Spring 2019

CSEN1083: Data Mining

Classification (2)

Seif Eldawlatly

Classification (2)

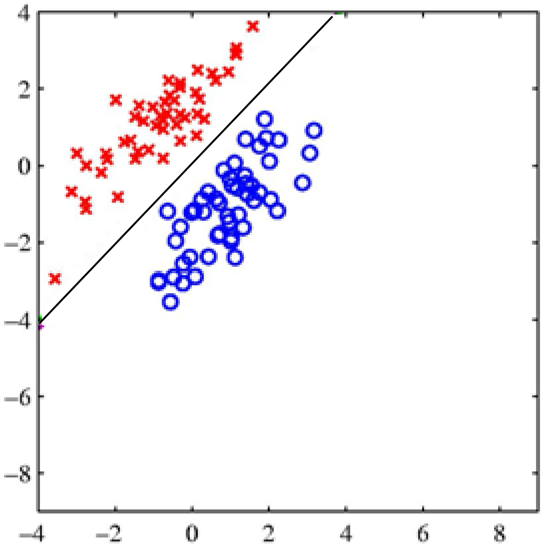
- Reference for this Lecture:

“Pattern Recognition and Machine Learning,” Christopher M. Bishop, Springer, 2006

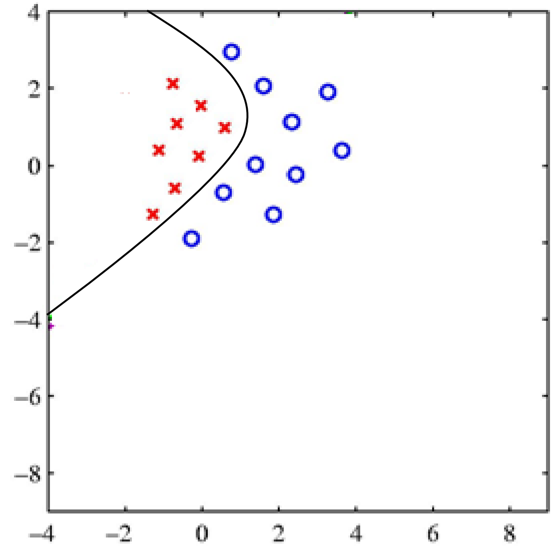
Linear vs. Non-linear

- Decision Boundary

Linearly Separable



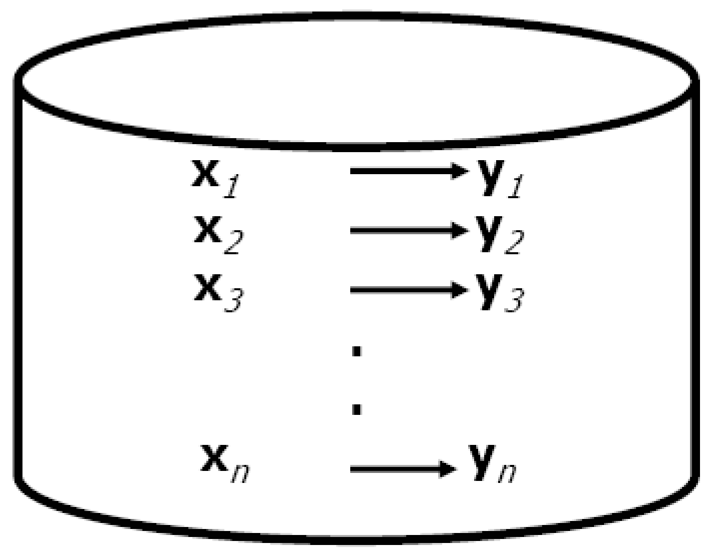
Non-linearly Separable



Instance-based Learning

- Each time a new instance is encountered, its relationship to previously stored instances is examined

- Disadvantage: Computation cost is high
  - To classify a new point, search database for similar points and fit with local points



K-nearest Neighbor (KNN) Classifier

- Most basic instance-based method

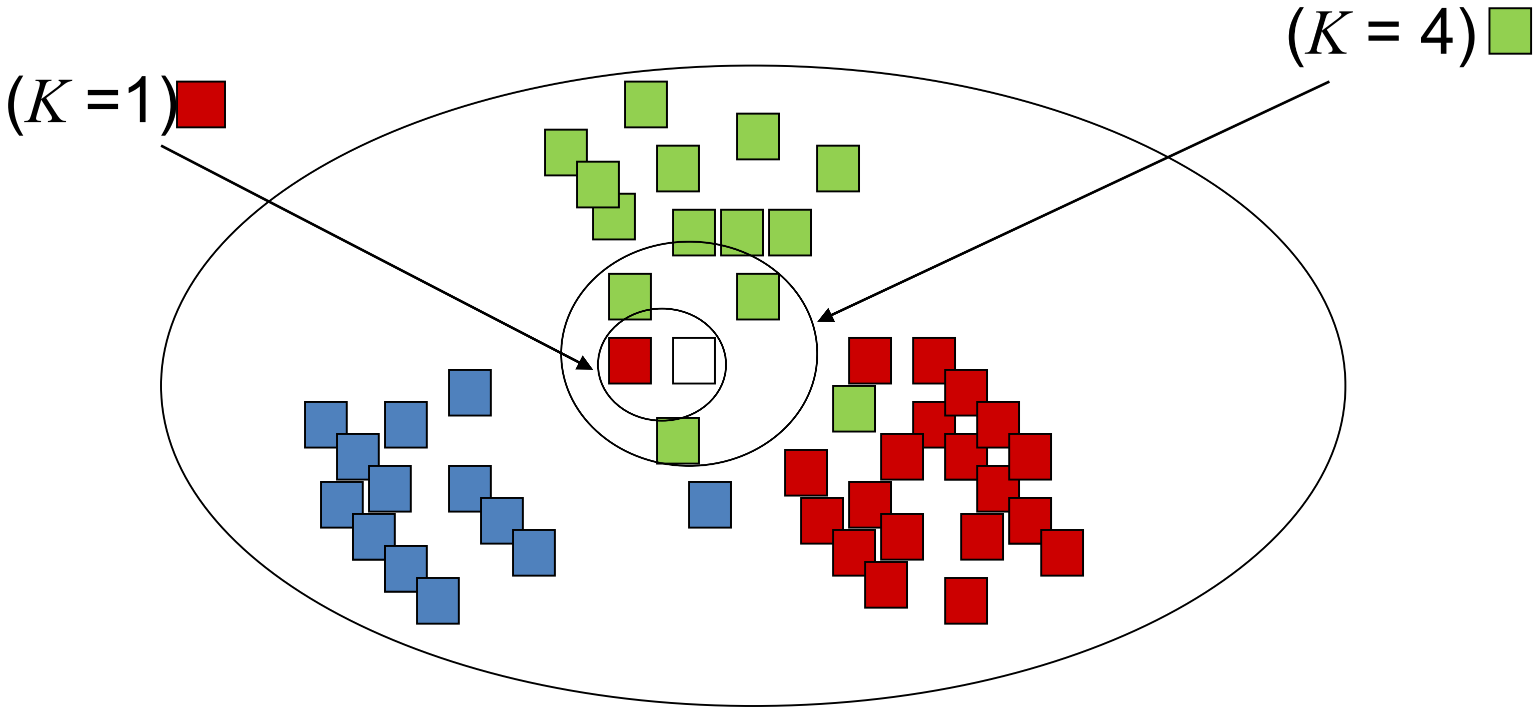
- Uses Euclidean distance to determine how dissimilar a pair of points are

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{r=1}^{n} (x_{ir} - x_{jr})^2}$$

- For any new input vector, the nearest K points are considered

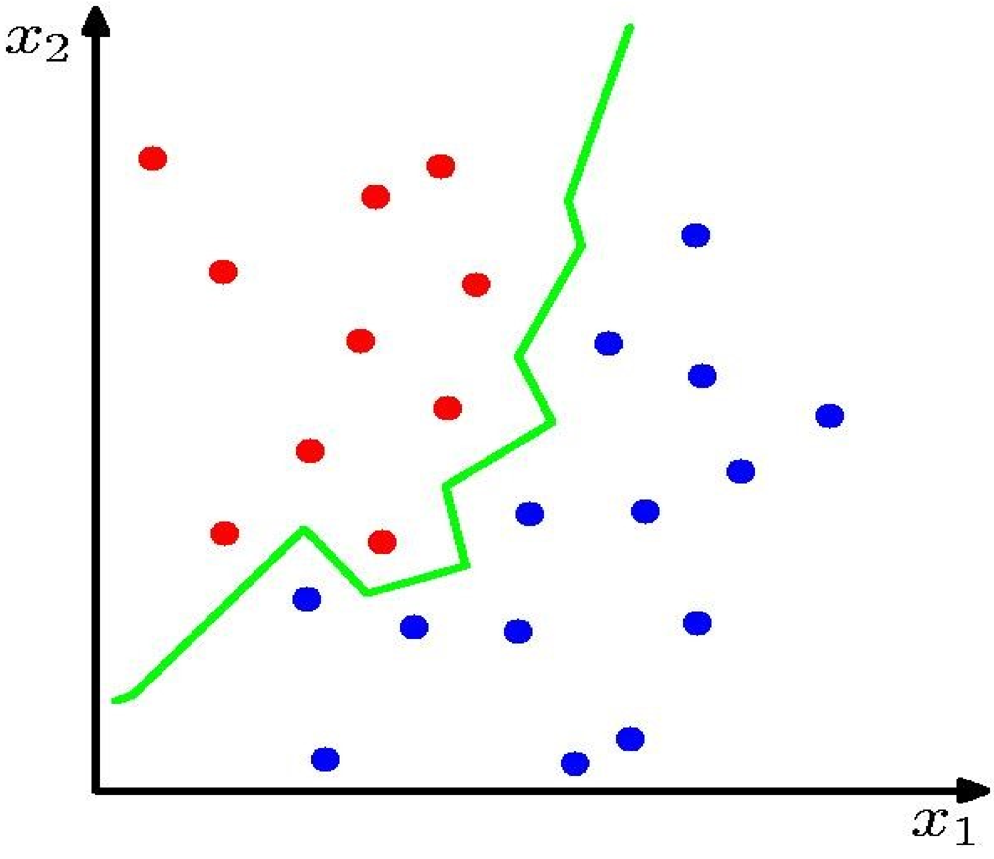
- A majority voting scheme is used to classify the new input vector

K-nearest Neighbor (KNN) Classifier



K-nearest Neighbor (KNN) Classifier

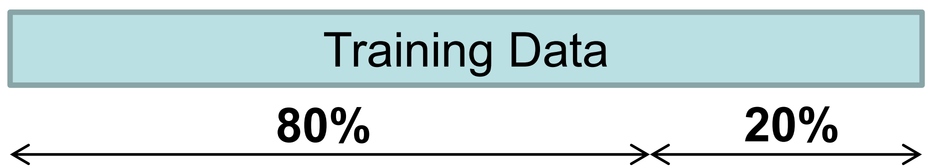
- A non-linear classifier



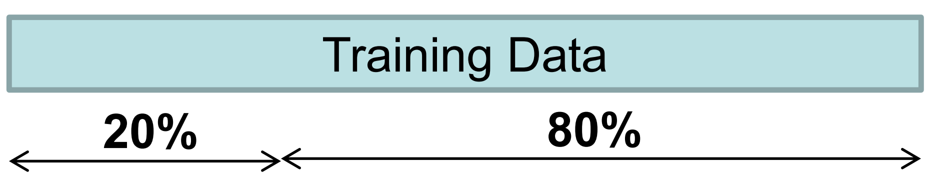
How to Choose *K*?

a) Cross-validation:

- 80% of training data for training and 20% for validation
- Find target value of the 20% part using the 80% and compute the corresponding error



$$\begin{cases} K = 1 \rightarrow \text{Error} = E_1 \\ K = 2 \rightarrow \text{Error} = E_2 \\ \vdots \\ K = M \rightarrow \text{Error} = E_M \end{cases}$$



$$\begin{cases} K = 1 \rightarrow \text{Error} = E_1 \\ K = 2 \rightarrow \text{Error} = E_2 \\ \vdots \\ K = M \rightarrow \text{Error} = E_M \end{cases}$$

The partitioning and validation process is repeated a number of times (for example 10 times) with different partitioning

How to Choose *K*?

a) Cross-validation:

- Find $K = k^*$ that minimizes the average error for the validation data

$$k^* = \arg\min_{k} \overline{E_k} \qquad \text{, where} \qquad \overline{E_k} = \frac{1}{L}\sum_{l=1}^{L} E_l$$

k = 1, 2, …, M, where M is the maximum number of neighbors

L is the total number of partitionings examined

- The obtained K is then used to classify the test data

How to Choose *K*?

b) Leave-one-out method

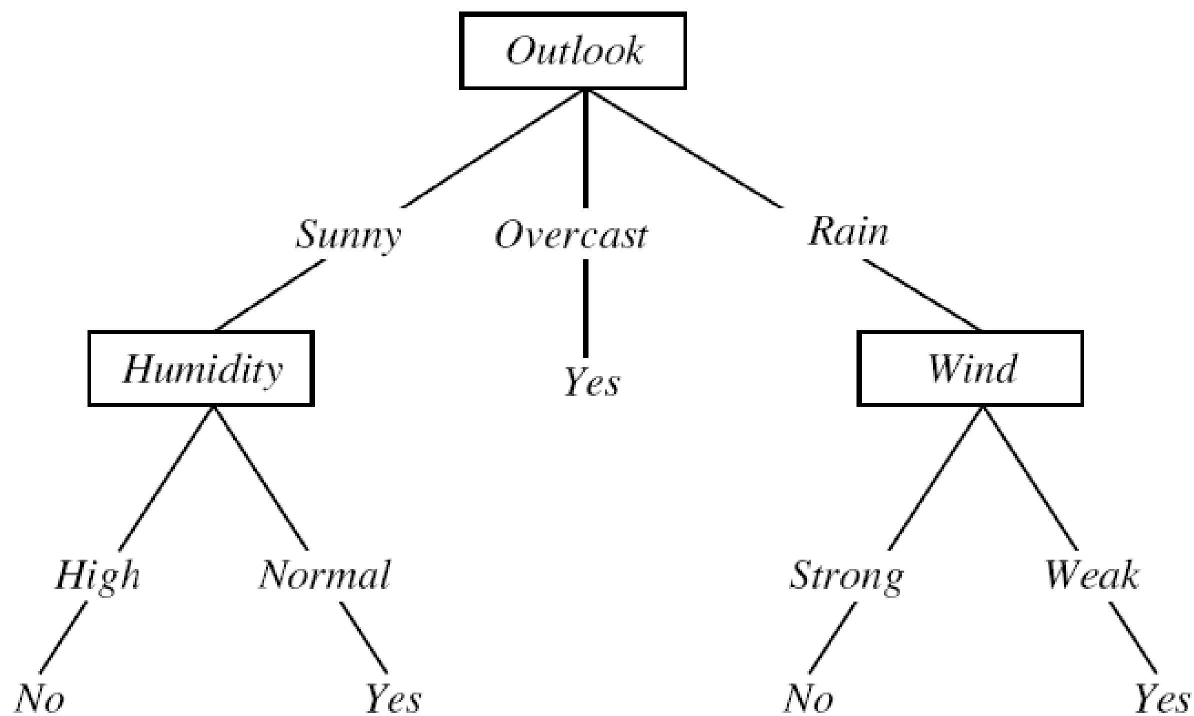
This method is equivalent to the previous cross-validation but with 1 validation point at a time

- For $k = 1, 2, \ldots, K$
  - $err(k) = 0$
  - For $i = 1, 2, \ldots, n$
    - Predict the class label $\widehat{y}_i$ for $\mathbf{x}_i$ using the remaining data points
    - $err(k) = err(k) + 1$ if $\widehat{y}_i \neq y_i$
- Output $k^* = \underset{1 \leq k \leq K}{\arg\min}\, err(k)$

Decision Tree Learning

- Decision Tree Learning: A method for approximating discrete-values target functions
- Decision Tree for Playing Tennis

  Play Tennis = {Yes, No}



Decision Tree Learning

- Decision Tree Representation:
  - Each internal node tests an attribute
  - Each branch corresponds to an attribute value
  - Each leaf node assigns a classification

- The Play Tennis decision tree corresponds to the expression

$$(\text{Outlook} = \text{Sunny} \wedge \text{Humidity} = \text{Normal})$$
$$\vee\ (\text{Outlook} = \text{Overcast})$$
$$\vee\ (\text{Outlook} = \text{Rain} \wedge \text{Wind} = \text{Weak})$$

Decision Tree Learning

- How to build the decision tree?

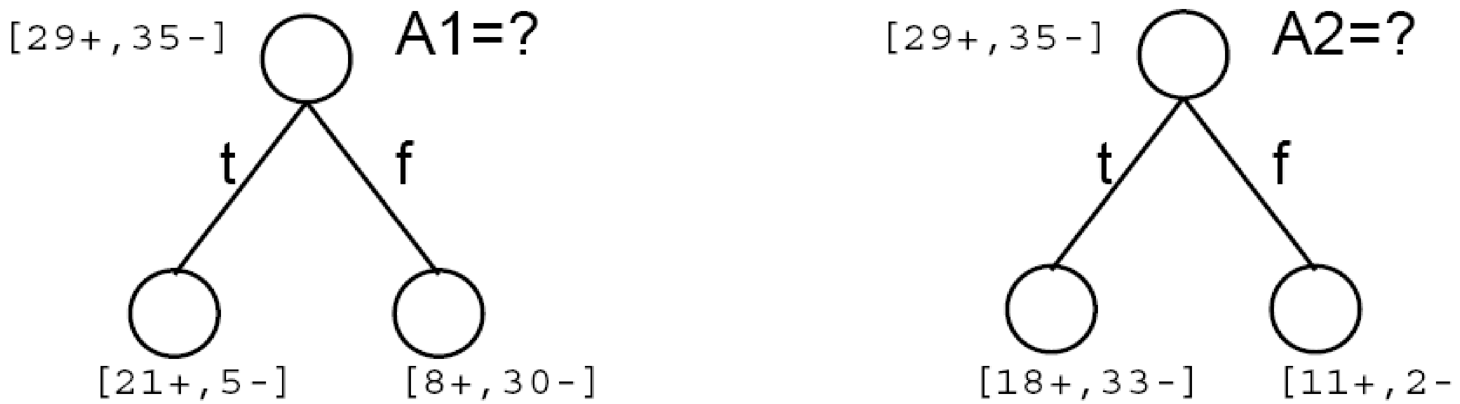
  Using ID3 (Iterative Dichotomiser 3) Algorithm

ID3 (Examples, Target_Attribute, Attributes)
- *Create a root node for the tree*
- *If all examples are positive, Return the single-node tree Root, with label = +.*
- *If all examples are negative, Return the single-node tree Root, with label = -.*
- *If number of predicting attributes is empty, then Return the single node tree Root, with label = most common value of the target attribute in the examples.*
- *Otherwise Begin*
  - *A = The Attribute that best classifies examples.*
  - *Decision Tree attribute for Root = A.*
  - *For each possible value,* v_i *, of A,*
    - *Add a new tree branch below Root, corresponding to the test A =* v_i.
    - *Let Examples(*v_i*) be the subset of examples that have the value* v_i *for A*
    - *If Examples(*v_i*) is empty*
      - *Then below this new branch add a leaf node with label = most common target value in the examples*
    - *Else below this new branch add the subtree ID3 (Examples(*v_i*), Target_Attribute, Attributes – {A})*
- *End*
- *Return Root*

Decision Tree Learning

- How to choose the attribute that best explains the data?

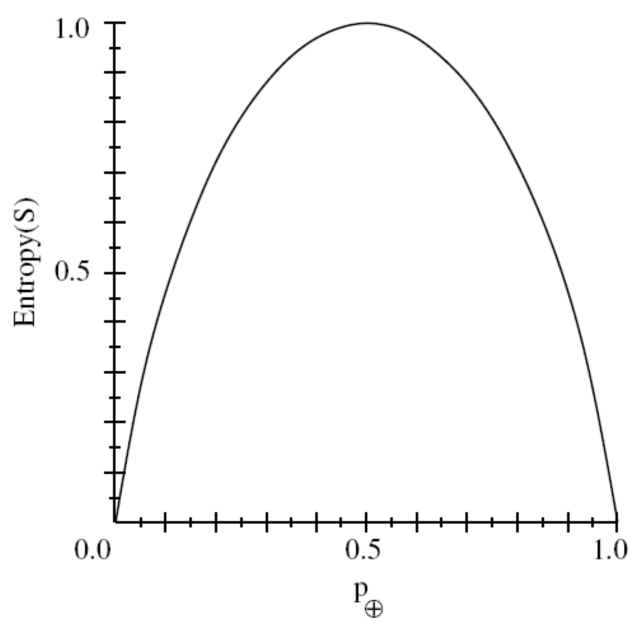
  Which attribute is better? (A1 or A2)



Decision Tree Learning

- To quantify which attribute is better, we define the *Entropy*
- The entropy measures the **impurity** of a sample of training examples *S*
- Let $p_{\oplus}$ be the proportion of +ve examples in *S*
- Let $p_{\ominus}$ be the proportion of –ve examples in *S*
- Entropy of *S* is defined by

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

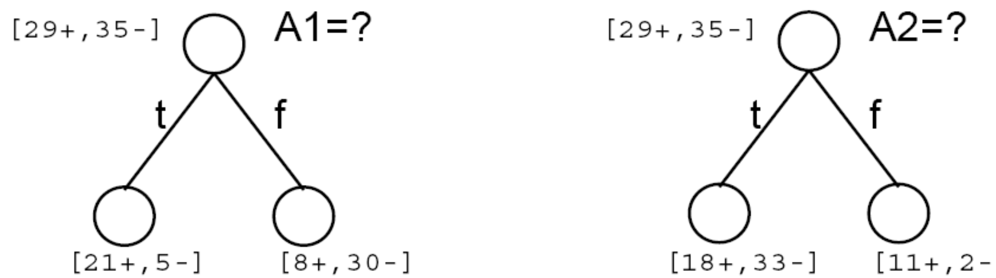


Decision Tree Learning

- We define the information gain as the expected reduction in entropy due to sorting on a certain attribute

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

- Which attribute is better?



Entropy(S) = -(29/64)$\log_2$(29/64) – (35/64)$\log_2$(35/64) = 0.99

Gain(S, A1) = Entropy(S) – (26/64)Entropy(S_t) – (38/64)Entropy(S_f)

For A1:

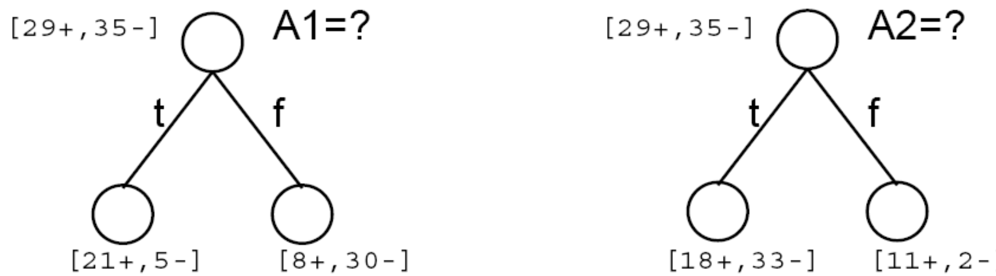
Entropy(S_t) = -(21/26)$\log_2$(21/26) – (5/26)$\log_2$(5/26) = 0.71

Entropy(S_f) = -(8/38)$\log_2$(8/38) – (30/38)$\log_2$(30/38) = 0.74

Gain(S, A1) = 0.99 – (26/64)0.71 – (38/64)0.74 = 0.26

Decision Tree Learning

- Which attribute is better?



$Entropy(S) = -(29/64)\log_2(29/64) – (35/64)\log_2(35/64) = 0.99$

$Gain(S, A2) = Entropy(S) – (51/64)Entropy(S_t) – (13/64)Entropy(S_f)$

For A2:

$\mathit{Entropy}(S_t) = -(18/51)\log_2(18/51) – (33/51)\log_2(33/51) = 0.94$

$\mathit{Entropy}(S_f) = -(11/13)\log_2(11/13) – (2/13)\log_2(2/13) = 0.62$

$Gain(S, A2) = 0.99 – (51/64)0.94 – (13/64)0.62 = 0.11$

Since Gain(S, A1) > Gain (S, A2), then using A1 is better than A2

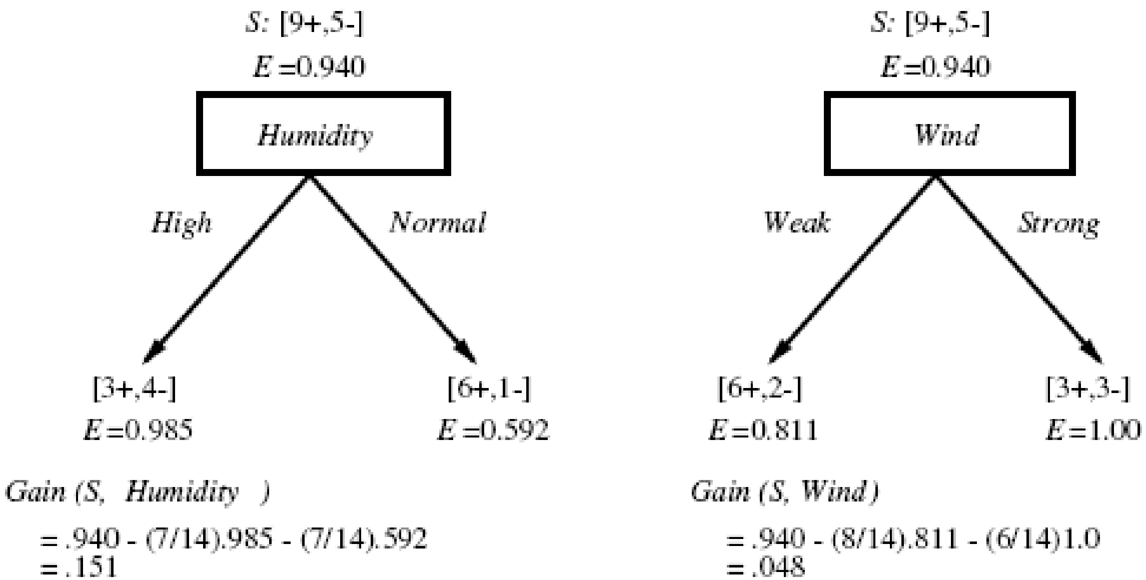
Decision Tree Learning

- Play Tennis Example: Data

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

Decision Tree Learning

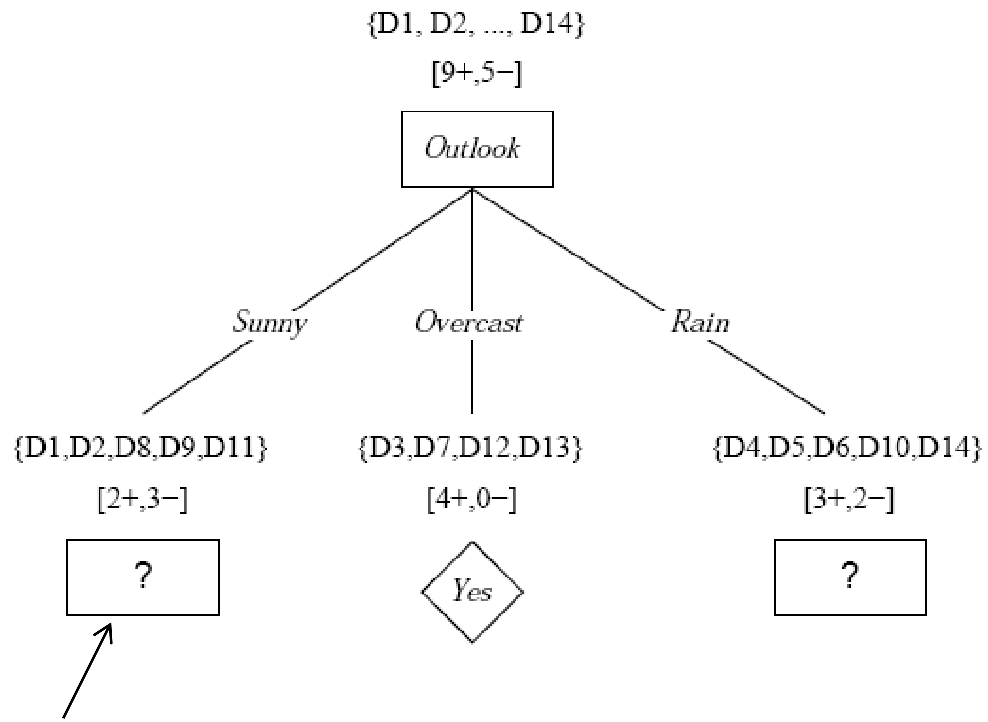
- Play Tennis Example: Root Node



- Also, Gain(S, Outlook) = 0.246 and Gain(S, Temperature) = 0.029
  Therefore Outlook is the root of the tree

Decision Tree Learning

- Play Tennis Example: Next Level



Which attribute should be tested here?

$$Gain\ (S_{sunny}, Humidity) = .970 - (3/5)\ 0.0 - (2/5)\ 0.0 = .970$$

$$Gain\ (S_{sunny}, Temperature) = .970 - (2/5)\ 0.0 - (2/5)\ 1.0 - (1/5)\ 0.0 = .570$$

$$Gain\ (S_{sunny}, Wind) = .970 - (2/5)\ 1.0 - (3/5)\ .918 = .019$$

So, it's Humidity

Decision Tree Learning

- Final Decision Tree