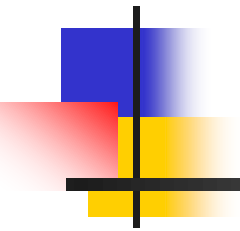
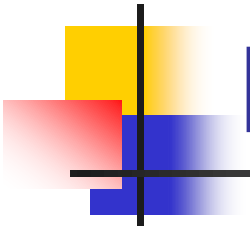


Learning, Induction, Decision Tree Generation

Halina Kwasnicka

halina.kwasnicka@pwr.edu.pl



Learning

- You know what does the *learning* mean, so try to define
- The first required element is change
- Forgetting is also a change (advantageous)
- A well-fed animal – an advantage (autonomous)
- A random change is autonomous (a result of experience)

Change (must be observed), but:

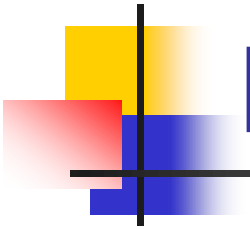
- **advantageous** (forgetting is also a change),
- **autonomous** (better feeding improves effectiveness),
- **on the basis of experience** (random change is autonomous).

Learning concept

Learning system is a system in which an **autonomous change** proceeds **on the basis of experience**, and this change causes **better functionality of the system**

Learning – a fuzzy concept

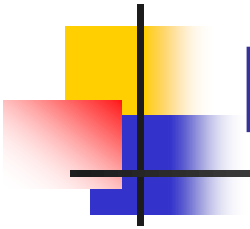
(a change can be objectively stated, but the rest are fuzzy, conventional)



Learning system – definition

„A computer program is said to **learn** from the experience ***E*** with respect to some class of tasks ***T*** and performance measure ***P***, if its performance at tasks in ***T***, as measured by ***P***, improves with experience ***E***"

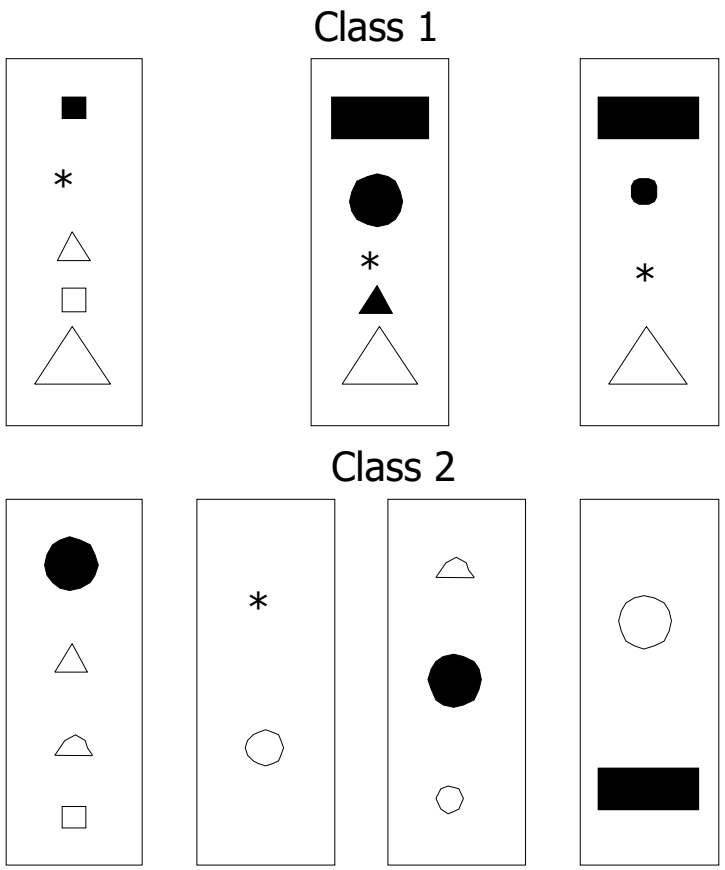
T.M. Mitchell „Machine Learning"



Examples

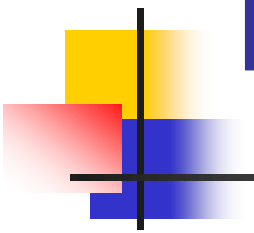
- Chess game problem
  - A class of tasks *T*: chess game
  - A measure of performance *P*: a percent of won games
  - Experiences *E*: real games with opponent
- Vehicle control problem
  - A class of tasks *T*: movement on public roads in four directions using only visual sensors
  - A measure of performance *P*: an average lenght of road without mistakes
  - Experiences *E*: a sequence of images and commands giving by human (gathered by observation of drivers)

Inductive learning



- R1: IF a rectangle contains a small black shape THEN Class 1
- R2: IF a rectangle contains five figures OR a small black circle THEN Class 1
- R3: IF a rectangle contains two figures OR white 'moon' THEN Class 2.
- ...
- Difficult task for computers ☺

Deduction versus induction

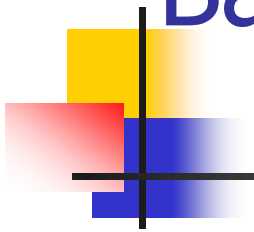


- Deduction – inference from general to specific (fact) – top-down:
  - Known: *All dogs have four legs,*
  - We know that: *Patcher is my dog,*
  - We deduct: *Patcher has four legs.*
- Induction – inference from specific (facts) to general – bottom-up:
  - We see: one leaf is green and thin, next leaf is green, thin and oval, next – green, thin, and oblong,
  - We induct: All leaves are green and thin.

Decision tree learning

- Decision tree learning is one of the most widely used and practical methods for inductive inference.
- Each node specifies a test of some attribute of the instance.
- Decision trees classify instances by sorting them down the tree from the root to some leaf node. All non-leaf nodes contain attribute names, leaf nodes contain class values, each branch corresponds to the attribute value.
- ID3, ASSISTANT, C4.5, C5.0, ...

Basic ID3 algorithm (Quinlan 1986)



- ID3 learns decision trees by constructing them top-down.
- Important question: which attribute should be tested at the root of the tree?
  - Each instance attribute is evaluated using a statistical test to determine how well it classifies the training examples.
  - The best attribute is selected and used as the test at the root node of the tree.
- A descendant of the root node is then created for each possible value of this attribute, and the testing examples are sorted to the appropriate descendant node.
- The entire process is then repeated using training examples associated with each descendant node to select the best attribute to test at that point in the tree.

ID3 – Which attribute is the best classifier?

- Information gain – a statistical property, that measures how well a given attribute separates the training examples according to their target classification.
- We consider data with logic target function (two classes): Entropy (S) $\equiv -p_+ \log_2 p_+ - p_- \log_2 p_-$,
- Assuming:
  - A is a root of tree,
  - A is an attribute with *n* values $A_1, \ldots, A_n$,
- Set *S* is divided into *n* disjoined subsets $S_1, \ldots, S_n$, each S_i contains examples with value of A equal to A_i.

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

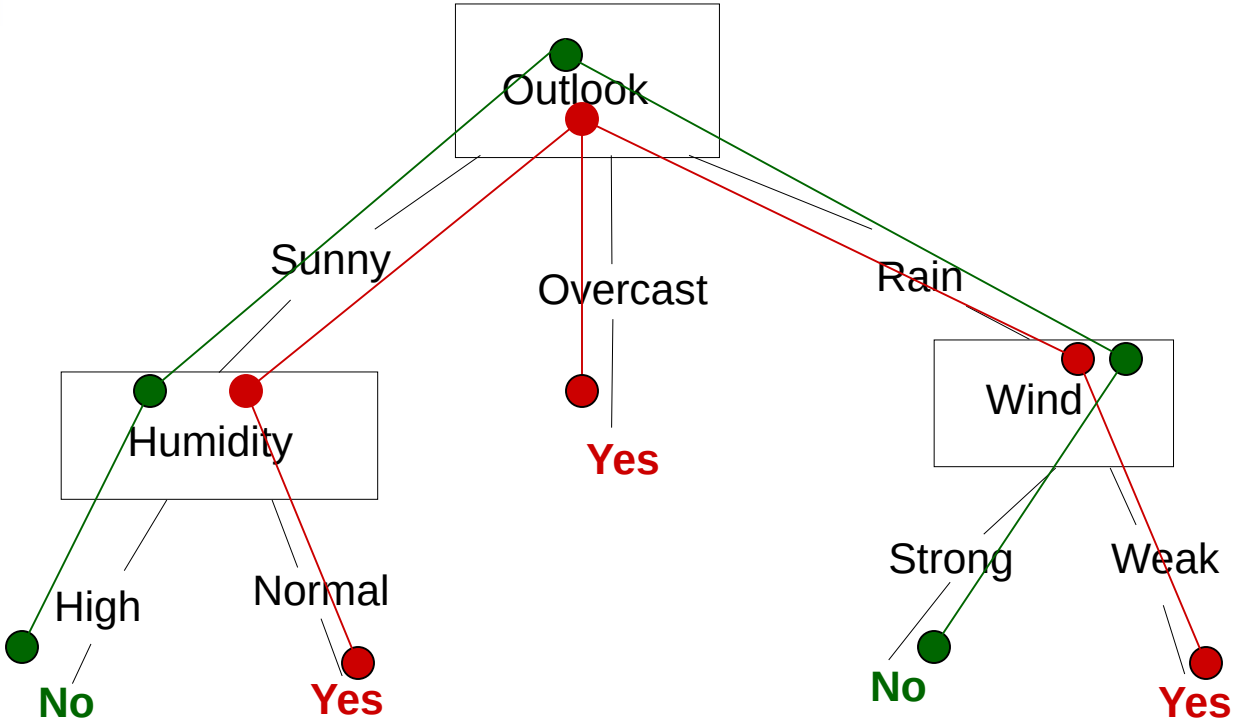
Where *Values*(A) is a set of all possible values of A

ID3 – an illustrative example

S:

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*Decision*
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	**Yes**
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**

Resulting tree



(Outlook = Sunny $\wedge$ Humidity = Normal) $\vee$ (Outlook = Overcast) $\vee$ (Outlook = Rain $\wedge$ Wind = Week)

$\rightarrow$ Decision=Yes

(Outlook = Sunny $\wedge$ Humidity = High) $\vee$ (Outlook = Rain $\wedge$ Wind = Strong) $\rightarrow$ Decision=No

Tree generation using ID3

- S = [9+,5–]
- Entropy(S)= $-p_+\log_2 p_+ - p_-\log_2 p_- = 9/14 \cdot \log_2(9/14) - 5/14 \cdot \log_2(5/14) = 0.940$
- Which attribute should be tested first in the tree?
- We calculate *information gain* for each candidate attribute (*Outlook*, *Temperature*, *Humidity*, *Wind*):
- *Values*(*Wind*)={*Weak*,*Strong*}: *SWeak* ← [6+,2–]; *SStrong* ← [3+,3–]
- Gain for *Wind*: $E(S) - (8/14) \cdot E(SWeak) - (6/14) \cdot E(SStrong) =$
  $= 0.940 - (8/14) \cdot 0.811 - (6/14) \cdot 1.00 = 0.048$
- We must calculate information gains for all attributes. We obtain:

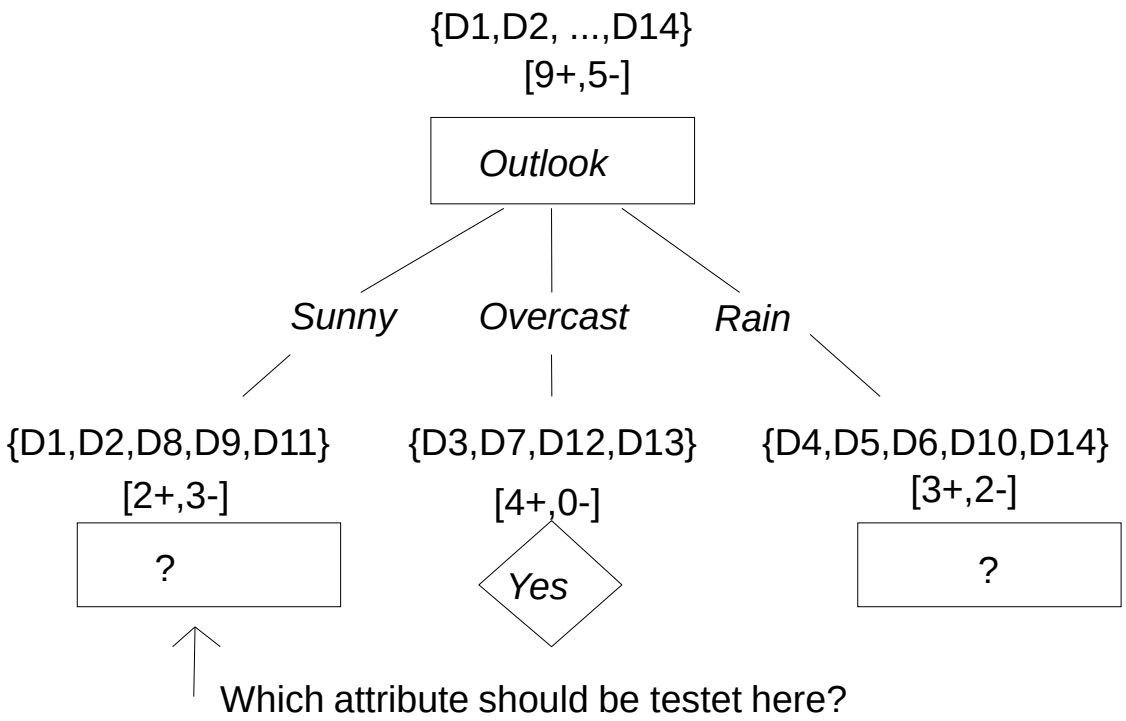
  Gain(S,Outlook) = 0,246

  Gain(*S*,*Humidity*) = 0,151

  Gain(*S*,*Wind*) = 0,048

  Gain(*S*,*Temperature*) = 0,029

A partial tree



S_{SUNNY}={D1,D2,D8,D9,D11}

Gain(S_{Sunny},Humidity)=0,970-(3/5)0,0-(2/5)0,0=0,970

Gain(S_{Sunny},Temperature)=0,970-(2/5)0,0-(2/5)0,0-(1/5)0,0=0,570

Gain(S_{Sunny},Wind)=0,970-(2/5)1,0-(3/5)0,918=0,019

ID3 - pseudocode

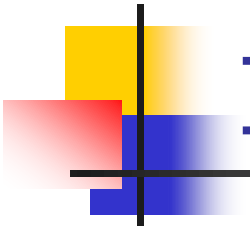
ID3(*Examples*, *Target_attribute*, *Attributes*)

Examples are the training examples. Target_attribute is the attribute whose value is to be predicted by the tree. Attributes is a list of other attributes that may be tested by the learned decision tree. Returns a decision tree that correctly classifies the given Examples.

- 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 *Attributes* is empty, Return the single-node tree *Root*, with label = most common value of *Target_attribute* in *Examples*
- Otherwise Begin
  - $A \leftarrow$ the attribute from *Attributes* that best* classifies *Examples*
  - The decision attribute for *Root* $\leftarrow A$
  - 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 value v_i for A
    - If $Examples_{v_i}$ is empty
      - Then below this new branch add a leaf node with label = most common value of *Target_attribute* in *Examples*
      - Else below this new branch add the subtree ID3($Examples_{v_i}$, *Target_attribute*, *Attributes* − {A}))
- End
- Return *Root*

* The best attribute is the one with highest *information gain*, as defined in Equation (3.4).

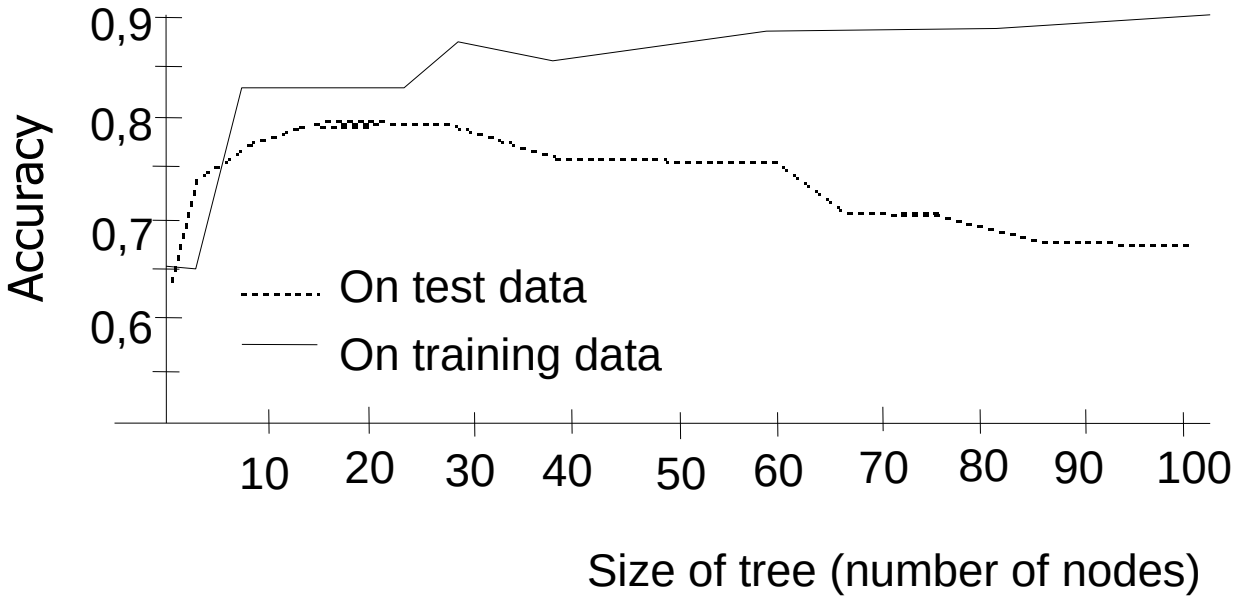
TABLE 3.1
Summary of the ID3 algorithm specialized to learning boolean-valued functions. ID3 is a greedy algorithm that grows the tree top-down, at each node selecting the attribute that best classifies the local training examples. This process continues until the tree perfectly classifies the training examples, or until all attributes have been used.



ID3 – a short summary

- ID3's hypothesis space of all decision trees is a *complete* space of finite discrete-valued functions, relative to the available attributes.
- ID3 maintains only a single current hypothesis as it searches through the space of decision trees. It does not have the ability to determine how many alternative decision trees are consistent with the available training data.
- ID3 does not perform backtracking, therefore it converges to the local optimum (produces locally optimal solution).
- ID3 uses all training examples and some statistical properties of the examples, and it is less sensitive to errors in an individual training example.

Avoid Overfitting the Data



Next presentation

Learning a set of rules

IF shape=block THEN Yes

IF size=average THEN Yes