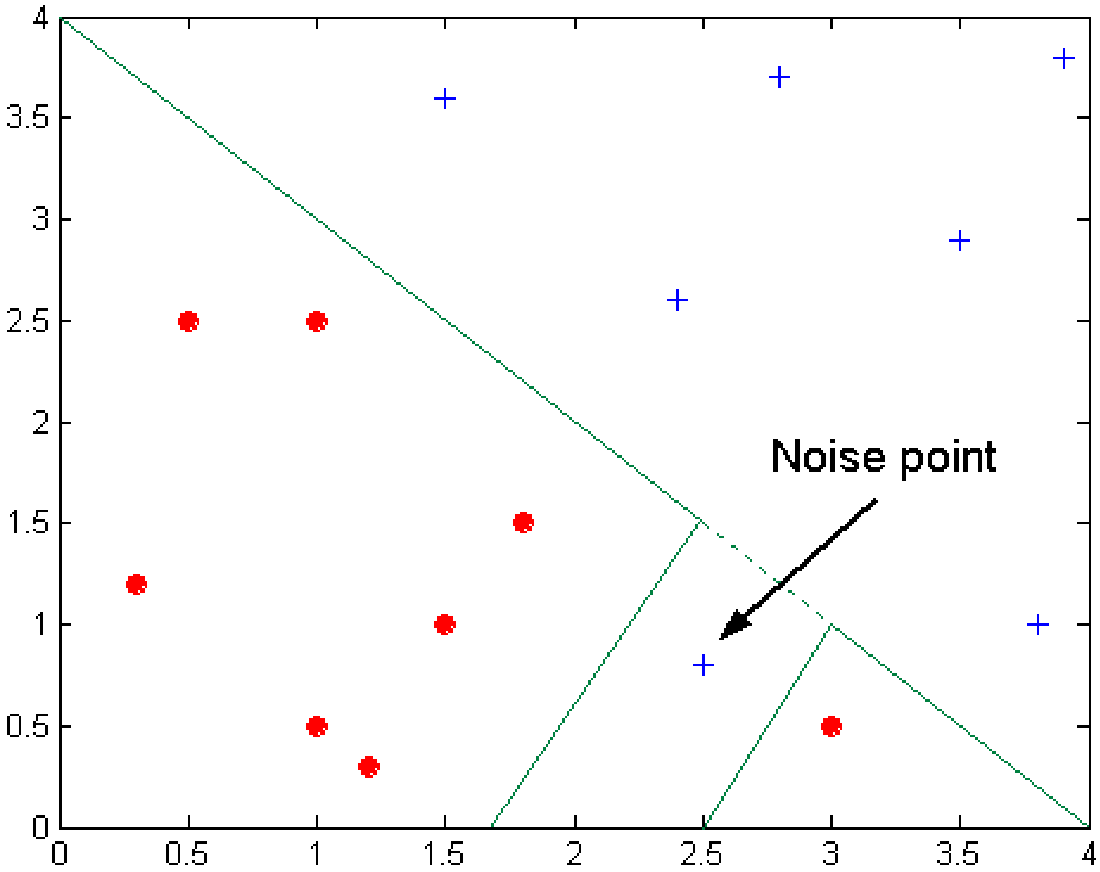
- With Overfitting the model's learns the patterns in the training data very well but the model learnt cannot predict newly arriving data well
- In other words accuracy on training dataset is high but accuracy drops drastically on newly arriving data – training set accuracy >> test set accuracy
- In the case of the Decision tree method the tree developed is too detailed (too large in size)
- Overfitting is generally caused by:

1. Noise (errors in assigning class labels) in the training dataset
2. Lack of sufficient data to capture certain types of patterns

Underfitting and Overfitting

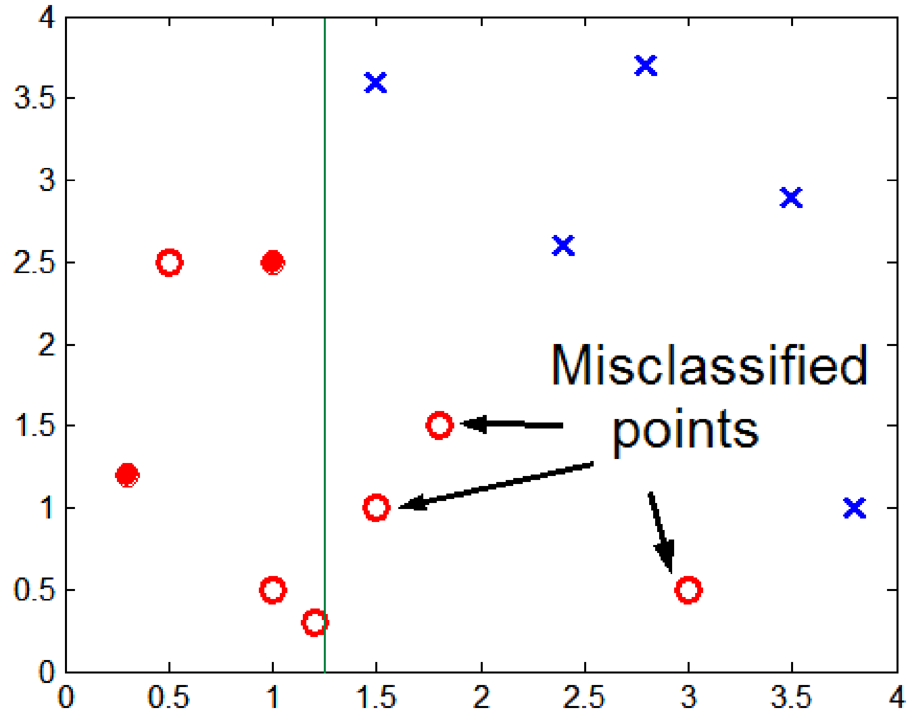


Overfitting due to Noise



Decision boundary is distorted by noise point

Overfitting due to Insufficient Examples



Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

- Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary

- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records

- Need new ways for estimating errors

Methods for estimating the error

- Re-substitution errors: error on training (Σ **e(t)**)
- Generalization errors: error on testing (Σ **e'(t)**)

- Methods for estimating generalization errors:
  - Optimistic approach: **e'(t) = e(t)**
  - Pessimistic approach:
    - For each leaf node: **e'(t) = (e(t)+0.5)**
    - Total errors: **e'(T) = e(T) + N × 0.5** (N: number of leaf nodes)
    - For a Tree with 30 leaf nodes and 10 errors on training (out of 1000 instances):

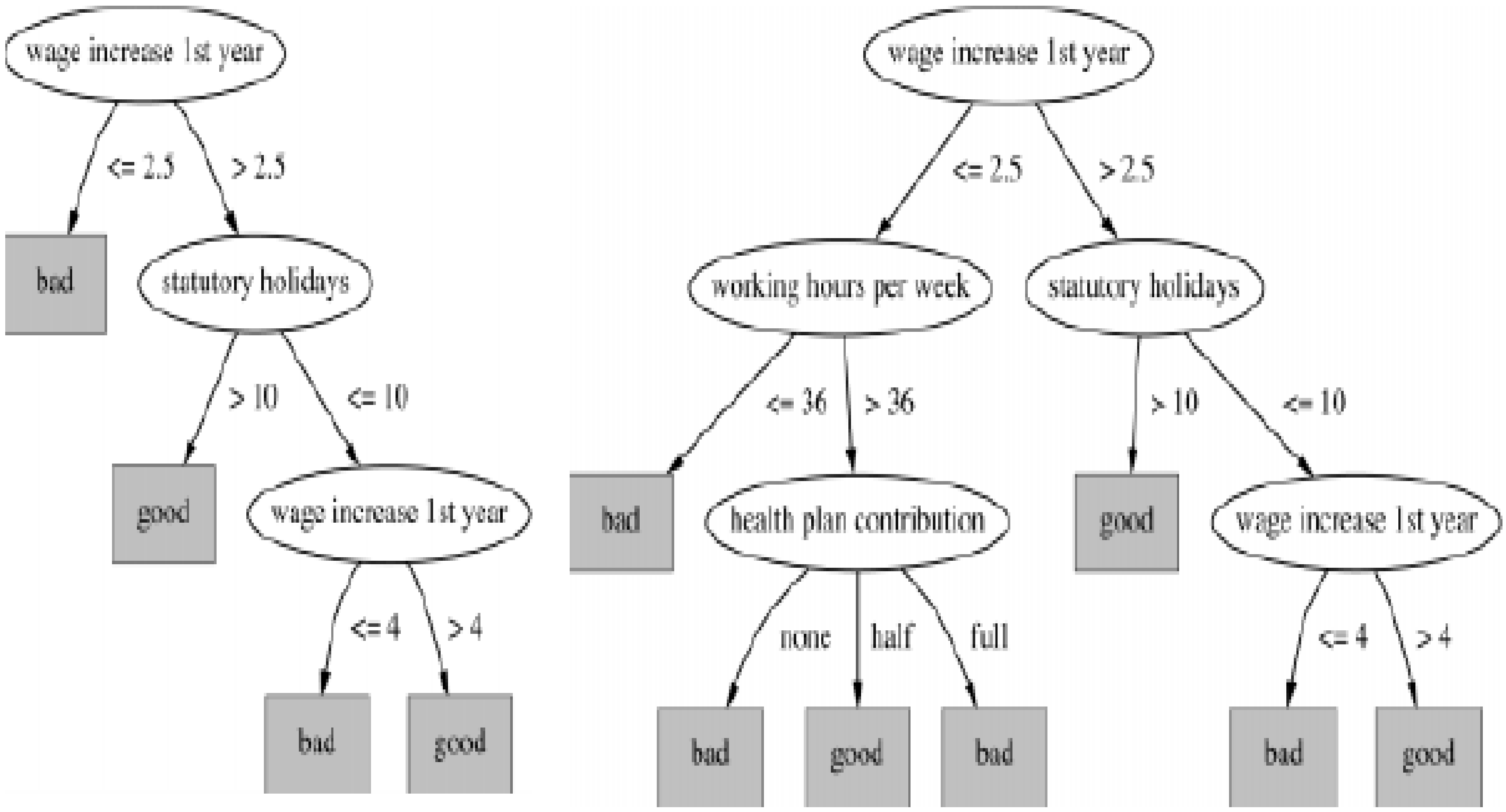
    Training error = 10/1000 = 1%

  Generalization error = (10 + 30×0.5)/1000 = 2.5%
  - Reduced error pruning (REP):
    - uses ***validation data set*** to estimate generalization error

How to Address Overfitting…

- Post-pruning
  - *Grow decision tree to its entirety*
  - *Trim the nodes of the decision tree in a bottom-up fashion*
  - *If generalization error improves after trimming, replace sub-tree by a leaf node.*
  - *Class label of leaf node is determined from majority class of instances in the sub-tree*

Decision Trees for the labour data



Occam's Razor

- Given two models of similar generalization errors, one should prefer the simpler model over the more complex model

- For complex models, there is a greater chance that it was fitted accidentally by errors in data

- Therefore, one should include model complexity when evaluating a model

Example of Post-Pruning

Class = Yes	20
Class = No	10
Error = 10/30	

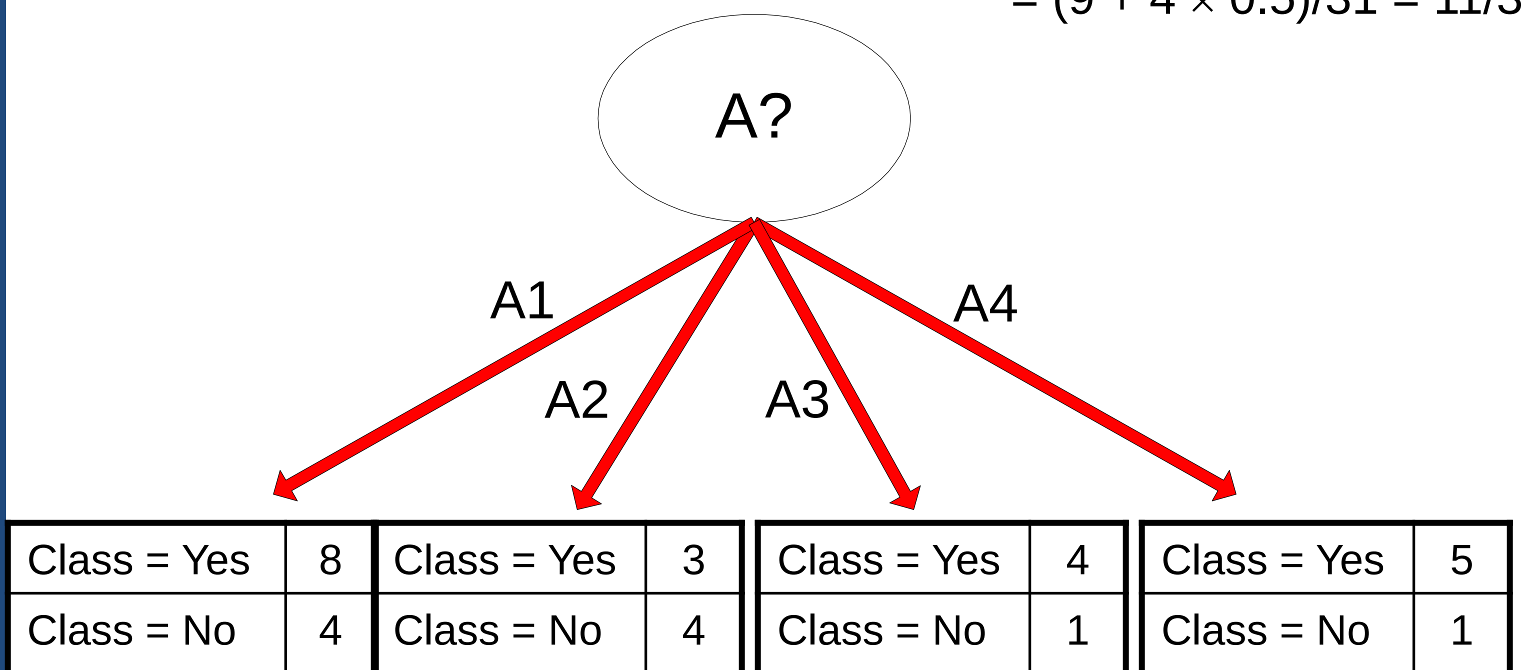
Training Error (Before splitting) = 10/30

Pessimistic error = (10 + 0.5)/31 = 10.5/31

Training Error (After splitting) = 9/30

Pessimistic error (After splitting)

$$= (9 + 4 \times 0.5)/31 = 11/31$$



Discussion

- Algorithm for top-down induction of decision trees ("ID3") was developed by Ross Quinlan
  - *ID3 only deals with nominal attributes*
  - *Led to development of C4.5, which can deal with numeric attributes, missing values, and noisy data*
- Similar approach: CART
- There are many other attribute selection criteria! (but with almost no difference in accuracy of result.)

Decision Tree Based Classification

- Advantages:
  - *Inexpensive to construct*
  - *Extremely fast at classifying unknown records*
  - *Easy to interpret for small-sized trees*
  - *Robust to noise (especially when methods to avoid overfitting are employed)*
  - *Can easily handle redundant or irrelevant attributes (unless the attributes are interacting)*
- Disadvantages:
  - *Space of possible decision trees is exponentially large. Greedy approaches are often unable to find the best tree.*
  - *Does not take into account interactions between attributes*
  - *Each decision boundary involves only a single attribute*

References

1. *Data Mining: Practical Machine Learning Tools and Techniques (3rd edition)* / Ian Witten, Eibe Frank; Elsevier, 2011, Chapter 4