“Induction of Decision Trees” by Ross Quinlan

Papers We Love Bucharest

Stefan Alexandru Adam

27th of May 2016

TechHub

Short History on Classification Algorithms

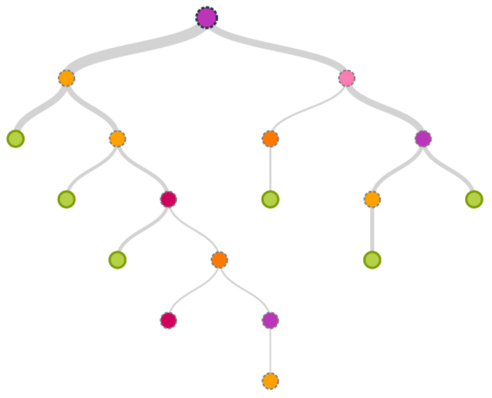
- Perceptron (Rosenblatt, Frank 1957)
- Pattern Recognition using K-NN (1967)
- *Top down induction of decision trees (1980's)*
- *Bagging (Breiman 1994)*
- *Boosting*
- SVM

TDIDT family of learning systems

A family of learning systems which solves a classification problem using a decision tree.

A classification problem can be :

- diagnosis of a medical condition given symptoms
- determining the Gain/Lose possible values of a chess position
- determining from atmospheric observation if it will snow or not



TDIDT family of learning systems

Decision trees are a representation of a classification

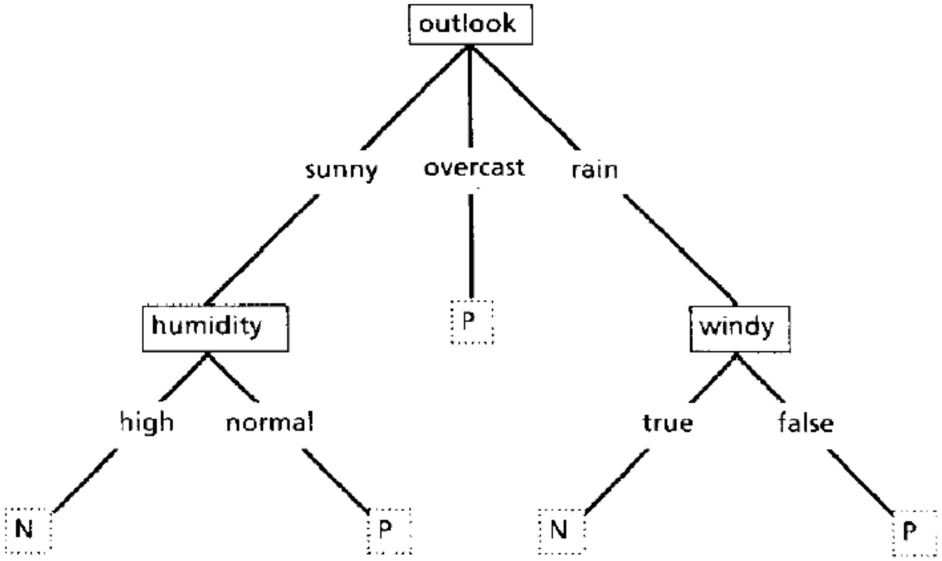
- The root is labelled by an attribute
- Edges are labelled by attribute values
- Each leaf is labelled by a class
- Is a collection of decision rules

Classification is done by traversing the tree from the root to the leaves.

TDIDT family of learning systems

PlayTennis: training examples

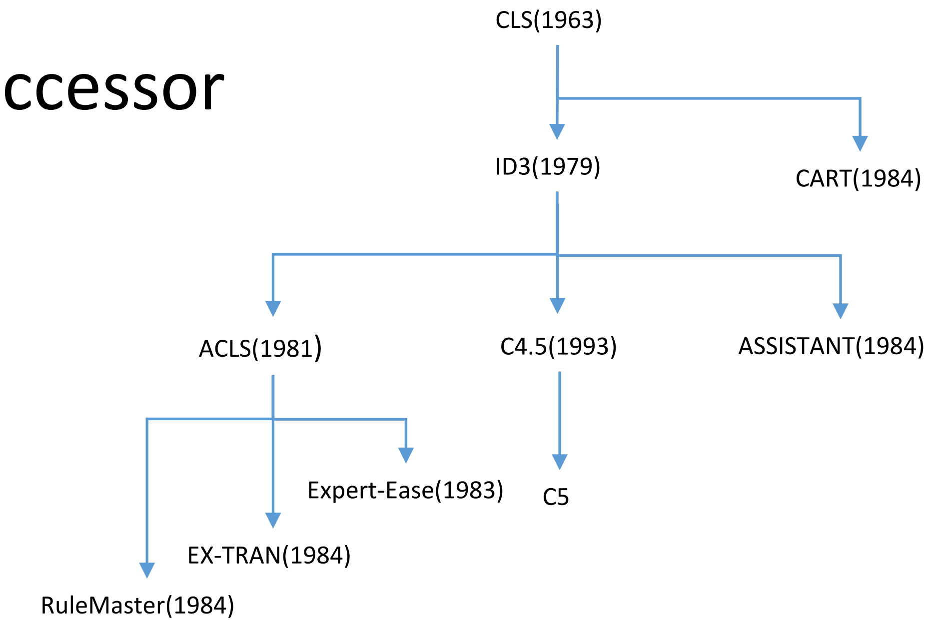
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal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



TDIDT family of learning systems

Important decision tree algorithms:

- ID3 (Iterative Dichotomiser 3) and successor
- CART binary trees (Classification and regression trees)



Induction Task

Given a training set of n observations $(x_{i,}y_i)$

$$\mathrm{T(X,Y)} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,s} \\ \vdots & \ddots & \vdots \\ x_{1,n} & \cdots & x_{n,s} \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

, x_i has s attributes and $y_i \in \{C_1, \ldots C_m\}$

Develop a classification rule(decision tree) that can determine the class of any objects from its values attributes. It usually has two steps:

- ❑ growing phase. The tree is constructed top-down
- ❑ pruning phase. The tree is pruned bottom-up

Induction Task – Algorithm for growing decision trees

GrowTree(observations, node)

1. If all observations have the same class C// node is a leaf

   node.Class = C

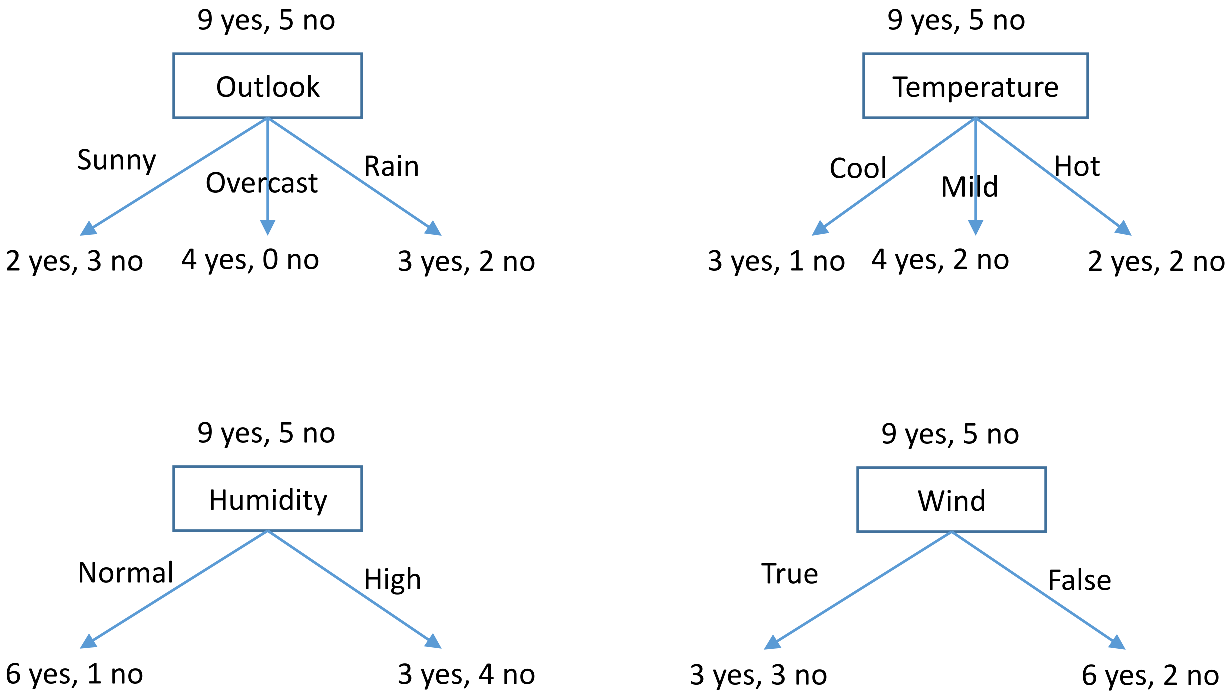
   return
2. bestAttribute = FindBestAttribute(observations) // identify best attribute which splits the data
3. Partition the observations according with bestAttribute into k partitions
4. For each partitions $P_{i=1:k}$

   ChildNode = new Node(node)

   GrowTree(P_i, ChildNode) // recursive call

Induction Task – Choosing best attribute

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No



Induction Task – Measure Information

Identify the best attribute for splitting the data.

Create a score for each attribute and choose the one with the highest value. This score is a measure of ***impurity***

Possible scores:

- Entropy (information Gain) used in ID3, C4.5
    - Gain Ratio
- Gini index used in CART
- Twoing splitting rule in CART

Entropy measure – Information Gain

Given a discrete random variable X with possible outcomes $\{x_1, \ldots, x_n\}$ and probability P

$\mathrm{H}(X) = E\left[-log_2\left(P(X)\right)\right]$ bits, represents the entropy of X

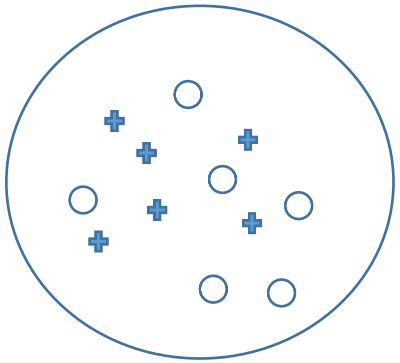
bit = the unit of measure for Entropy

The entropy measures the impurity (disorder) of a system.

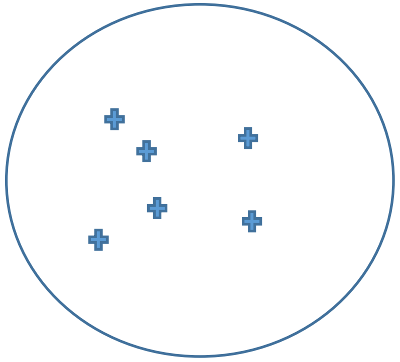
For a subset S(n) which correspond to a node n in a decision tree

$$\mathbf{H}(\boldsymbol{S}) = -\sum_{i=1}^{m} \boldsymbol{p}(\boldsymbol{C_i})\, \boldsymbol{log_2}\left(\boldsymbol{p}(\boldsymbol{C_i})\right) \text{ where}$$

$$p(C_i) = \frac{count\ of\ items\ in\ S\ with\ class\ C_i}{total\ number\ of\ items\ in\ \ S}$$



Maximum impurity



Zero impurity

Entropy measure – Information Gain

Given a subset S and attribute A with possible outcomes $\{V_1, \ldots, V_l\}$ we define the information gain

$\mathbf{G}(\boldsymbol{S}, \boldsymbol{A}) = \boldsymbol{H}(\boldsymbol{S}) - \sum_{i=1}^{l} \boldsymbol{p}(\boldsymbol{V_i})\, \boldsymbol{H}\left(\boldsymbol{S_{V_i}}\right)$, where

$p(V_i) = \frac{count\ in\ S\ where\ value(A) = V_i}{count\ of\ items\ in\ S}$, prob. of value(A) = V_i in S

S_{V_i} represents the subset from S where value(A) = V_i

$SplitInfo(S, A) = \ -\sum_{1}^{l} p(V_i)\, log_2(p(V_i))$

$$\boldsymbol{G_{ratio}}(\boldsymbol{S}, \boldsymbol{A}) = \frac{\boldsymbol{G}(\boldsymbol{S}, \boldsymbol{A})}{\boldsymbol{SplitInfo}(\boldsymbol{S}, \boldsymbol{A})}$$

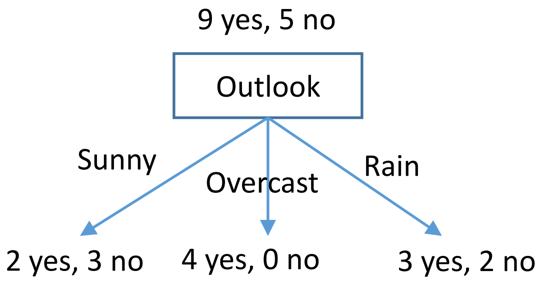
ID3 – decision tree

GrowID3(S, node, attributes)

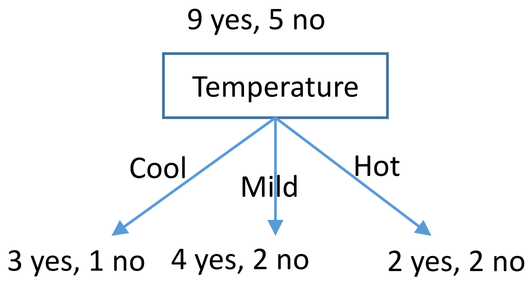
1. If all items in S has the same class C
   node.Class = C
   return
2. For each attribute A in attributes compute $G(S, A)$
   $A_{split} = argmax_A(G(S,A)), A_{split} \in \{V_1, \dots, V_l\}$
3. Compute S_{V_i} the subset from S where value(A) = V_i
4. attributes.remove(A_{split})
5. For each subset partitions $S_{V_{i=1:l}}$
   ChildNode = new Node(node)
   GrowID3 (P_i, ChildNode) // recursive call

ID3– Example

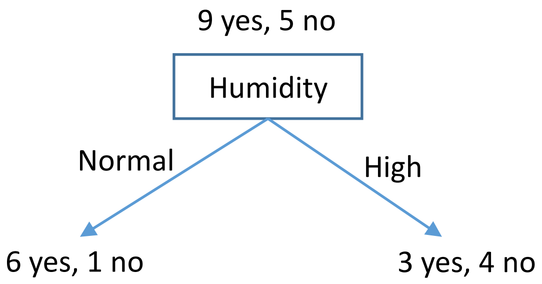
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No



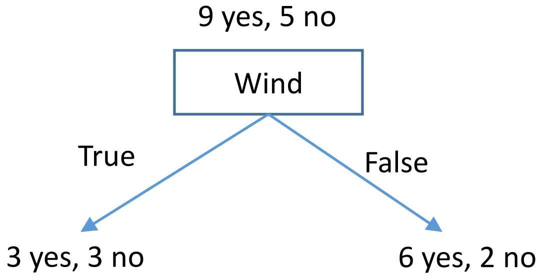
G=0.246 bits



G=0.029 bits



G(Humidify)=0.152 bits

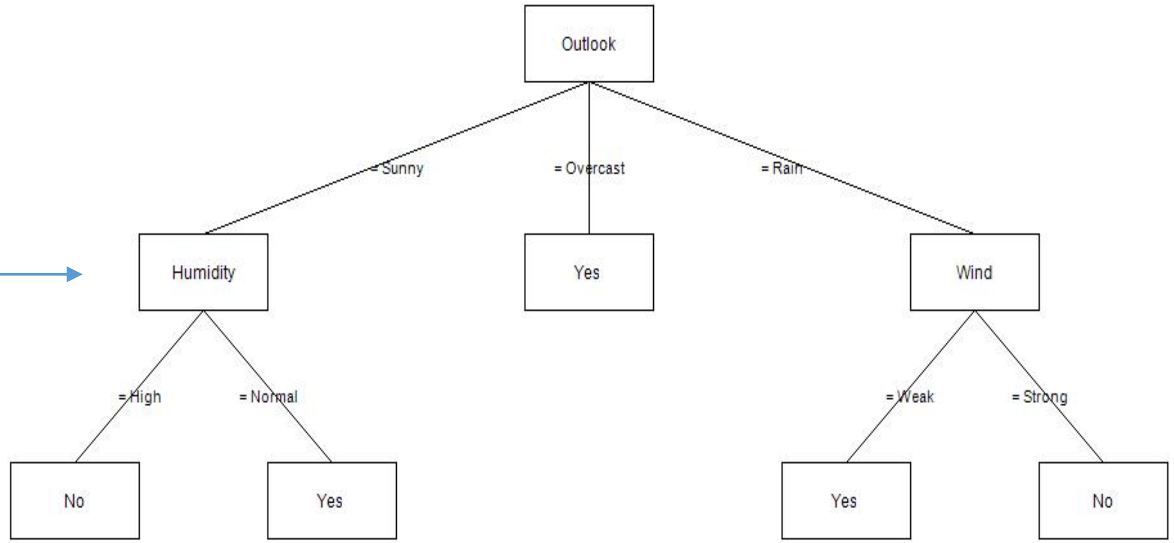


G(Wind)=0.048 bits

Ex: G(Outlook) = - 9/14log(9/14) -5/14log(5/14) – (5/14 * (-2/5 log(2/5) – 3/5 log(3/5)) + 4/14 * (-4/4 log(4/4) - 0* log(0)) + 5/14 * (-3/5log(3/5) -2/5log(2/5))) = 0.246 bits

ID3 - Example

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

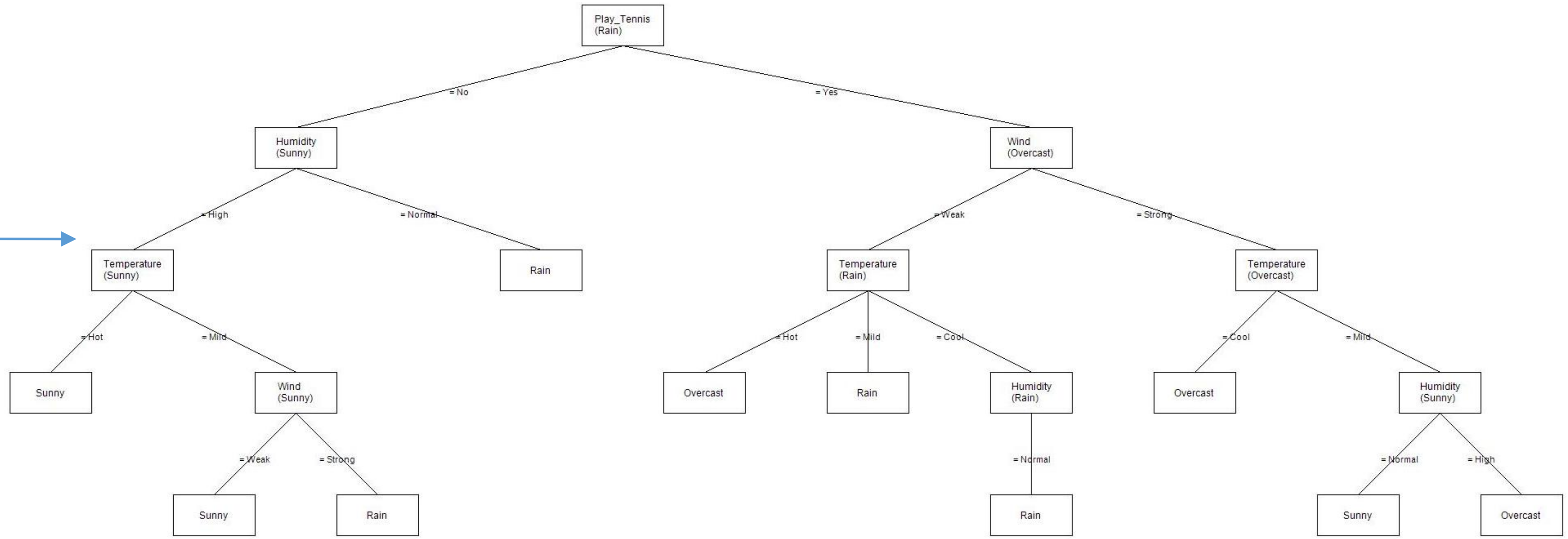


What if X'=(Outlook=Sunny, Temperature=Cool, Humidity=High, Wind=Strong) = > No

ID3 – handling unknown values

- Consider the attribute with the unknown value as the class feature and identify the missing value using the ID3 algorithm

Day	Play Tennis	Temperature	Humidity	Wind	Outlook
Day1	No	Hot	High	Weak	?
Day2	No	Hot	High	Strong	Sunny
Day3	Yes	Hot	High	Weak	Overcast
Day4	Yes	Mild	High	Weak	Rain
Day5	Yes	Cool	Normal	Weak	Rain
Day6	No	Cool	Normal	Strong	Rain
Day7	Yes	Cool	Normal	Strong	Overcast
Day8	No	Mild	High	Weak	Sunny
Day9	Yes	Cool	Normal	Weak	Sunny
Day10	Yes	Mild	Normal	Weak	Rain
Day11	Yes	Mild	Normal	Strong	Sunny
Day12	Yes	Mild	High	Strong	Overcast
Day13	Yes	Hot	Normal	Weak	Overcast
Day14	No	Mild	High	Strong	Rain



Suppose the first value from the Outlook is missing.

? = class(PlayTennis=No, Temperature=Hot, Humidity=High, Wind=Weak) = Sunny

ID3 – overfitting

To avoid overfitting the negligible leaves are removed. This mechanism is called pruning.

Two types of pruning:

- Pre-Pruning
  - is not so efficient but is fast because is done in the growing process
  - based on thresholds like (maximum depth, minimum items classified by a node)
- Post – Pruning
  - done after the tree is grown
  - prunes the leaves in the error rate order (much more efficient)

ID3 - summary

- Easy to implement
- Doesn't tackle the numeric values (C4.5 does)
- Creates a split for each Attribute value (CART trees are binary tree)
- C4.5 the successor of ID3 was considered the first algorithm in the "Top 10 Algorithms in Data Mining" paper published by Springer LNCS in 2008

Decision tree – real life examples

- Biomedical engineering (decision trees for identifying features to be used in implantable devices)
- Financial analysis (e.g Kaggle Santander customer satisfaction)
- Astronomy(classify galaxies, identify quasars, etc.)
- System Control
- Manufacturing and Production (quality control, semiconductor manufacturing, etc.)
- Medicine(diagnosis, cardiology, psychiatry)
- Plant diseases (CART was recently used to assess the hazard of mortality to pine trees)
- Pharmacology (drug analysis classification)
- Physics (particle detection)

Decision tree - conclusions

- Easy to implement
- Top classifier used in ensemble learning (Random Forest, Gradient boosting, Extreme Trees)
- Widely used in practice
- When trees are small they are easy to understand