Mohamed Boudiaf University of M'sila
Faculty of Mathematics and Computer Science

Introduction to Artificial Intelligence

Chapter 4. Machine Learning

Dr. SAID KADRI

Associate Professor "Class A"

Department of Computer Science, Faculty of Mathematics and Informatics,

University Mohamed Boudiaf of M'sila

E-mail:kadri.said28@gmail.com

Website:https://kadrisaid28.wixsite.com/sgadri

2020 – 2021

Machine Learning

- Machine learning is a subfield of AI.
- The objective of ML is to reproduce the same behavior of a human.
- The main idea of ML is to predict the future based on the past.

Traditional Programming

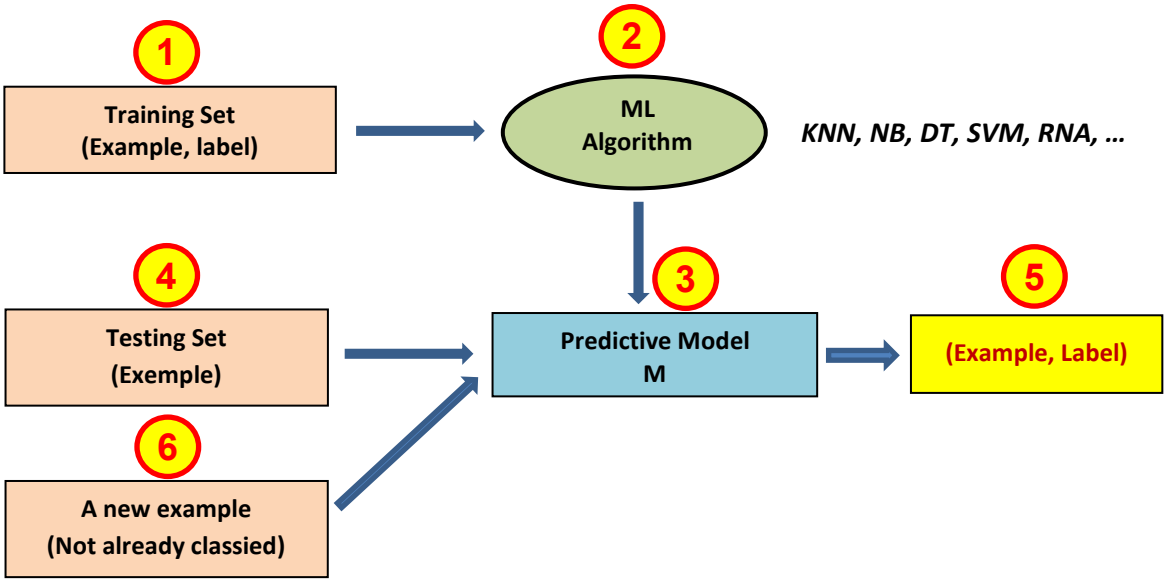
Machine Learning

Example:

➢ Predict how Omar will choose a movie he has not seen before, based on his reviews (likes, queries, …) of movies he has seen before (the passive).

==> **Predicting the future based on the past.**

Machine Learning Process (Classification problems)



Main Steps

(1) Take some training data (training set) and use them to build the prediction model (A model M /A function F).

(2) Apply An ML algorithm on the training dataset to build a predictive model

(3) Build a predictive model.

(4) This model will be evaluated on the testing data (test set).

(5) The learning algorithm is considered valid and efficient if its performance on the test set is high.

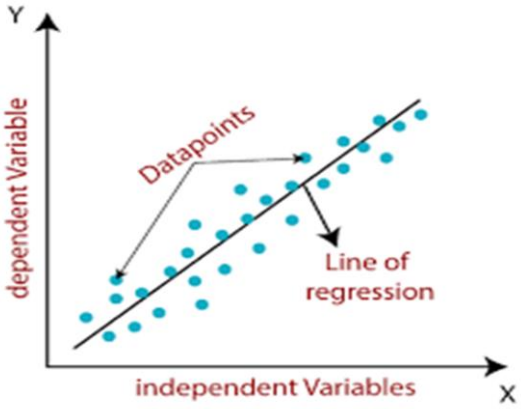
(6) Then we use the model validated on the test set to classify a new example.

Examples of typical prediction problems

1. Regression: trying to predict a real value.

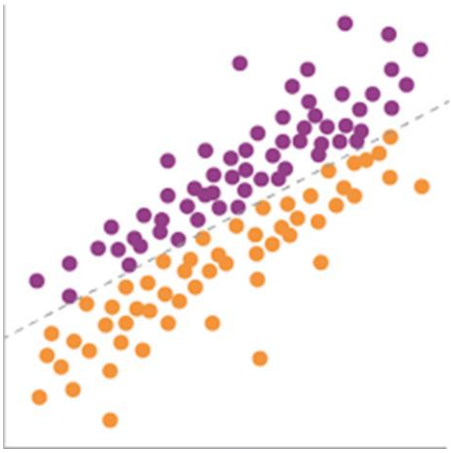
==> Example: predicting the value of the stock tomorrow based on its past performance (statistics).

X	5	10	12	17	24	45
Y	12	22	26	36	?	?



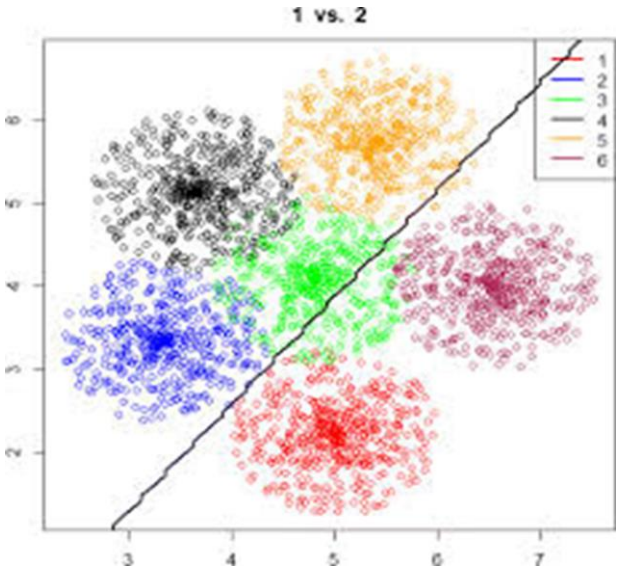
2. Binary classification: trying to predict a simple answer (Yes/No).

==> Example: predicting whether the student Mourad prefers a subject or not (email filtering/Fake detection).



3. Multi-class classification: try to classify an example into one class among many classes.

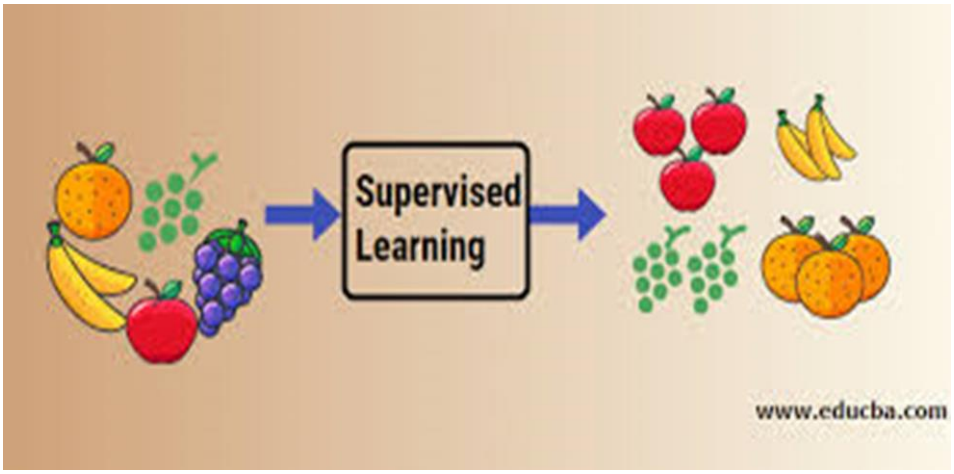
==> Example: predicting the topic of an article (Politics, Sport, Economy, Religion, Cinema, Hobbies, etc.).



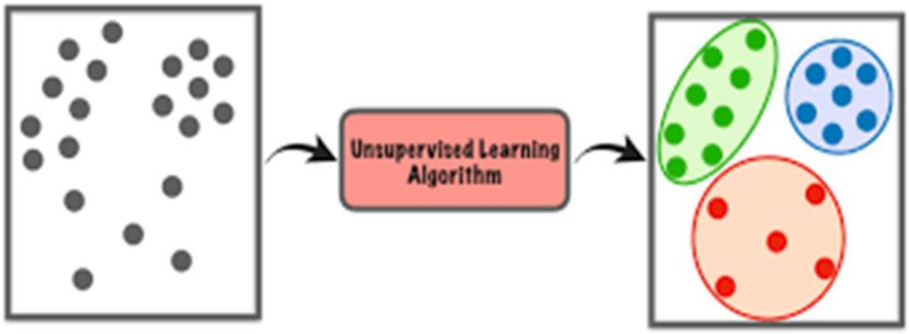
4. Multi-Labels classification: try to classify an example in several classes at the same time. For example, classifying a book in both classes (politics, history) at the same time.

Types of learning

1. Supervised learning: the classification is performed on predefined classes (categorization)

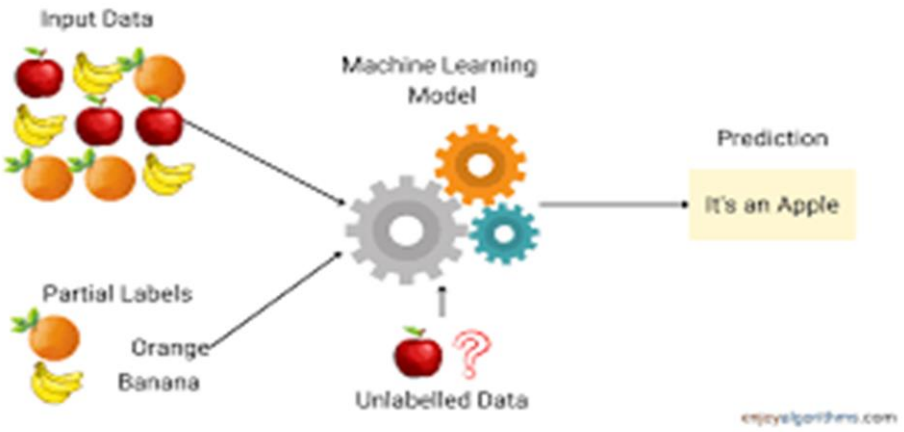


2. Unsupervised learning: the classification is made on non-predefined classes.

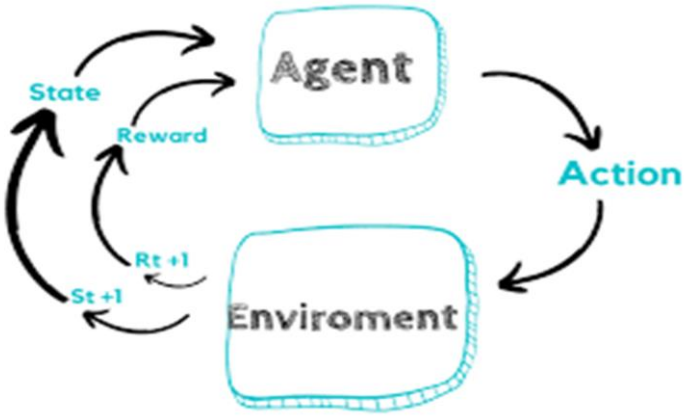


3. Semi-supervised learning: the classification is made on data whose classes are predefined, but also on data whose classes are not predefined.

==> Combining the two types of learning to improve the quality of classification and learning.

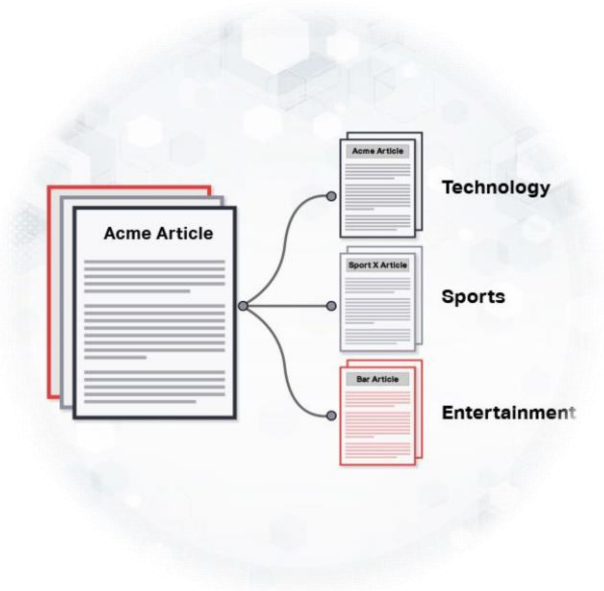


4. Reinforcement learning: Reinforcement learning consists of learning from successive experiences in contact with the environment in order to find the best solution (A kind of learning guided by the reaction of the environment).



Classification

Assigns one or more elements to a class



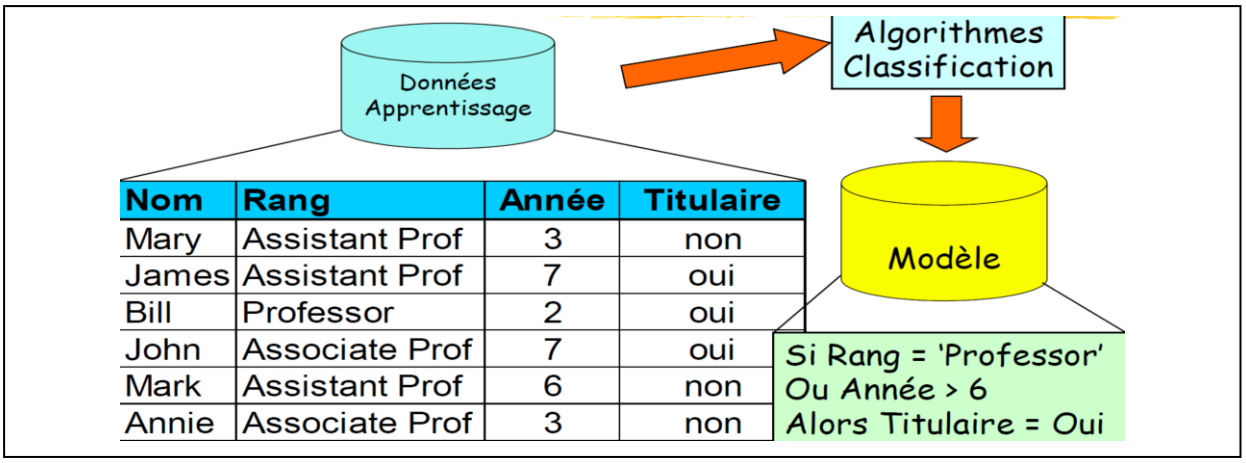
Classification process

We distinguish two steps:

Step 1: Construction of the prediction model from the training set given at the start (the passive)

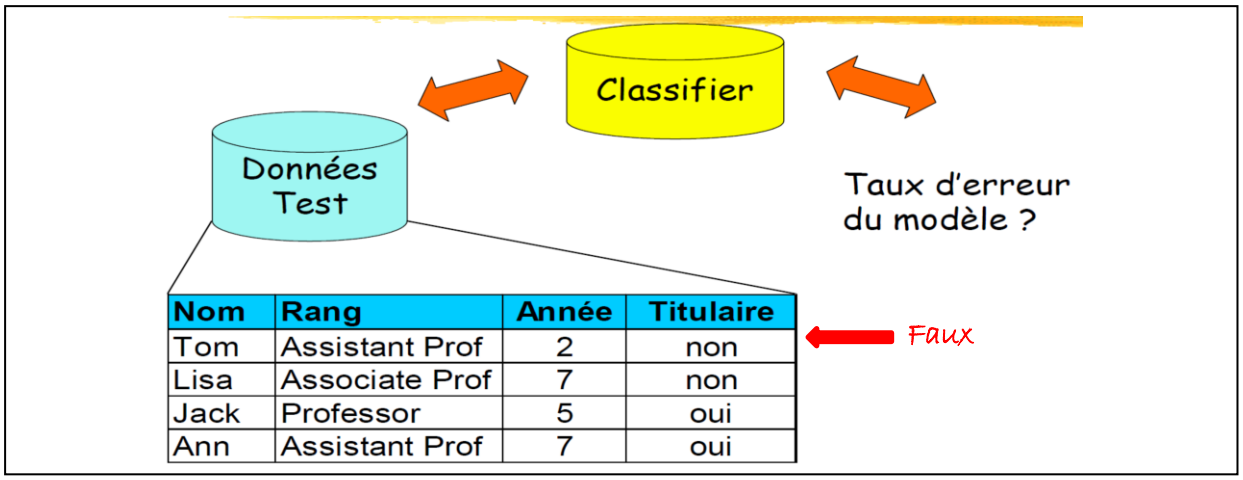
Step 2: Using the model built on the test set to test its accuracy, then use it in classifying new data (the current/present).

Example: Building a predictive model from data



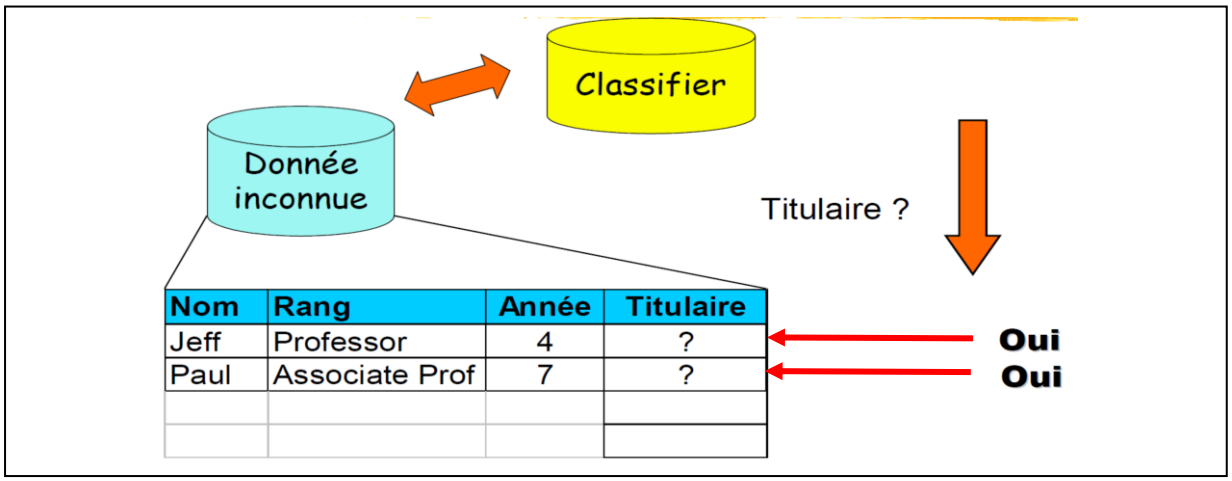
Nom	Rang	Année	Titulaire
Mary	Assistant Prof	3	non
James	Assistant Prof	7	oui
Bill	Professor	2	oui
John	Associate Prof	7	oui
Mark	Assistant Prof	6	non
Annie	Associate Prof	3	non

Example: Using the built model (Validation Model)



Nom	Rang	Année	Titulaire
Tom	Assistant Prof	2	non
Lisa	Associate Prof	7	non
Jack	Professor	5	oui
Ann	Assistant Prof	7	oui

Example: Using the model (classification of new elements)

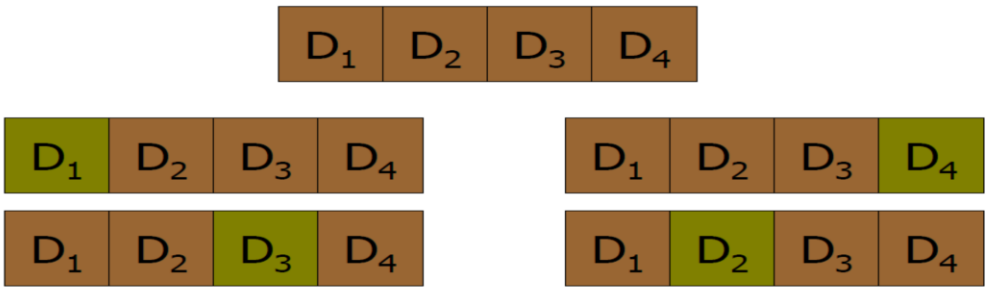


Nom	Rang	Année	Titulaire
Jeff	Professor	4	?
Paul	Associate Prof	7	?

Remarks :

- For the training set each instance (example) is given in the form of a pair (instance, class), i.e., the class of the element is known.
- For the test set the class of each instance (example) is unknown ==> will be determined at the end of the classification process based on the predictive model built, and the same for a new element.
- The classification is carried out on the new instances with a possible error rate which reflects the performance of the constructed model (error rate = rate of incorrectly classified instances).
- Generally, 2/3 of the data is used for training (model building) and 1/3 is used for testing (model validation).
- One of the methods used for model validation is the cross-validation method which consists of:

- Divide data into k subsets
- Use (k-1) subsets as training data and one (01) subset as testing data.



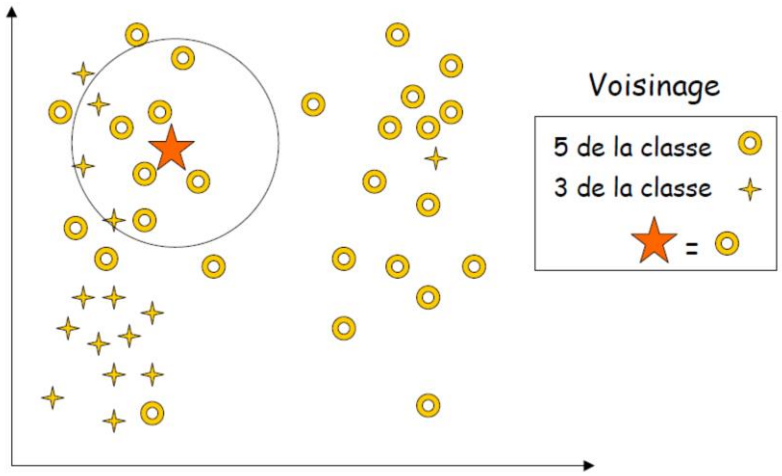
Machine Learning Algorithms Used for Classification Task

Several algorithms are used in Machine Learning, including:

- K-NN method (k nearest neighbors)
- Artificial Neural networks
- Decision Trees DT
- Bayesian methods (Ex : Naive Bayes NB)
- Vector Machine Support VMS
- AdaBoost
- Linear Regression LR.
- Logistic Regression LR.
- Genetic Algorithms.
- Others

1. K-NN algorithm (k Nearest Neibors)

- Its principle is as follows: the class of an element e_i is that of the k closest elements to e_i.
- Model = training set (many examples) + a distance calculation function (similarity) + a class choice function based on the classes of the closest neighbors (majority class).
- Several similarity metrics are used (***Inner Product, Euclidean, Cosine, Dice, Jaccard, Minkowski ...***).
- The element is assigned to the majority class.



kNN *(K-nearest neighbors) algorithm*

Objective *: assign a class to a new instance*

Beginning

Input data:

- *A training set of m examples classified as a pair (x, c(x))*
- *A new example*

Treatment :

- *Determine the k nearest neighbor examples of y*
- *VScombine the classes of these k examples into a class c*

Output Results: *the class of y is c(y) = c*

END

Remarks

- The simplest solution is to search for a single nearest neighbor (k = 1) and assign its class to the new element x, we speak of a 1-NN algorithm.
- It is preferable to take an odd value of k.
- For k > 1, we assign the class which contains the maximum number of neighbors to the new element (majority vote).

Advantages of the Algorithm

- Easy to implement.
- Clear results.
- Valid for all types of data.
- Addition of new data does not require model reconstruction.

Disadvantages of the Algorithm

- No learning, but distance calculation.
- Requires a large number of attributes (examples/Samples).
- Costly in terms of memory and time (model storage and distance calculation).

2. Decision Tree:

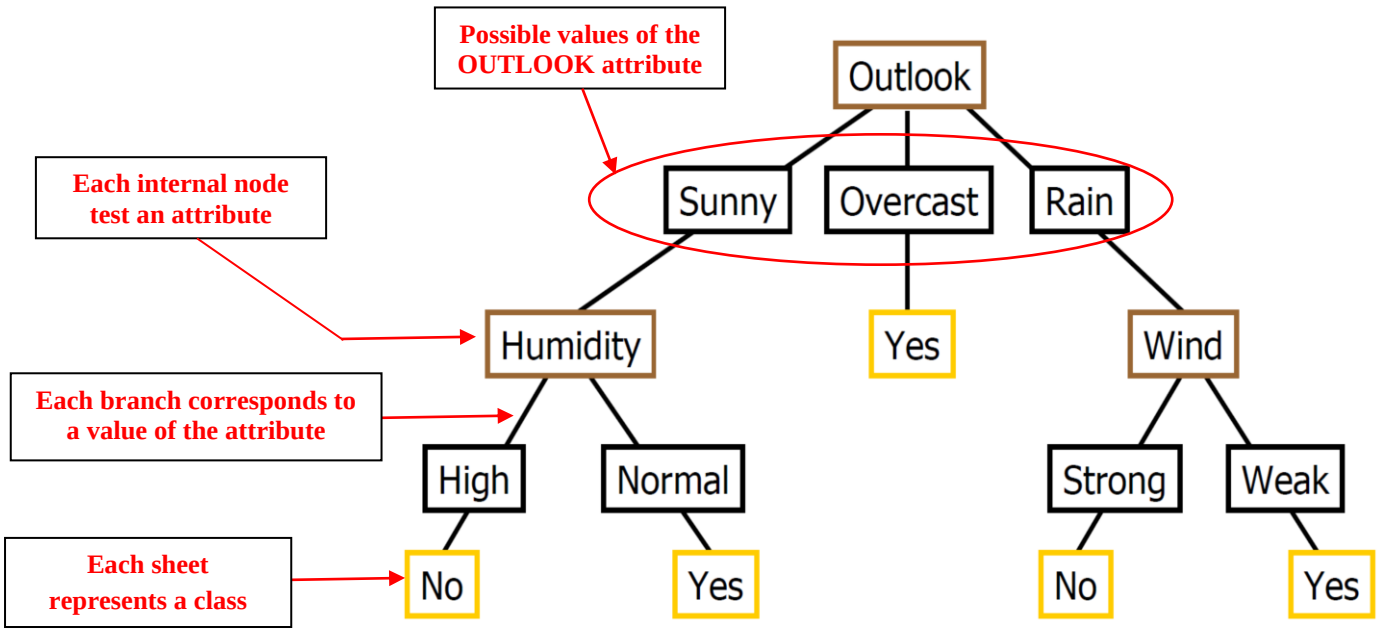
- The decision tree is a classic learning model (algorithm). It is related to the fundamental notion of computing (divide and conquer).
- Decision trees can be applied to several learning problems, the simplest is binary classification.
- A tree = Graphical representation of a classification process.
- The generation of decision trees is done from the data.

Example 01: Construction of a decision tree from a learning base

Outlook	Temperature	Humidity	Windy	Class Play tennis? (yes/No)
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rain	Mild	High	False	Yes
Rain	Cool	Normal	False	Yes
Rain	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rain	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	HIgh	True	Yes
Overcast	Hot	Normal	False	Yes
Rain	Mild	High	True	No

Learning base

From the previous learning base, we can construct the following decision tree:



Transition from the decision tree to classification rules

Example :

If (outlook=sunny) And (humidity=normal) Then (Yes: play tennis)

Remarks

- A rule is generated for each path of the tree (from root to leaf)
- The (attribute-value) pairs of a path form a conjunction
- The terminal node (the leaf) represents the predicted class
- Rules are generally easier to understand than trees

For the previous decision tree, we will have the following set of rules:

R1: If (Outlook=Sunny)∧(Humidity=High) Then PlayTennis=No

R2: If (Outlook=Sunny)∧(Humidity=Normal) Then PlayTennis=Yes

R3: If (Outlook=Overcast) Then PlayTennis=Yes

R4: If (Outlook=Rain)∧(Wind=Strong) Then PlayTennis=No

R5: If (Outlook=Rain)∧(Wind=Weak) Then PlayTennis=Yes

Passage from classification rules to a decision tree

R1: (Ristourne = Yes)⇒No

R2: (Ristourne = No)

and (Situation in {Single, Divorced})

and (Tax < 80K)⇒No

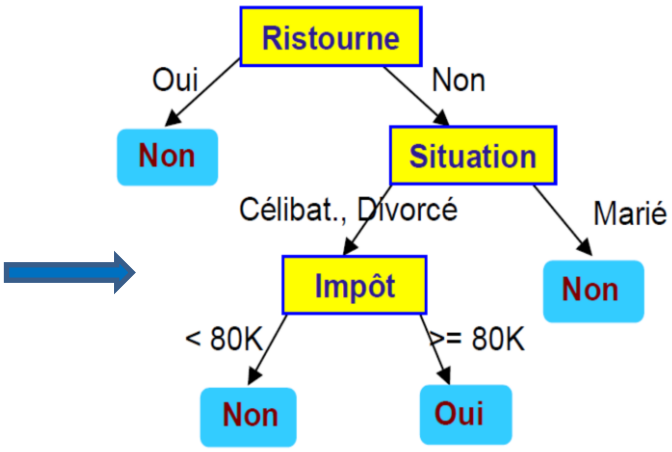
R3: (Ristourne = No)

and (Situation in {Single, Divorced})

and (Tax >= 80K)⇒Yes

R4: (Ristourne = No)

and (Situation in {Married})⇒No



Construction of the decision tree

There exist two phases:

Phase 1: Tree construction (from data)

- Tree can reach a large size

Phase 2: Pruning the tree

- Identify and remove branches that represent "noise" or not important ➔Improve success rate

Decision tree construction process (Phase 1)

- Initially, all learning instances are at the root of the tree.
- Select an attribute and choose a separation test (split) on the attribute, which "best" separates the instances (eg: Outlook, Humidity, Wind).
- The selection of attributes is based on a heuristic or a statistical measure (Ex: Information gain, Gini Index, the chi-square law, etc.).
- Partition the instances between the child nodes according to the satisfaction of the logical tests
- Process each child node recursively.
- Repeat until all nodes are terminals. A current node is terminal if:

- There are no more attributes available.
- The node is "pure", i.e., all instances belong to the same class,
- The node is "almost pure", i.e., the majority of instances belong to a single class (Ex: 95%)
- Having a minimum number of instances per branch (Ex: C5 algorithm avoids tree growth, k=2 by default)

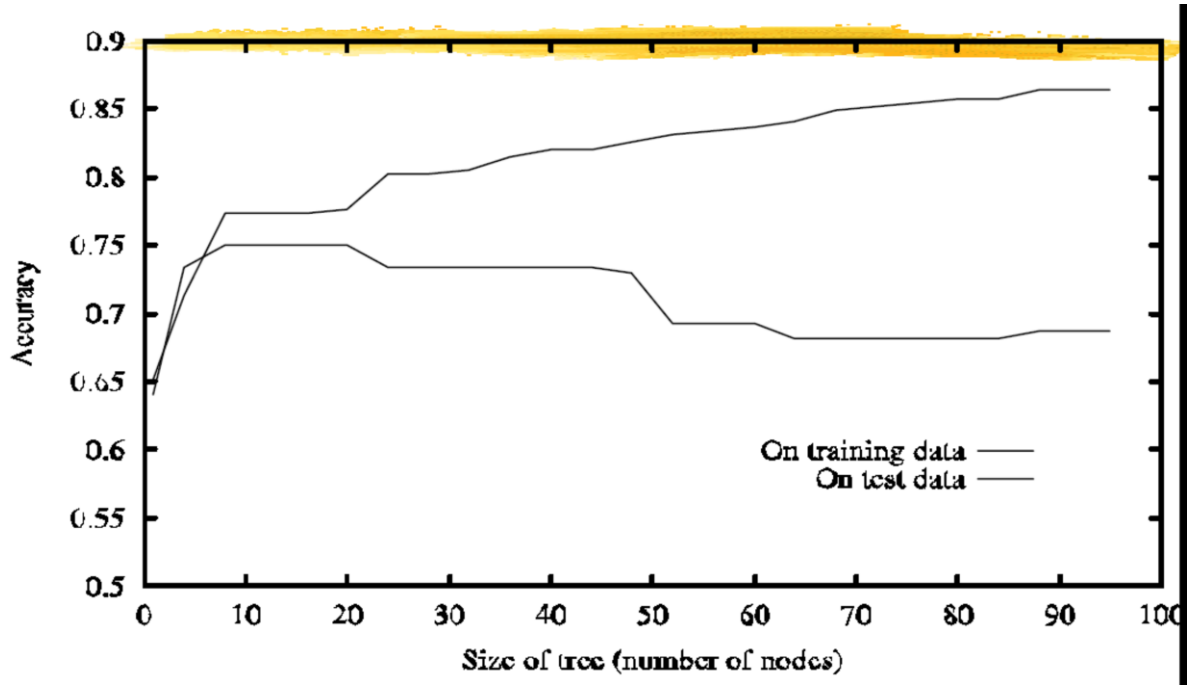
- Label the terminal node by the majority class.

Phase 2: Prun the obtained tree (pruning)

- Remove subtrees that do not improve classification accuracy ➔ have a tree with better generalization power, even if we increase the error on the training set.
- Avoid the problem of over-specialization or over-fitting, i.e., we have learned the learning set "by heart", but we are not able to generalize.

Reasons for over-specialization (Over-fitting)

- Less training data.
- Noise and exceptions on data (the dataset is not perfect).



How to avoid overfitting?

Two techniques can be used:

1. ***Pre-pruning:*** Prematurely stopping tree construction (avoid adding noisy branches).
2. ***Post-pruning:***
   - Remove branches from the fully grown tree.
   - Convert the tree into rules; prune rules independently (C4.5)

Construction of a decision tree: Summary

- Evaluation of different connections for all attributes.
- Selection of the "best" connection and the best attribute.
- Partition data between childs.
- Building in width (C4.5) or depth (SPLIT)

Critical Questions:

- How formulate connection tests ?
- How select attributes (Selection measures)?

Some algorithms for decision trees

- Several algorithms for constructing decision trees have been developed, notably: ID3, C4.5, C5 (CART), CHAID
- Main difference: attribute selection measure – branching criterion (split)

Attribute selection measures when constructing a DT

1. Information Gain (for algorithms: ID3, C4.5)
2. Gini Index (CART)
3. $\chi 2$ statistical contingency table (CHAID)
4. G-statistic

1.Information gain

Principle: Select the attribute with the greatest information gain

- Let P and N be two classes (Eq: P=Yes/N=No) and S be a set of instances with p elements of P and n elements of N.
- The information necessary to determine whether an instance taken at random is part of P or N and we note ***I(p, n)*** or also called the classification entropy ***E*** which measures the homogeneity of the examples is given by the following formula:

$$\boldsymbol{Entropie}\ (\boldsymbol{S}) = \boldsymbol{Entropie}\ (\boldsymbol{p},\boldsymbol{n}) = \boldsymbol{I}(\boldsymbol{p},\boldsymbol{n}) = -\frac{\boldsymbol{p}}{\boldsymbol{p+n}}\boldsymbol{Log}_2\left(\frac{\boldsymbol{p}}{\boldsymbol{p+n}}\right) - \frac{\boldsymbol{n}}{\boldsymbol{p+n}}\boldsymbol{Log}_2\left(\frac{\boldsymbol{n}}{\boldsymbol{p+n}}\right)$$

We note the following:

- S: Set of examples |S| = N = p+n
- p: number of positive examples (classified Yes)
- n: number of negative examples (classified No)
- E(s) = 0 ==> all examples belong to the same class
- E(s) = 1 ==>∃as many examples (p) as examples (n) (50% for each class)

- Information Gain (Information Gain IG) calculates the expected reduction on entropy E(s) if an attribute A is used.

⇨ A classification algorithm based on DT calculates IG for each attribute Ai then chooses the one which reduces the entropy the most, i.e., the one which will allow the remaining examples to be separated as clearly as possible.

$$Information\ Gain\ IG\ (S,A) = E(S) - Somme\ des\ valeurs\ de\ l'attribut$$

$$= E(s) - \sum |S_{V_i}| * E(S_{V_i})/|S|$$

With:

S: set of input examples

Ai: The Used attribute (The current)

Sv: the subset of S whose attribute Ai has the value V

Example:

S = {9+, 5-} With: + (play tennis / YES)

- (do not play tennis / NO)

➔The needed information to classify an example is given by the entropy E

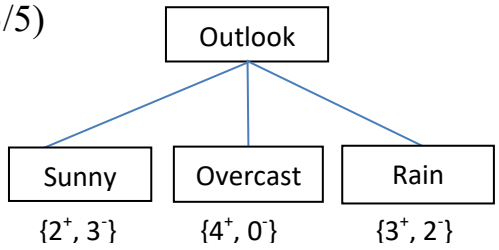
$E(S) = I(p, n) = -(p/N)*Log_2(p/N) - (n/N)*Log_2(n/N)$ (1)

$= - 9/14*Log2(9/14)-5/14*Log2(5/14)$

$= \mathbf{0.940}$

$E(sunny) = I(2^+, 3^-) = -2/5*Log_2(2/5) – 3/5*Log_2(3/5)$

$= 0.971$



$E(Overcast) = I(4^+, 0^-) = 0$

$E(Rain) = I(3^+, 2^-) = 0.971$

➔Gain (S, Outlook) $= E(s) - E(outlook) = E(s) - \sum|S_{V_i}| * E(S_{V_i})/|S|$

= 0.940 – (|Sv = sunny|*E(Sv = sunny)/|S| + |Sv = Overcast|*E(Sv = Overcast)/|S| +

|Sv = rain|*E(Sv = Rain)/|S| (2)

= 0.940 – (5/14 * 0.971 + 4 /14* 0 + 5 /14* 0.971) = 0.940 – (0.3468 + 0 + 0.3468)

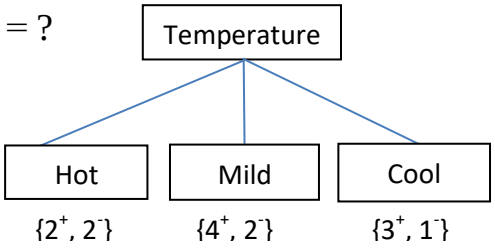
= 0.940 – 0.694 =**0.246** (E(Outlook) = 0.694)

In a similar way, we calculate: (by applying (1))

$E(Hot) = I(2^+, 2^-) = -2/4*Log_2(2/4)–2/4*Log_2(2/4) = ?$

$E(Mild) = I(4^+, 2^-) = ?$

$E(Cool) = I(3^+, 1^-) = ?$

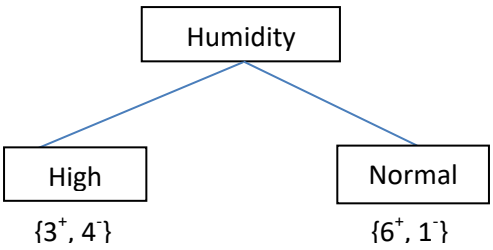


GI (S, Temperature) = **0.029** (by applying (2))

$E(High) = I(3^+, 4^-) = 0.985$ (applying (1))

$E(Normal) = I(6^+, 1^-) = 0.592$

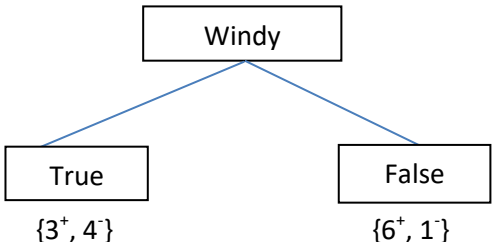
GI (S, Humidity) =**0.151** (by applying (2))



E(True) = $I(3^+, 4^-) = 0.811$ (applying (1))

E(False) = $I(6^+, 1^-) = 1.0$

IG (S, Windy) =**0.048** (by applying (2))



==> Best attribute for selection in the first phase is "Outlook" attribute (having maximum value of IG)

Remarks

- In the same way, other attributes are selected in the next phases of tree construction.
- Other variants of the DT algorithm (C4.5) use other selection measures such as: Gain Ratio with:

$$GR(A) = \frac{IG(A)}{E(A)}$$

Exp:

$$GR(Outlook) = \frac{IG(Outlook)}{E(Outlook)} = \frac{0.246}{0.694} \approx 0.354$$

Decision Trees – Advantages

- Understandable for any user (readability of the result <<rules–tree>>)
- Justification of an instance classification (fellow the path: root➔leaf)
- Valid for any type of data.
- Robust to noise and missing values.
- Attributes appear in order of relevance ➔ pre-processing task (attribute selection)
- Fast classification (accessing a path in a tree).
- A tool available in most data mining environments.

Decision Trees – Disadvantages

- Sensitive to the number of classes: performance degrades for multiclass classification (DT not recommended).
- Scalability over time: if the data evolves over time, it is necessary to restart the learning phase.

4. Bayesian Methods (Probabilistic Methods)

Idea: Predict multiple hypotheses set in advance, based on their probabilities.

How to classify?

- Calculate the probability P(C/X): probability that the tuple (instance) (X=<x1,…,xk> belongs to the class C).
  Ex: P(class=N | outlook=sunny, windy=true,…)
- Assign to an instance X the class C such that P(C|X) is maximum

Bayes' Theorem

$$P(C_i|X) = \frac{P(X|C_i).P(C_i)}{P(X)} \qquad (1)$$

With :

- P(X): is a constant for all classes (probability of appearance of the instance X in the database).
- $P(C_i)$ = Relative frequency of the instances of the class *Ci*
- $P(C_i|X)$ ***is maximum*** ➔ $P(X|C_i).P(C_i)$ ***is maximum***

Issue : *Calculating* $P(X|C_i)$ *is not feasible!*

Naive Hypothesis:

$$P(x_1, \ldots, x_k|C_i) = P(x_1|C_i)\cdot\ldots\cdot P(x_k|C_i) \qquad (2)$$

With :

- $P(x_i|C_i)$ is estimated as the relative frequency of instances having the value x_i (i-th attribute) in the class C_i.

Example: let's take the example seen previously (with decision trees)

Num	Outlook	Temperature	Humidity	Windy	Class Play tennis? (yes/No)
1	Sunny	Hot	High	False	No
2	Sunny	Hot	High	True	No
3	Overcast	Hot	High	False	Yes
4	Rain	Mild	High	False	Yes
5	Rain	Cool	Normal	False	Yes
6	Rain	Cool	Normal	True	No
7	Overcast	Cool	Normal	True	Yes
8	Sunny	Mild	High	False	No
9	Sunny	Cool	Normal	False	Yes
10	Rain	Mild	Normal	False	Yes
11	Sunny	Mild	Normal	True	Yes
12	Overcast	Mild	HIgh	True	Yes
13	Overcast	Hot	Normal	False	Yes
14	Rain	Mild	High	True	No

Let's apply the Naïve Bayes algorithm as follows:

- We have two classes: p(Yes), n(No)
- We have 04 attributes: *Outlook*, *Temperature*, *Humidity*, *Windy*
- We can calculate the two probabilities: $P(p) = 9/14$; $P(n) = 5/14$

  $P(v_i|c_j)$: probability of appearance of the value v_i in tuples belonging to class ci ➔ $= P(v_i|c_j = Nb.Occ(vi) \in c_i/Total.Occ(c_i)$

Ex : $P(outlook=overcast|yes)=Nb.Occ(overcast) \in c_i=yes/Total.Occ(yes)=4/9$

- By applying the previous formula, we obtain the following table:

Attribute 1: Outlook {sunny, overcast, rain}		Attribute 2: Temperature {Hot, Mild, Cool}		Attribute 3: Humidity {High, Normal}		Attribute 4: Windy {True, False}	
P(class=P)	**P(class=N)**	**P(class=P)**	**P(class=N)**	**P(class=P)**	**P(class=N)**	**P(class=P**	**P(class=N)**
P(sunny|p)=2/9	P(sunny|n)=3/5	P(hot|p)=2/9	P(hot|n)=2/5	P(high|p)=3/9	P(high|n)=4/5	P(true|p)=3/9	P(true|n)=3/5
P(overcast|p)=4/9	P(overcast|n)=0	P(mild|p)=4/9	P(mild|n)=2/5	P(normal|p)=6/9	P(normal|n)=1/5	P(false|p)=6/9	P(false|n)=2/5
P(Rain|p) = 3/9	P(Rain|n) = 2/5	P(cool|p)=3/9	P(cool|n)=1/5				

Let the instance $X=<x_1, x_2, x_3, x_4>$ = <Rain, Hot, High, False>

Calculate: ***P(X\|class = p)?*** ***P(X\|class = n) ?***

P(X\|class = p) = P(X|class = p)·P(class = p)

= P($<x_1, x_2, x_3, x_4>$|class=p)·P(class=p)

= P(x_1|p). P(x_2|p). P(x_3|p). P(x_4|p). P(class=p) (according to (2))

= P(Rain|p). P(Hot|p). P(High|p). P(False|p).P(class=p)

= 3/9*2/9*3/9*6/9.9/14 = 1/3*2/9*1/3*2/3*9/14=$\frac{1*2*1*2*9}{3*9*3*3*14}$

= $\frac{1*2*1}{3*3*7} = \frac{2}{63} \approx 0.0317$ (1)

P(X\|class = n) =P(X|class = n)·P(class = n)

= P($<x_1, x_2, x_3, x_4>$|class = n)·P(class = n)

= P(x_1|n). P(x_2|n). P(x_3|n). P(x_4|n). P(class = n)

= P(Rain|n). P(Hot|n). P(High|n). P(False|n).P(class = n)

= 2/5*2/5*4/5*2/5.5/14 =$\frac{2*2*4*2*5}{5*5*5*5*14}$

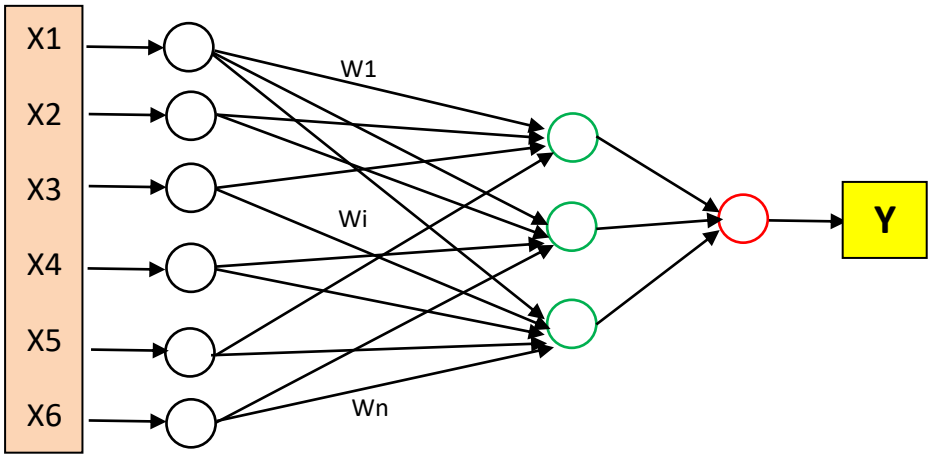
= $\frac{2*2*4}{5*5*5*7} = \frac{16}{875} \approx 0.0183$ (2)

From (1) and (2), the instance X= <Rain, Hot, High, False> can be classified in the class C_1 = p because:

P(X|class=p)·P(class=p) > P(X|class=n)·P(class=n) <==> 0.0317>0.0183

3. Artificial Neural Networks ANNs

- An ANN simulates the biological nervous system.
- A neural network is composed of several interconnected neurons.
- A weight is associated with each arc.
- Each neuron is associated with an output value.



Characteristics of ANNs

- Learning capacity: learn and change their behavior based on any new experience (sample).
- Allow to automatically discover complex models.
- There exist several neural network models: MLP (Multi-Layer Perceptron), RBF (Radial Basis Function), Kohonen, ...
- Classification method: Adjust weights using error

Error = Desired value – Current value.

ANN learning methods

- Method of Backpropagation of the gradient
- Kohonen method
- RBF method (Radial Basis Function)
- Probabilistic neural networks
- ART (Adaptive Resonance Theory)

Some guidelines for building an ANN model

- Inputs' representation
- Number of input nodes: corresponds to the dimension of the problem data (Nb.attributes or their coding).
- Determine the number of layers and the number of nodes in each layer
- Number of hidden layers: Adjusted during the training stage.
- Number of nodes per layer: the number of nodes per layer is at least two, and at most equal to the number of input nodes.
- Number of output nodes: depending on the number of classes associated with the application.
- Rich network➔ great power of expression (Ex. 4-2-1 is less powerful than 4-4-1)
- Warning: Choose a rich architecture but not too much – heavy ==> Over-specialization problem (Overfitting)

Learning an ANN

➢ ***Principle objective:*** obtain a set of weights that allow most instances in the training set will be correctly classified.

➢ ***Steps :***

- Initial weights are randomly generated
- Input vectors are processed sequentially by the network
- Calculation of activation values of hidden nodes.
- Calculation of the output vector.
- Error calculation [***Desired output – Current output***].
- The weights are updated based on the error.

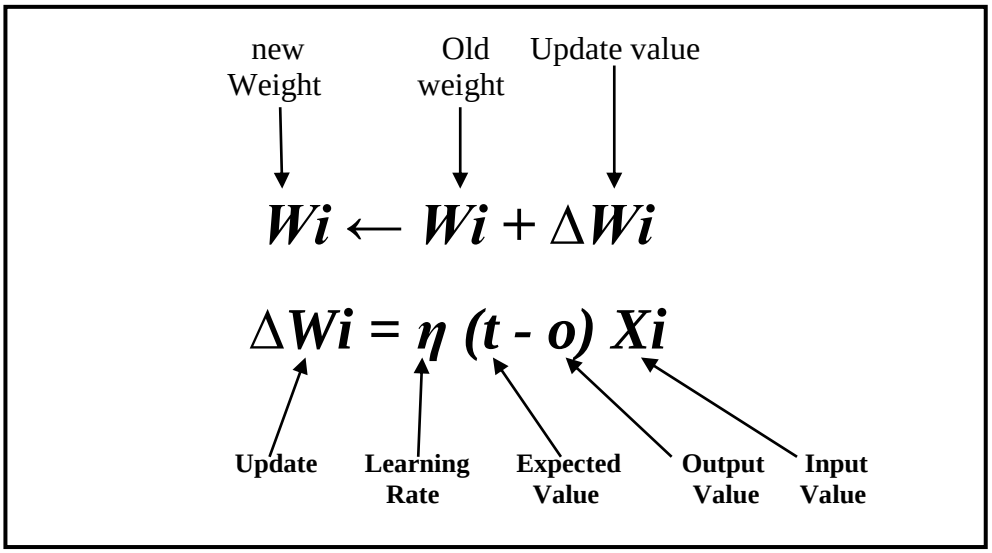
$$\boldsymbol{E(MLP) = \frac{1}{2}\sum_{x \in S}^{N}(d(x) - a(x))^2}$$

With :

x: an example of the learning base S

d(x): desired output a(x): output calculated by the ANN

- Weights are updated using the error.
- Learning is done according to Hebb's law (backpropagation law)



- The learning rate parameter $\eta \in [0,1]$ influences the update of the weights. (Great value ➔Strong modification; Small value ➔ minimal modification)
- Stopping Criterion: Tolerance defines the target error.
- and/or Number of well-classified instances (threshold)

Network Pruning

- *N* input nodes, *h* hidden layers, and *m* output nodes *h(m+n)* arcs (weights)

⇨ ***Pruning:*** Remove edges and nodes that do not affect the network error rate. Avoid the problem of over-specialization (over-fitting).

- This allow to generate concise and clear rules.

ANN Neural Networks : Advantages

- Generally, give good error rate.
- Tool available in data mining environments.
- Noise resistance➔pattern recognition (voice, images on a retina, etc)
- Quick classification (when the network is built)
- Combination with other methods (e.g. decision tree for attribute selection)

ANN Neural Networks : Disadvantages

- Very long learning phase.
- Several parameters (architecture, synaptic coefficients, etc.)
- Low explanatory power (black box)
- Not easy to incorporate domain knowledge.
- Scalability over time (learning phase)