Compact Course in Data Mining

Data Mining Algorithms

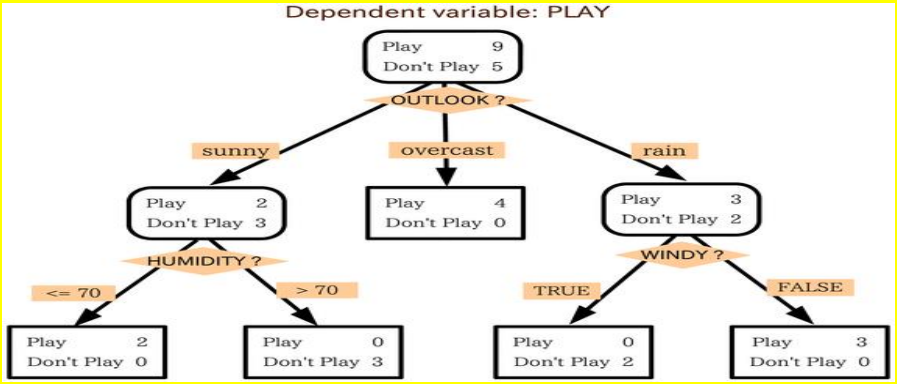
Professor Dr. Gholamreza Nakhaeizadeh

Some Data Mining algorithms

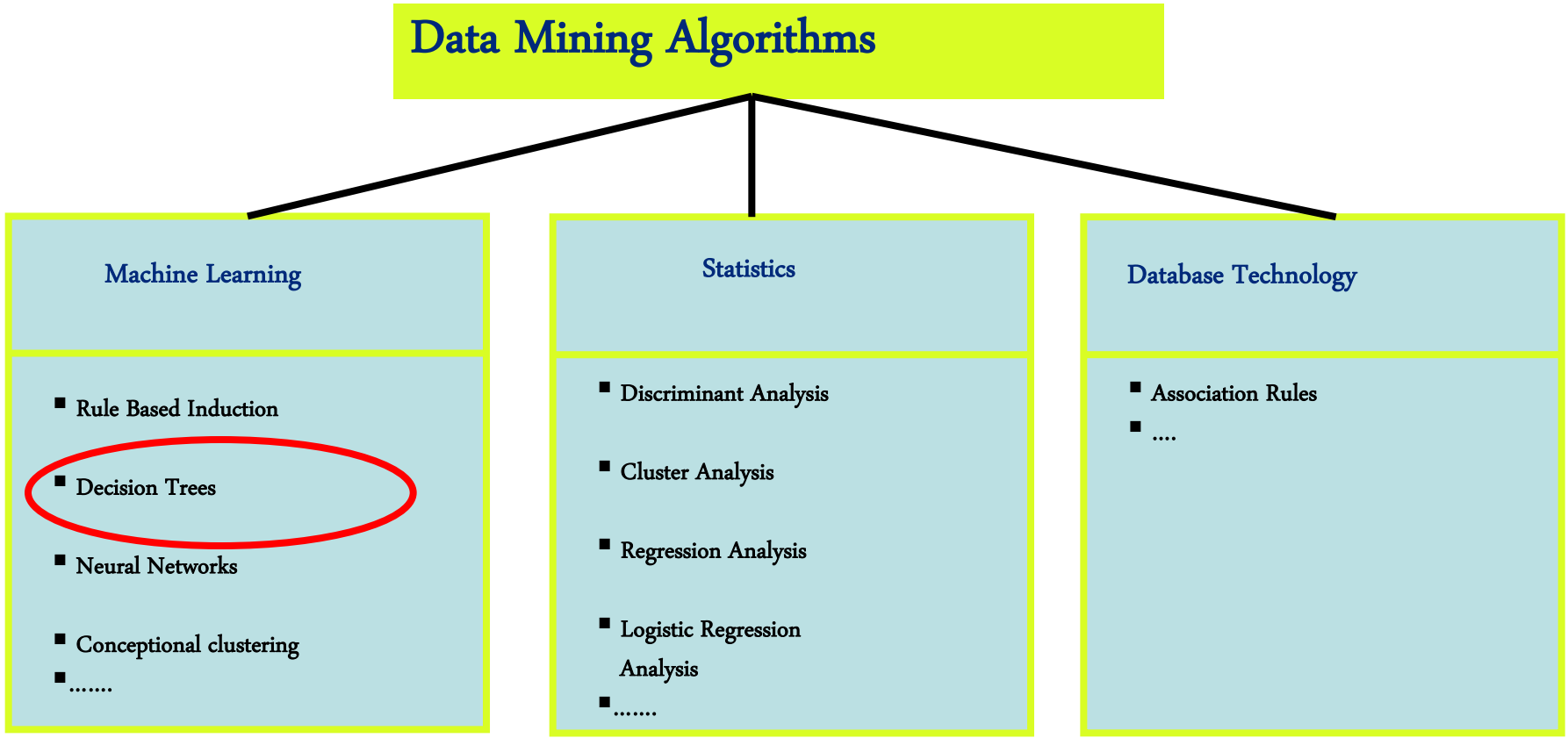
- Supervised Learning
  - Tree Family
    - Decision Trees
    - Regression Trees
    - Model Tree
  - Artificial Neural Networks
  - K-Nearest Neighbors
  - Naïve Bayes
  - Linear Regression

- Unsupervised Learning
  - Clustering
    - K-Means
    - K-Medoids
  - Association Mining
    - Apriori

Decision Trees



Artificial Neural Networks



Decision Trees (DT)

Introduction

- DT are classification tools
- Class Variable (Target Variable): Nominal
- Attributes: Nominal or continuous-valued
- Top Down construction based on heuristic methods by using training data (Greedy instead completely search : tends to find good solutions quickly, but not always optimal ones)

Construction of the Decision Trees

Three often applied Attribute selection measures:

- **Information Gain**
- **Gain Ratio**
- **Gini Index**

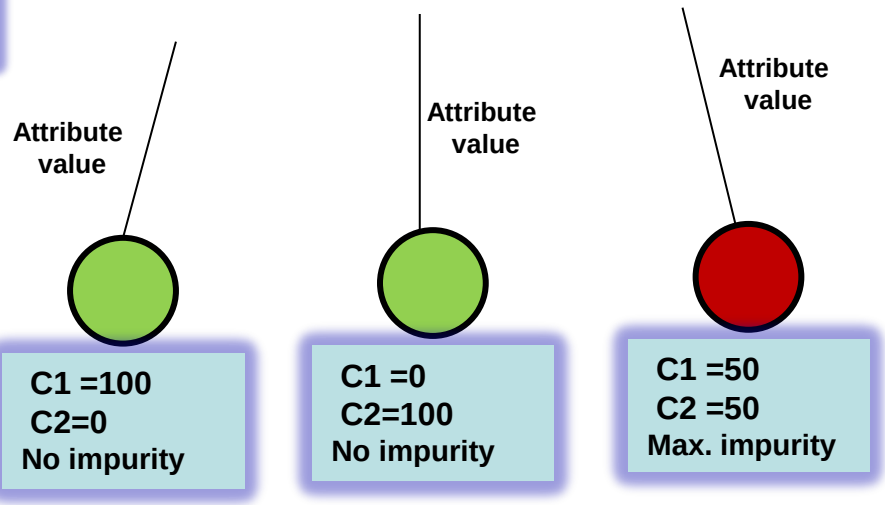
They are all based on the "Impurity" in the nodes of DT

Example : Number of observation = 100

C_1 = Number of observations in class 1

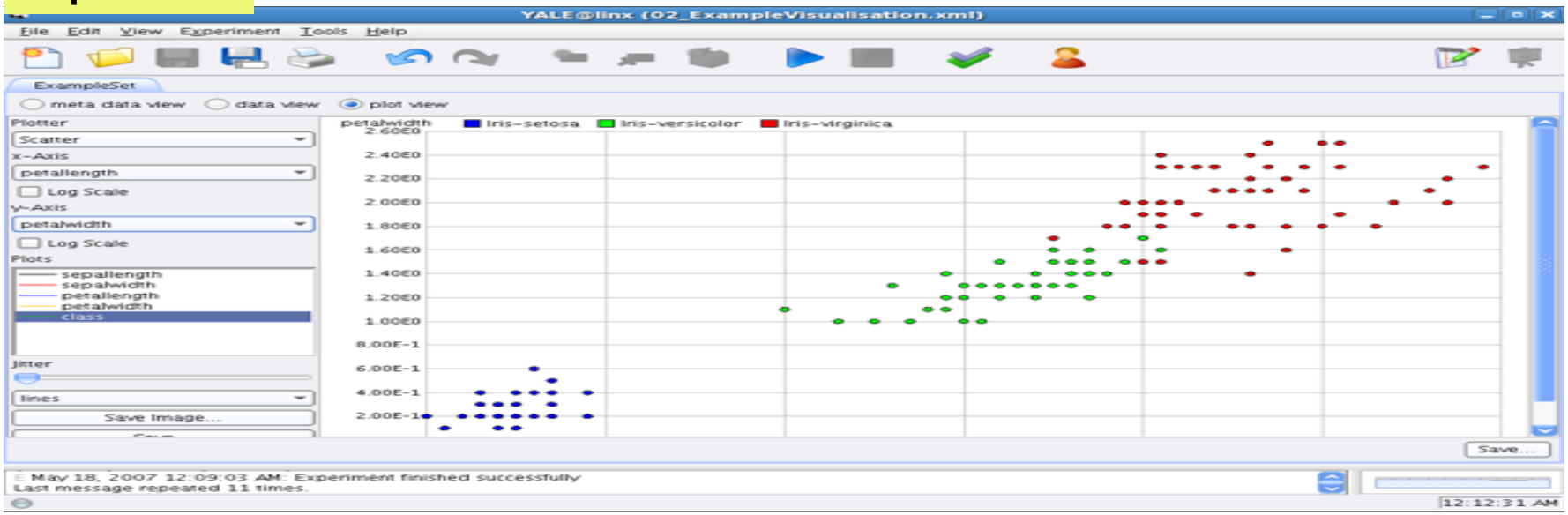
C_2 = Number of observations in class 2

$$C_1 + C_2 = 100$$



Practical Work, Decisin Trees

RapidMiner



Use: Churn.aml
W-J48

Attribute selection measures in Decision Trees

information Gain (IG)

IG is based on information theory due to Shannon
What does it means?

Example :
Finding a certain number between 1 and 1000 by asking question

1. Alternative

Choose randomly a number between 1 and 1000 and ask whether it is the right one. No optimal method, because in the worst case 999 questions are needed to find the right number.

In this case, if the first answer is “no”, the IG has been very little. Because, there are still 999 alternative numbers between them the number we are looking for is.

Attribute selection measure in Decision Trees

information Gain (IG) (continues)

Second alternative

The first question should be: is the number ≤ 500?

The IG of this question is too higher because after the answer we have to search between 500 numbers instead of 1000.

If the answer of this question is positive, the next question is, as it may be expected: Is the number ≤ 250? and so on .

In this example, IG of each new question is equal to the amount of information one gains by asking this question.

Higher the IG of a question (an attribute) → quicker to reach the goal.

Attribute selection measure in Decision Trees

information Gain (IG) (continues)

Definition of Entropy

Y: a random variable and **$P(Y=b_1)=p_1,\ldots\ldots\ldots\ldots P(Y=b_m)=p_m$**

Entropy of Y:

$$I(Y)= - p_1 \operatorname{Log}_2 p_1 - p_2 \operatorname{Log}_2 p_2 - \cdots\cdots\cdots\cdots\cdots - p_m \operatorname{Log}_2 p_m = -\sum_{i=1}^{m} p_i \operatorname{Log}_2 p_i \quad (1)$$

I(Y) is the expected information needed to find a certain value in distribution of Y

Attribute selection measure in Decision Trees

information Gain (IG) (continues)

Remark1

Definition (1) is due to Shannon in conjunction with information theory and aims to find the number of needed bits to communicate a messages (for this reason the base of used logarithm is 2)

Remark 2

(In example for m=1000 we get pi = 1/1000) and

I (Y) in (1) is equal nearly to 10 which is the average number of the

question that one needs to find a certain number

between 1 and 1000

What about m=2 and m=3 ?

Attribute selection measure in Decision Trees

information Gain (IG) (continues)

Conditional Entropy

Example: X Income Y Football Fan

Using relation **(1)** results to:
I(X) = 1.5 and I(Y) = 1
Moreover:
P (Football Fan = yes) = 0.5
P (Income= high) = 0.5
P (Income high and Football Fan = no) = 0.25
P (Football Fan = yes | Income = medium) = 0

X	Y
high	yes
medium	no
low	yes
high	no
high	no
low	yes
medium	no
high	yes

Table 2 : Football Fan

Entropy of Y for X = b: I (Y | X = b)

Using this definition and **(1)** leads to:

$$\left.\begin{array}{l} I(Y \mid X = high) = 1 \\ I(Y \mid X = medium) = 0 \\ I(Y \mid X = low) = 0 \end{array}\right\} \quad (2)$$

Attribute selection measure in Decision Trees

information Gain (IG) (continues) **Conditional Entropy**

Generally:

$$I(Y|X) = \sum_i p(X = bi)\, I(Y \mid X = bi) \quad \textbf{(3)}$$

Called average conditional entropy

From **Table 2** and relation **(2)** we can get:
And from this table:
I(Y|X) = 0.5*1+0.25*0+0.25*0 = 0.5

X= bi	P(X=bi)	I(Y| X=bi)
high	0.5	1
medium	0.25	0
low	0.25	0

we have seen already I(Y) = 1 and now by using the values of X we have got I(Y|X) = 0.5 ⟶ the needed information reduced to half and we have got 1- 0.5 = 0.5 **"Information Gain"**

Attribute selection measure in Decision Trees

information Gain (IG) (continues)

Generally:

$$IG(Y|X) = I(Y) - I(Y|X) \qquad (4)$$

(4) called information gained by using X

Inserting (3) in (4) leads to:

$$IG(Y|X) = I(Y) - \sum p_i (X = bi)\, I(Y \mid X = bi) \qquad (5)$$

In the relation (5), like before p (x=bi) can be approximated by Ni/N, where Ni is the frequently of the value xi in X.

(5) Is one of the measures that has been used for attribute selection in Decision trees. The Decision Tree algorithms ID3 e.g. uses this measure

Example: Computer buyers

Nr.	Age	Income	Student?	Credit Rating	Buys Computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle aged	medium	no	excellent	yes
13	middle aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

- Source: Jiawei Han, et al. 2006

Table:1
Computer buyers

Attribute selection measure in Decision Trees

information Gain (IG) (continues)

Example : Computer buyers

- Two classes: Buys Computer (yes or no)
- C1: class 1 (yes), C2: class 2 (no)
- N1: Numbers of tuples in C1= 9
- N2: Numbers of tuples in C2 = 5
- N=N1+N2=14
- p1: probability that a tuple belongs to C1
- p2: probability that a tuple belongs to C2
- Probability should be approximated by the portions
- Thus: p1=N1/N = 9/14 and p2= N2/N = 5/14

Using relation **(1)** results to

$$I(Y) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.94$$

This is the Expected information (entropy) needed to classify a tuple.

Nr.	Buys Computer
1	no
2	no
3	yes
4	yes
5	yes
6	no
7	yes
8	no
9	yes
10	yes
11	yes
12	yes
13	yes
14	no

Construction of Decision Trees

Root selection: using the attribute with highest information gain

Example: Computer Buyers

In the following we show the target variable (computer Buyers) with Y and the attributes (Age, Income..) with X

Now we calculate IG (Y) regarding attribute **age**

$IG_{age}(Y) = I(Y) - I(Y|\text{ age})$ **(6)**

with

$I(Y|\text{ age}) = p(\text{youth})* I(Y \mid \text{age=youth}) + p(\text{middle aged})* I(Y \mid \text{age=middle aged}) + p(\text{senior})* I(Y \mid \text{age=senior})$ **(7)**

we have seen already that I(Y)=0,94 and for Attribute age we have
p (youth) = 5/14, p (senior)= 5/14 and
p (middle aged) = 4/14

X Y

Nr.	Age	Income	Student?	Credit Rating	Buys Computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle aged	medium	no	excellent	yes
13	middle aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Root selection: continues

On the other hand :
p (Y=no | age=youth)=3/5
p (Y= yes | age = youth) = 2/5

It means.
I (Y | age=youth) =
-3/5 Log (-3/5) -2/5 Log (-2/5)= 0,968 **(8)**

From (7) and (8) we get:

P(youth)* I (Y | age= youth) = 5/14 * 0.968 = 0,346

In the same way we can calculate the other components of (6) :
IG (Y) = I (Y) – I (Y | age) = 0.246
age

and for the other attributes:
IG (Y) = 0.029
income
IG (Y) = 0.151
student
IG (Y) = 0.048
Credit Rating

Nr.	Age	Income	Student?	Credit Rating	Buys Computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle aged	medium	no	excellent	yes
13	middle aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

IG of the attribute *age* is at the highest; Splitting of the DT starts by using this attribute as the root and its values (senior, middle aged, and youth) as the first branches of the tree

Construction of Decision Trees

splitting

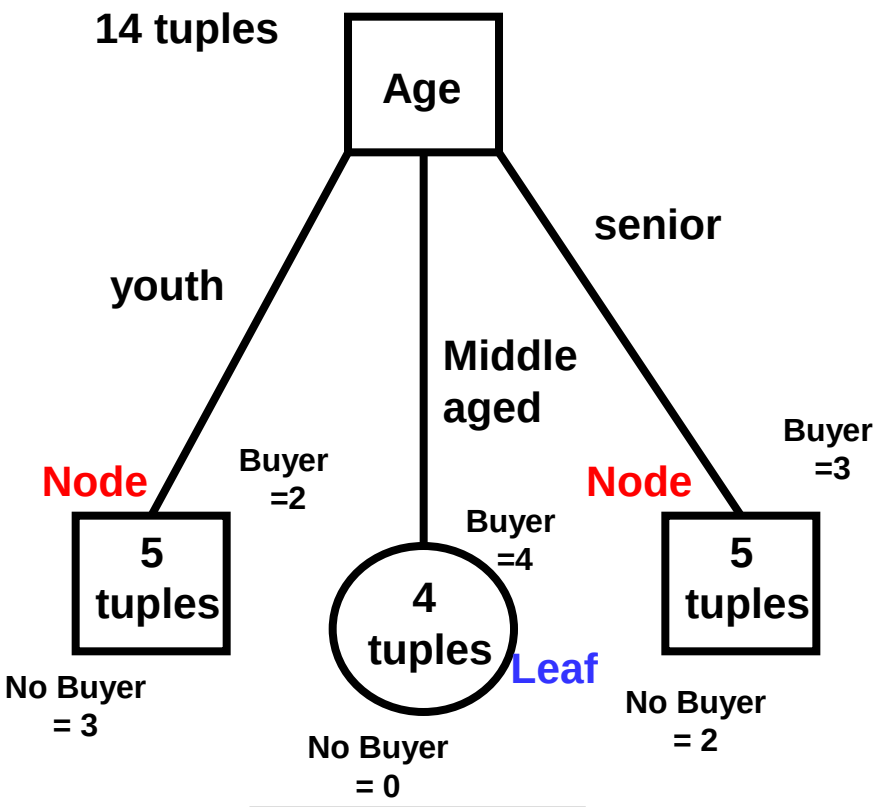
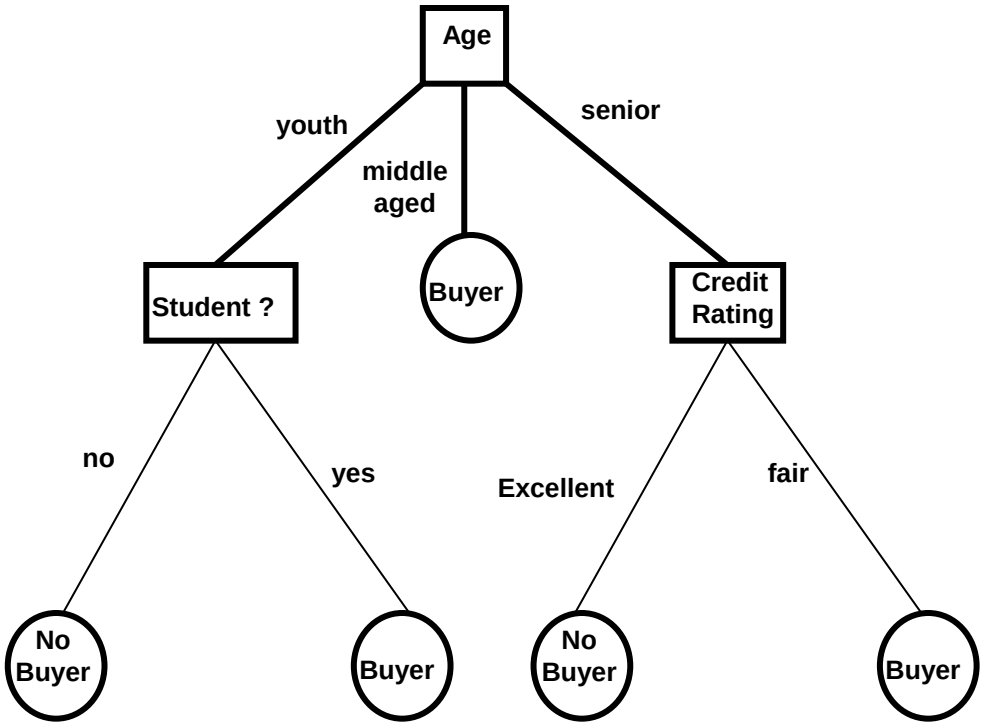


Fig 1, first splitting of DC

Nr.	Age	Income	Student?	Credit Rating	Buys Computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle aged	medium	no	excellent	yes
13	middle aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Construction of Decision Trees

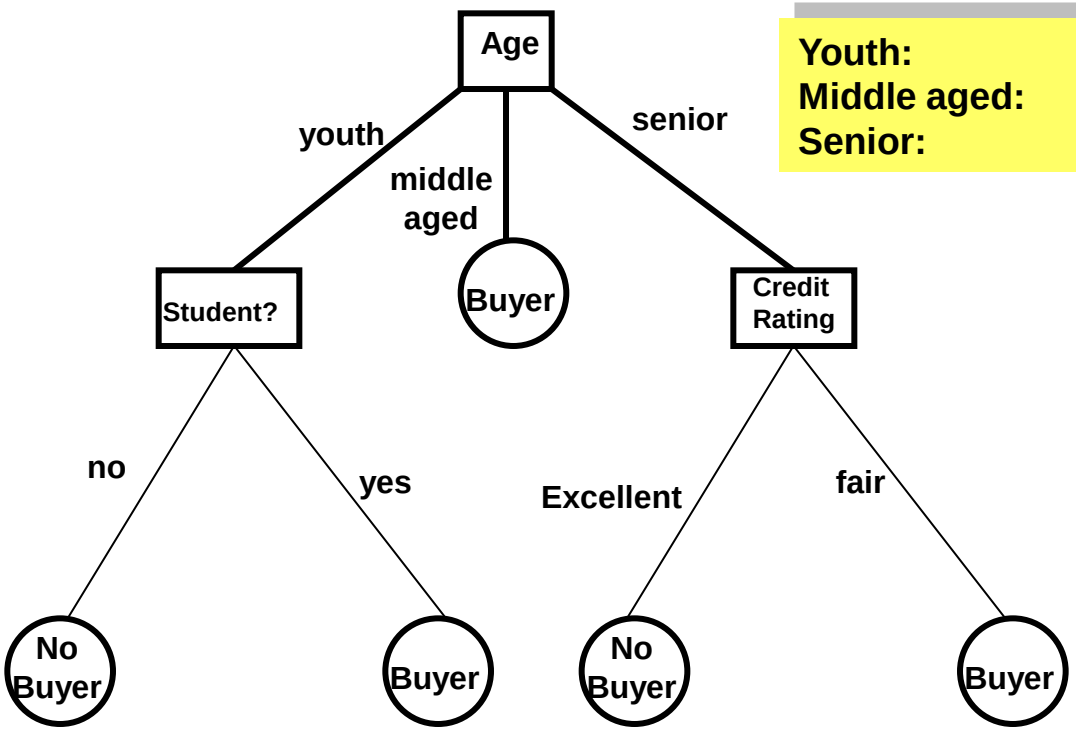
Splitting (continues)



Nr.	Age	Income	Student?	Credit Rating	Buys Computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle aged	medium	no	excellent	yes
13	middle aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Notice: Attribute income wasn't use

Decision Trees as classifiers for new tuples



Youth: <= 30
Middle aged: 30=< <=40
Senior: > 40

1. Age 37 = buyer
2. Age 55 with excellent credit rating = no buyer
3. Age 18 but no student = no buyer
4. …..

Construction of Decision Trees

Splitting of continuous - valued attributes

Example: monthly income of 10 individuals

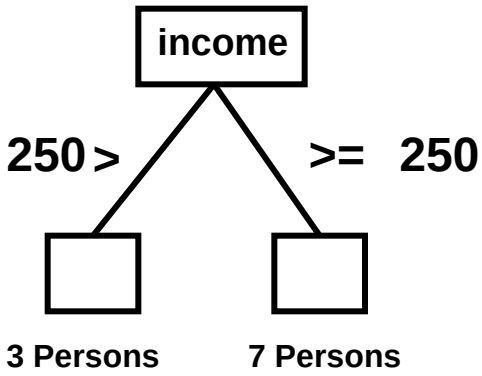
Income	700	300	200	200	300	200	500	300	700	500

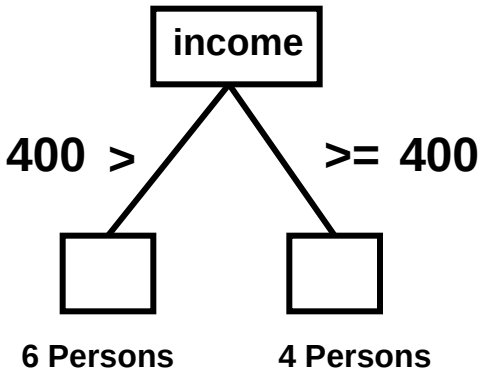
Step 1: Sort the attribute values increasing and calculate the average of each two neighbors as possible threshold

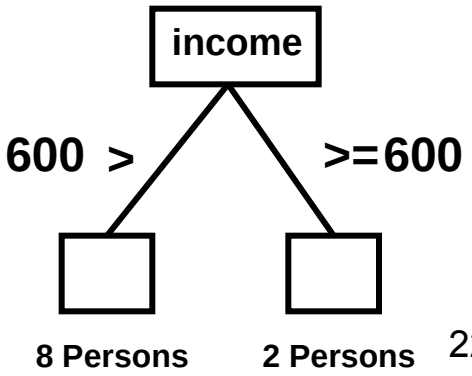
Attribute value	200	300	500	700
Average	250	400	600	

Step 2: Split the node using the averages as alternative thresholds

Step 3: Calculate for each split the IG (income) and choose the one with highest IG





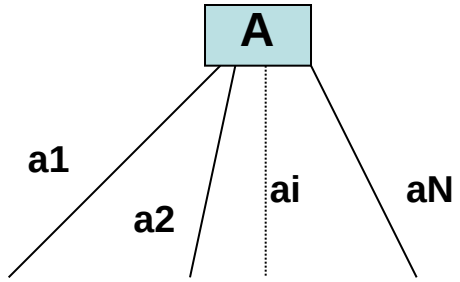


Attribute selection measure in Decision Trees

Other selection measure Gain Ratio

IG has a significant drawback:
it does not take into account the number of attribute values

Suppose that we have just N tuples and between the attributes we have a discrete-valued attribute A with values a1, a2,…ai,.. aj,….aN with ai ≠ aj for ∀ i and j In this case we would have by splitting using A, so many partitions as tuples namely N:



For this case $I(Y \mid A) = -\sum_{i}^{N} 1/N \text{ Log } (Y \text{ I } ai) = 0$; **Regarding:** $IG(A) = I(Y) - I(Y|A)$

means A would have the maximal IG .
This extreme case shows very well that the IG prefers selection attributes with a large number of partitions.

Attribute selection measure in Decision Trees

Other selection measure

Gain Ratio (continues)

To overcome this problem Quinlan suggests for C4.5 (extension of ID3 algorithm) using Gain Ratio instead of Information Gain.

Gain Ratio is defined as: GR(A) = IG (A)/I (A)

with

$$I(A) = -\sum_{i}^{n} n_i/N * Log_2 (n_i/N)$$

n_i/N is the portion of the tuples with attribute value a_i

Nr.	Age	Income	Student?	Credit Rating	Buys Computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle aged	medium	no	excellent	yes
13	middle aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

a_1 = high $n_1/N = 4/14$
a_2 = medium $n_2/N = 6/14$
a_3 = low $n_3/N = 4/14$

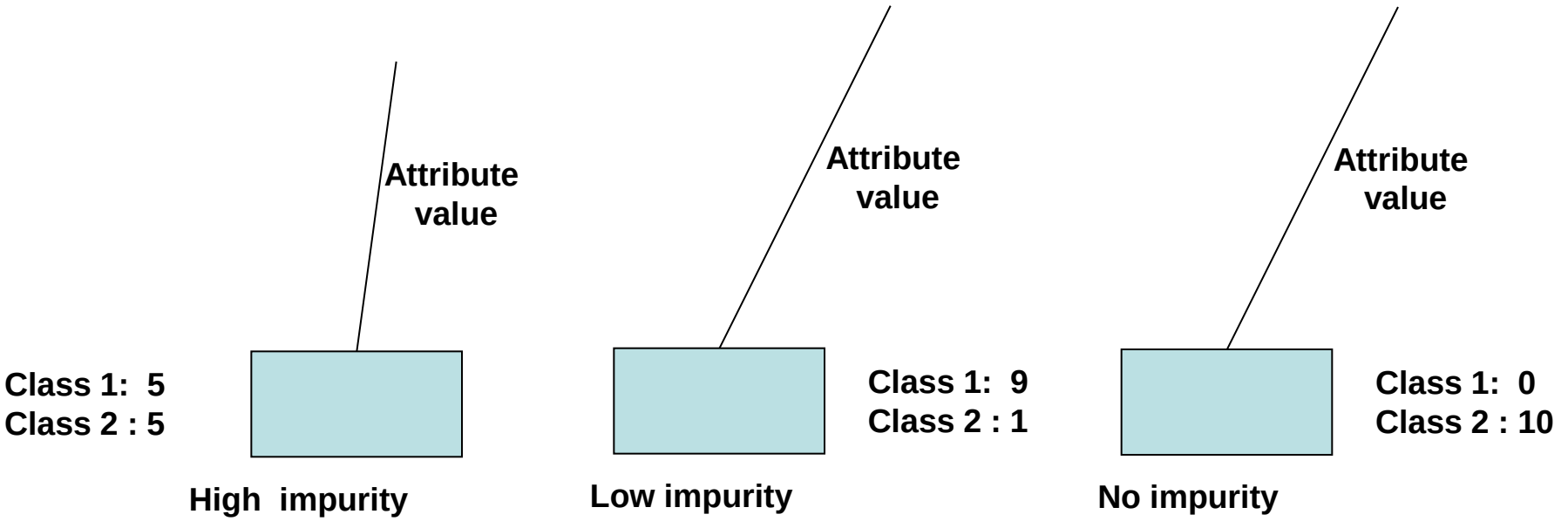
→ **I (A) = - 4/14 * Log 4/14 – 6/14 * Log 6/14 – 4/14 Log 4/14 = 1.557**

we had calculated IG (income) = 0. 029 thus : GR(A) = 0.029/1.557 = 0.019

Attribute selection measure in Decision Trees

Gini Index (Gini)

Based on impurity in the nodes



Attribute selection measure in Decision Trees

Gini Index (Gini) **Gini Index of a node**

$$Gini(k) = 1 - \sum_{j} (n_j / n | k)^2$$

n = Number of tuples at the node k

n_j = Number of the tuples belong to the class j at the node k

n_j / n = relative frequently of the class j at the node k

For two classes:

$n_1 = 0 \quad n_2 = n$ → Gini = 0 → lowest impurity

$n_1/n = ½ \quad n_2/n = ½$ → Gini = 1/2 → highest impurity

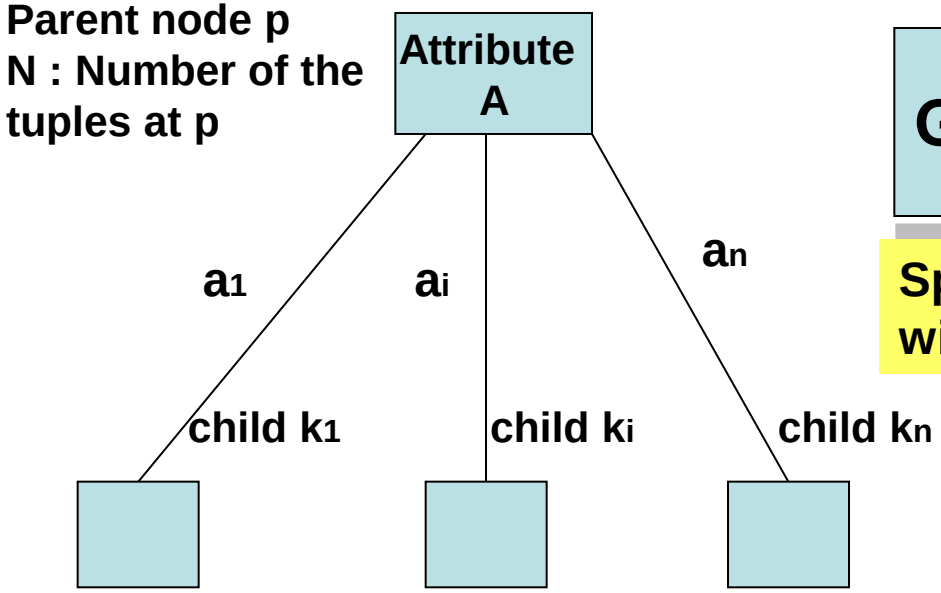
Further examples:

$n_1/n = 1/6 \quad n_2/n = 5/6 \quad Gini = 1 - (1/6)^2 - (5/6)^2 = 0.278$

Attribute selection measure in Decision Trees

Gini Index

Gini Index as attribute selection measure



$$Gini(A) = \sum_{i}^{n} (N_i/N)\, Gini(k_i)$$

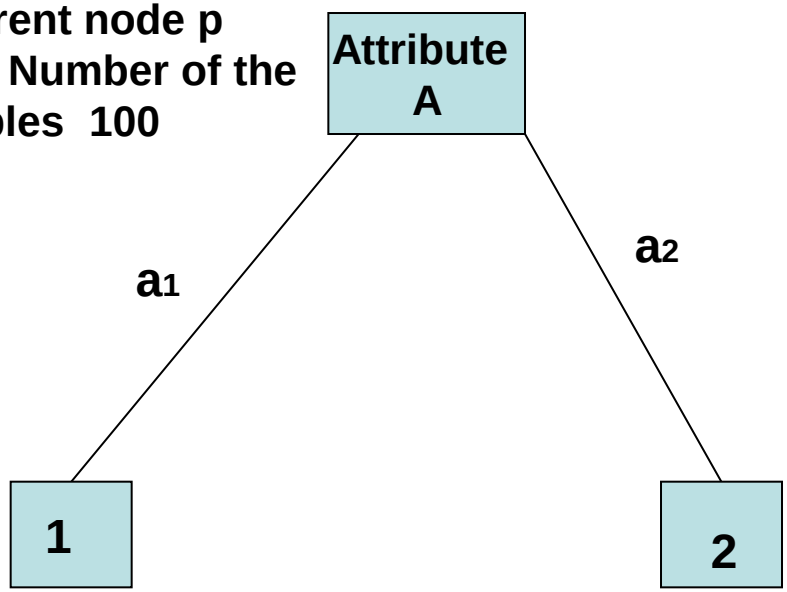
Splitting will be done for the attribute with minimal GINI

Some of DT algorithms among them CART use Gini Index

Attribute selection measure in Decision Trees

Gini Index | **Example**

Parent node p
N : Number of the tuples 100



$N_1 / N = 40/100 = 0.4$

$N_2 / N = 60/100 = 0.6$

$$Gini(A) = \sum_{i}^{n} (N_i/N)\, Gini(k_i)$$

Gini (A) = 0.4 * 0.5 + 0.6 * 0.278
=0.367

$N_1 = 40$
C1= 20, C2= 20

$N_2 = 60$
C1 = 10 C2= 50

$$Gini(k) = 1 - \sum_{j} (n_j / n|\, k)^2 \rightarrow$$

$$Gini(1) = 1-(20/40)^2-(20/40)^2 = 0,5$$

$$Gini(2) = 1-(10/60)^2-(50/60)^2 = 0,278$$

Construction of Decision Trees

Overfitting

Overfitting means: DT can classifies the training data with a relative high accuracy rate but not the test data. It means the DC is not able to generalize

Solution: Tree Pruning

- Pre-pruning
- Post-pruning

•Pre-pruning: Stop growing of the tree in the early stages

Stop Criteria:
- at pure nodes or nodes with high degree of purity
- small number of tuples at a node
- no more improving of accuracy rate by more growing

……

Construction of Decision Trees

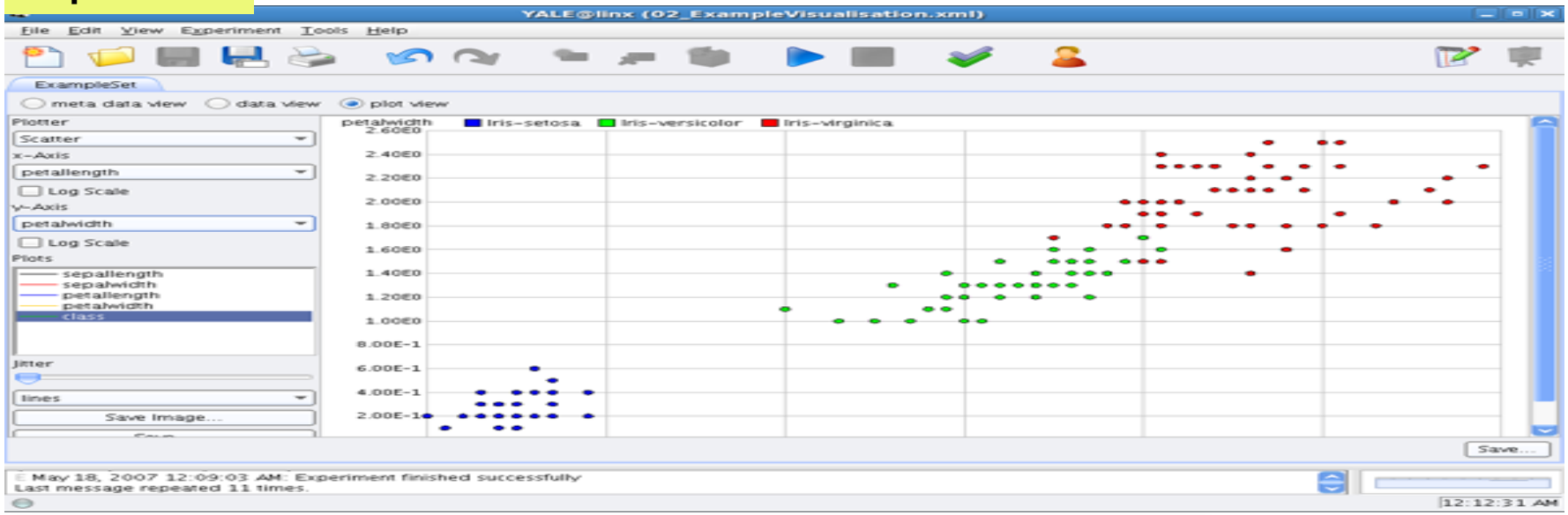
Overfitting

Post-pruning :

- **Produce a full grown tree**
- **Prune this tree in different depths to produce a set of pruned trees**
- **Select the best one using a "validation" data set**

Construction of Decision Trees

RapidMiner



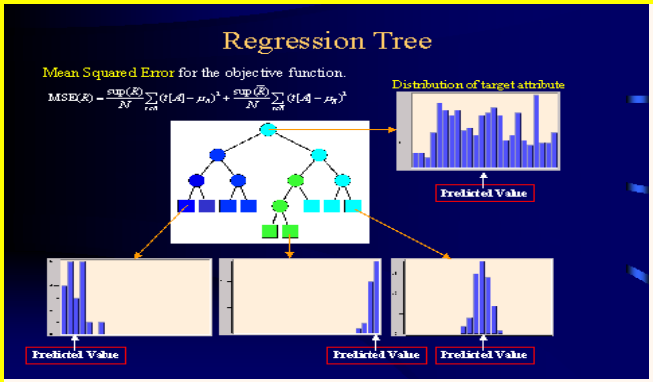
Practical Work: DT without Pre-Pruning, Dataset = Churn
Algorithm= W-J48, Set Tree Parameter U= yes

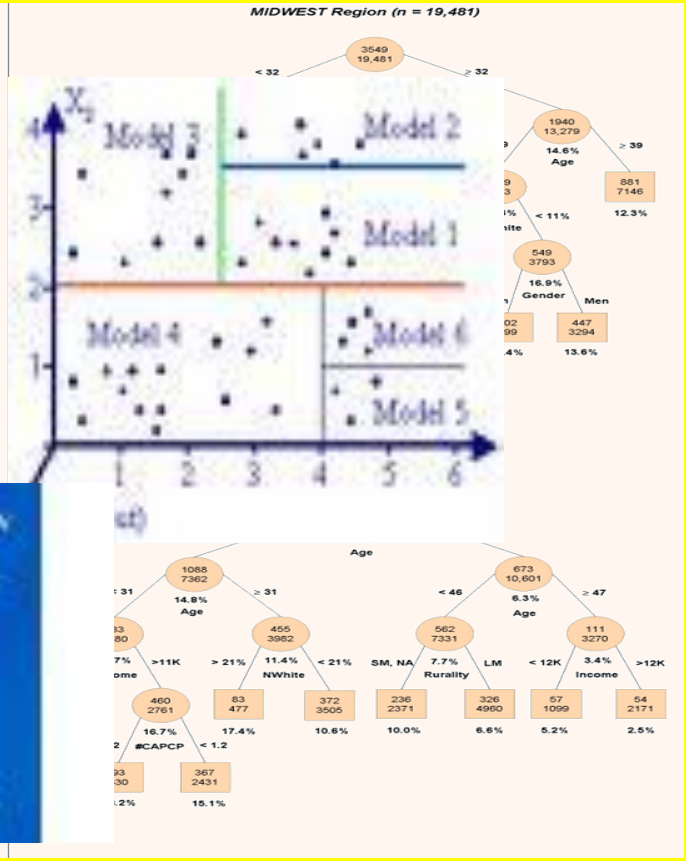
Weakness and Strength of Decision Trees

- **Strength**
  - Produce understandable classification rules with reasonable accuracy rates
  - Decision trees can be constructed relatively fast
  - Decision trees indicate clearly which attributes are most important for classification

- **Weakness**
  - By using of decision trees only descriptive analysis of data is possible
  - Discretization of continuous-valued is necessary
  - They are not appropriate for time series analysis and prediction

Regression and Model Trees



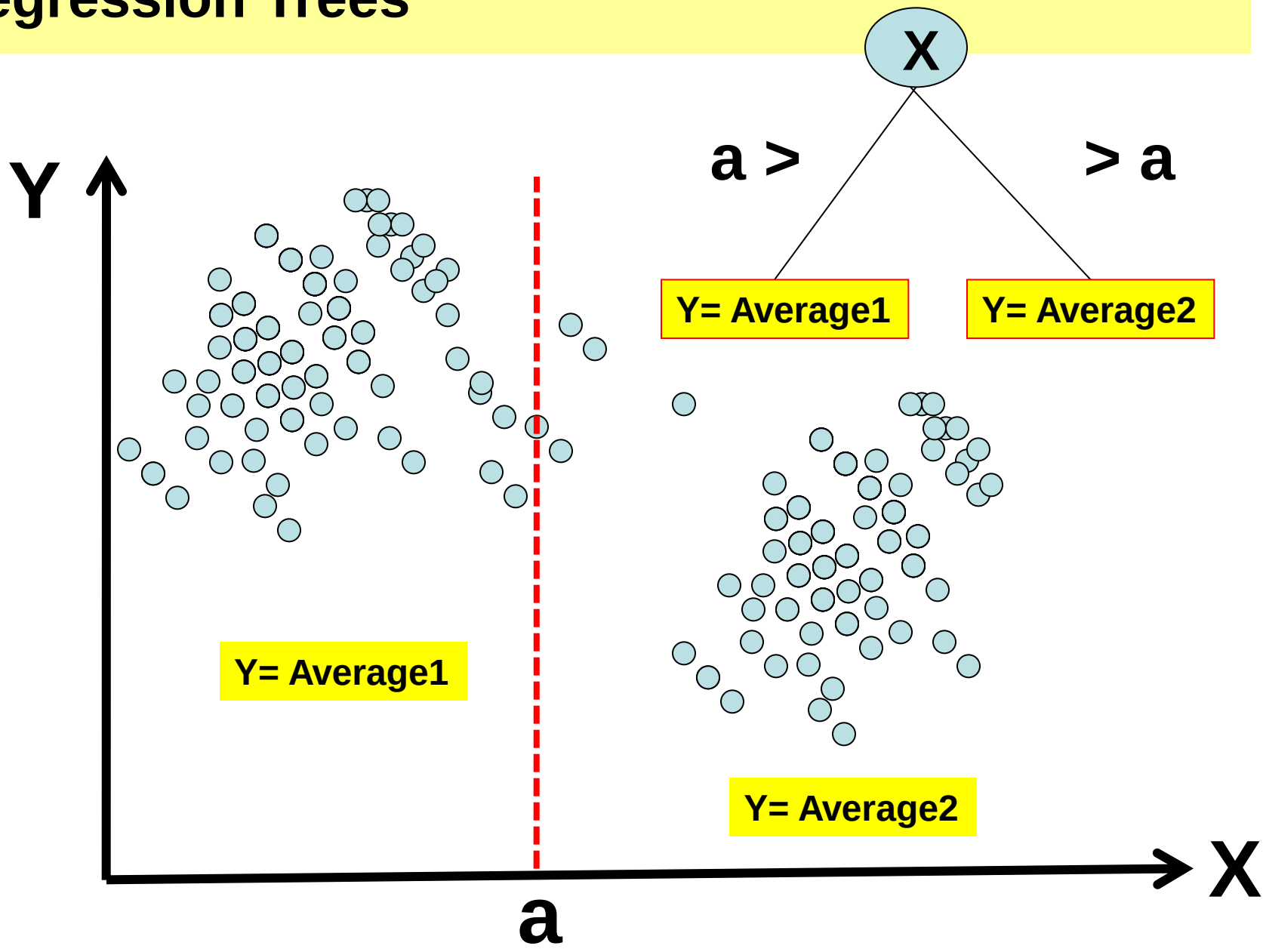


Regression Trees

General remarks

- The structure of the training data for RTs is the same as DT
- One or more attributes and a single target variable
- Attributes can be numerical or nominal
- In contrast to Decision Trees, the target variable is numerical
- The value at each leaf of the tree is the average of the target attribute for all instances arrive to this node
- RT are prediction tools

Regression Trees



Regression Trees

Choosing split attribute

In each node the algorithm chooses the attribute that leads to the nodes with smallest sum of the squared deviations from the mean

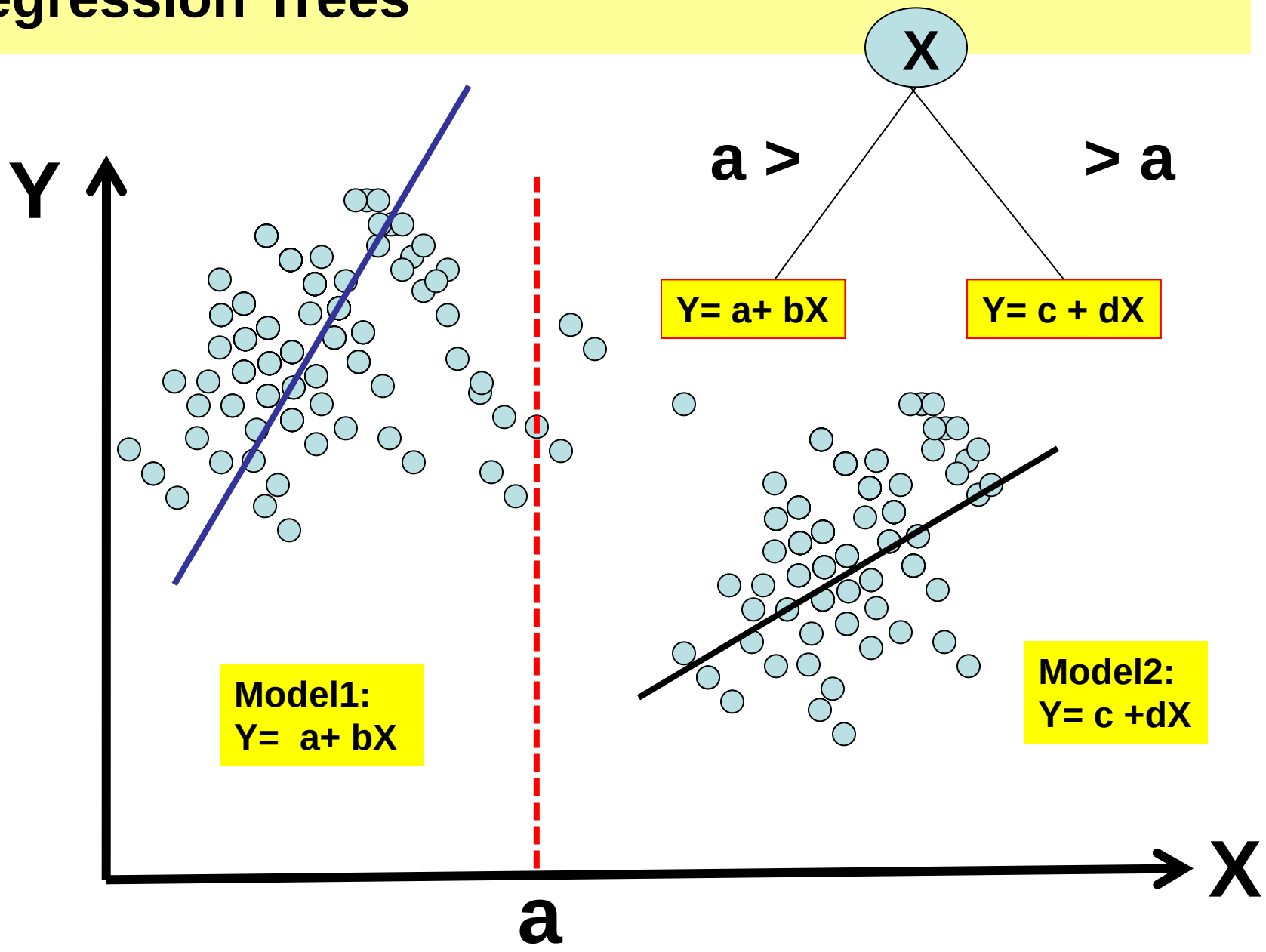
Strength and Weakness

- Have most advantages and disadvantages of the decision trees
- Can do prediction
- The predicted value can not be outside of the range of the target variable
- This a very hard disadvantages for steady increasing and decreasing target variable

Model Trees

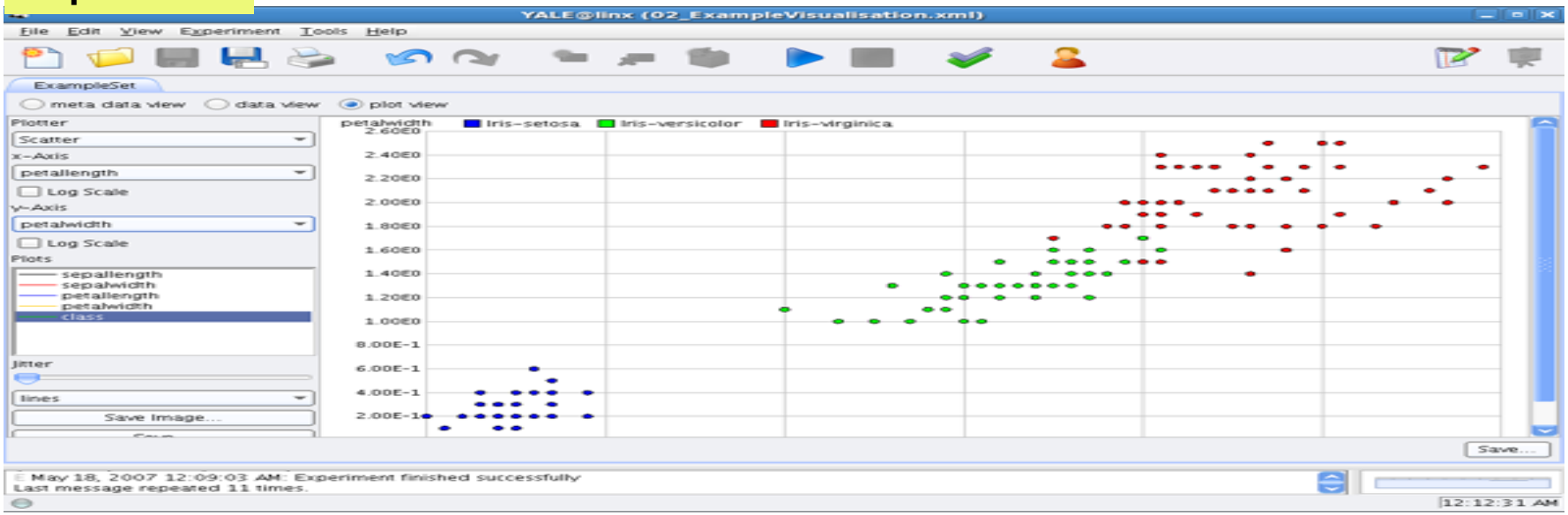
- They are a combination of Regression Trees and linear regression
- Instead of computing average in each leaf of the tree a Linear Regression is fitted
- Application of Model Trees leads, generally, to reducing the number of nodes in the tree.
- They lead, generally, to a better performance as Linear Regression and Regression Trees
- They are prediction tools

Regression Trees

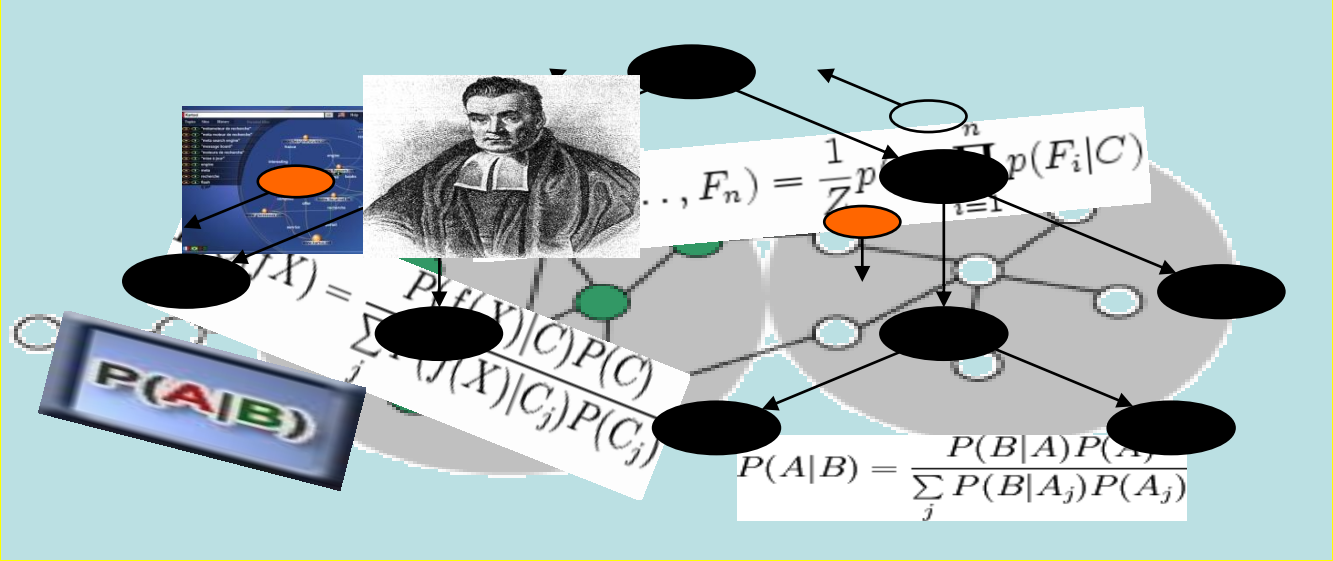


Model Trees, Practical work

RapidMiner



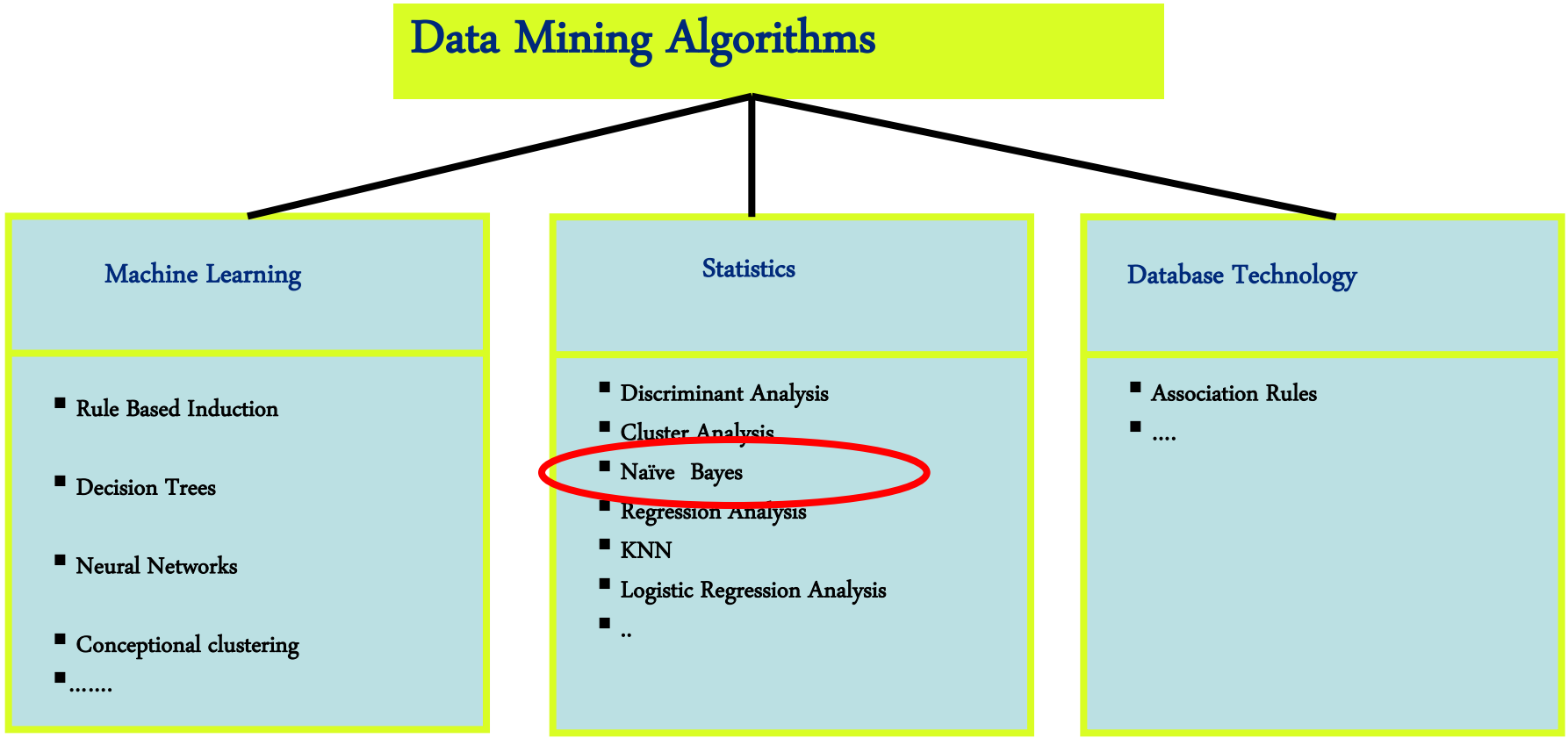
Naïve Bayes



Application of Bayesian Statistics in Classification

Professor Dr. Gholamreza Nakhaeizadeh

Artificial Neural Networks



Naïve Bayes

Introduction

- In classification tasks, assigning the class value to an observation is often stochastic and not deterministic

Reasons:

- Noisy data
- Some of the relevant attributes are not considered that are stochastic

To handle such situations, one needs stochastic approaches

- Naïve Bayes
- Logistic Regression
- …..

Example:

income	car	gender	class
2000	yes	F	good
2000	yes	F	bad

Naïve Bayes

Bayes-Theorem

X and Y : Random variables

P (X,Y) : joint probability of X and Y

P(XIY) : Conditional probability of X given Y

Then: P(X,Y) = P(YIX) . P(X) = P(XIY) . P(Y)

$$P(YIX) = \frac{P(XIY) \cdot P(Y)}{P(X)} \qquad (1)$$

Relation (1) is known as Bayes-Theorem

Naïve Bayes

Application of Bayes-Theorem in classification

X : Attributes Vector, Y: Class Vector

X and Y : Random variables

P (YIX) : Posterior probability, P(Y) : Prior probability

Bayesian Classification Task

- Building the Classifier:
  Learning P(YIX) by using data on X and Y

- Classification of new tuples:
  To each new tuple X´ assign the class value that maximizes p(YIX')

Naïve Bayes

Application of Bayes-Theorem in classification

How can we compute P (YIX) ?

By using the Bayes-Theorem

$$P(YIX) = \frac{P(XIY) \cdot P(Y)}{P(X)}$$

- P(x) is independent of Y and can be ignored
- Computing of P(Y) can be done easily by using the observations on Y
- To compute P (XIY) there are different alternative:

Naïve Bayes is one of them

Naïve Bayes

Naïve Bayes Classifier

- **Goal: Estimating the class conditional probability**
- **(Naïve) Assumption: given the class label Y the attributes are conditionally independent**

$$P(X|Y=y) = \prod_{i=1}^{m} P(X_i | Y=y), \quad X = (X1, X2, \ldots Xm) \qquad (2)$$

Instead of computing the joint conditional probability of X, it is just necessary to compute the probability of each Xi given Y

Thus, from (1) and (2) we have

$$P(Y|X) = \frac{P(Y) \cdot \prod_{i=1}^{m} P(Xi \mid Y)}{P(X)} \qquad (3)$$

Naïve Bayes Classification Rule: Assign to the new Vector X´ the class that maximizes the numerator of (3)

Naïve Bayes

Naïve Bayes Classifier

$$P(Y|X) = \frac{P(Y) \cdot \prod_{i=1}^{m} P(X_i | Y)}{P(X)}$$

$\alpha = 1/P(X) \rightarrow$ constant

$$P(Y|X) = \alpha \; P(Y) \cdot \prod_{i=1}^{m} P(X_i | Y) \qquad (4)$$

Naïve Bayes

Example

Source: http://www-users.itlabs.umn.edu/classes/Spring-2006/csci5523/index.php?page=lecture%20slides

Tid	Refund	Marital Status	Taxable Income	Evade
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

Y = Evade
X= (Refund, Marital Status, Taxable Income)

New Record X':
(Refund= no,
marital status=married,
taxable income = $ 120 K)

Based on the training data, we compute

P (yes I X') and P(no I X')

The new record is classifies as "yes" if

P (yes I X') > P(no I X')

Otherwise it is classified as "no"

Naïve Bayes

Estimation conditional probabilities P (XIY)

A. Attribute is nominal

1. Choose a value of Y
2. Determine the values of the nominal attribute X that correspond to this selected value of Y
3. Determine the fraction of these values

Example:

1. We choose Y= Evade = No

2. The values of the attribute "Refund" that corresponds to Evade=no are:

Refund	Evade
Yes	No
No	No
No	No
Yes	No
No	No
Yes	No
No	No

3. Determining of the fractions

P (X = Yes I Y= No) = 3/7

P (X = No I Y = No) = 4/7

Tid	Refund	Marital Status	Taxable Income	Evade
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

Naïve Bayes

Estimation conditional probabilities P (XIY)

B. Attribute is continuous-valued

Alternative 1:
Discretization of the continuous-valued Attribute. The rest of the procedure is similar to the case A

Alternative 2:
Assume a certain conditional distribution for the cotinuous-valued attribute(e.g. normal distribution)

$$P(X_i \mid Y_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(X_i - \mu_{ij})^2}{2\sigma_{ij}^2}}$$

The distributions parameters can be estimated by using the observations on X and Y

Naïve Bayes

Estimation conditional probabilities P (XIY)

B. Attribute is continuous-valued

Example:

Taxable Income	Evade
125	No
100	No
70	No
120	No
60	No
220	No
75	No

Tid	Refund	Marital Status	Taxable Income	Evade
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

$\bar{X} = (125 + 100 + 70 + \ldots + 75) / 7 = 110$

$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (Xi - \bar{X})^2$

$s^2 = 17850 / 6 = 2975$

$S = \sqrt{2975} = 54.54$

$$P(X_i \mid Y = \text{No}) = \frac{1}{\sqrt{2\pi}\,.\,54.54} e^{-\frac{(X_i - 110)^2}{2.\,2975}}$$

Xi = 120 ⟶ P (Xi I Y= No) = 0.0072

Source: http://www-users.itlabs.umn.edu/classes/Spring-2006/csci5523/index.php?page=lecture%20slides

Naïve Bayes

Example:

Determine the class of a new Record X:
(Refund= no, marital status=married, taxable income = $ 120 K)

Tid	Refund	Marital Status	Taxable Income	Evade
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

P (Y= No) = 7/10 P (Y = Yes) = 3/10

P (XI Y=No) =
P (Refund=No I Y=No). P(status=married I Y=No). P (T. Income = 120 IY=No)

Refund	Evade
Yes	No
No	No
No	No
Yes	No
No	No
Yes	No
No	No

→ **P (Refund = No I Y= No) = 4/7**

P (T. Income = 120 I Y= No) = 0.0072

(see the last slide)

M. status	Evade
single	No
married	No
single	No
married	No
married	No
divorced	No
married	No

→ **P (Status = married I Y= No) = 4/7**

P (X I Y=No) = 4/7 . 4/7 . 0.0072 = 0.0024

From (4) we have: P (No I X) = α P (Y= No) . P (X I Y= No) = 7/10 . 0.0024 α= 0.0016 α

Using the same method → P (Yes I X) = 0 → P (No I X) > P (Yes I X)

The class value of the new record is computed as No

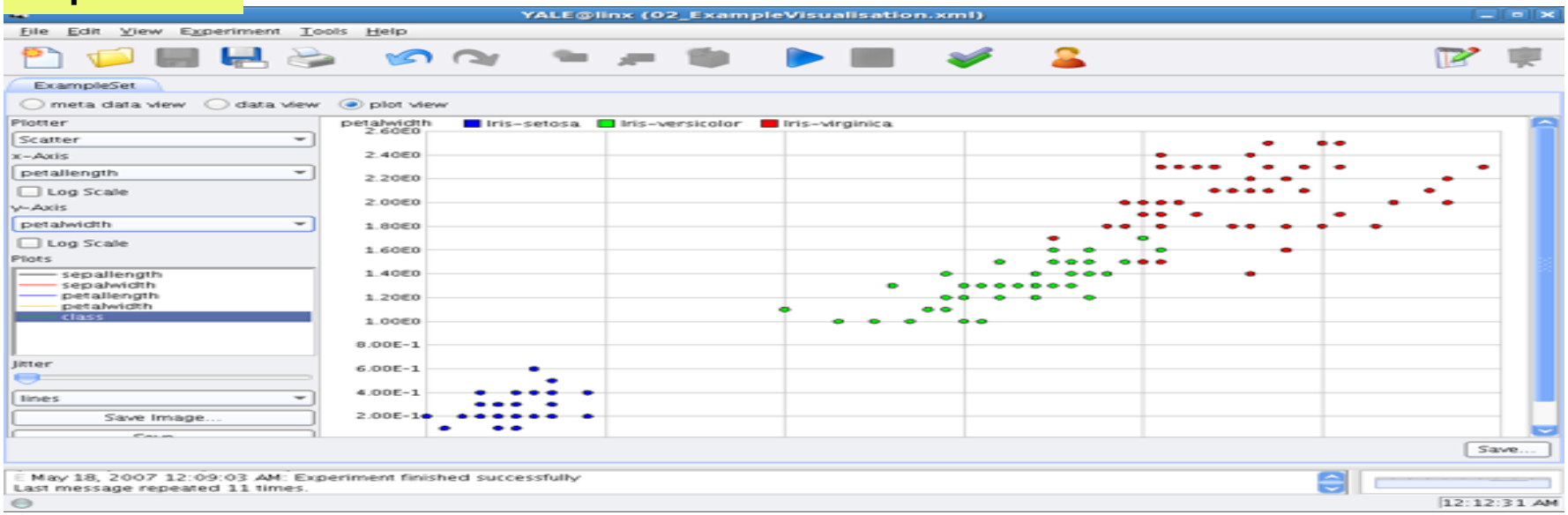
Naïve Bayes

Strength and Weakness

- Robust to noise and irrelevant attributes
- The results achieved in praxis have been very promising
- Independence assumption may not hold for some attributes

Practical Work

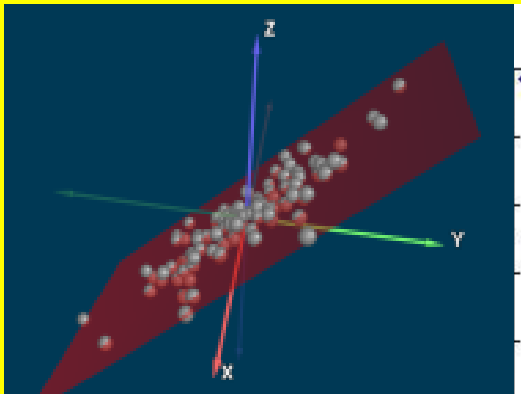
RapidMiner

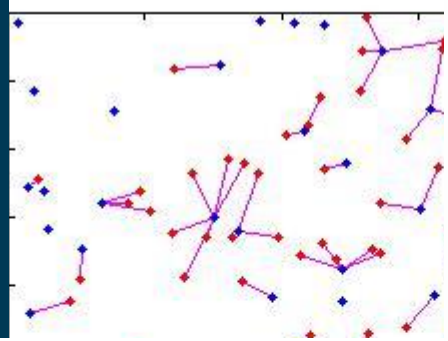


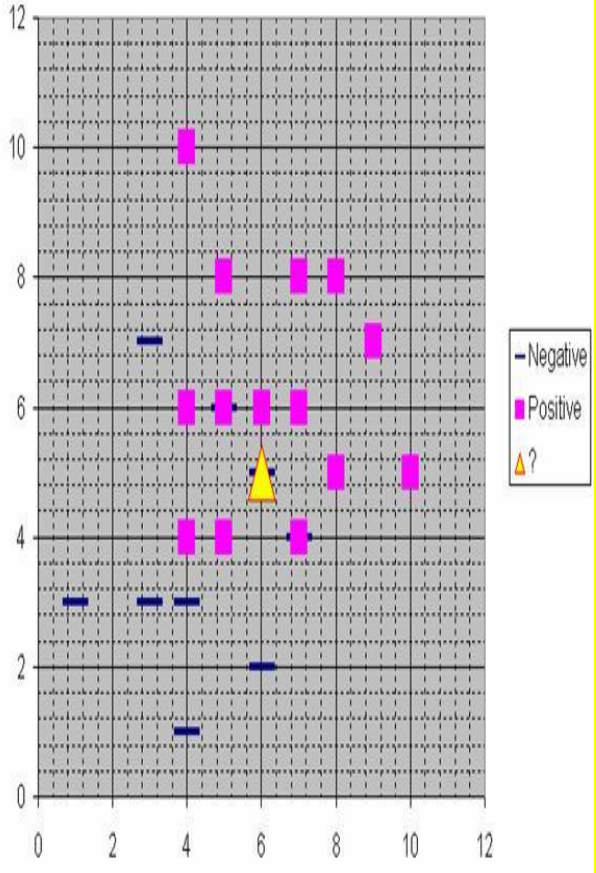
Load German_CreditTr

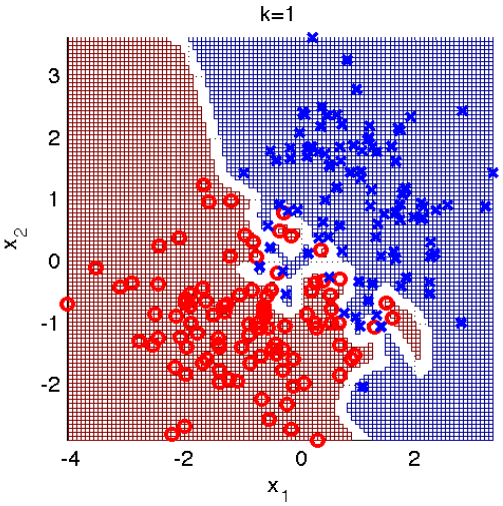
1. Compare the performance of Naïve-Bayes with other algorithms using Xvalidation

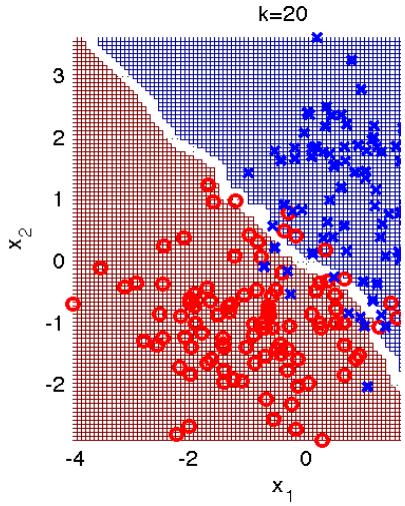
K-Nearest Neighbors



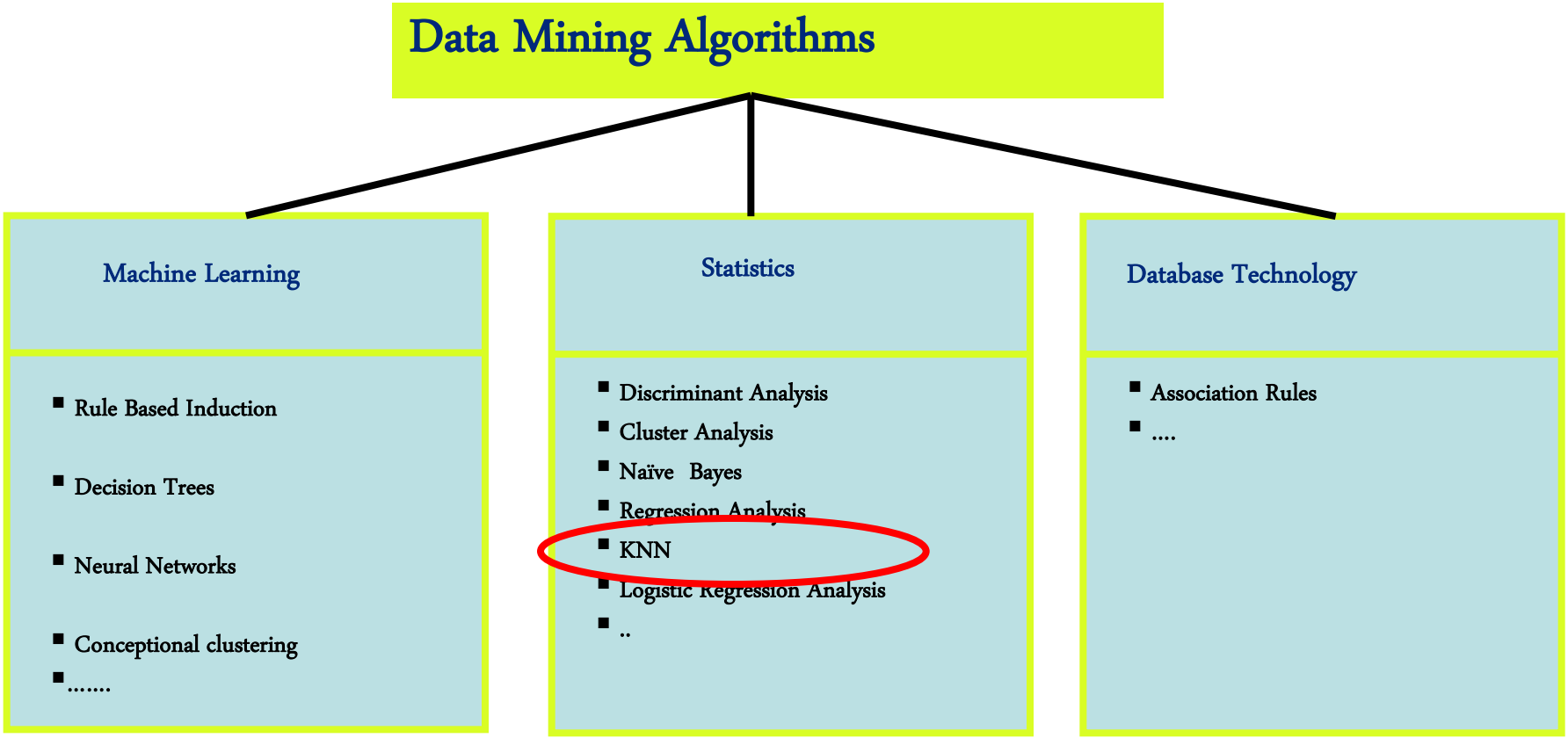








Artificial Neural Networks



K-Nearest Neighbors

General remarks

- **Belongs to Instance-Based Learning (Lazy Learning)**
- **Part of supervised learning**
- **Very simple**
- **Target Variable and attributes can be nominal or continues-valued**
- **KNN can be used for classification and prediction**

K-Nearest Neighbors

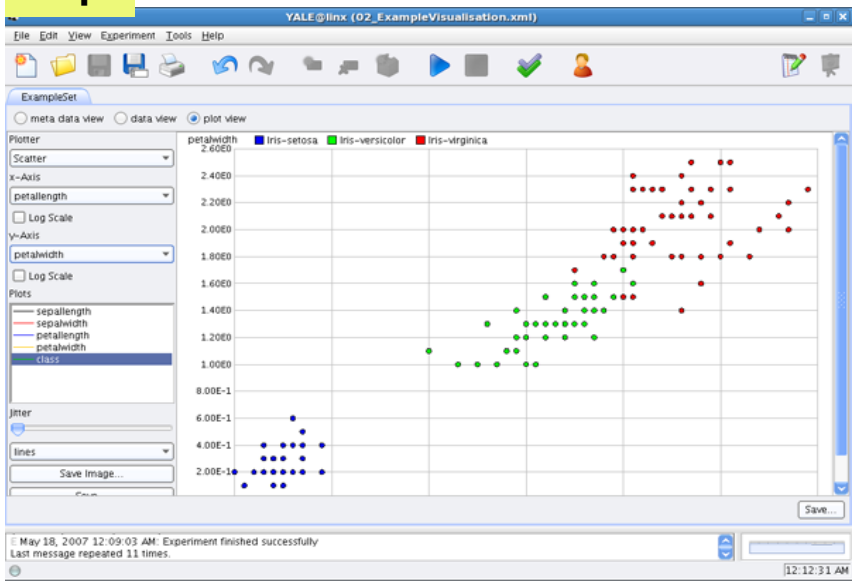
Basic Ideas

Based on minimum distance between the query instance and the training samples, the algorithm determines the K-nearest neighbors of the query instance

The class (prediction value) of the query is the class of the majority neighbors , mode (in the case of classification) or their average value or any other central tendency parameter (in the case of prediction)

K-Nearest Neighbors, Practical Work

RapidMiner



X1	X2	Y
4	3	+
1	3	+
3	3	+
3	7	+
7	4	+
4	1	+
6	5	+
5	6	+
3	7	+
6	2	+
4	6	-
4	4	-
5	8	-
7	8	-
5	6	-
10	5	-
7	6	-
4	10	-
9	7	-
5	4	-
8	5	-
6	6	-
7	4	-
8	8	-
6	5	?

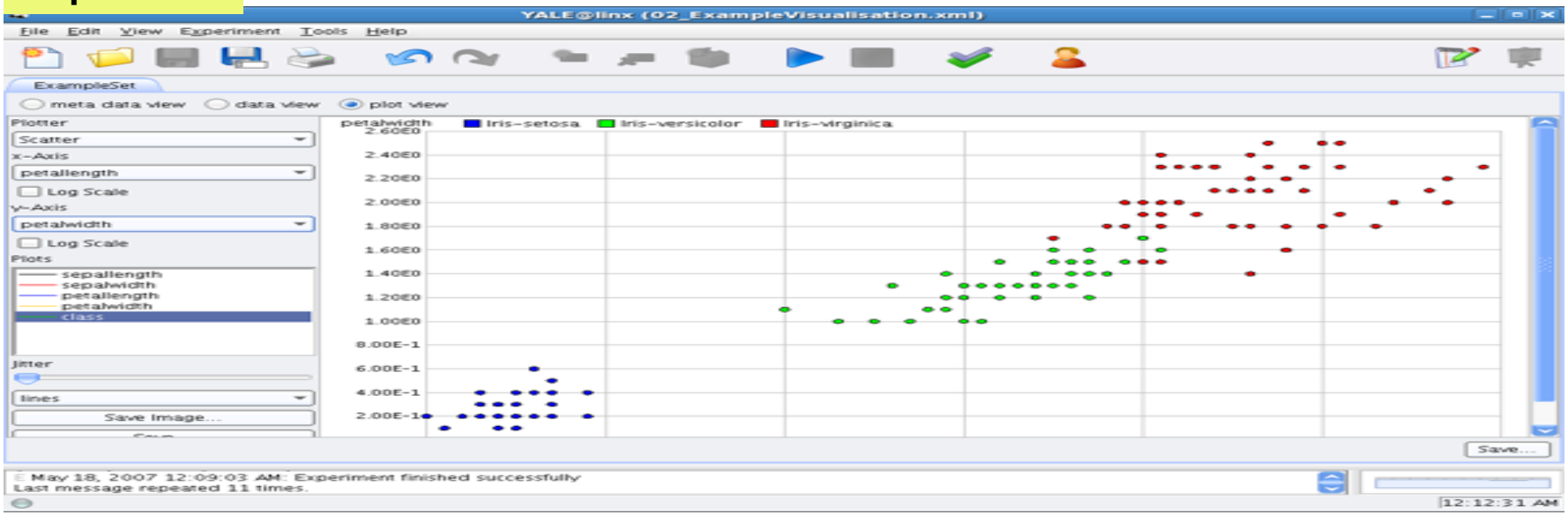
training data

new tuple (query)

Classification

Practical Work

RapidMiner



Load churn

1. Compare the performance of KNN with other algorithms using Xvalidation

Weakness and Strength of KNN

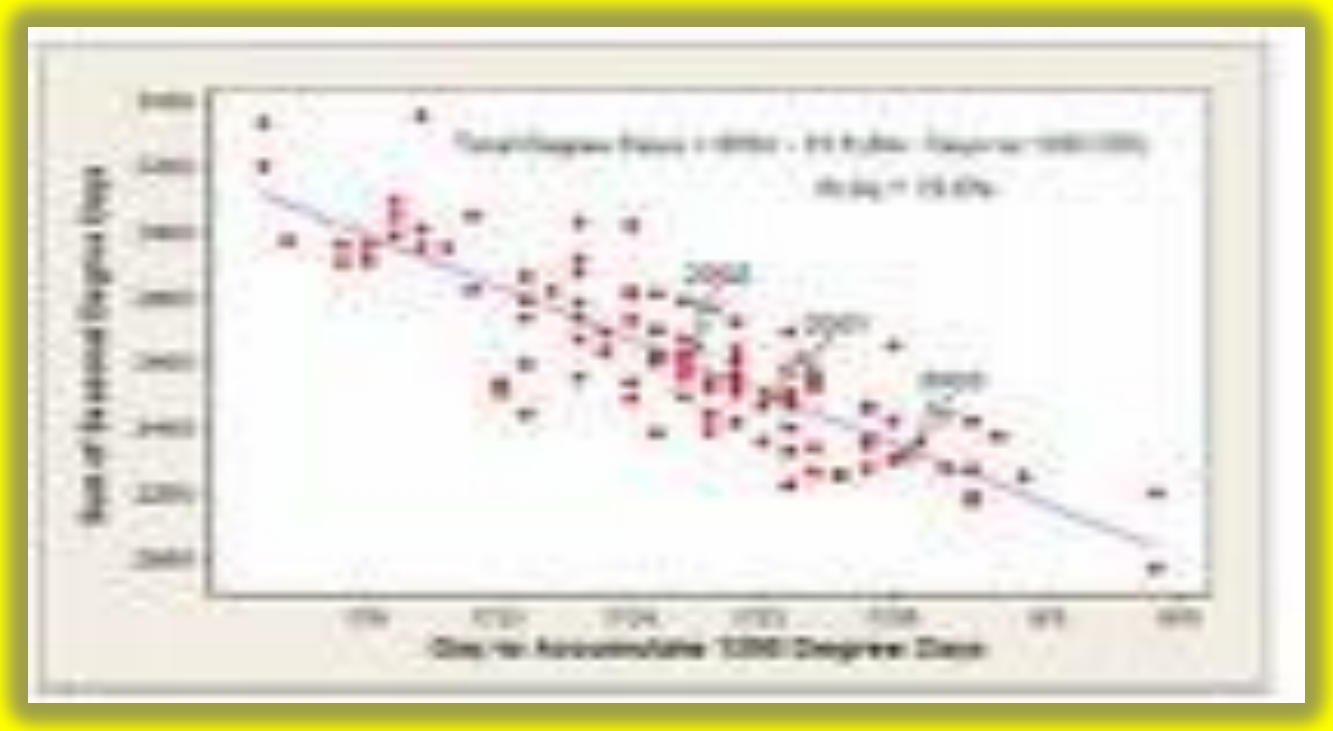
Weakness

- Need to determine value of parameter K
- Computation cost is quite high because we need to compute distance of each query instance to all training samples

Strength

- Robust to noisy training data
- Simple to implement

Teknomo, Kardi. K-Nearest Neighbors Tutorial. http:\\people.revoledu.com\kardi\ tutorial\KNN\

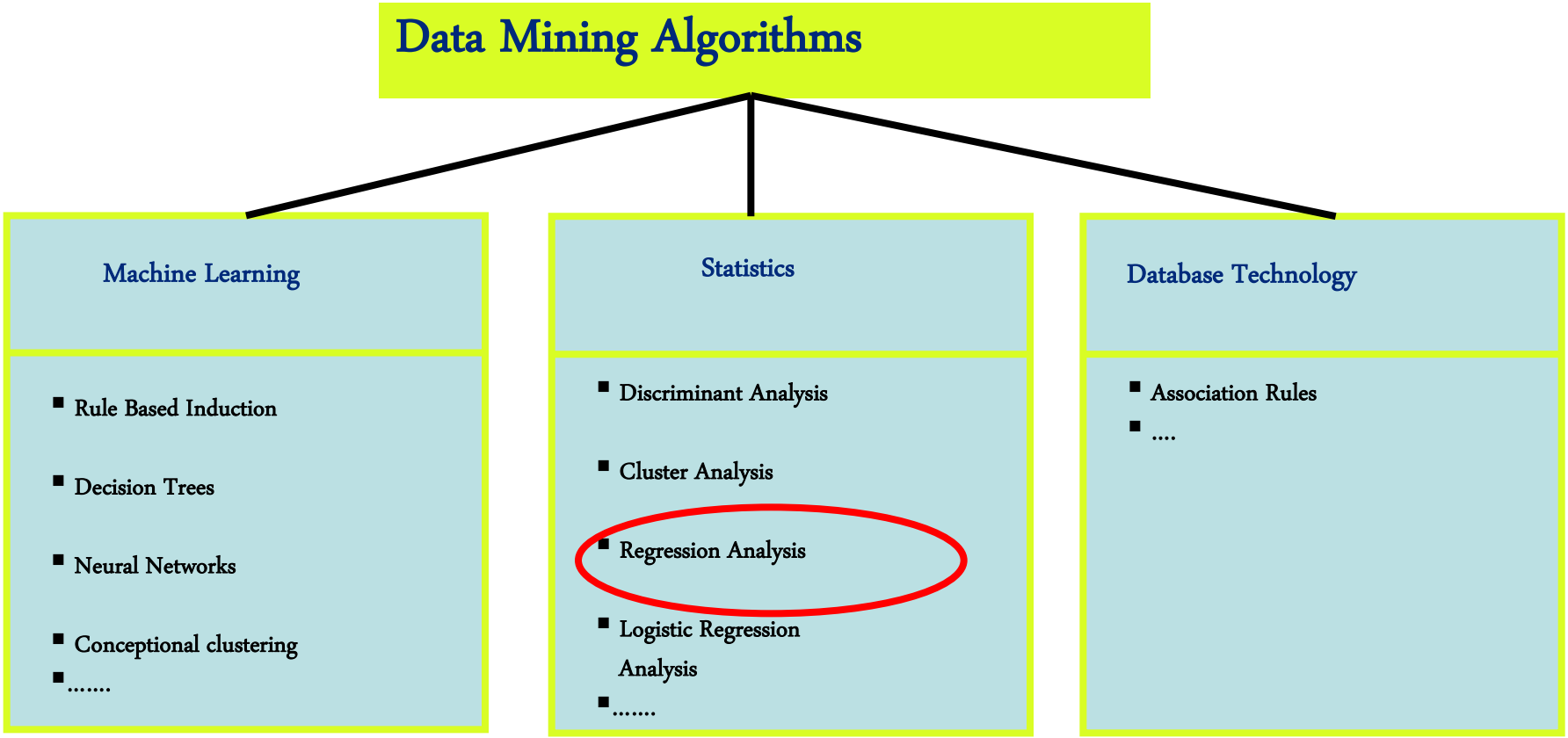


Regression Analysis

Additional Litrature

http://www2.chass.ncsu.edu/garson/PA765/regress.htm
http://www.statsoft.com/textbook/stmulreg.html
http://www.statsoft.com/textbook/glosfra.html?glosm.html&1

Regression Analysis



Regression Analysis

Introduction

- **Tools for prediction and causal analysis based on Supervised Learning**

- Regression function, **y = f (X),** maps a set of attributes **X** known also as **exogenous, independent or explanatory variables**

  into an output **y** known also as **endogenous dependent , response, or target variable** by learning from the tuples observed for X and y

Regression Analysis

Introduction

- The aim is to use the input data to perform the **best estimation** for **y** with **minimum error**
- **Time Series and Cross-Section aspects regarding prediction**
- Endogenous variable must be **continuous-valued** but the exogenous variables can be **nominal or continuous**
- Estimation of parameters and their Significant tests are based on **statistical methods**

Regression and Artificial Neural Networks

Estimation Method

- Error Function: Sum of **squared errors**

$$\sum_i [\, y_i - f(X_i) \,]^2$$

- Estimation on **the training data**, assessment on the **test data** or **validation data**

- In **Stepwise** regression **backward** and **forward** possible (like pruning in DT)

Regression Analysis

Examples of applications

- Prediction of the family consumption using other indicators like, income, price, family size, living place
- Prediction of stock market index by applying other economic indicators
- Prediction of the air temperature based on other atmospheric factors
- Trend Prediction

Regression Analysis

Single-Equation Linear Models

$$Y = \underbrace{ß_0 + ß_1 X}$$ (1)

ßs are the coefficients
ß0 : Constant or intercept
X=0 → Y= ß0

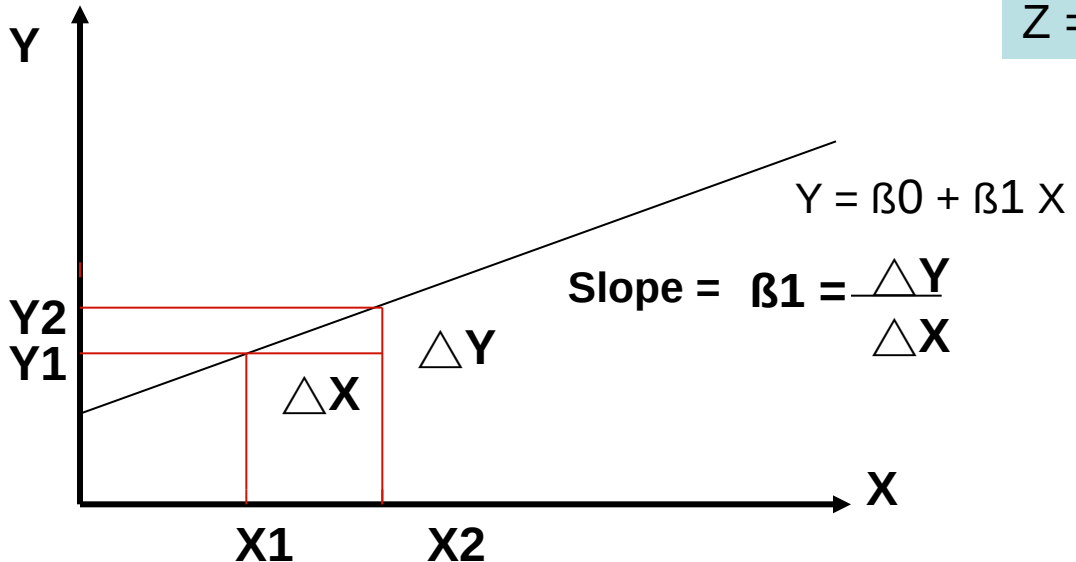
ß1 : slope coefficient

X increases by one unit → Y increases by ß1

Making nonlinear equations linear

$Y = ß_0 + ß_1 X^2$

$Z = X^2$ $\quad Y = ß_0 + ß_1 Z$



Regression Analysis

The stochastic Error Term

$$Y = ß0 + ß1\ X + Є \longrightarrow \text{Stochastic error term} \qquad (2)$$

$ß0 + ß1\ X$: deterministic component

Stochastic error term must be preset, because

- **All relevant explanatory variables are not considered**
- **Measurement error**
- **Misspecification of functional form**
- **Random shocks**
- **…..**

$$E(\epsilon | X) = 0 \qquad (3)$$

$$E(Y | X) = ß0 + ß1\ X \qquad (4)$$

Regression Analysis

Consideration of the observations

$Y_i = ß0 + ß1\ X_i + \epsilon_i$ (i = 1, 2, 3,……, n) n : Number of observations

Yi : the ith observation of the dependent variable

Xi : the ith observation of the independent variable

ϵ i : the ith *observation* of the stochastic error term

$$\left.\begin{aligned} Y1 &= ß0 + ß1\ X1 + \epsilon_1 \\ Y2 &= ß0 + ß1\ X2 + \epsilon_2 \\ &\cdots\cdots\cdots\cdots\cdots\cdots \\ Yn &= ß0 + ß1\ Xn + \epsilon_n \end{aligned}\right\}$$

The coefficients ß0 and ß1 do not change from observation to observation

Regression Analysis

General Case: Multivariate Regression Equation

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \ldots + \beta_m X_{mi} + \epsilon_i \qquad (5)$$

One unit increase in the independent variable Xk

↓

Change in the dependent variable Y is equal to ßk, holding constant the other independent variables

Regression and Artificial Neural Networks

Regression Analysis

Multivariate Linear Regression

$$Y_i = ß0 + ß1\ X1i + ß2\ X2i + \ldots. + ßm\ Xmi$$

Observations

x11 x12 …. x1j …. x1m Y1

……

xi1 xi2 …. xij …… xim Yi

……

xn1 xn2 …. xnj …. xnm Yn

$$X = \begin{pmatrix} 1 & x11 & x12 \ldots\ldots\ldots & x1m \\ 1 & x21 & x22 \ldots\ldots & x2m \\ \ldots & \ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots \\ 1 & xn1 & xn2 \ldots\ldots\ldots & xnm \end{pmatrix}$$

$$Y = \begin{pmatrix} Y_1 \\ Y2 \\ . \\ . \\ Yn \end{pmatrix} \quad ß = \begin{pmatrix} ß0 \\ ß1 \\ . \\ . \\ ßm \end{pmatrix}$$

Matrix notation

$$Y = X\ ß$$

Regression Analysis

Ordinary Least Square

In the regression equation $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$

The parameters ß0 and ß1 are unknown
they can be estimated by using the observations of Y and X

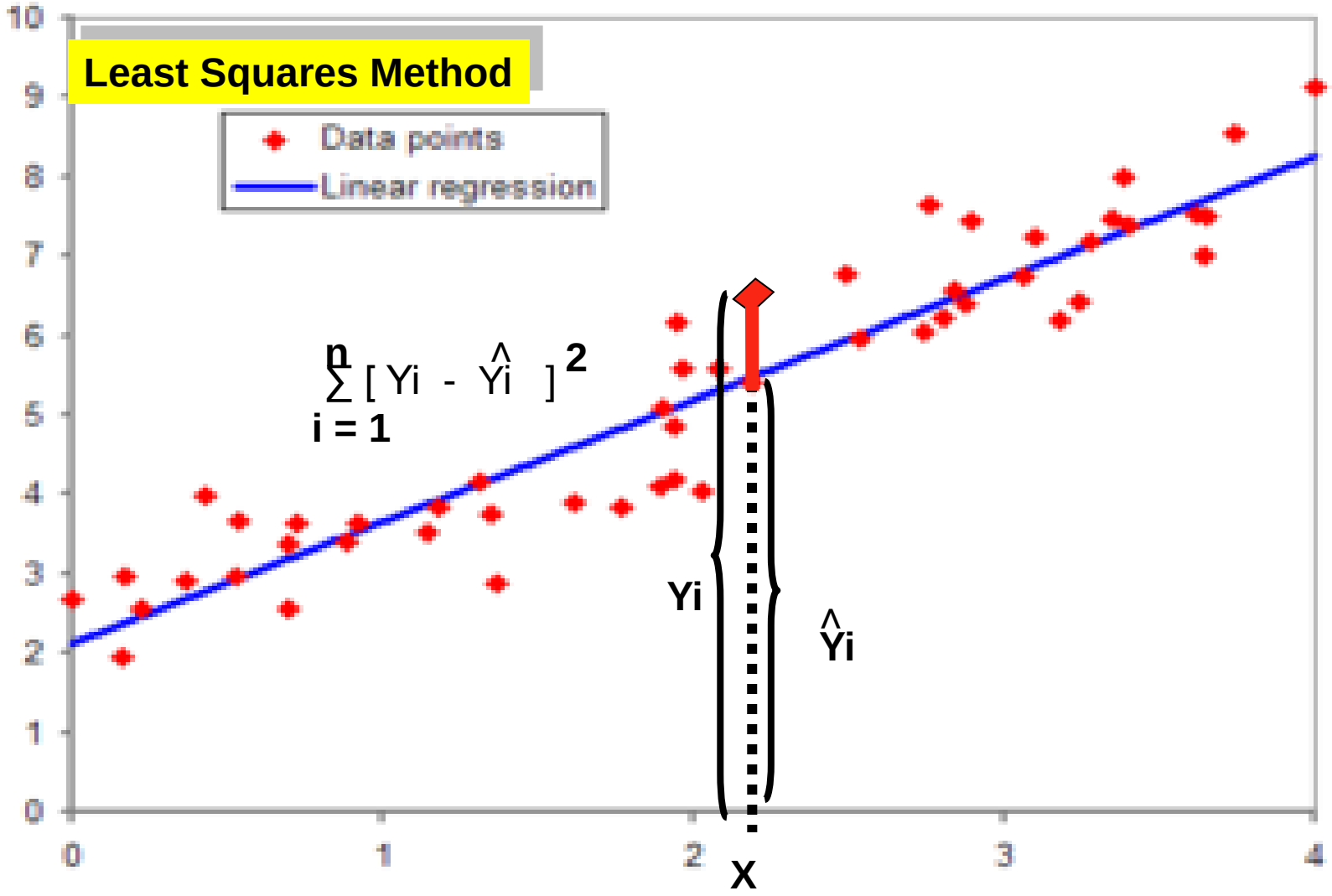
$\hat{\beta}_0$ and $\hat{\beta}_1$: Estimates of ß0 and ß1

$\hat{Y}_i$: Estimate of Yi

and $\hat{\epsilon}_i = Y_i - \hat{Y}_i$: residual

OLS: Determine $\hat{\beta}_0$ and $\hat{\beta}_1$ so that is minimized $\sum_{i=1}^{n} \hat{\epsilon}_i^2$

OLS is relatively easy and OLS – estimates have useful characteristics



Regression Analysis

OLS-estimates for single-equation linear model

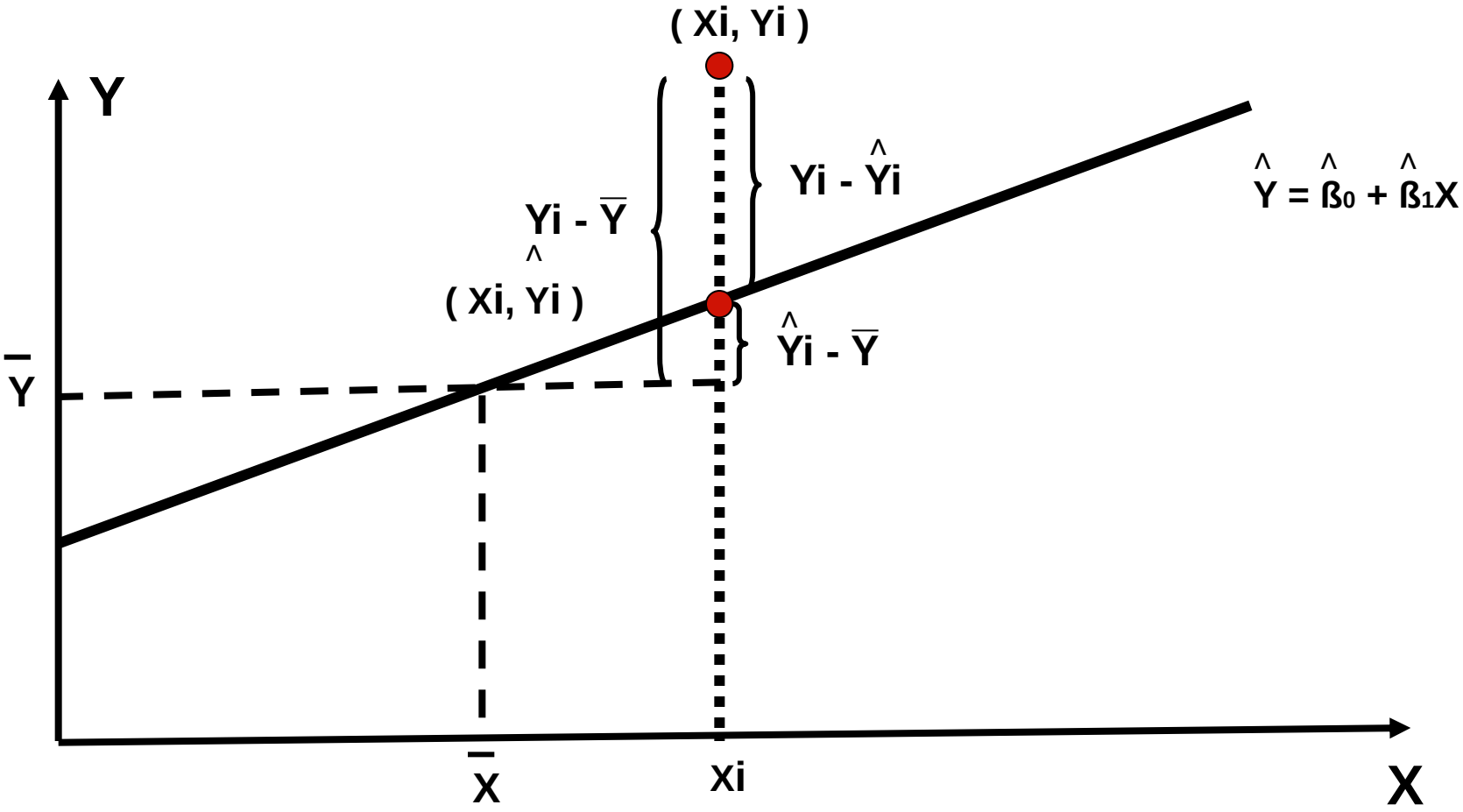
$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n} (X_i - \bar{X})^2}, \quad \hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X} \qquad (6)$$

with

$$\left.\begin{aligned} \bar{X} &= 1/n \sum_{i=1}^{n} X_i \\ \bar{Y} &= 1/n \sum_{i=1}^{n} Y_i \end{aligned}\right\} \quad (6)$$

Regression Analysis

Overall fit of the estimated regression | **Decomposition of Variance**



Regression Analysis

Overall fit of the estimated regression | **Decomposition of Variance**

$$\sum_i (Y_i - \bar{Y})^2 = \sum_i (\hat{Y}_i - \bar{Y})^2 + \sum_i (Y_i - \hat{Y}_i)^2 \quad (7)$$

Total Sum of Squares (TSS)

Explained Sum of Squares (ESS)

Residual Sum of Squares (RSS)

TSS = ESS + RSS

Smaller RSS to TSS → **better the estimated regression fits the data**

Regression Analysis

Overall fit of the estimated regression

Coefficient of Determination

TSS = ESS + RSS

$$R^2 = \frac{ESS}{TSS} = 1 - \frac{RSS}{TSS} = 1 - \frac{\sum (Yi - \hat{Y})^2}{\sum (Yi - \bar{Y})^2} \qquad (8)$$

From (7) and (8) $0 \leq R^2 \leq 1$ (9)

Value of R^2 close to one ⟶ excellent overall fit

Value of R^2 close to zero ⟶ very poor fit

Regression Analysis

Overall fit of the estimated regression

Adjusted Coefficient of Determination

R^2 is biased to the number of independent variables

More independent variables ⟶ higher R^2

Solution: Adjusted R^2

$$\bar{R}^2 = 1 - \frac{\sum (Yi - \hat{Y})^2 / (n - k - 1)}{\sum (Yi - \bar{Y})^2 / (n - 1)}$$

- Normally, $\bar{R}^2$ is used to compare the goodness of fit of regression equations with different numbers of independent variables
- $\bar{R}^2$ is not a percent but an index

k : Number of independent variables

Regression Analysis

Overall fit of the estimated regression | **Simple Correlation Coefficient**

$$r_{X,Y} = \frac{\sum [(X_i - \bar{X}) (Y_i - \bar{Y})]}{\sqrt{\sum (X_i - \bar{X})^2 \sum (\bar{Y}_i - Y)^2}} \qquad (10)$$

$$-1 \leq r \leq +1$$

X and Y are perfectly positively correlated, then r = +1

X and Y are perfectly negatively correlated, then r = -1

X and Y are totally uncorrelated, then r = 0

Regression Analysis

Simple linear regression model

Model assumptions

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i \qquad \text{for } i = 1, 2, \ldots n$$

Assumption 1:

$$E(\varepsilon_i \mid x) = 0 \qquad \text{for } i = 1, 2, \ldots n$$

Assumption 2:

$$V(\varepsilon_i \mid x) = \sigma^2 \qquad \text{for } i = 1, 2, \ldots n$$

→ **Variance of ϵ_i is constant for i=1,2,… n**

Homoscedasticity

(Heteroscedasticity)

Assumption 3:

$$E(\varepsilon_i \varepsilon_j \mid x) = 0 \qquad \text{for } i \neq j \qquad i, j = 1, 2, \ldots n$$

↓

ε_i and ε_j are not correlated

Regression Analysis

Simple linear regression model

Model assumptions

Assumption 4:

Sample-Var(x) = $S^2(x) = 1/n \sum_{i=1}^{n} (x_i - \bar{x})^2 > 0$

und

$\lim_{n \to \infty} \overline{x^2} < \infty$ $\quad \overline{x^2} = 1/n \sum_{i=1}^{n} x_i^2$

und

$\lim_{n \to \infty} S^2(X) > 0$

Aussumption 5:
The explanatory variables must be linearly independent

↓

no collinearity or multicollinearity

Under these 5 assumptions the OLS-Estimators are **Best Linear Unbiased Esimator (BLUE).**
It means that they are the efficient ones amongst the set of unbiased linear estimators

Assumption 6: (not always necessary)
For given x the error term ε is normally distributed

Regression Analysis

General Case: Multivariate Regression Equation

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots + \beta_k x_{ki} + \varepsilon_i$$

Regression and Artificial Neural Networks

Regression Analysis

Multivariate Linear Regression

$$y = w0 + w1\ x1 + w2\ x2 + \ldots\ldots + wm\ xm$$

Matrix notation

$$\mathbf{y} = \mathbf{X}\,\mathbf{w}$$

Observations

x11	x12	….	x1j	….	x1m	y1
……					
xi1	xi2	….	xij	……	xim	yi
……					
xn1	xn2	….	xnj	….	xnm	yn

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y2 \\ . \\ . \\ yn \end{pmatrix} \quad \mathbf{w} = \begin{pmatrix} w0 \\ w1 \\ . \\ . \\ wm \end{pmatrix}$$

$$\hat{W} = (X' X)^{-1} X' y$$

$$\mathbf{X} = \begin{pmatrix} 1 & x11 & x12 & \ldots\ldots\ldots & x1m \\ 1 & x21 & x22 & \ldots\ldots & x2m \\ \ldots & \ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots \\ 1 & xn1 & xn2 & \ldots\ldots\ldots\ldots & xnm \end{pmatrix}$$

Weakness and Strength of Regression Analysis

- **Strength**

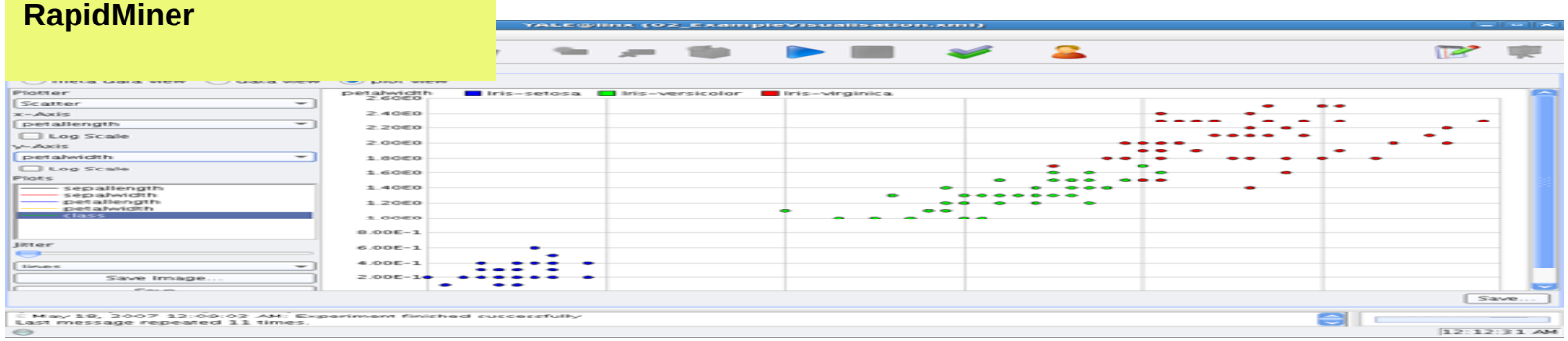
- Simple tool for prediction and causal analysis with high availability
  (Available in almost every statistical package)
- Estimation of parameters and their significant tests are based on statistical methods
- Forward and backward feature selection is possible

- **Weakness**

- Not appropriate for classification tasks
- Not appropriate for the case with a lot of nominal exogenous variable

Practical Work: Forecasting of Interest Rate

RapidMiner



Y= GDP
CO= Total Personal Consumption
I= Total Gross Private Investment
G= Government Purchases of Goods and Services
R= Interest rate
YD= Disposal Income

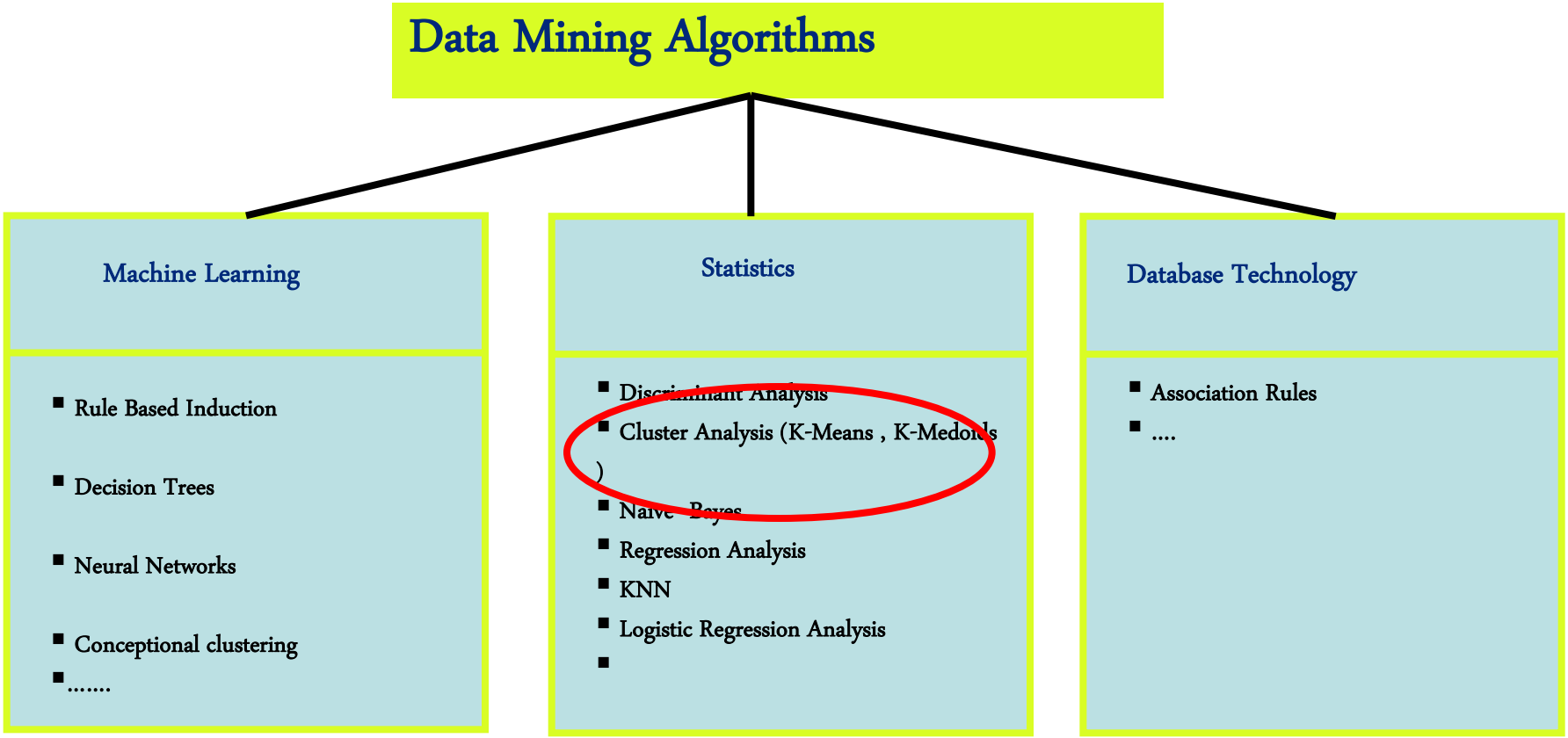
Load InterestRate dataset

Try Different algorithms: Linear Regression ANN, KNN, W-MP5, W-REPT-REE, REG,…

Use Xvalidation

Repeat the same with Location.aml

Clustering



K-Means Algorithm

Characteristics

- Made popular by J.B. MacQueen in 1967
- Very easy to implement
- Based on unsupervised learning
- Attributes must be numerical
- The number of clusters (K) must be fixed in advance
- Partitional Clustering, not hierarchical
- The main idea is based on the so-called cluster centroids (centre point)

K-Means Algorithm

Basic Algorithms

- Specify K
- Select K points in the dataset as initial centroids
- **Repeat**
- Form K clusters by assigning each point of the dataset to the closest centroid
- Recompute the centroid of each cluster
- **Until** The centroids don't change

Source: Tan, Steinbach, Kumar

K-Means Algorithm

Example: Clustering with two numerical attributes

K=5

Next step:

Select 5 points in the dataset as initial centroids

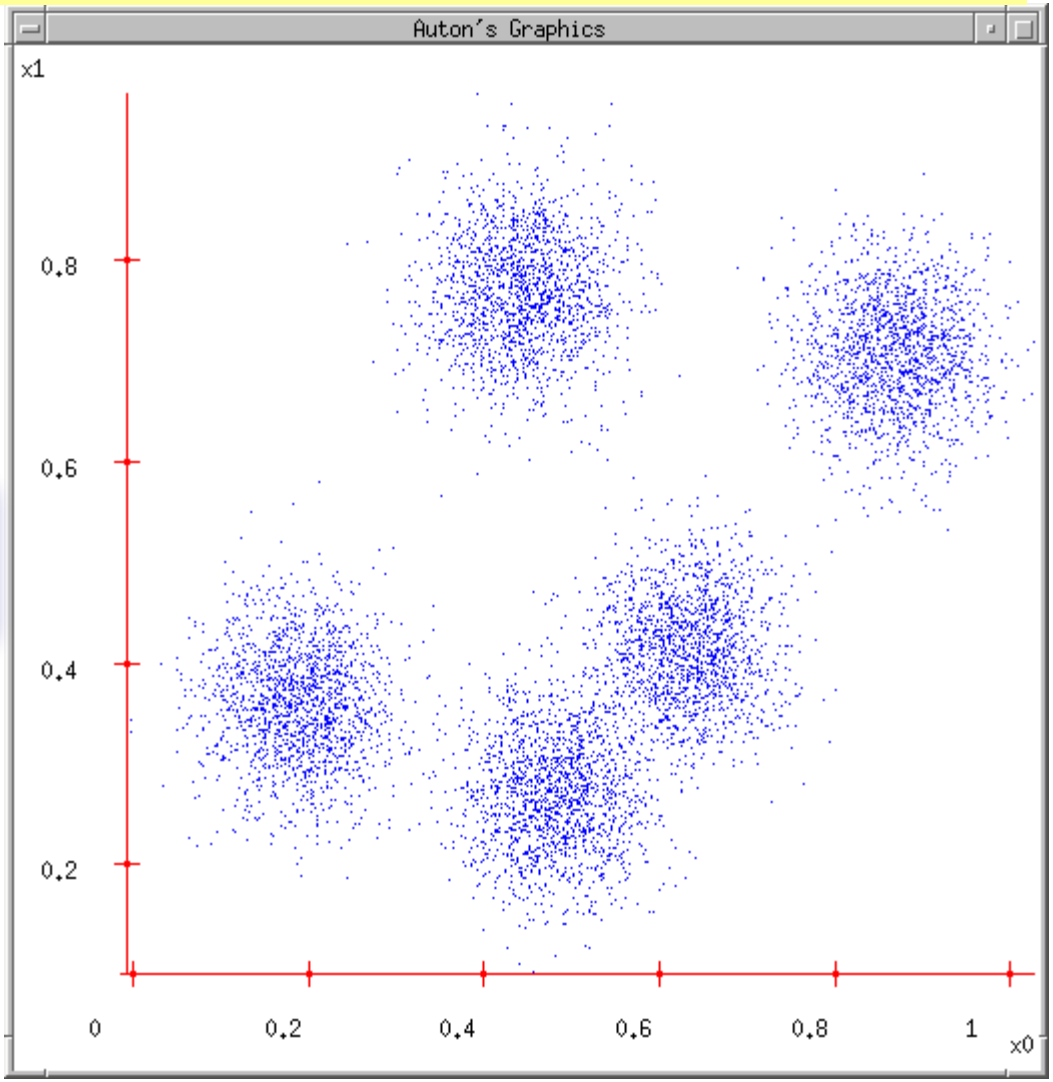
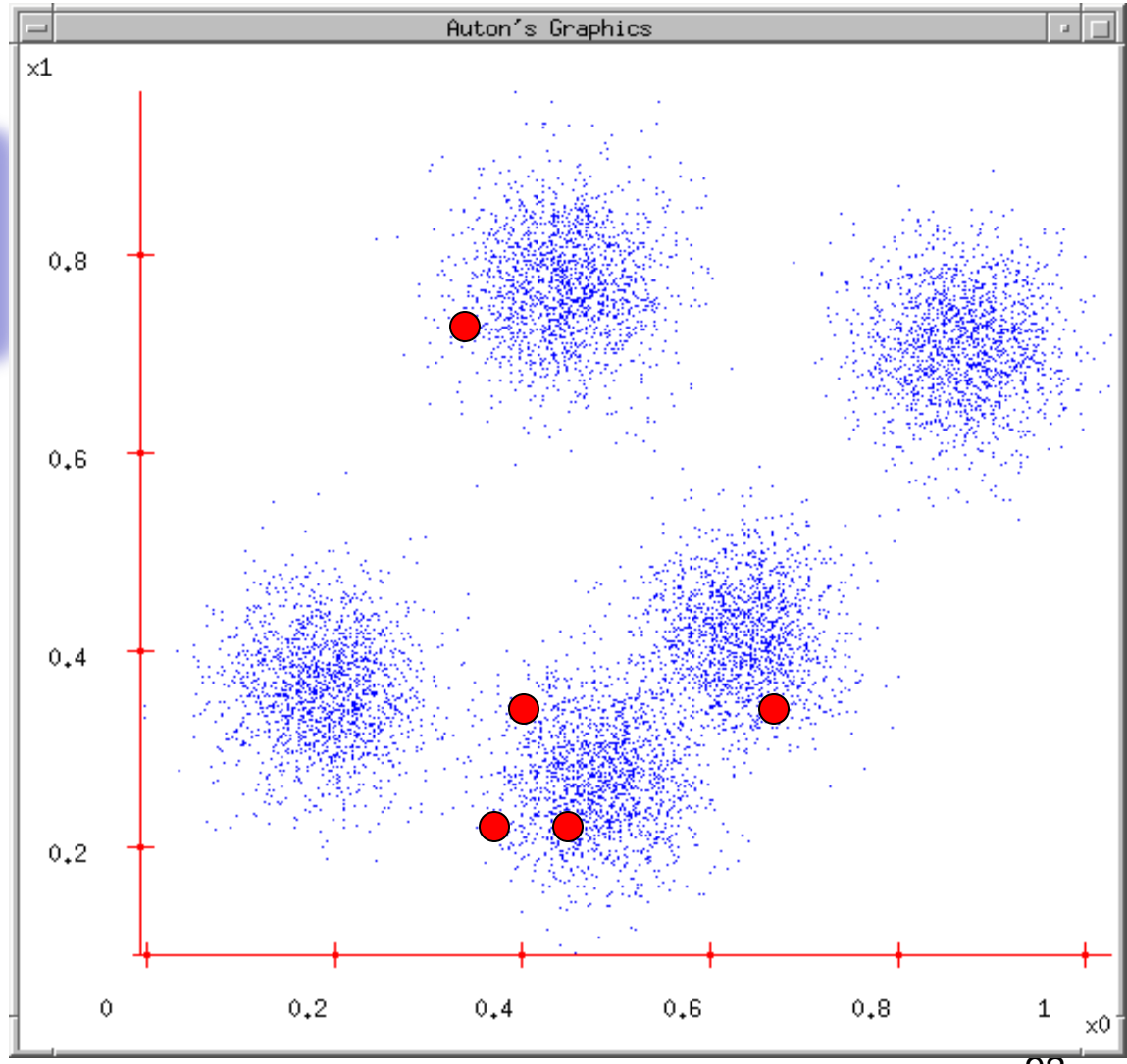


Figure based on: http://www.autonlab.org/tutorials/kmeans.html

K-Means Algorithm

Next step:

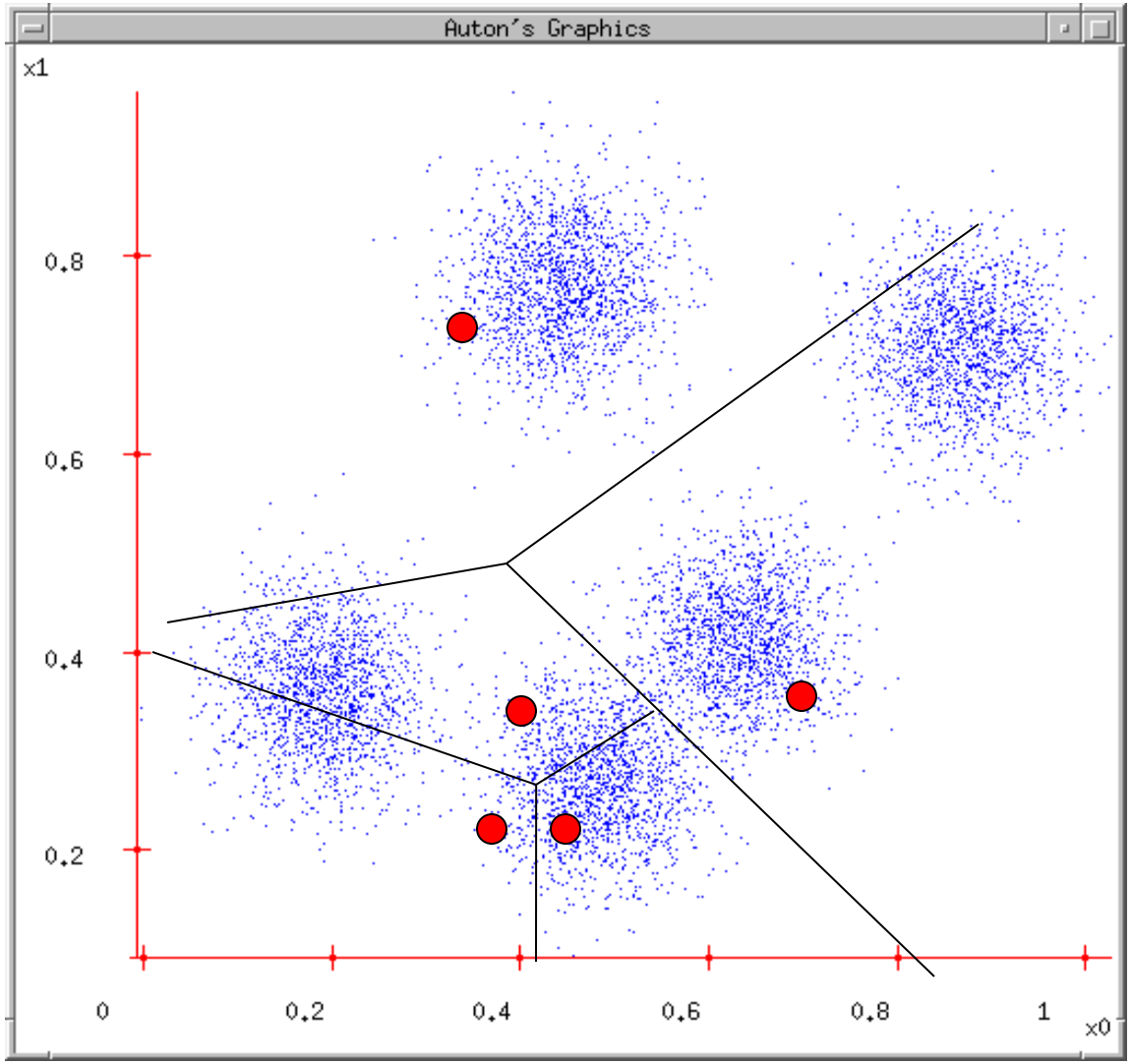
Form 5 clusters by assigning each point of the dataset to the closest centroid



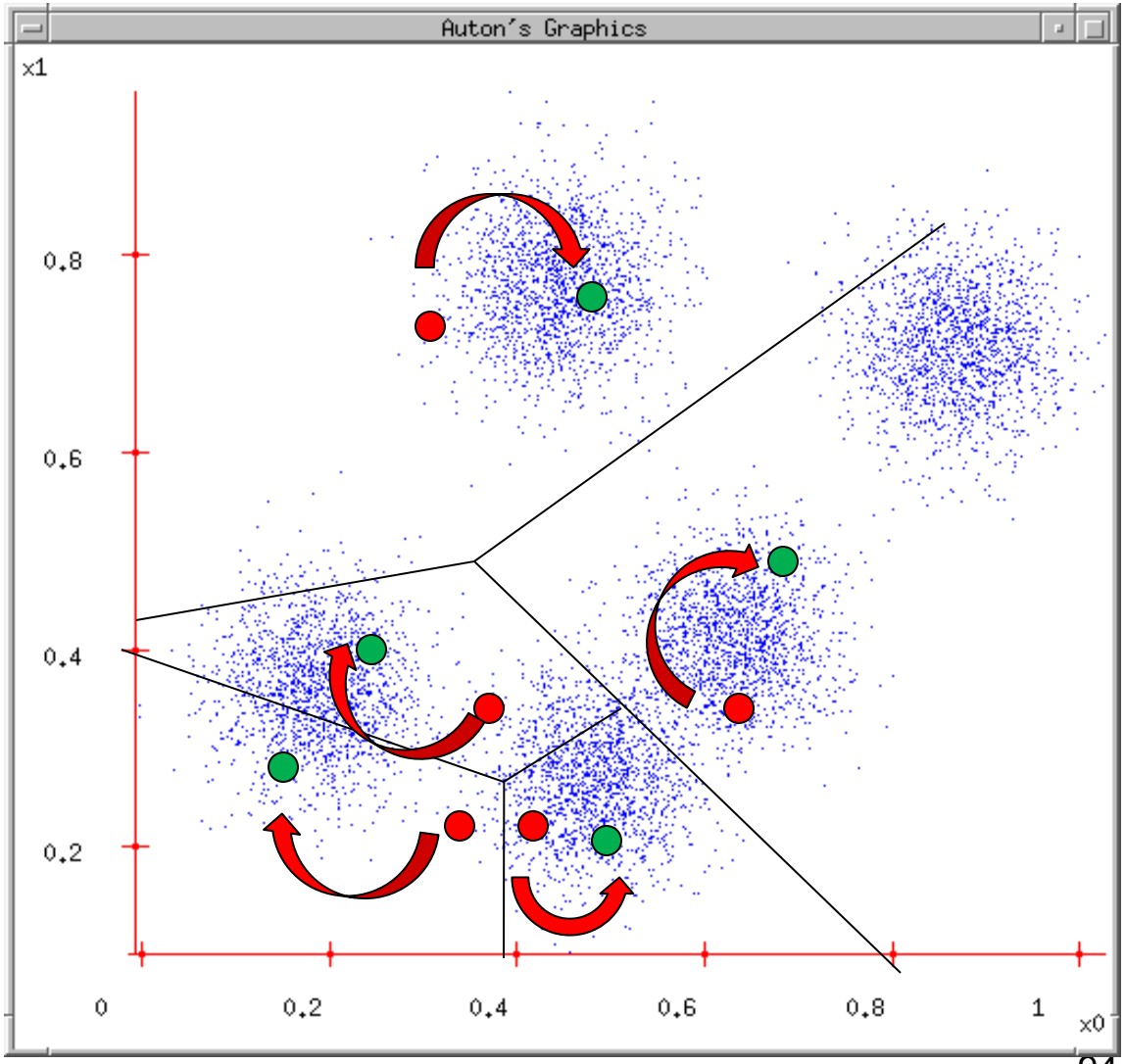
K-Means Algorithm

Next step:

Recompute the centroid of each cluster



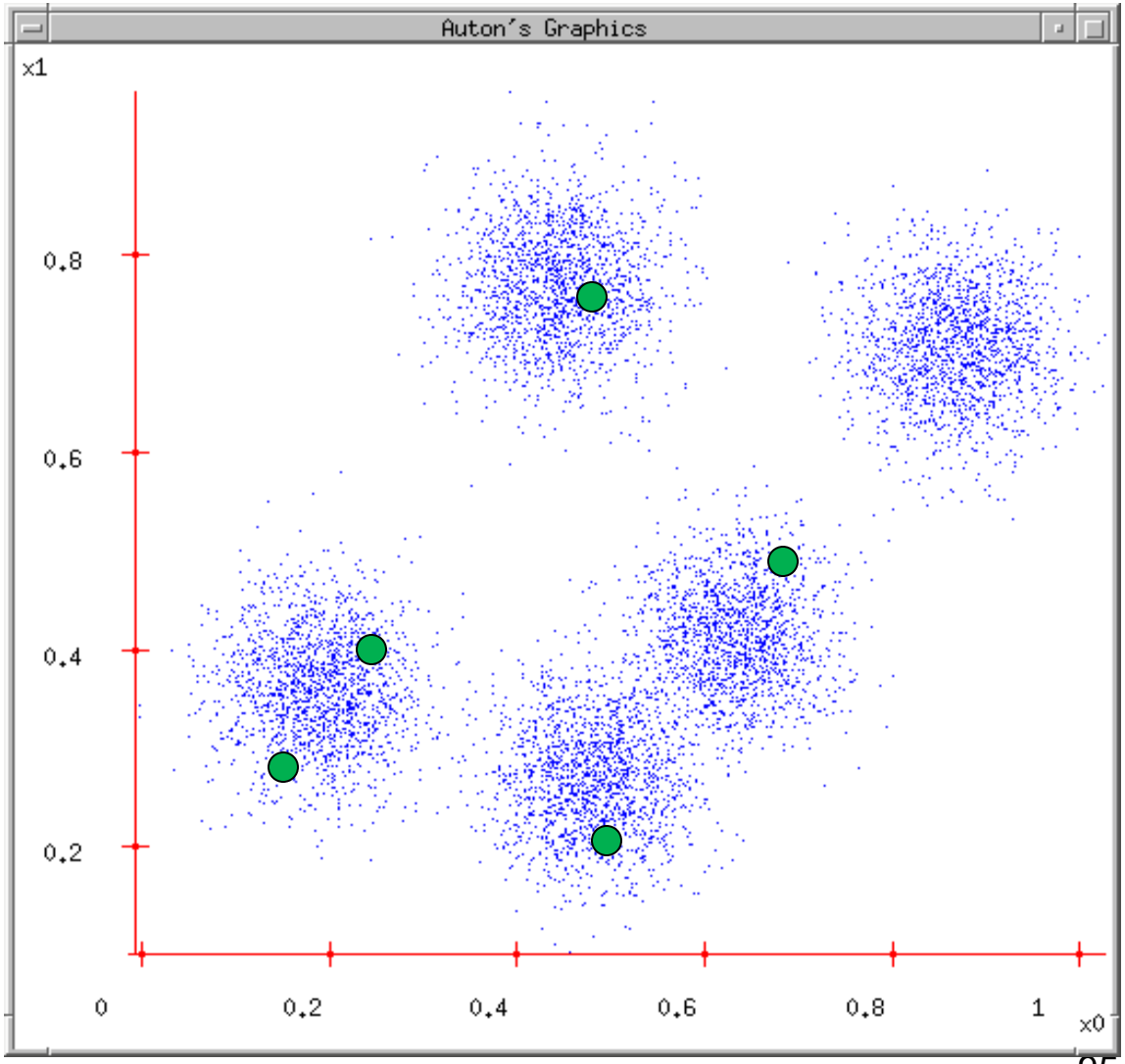
K-Means Algorithm



K-Means Algorithm

Repeat

Until The centroids don't change



K-Means Algorithm

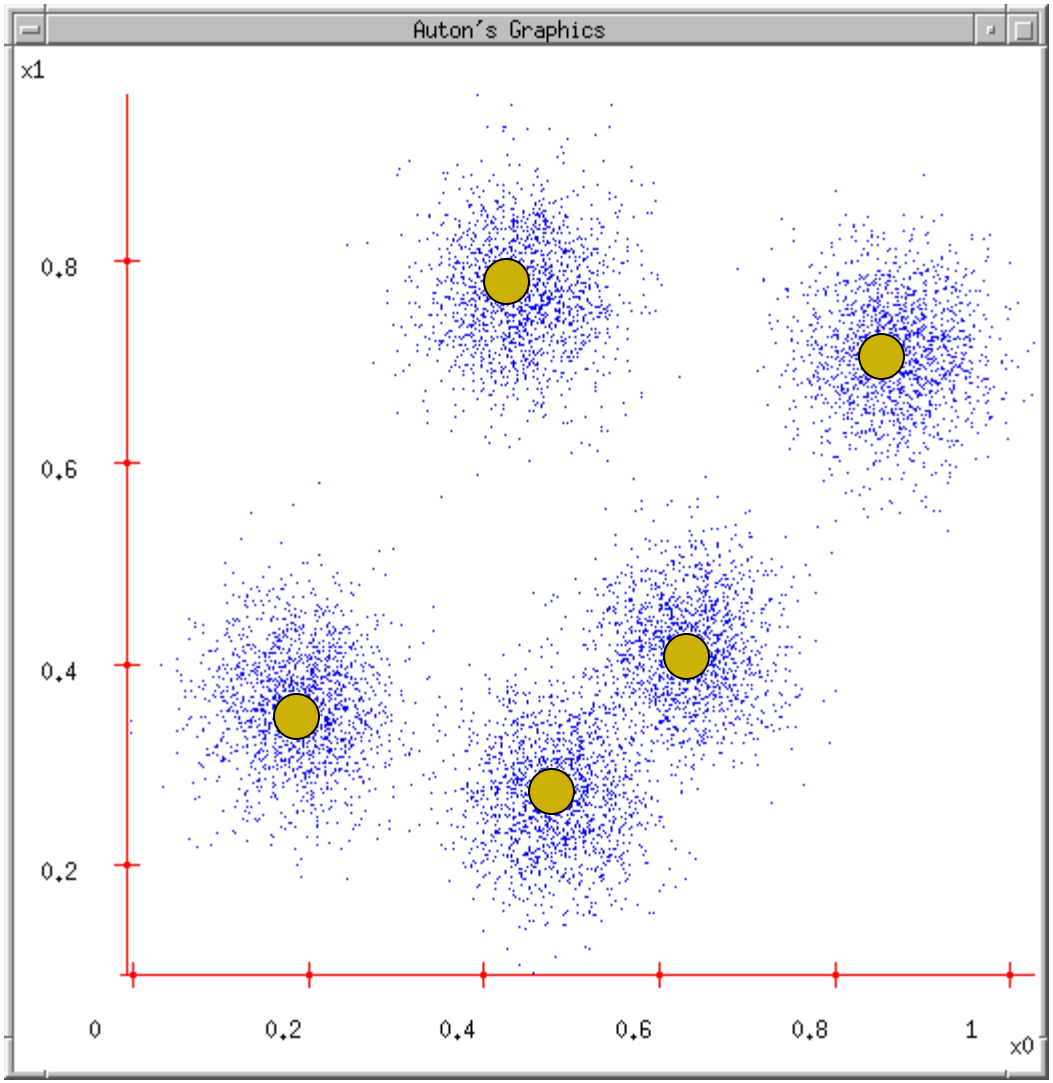
.....

.....

.....

iterations

Final Situation



K-Means Algorithm

Remarks:

- Euclidean distance are often used to measure the closeness of the cluster members and the Centroids
- Attributes average is usually used as the centroid
- Usually the procedure converges after a few Iterations
- There are some evaluations methods to measure the goodness of the clusters but in many cases the background knowledge of the user is very essential as an evaluation criterion

K- Means Algorithm, Weakness

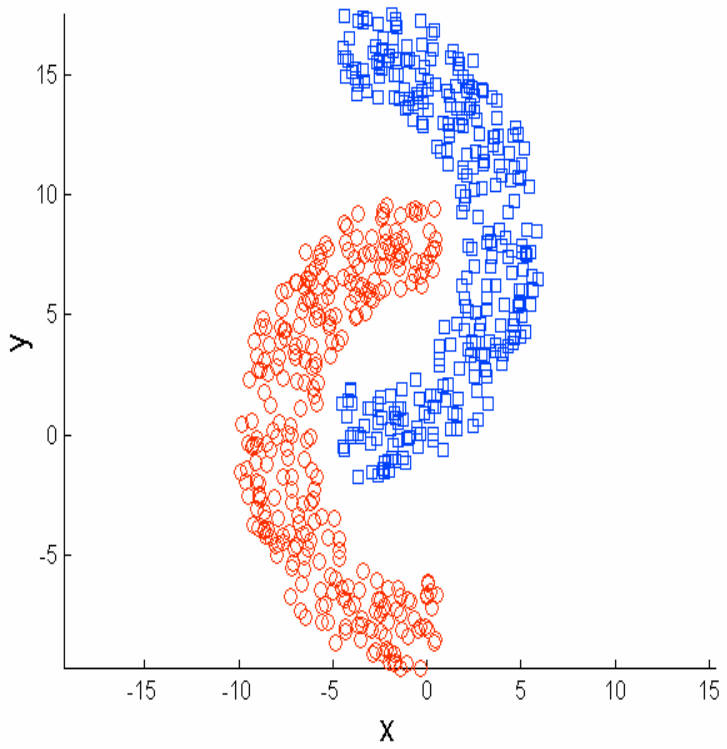
Weakness

- K-means has problem when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes (see the next slide)
- K-means has problem when the data contains outliers
  using median instead mean can help
- K-means is strongly sensitive to the initial
  randomly selected centroids, this effect can be , however,
  reduced by multiple runs using different starting values

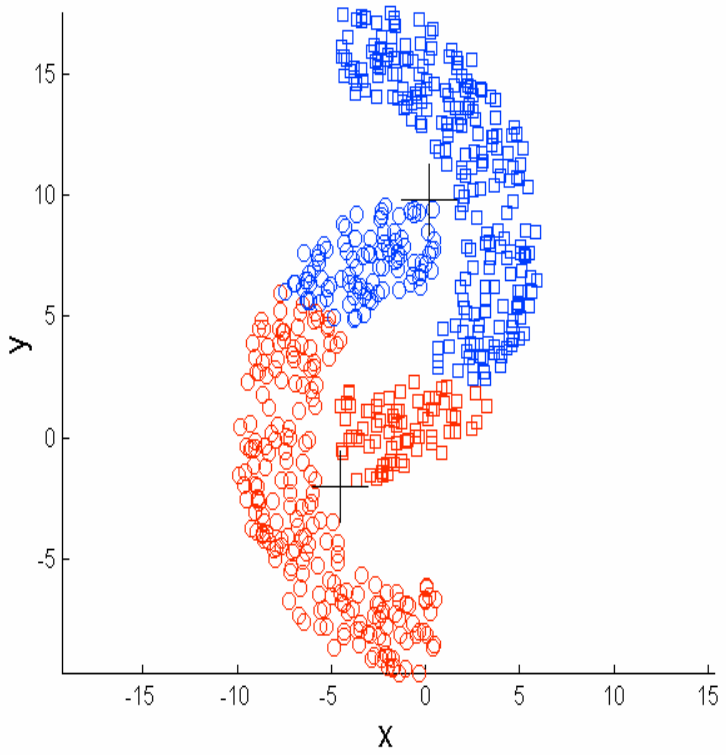
Strengths

- Work good for globular shapes
- Usually very fast
- Easy to understand and to implement

K-Means Algorithm and non globular shapes



Original Points



K-means (2 Clusters)

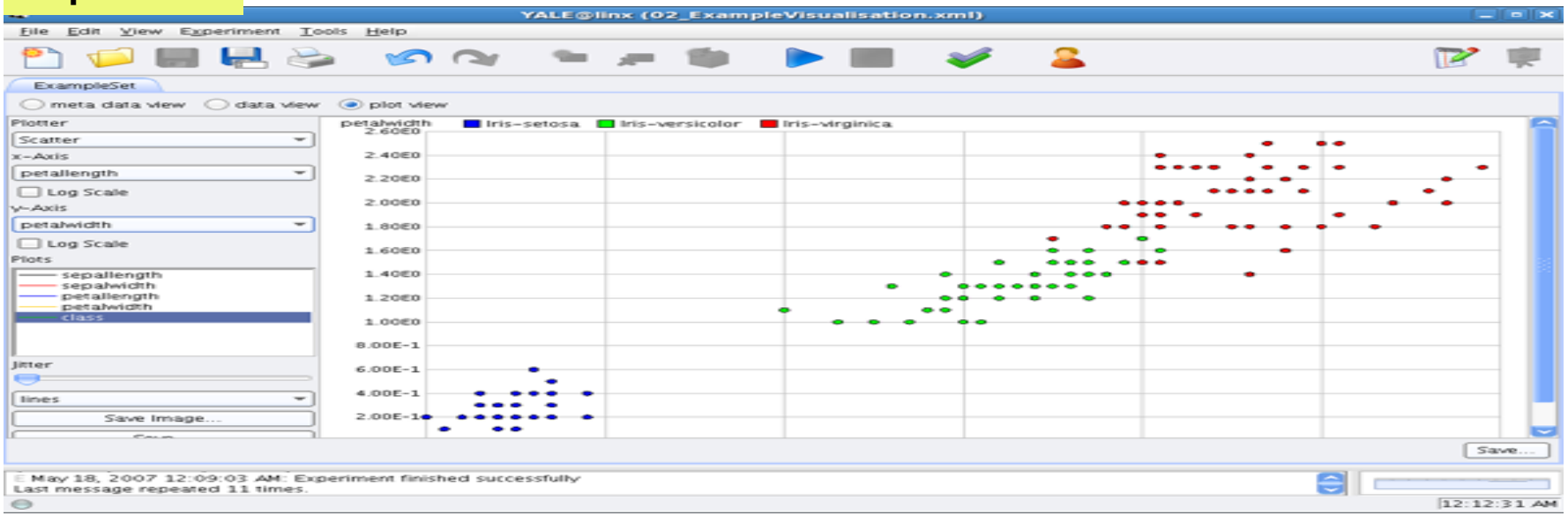
Source: Tan, Steinbach, Kumar

K- Medoids Algorithm

- K-medoids is very similar to K-Means
- It is appropriate for the cases in which a mean or median can not be calculated
- In contrast to means, medoids are members of the datasets
- Construction of the clusters is like K-means
- Attributes can be nominal , numerical or both

K-Means, K-Medoids

RapidMiner

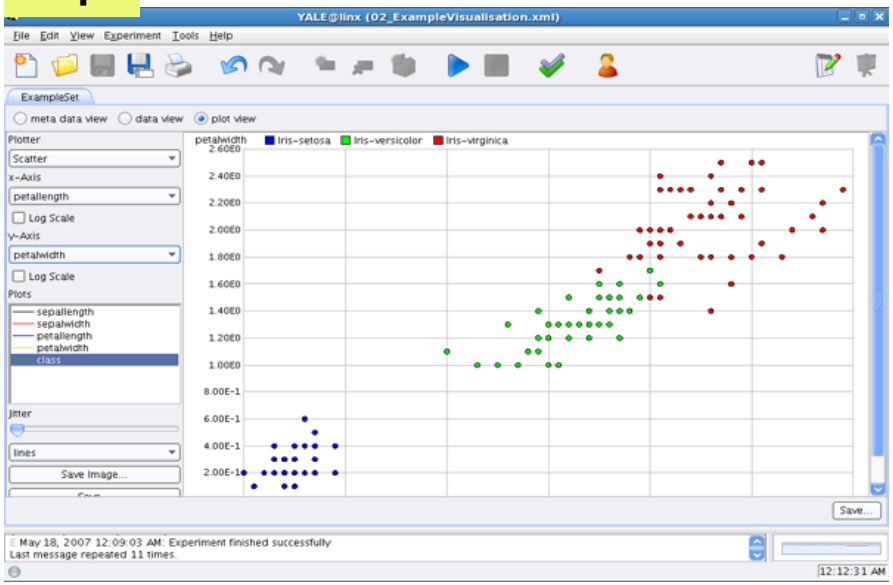


Practical Work
Rainy days, German_Credit_Tr

K-Medoids

Practical Work
Team profit

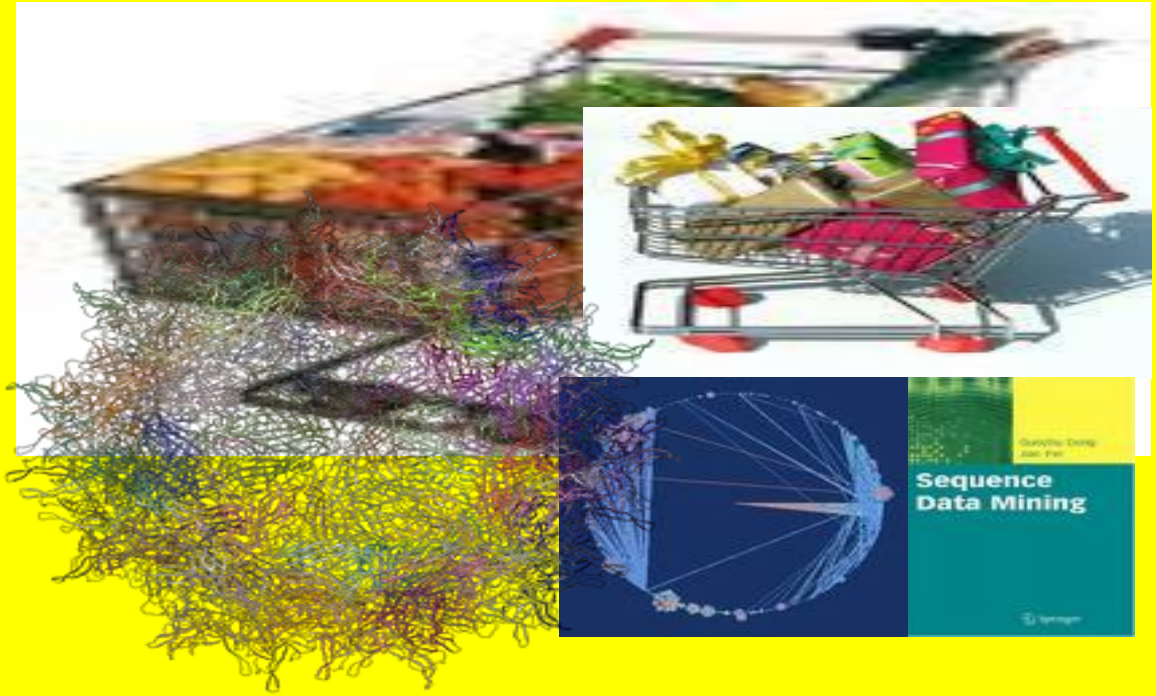
RapidMiner



Load Team Profit From IO→ Generator

Use the Data for Classification and Clustering

Association Mining

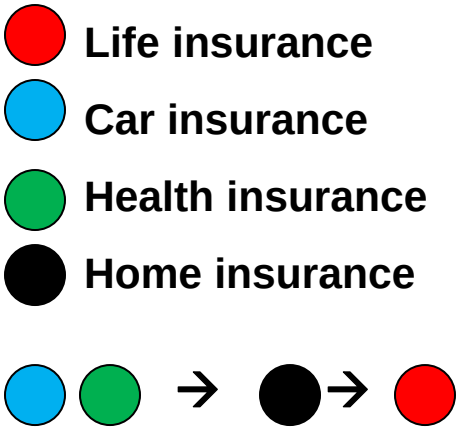


Task Identification: Dependency Analysis (Association Detection)

Remember from Part 2

Examples:

- **More Income → more consumption**
- **Today water pump defect → in one month defect in engine**



{ milk, meat } → {orange Juice }

Data Mining can help to find such relations

Some criterion to determine dependency

Determining attribute importance by criteria like:

- **Information Gain**
- **Gini-Index**
- **Pearson Chi-Square**
- **Correlation coefficient**
- **Akaike information criterion (AIC)**
- **….**

Remember from Part 3

Mining Association Rules

Mining Association Rules

Mining Frequent Pattern

Frequent patterns:

- { Milk, eggs, butter } often observed → **Frequent itemset**
- { buying computer , buying printer (after 2 months), buying scanner (after 6 months) }

often observed over the time → **Frequent Sequential pattern**

Frequent itemsets and frequent sequential patterns play a very import role in Mining Association

Famous application: Market Basket Transaction

Example

TID	Items
1	bread, milk
2	bread, meat, orange juice, eggs
3	milk, meat, orange juice, cola
4	bread, milk, meat, orange juice
5	bread, milk, meat, cola

→

{ Milk } → { meat }

{ Meat } → { Orange juice }

} **Association rules**

The rules show that apparently there is a strong relationship between buying of milk and meat as well meat and orange juice

Mining Association Rules

Association Rules (AR)

Problems in AR-Mining:
- AR-mining from large datasets is pretty time consuming
- mined Associations could be spurious because may happen by chance

Binary representation of market basket data

TID	bread	milk	meat	Orange juice	eggs	cola
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1
Total	**4**	**4**	**4**	**3**	**1**	**2**

→ ignore quantity, Price, expiration date, supplier, ingredient etc.

Notations:

$I = \{ i_1, i_2, \ldots i_m \}$ set of all items

$T = \{t_1, t_2, \ldots t_N \}$ set of all transactions

t_i contains a subset of items of I

$\{ i_1, i_2, .. i_k \}$: k-itemset

Example: { milk, meat, eggs } : 3-ietemset

X: Itemset

ρ(X) = number of transactions contin X

Example: in the table

ρ {bread, milk } = 3 ρ {eggs, cola } = 0

Mining Association Rules

Association Rules (AR)

Support and Confidence of an AR-Rule

Definition:
X → Y (X is associated to Y) is called an AR
X and Y are disjoint itemsets : X ∩ Y = ø

Definition (support and confidence of an AR-Rule)

$$\text{Support , } s(X \rightarrow Y) = \frac{\rho(X \,\&\, Y)}{N}$$

Percentage of the transactions contain both X and Y in the whole transactions

Probability of (X & Y) appear together

$$\text{Confidence, } c(X \rightarrow Y) = \frac{\rho(X \,\&\, Y)}{\rho(x)}$$

Percentage of the transactions containing both X and Y in the transactions contain X

Conditional probability of Y by given X

$$\text{Lift}(X, Y) = \frac{p(X \,\&\, Y)}{p(X) \,.\, p(Y)} \sim \frac{\text{Sup}(X \rightarrow Y)}{\text{Sup}(X) \,.\, \text{Sup}(Y)}$$

Mining Association Rules

Association Rules (AR)

Support and Confidence of an AR-Rule

Example

TID	bread	milk	meat	Orange juice	eggs	cola
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1
Total	**4**	**4**	**4**	**3**	**1**	**2**

Rule:
{ milk, meat } → {orange Juice }

$$X \rightarrow Y$$

X = { milk, meat } Y= {orange Juice}

$$\rho (X\& Y) = \rho \{ \text{milk, meat, orange juice} \} = 2$$

$$\rho (X) = \rho \{ \text{milk, meat} \} = 3$$

$$\text{Support , } s(X \rightarrow Y) = \frac{\rho (X \& Y)}{N} = 2/5 = 40\%$$

$$\text{confidence , } c(X \rightarrow Y) = \frac{\rho (X \& Y)}{\rho (X)} = 2/3 = 67\%$$

Rule:
{meat, orange juice } → {eggs}

S= 20%, c= 33%

Rule:
{bread} → {milk}

S = 60% C = 75%

Rule:
{eggs} → {cola}

S = 0% c= 0%

Mining Association Rules

Association Rules (AR)

AR-Discovery

Definition:

Given: a set of transactions T **Find: Association Rules having :**

Support* ≥ *sup_min* *and* *Confidence* ≥ *conf_min

***sup_min*: given support threshold** ***conf_min* : given confidence threshold**

Methods of AR-Mining

Brute-force approach: calculate support and confidence for every possible rules
Problem: many many rules
A dataset with 10 items would generate 57000 rules;
a department store could have more than 10.000 items

Mining Association Rules

Association Rules (AR)

AR-Discovery

Rule Pruning before computing support and confidence

Example: Consider the itemset

{ orange juice, meat, milk }

the following AR-Rules involve the same Itemset:

{ orange juice, meat } → { milk }
{ orange juice, milk } → { meat }
{ meat, milk } → {orange juice}
{orange juice → { meat, milk }
{ milk } → { orange juice, meat }
{ meat } →{ orange juice, milk }

TID	bread	milk	meat	orange juice	eggs	cola
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1
Total	**4**	**4**	**4**	**3**	**1**	**2**

→ **Have the same Support 40%**

It means : if we define a *sup_min* of e. g. 50% , after calculating the support of the first rule (40%) we see that we can prune all the others rule before we calculate their support and confidence

Mining Association Rules

Association Rules (AR) **AR-Discovery**

Viewing the AR-Mining as a two steps Process:
(adopted by many AR-Mining algorithms)

1. **Frequent Itemset Generation (FIG)**
2. **Rule Generation**

The aim of FIG is to find all itemsets with support ≥ *sup_min*
Such itemsets called *frequent itemsets* (sometimes large itemsets)

The aim of Rule Generation is to extract from frequent itemsets the rules with Confidence ≥ *conf_min*; such rules are called strong rules

In the past years a lot of attempts put to find efficient methods for generating the frequent itemsets

Mining Association Rules

Association Rules (AR) | AR-Discovery | Frequent itemset generation

Candidate Itemset

Generally for an itemset with n items, potentially $2^n - 1$ candidate itemsets can be generated

Example

Consider itemset { a, b, c, d, e }

n=5 number of candidat itemsets = 31

a b c d e

ab ac ad ae bc bd be cd ce de

abc abd abe acd ace ade bcd bce bde cde

abcd abce abde acde bcde

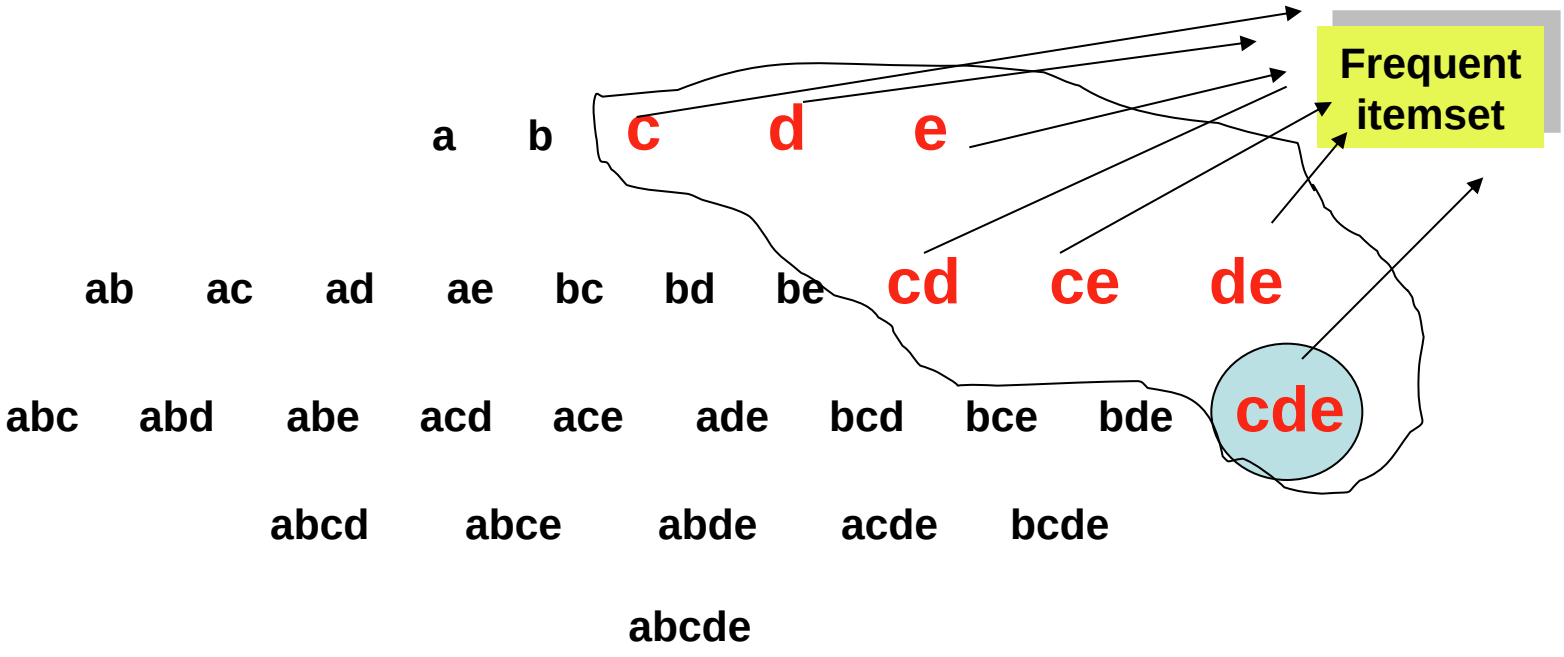
abcde

Mining Association Rules

Association Rules (AR) | AR-Discovery | Reduce candidate itemsets

Apriori – Principal (1)

- All of the subsets of a frequent itemset must be frequent itemsets too

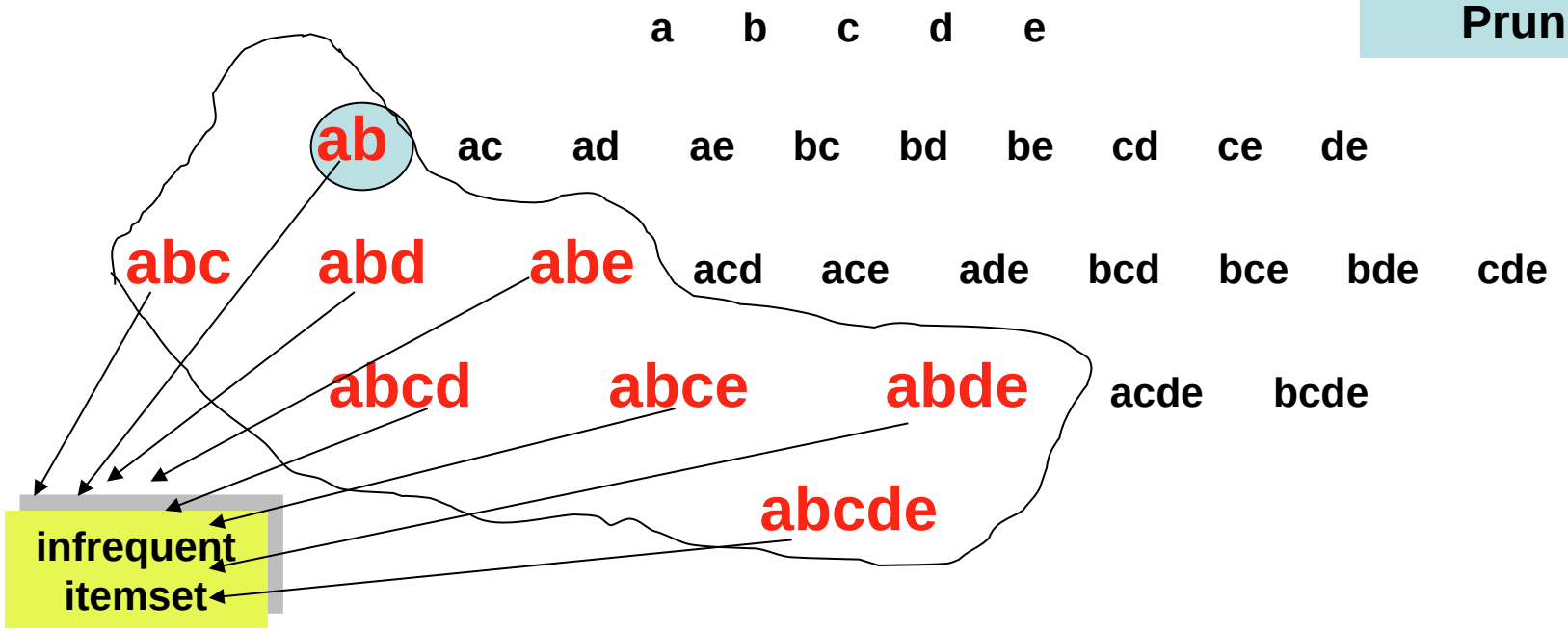


Mining Association Rules

Association Rules (AR) | AR-Discovery | Reduce candidate itemsets

Apriori – Principal (2)
- All of the supersets of an infrequent itemset must be infrequent itemsets too

Support-based Pruning



Mining Association Rules

Association Rules (AR) | **AR-Discovery** | **Apriori-Algorithm (AA)**

Frequent itemset generation in AA

Item	Count
Orange juice	3
bread	4
cola	2
meat	4
milk	4
eggs	1

Itemset	Count
{orange juice, bread}	2
{orange juice, meat}	3
{orange juice, milk}	2
{bread, meat}	3
{bread, milk}	3
{meat, milk}	3

Itemset	Count
{bread, meat, milk}	2

Example

TID	Items
1	bread, milk
2	bread, meat, orange juice, eggs
3	milk, meat, orange juice, cola
4	bread, milk, meat, orange juice
5	bread, milk, meat, cola

Given: sup_min = 60% ~ min support count = 3

Candidate itemsets (up to size 3)

Brute-force strategy: $\binom{6}{1} + \binom{6}{2} + \binom{6}{3} = 6 + 15 + 20 = 41$

Apriori principal: $\binom{6}{1} + \binom{4}{2} + 1 = 6 + 6 + 1 = 13$

Mining Association Rules

Association Rules (AR) | AR-Discovery | Apriori-Algorithm (AA)

Rule generation in AA

$$\text{Conf}(X \rightarrow Y) = \frac{\rho(X \,\&\, Y)}{\rho(x)} = \frac{N * \text{Support}(X \,\&\, Y)}{N * \text{Support}(X)} = \frac{\text{Support}(X \,\&\, Y)}{\text{Support}(X)}$$

Notes:

1- Rule generation in AA is less computing time consuming as frequent itemsets generation, because the needed supports are already calculated

Mining Association Rules

Association Rules (AR) | **AR-Discovery** | **Apriori-Algorithm (AA)**

Rule generation in AA

Example: Given: conf_min = 80%

Item	Count
orange juice	3
bread	4
meat	4
milk	4

Itemset	Count
{orange juice, meat}	3
{bread, milk}	3
{bread, meat}	3
{meat, milk}	3

Itemset	Count
{meat, orange juice}	3

We consider the frequent itemset

Conf of { meat } → { orange juice} = 3/4 = 75%

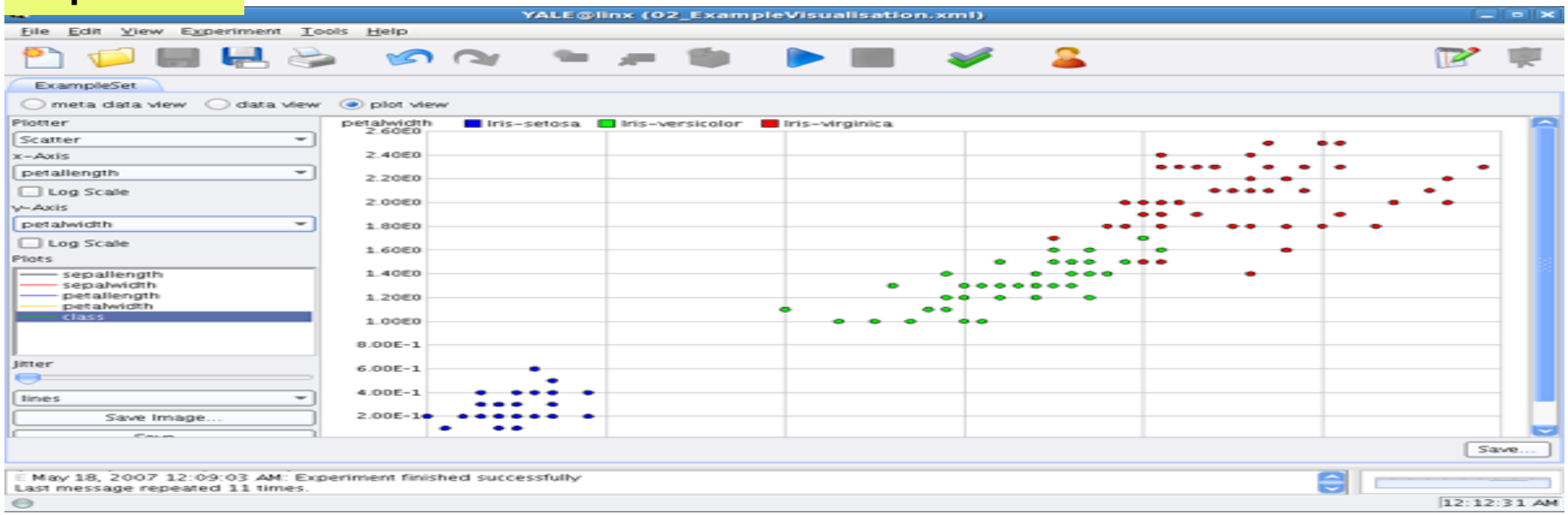
Conf of { orange juice } → { meat } =3/3 = 100%

Generated AR from the { meat, orange juice }

{ orange juice} → { meat }

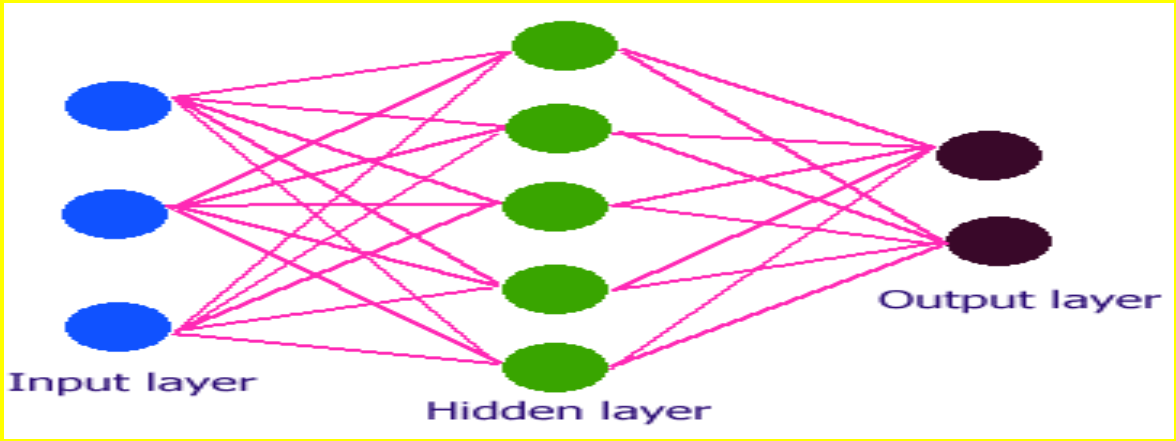
Practical Work : Association Rules

RapidMiner



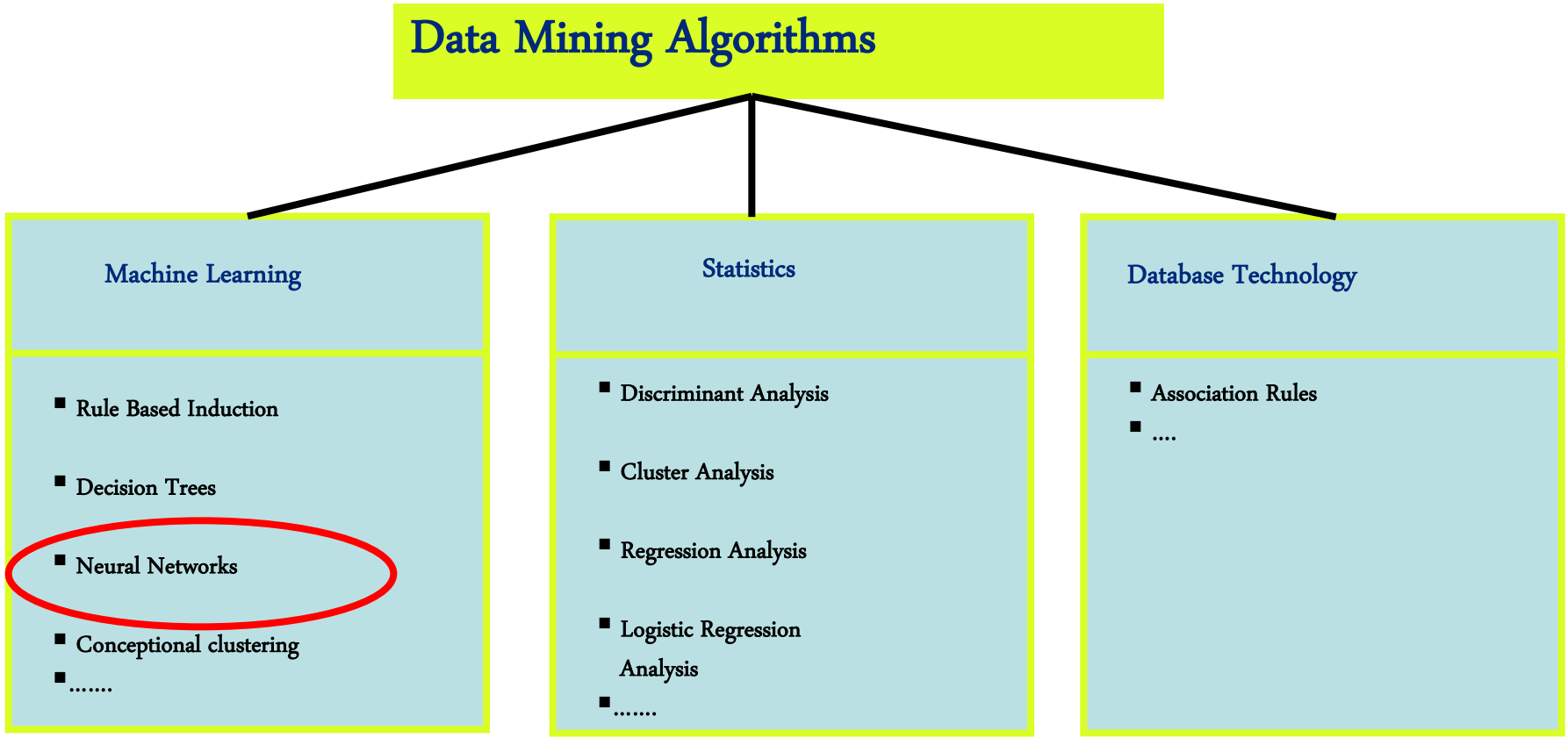
Association_Bookstore.xml

Load : Association_Bookstore



Artificial Neural Networks

Artificial Neural Networks



Artificial Neural Networks

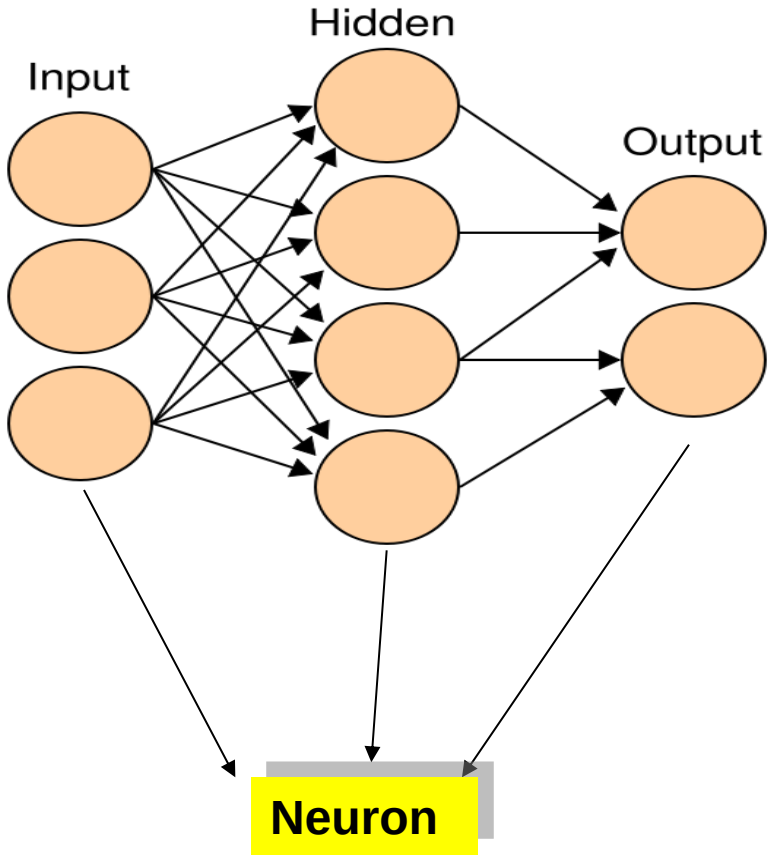
General remarks

- **Able to perform supervised AND unsupervised learning**
- **Belong, generally speaking, to relatively complicated class of learning algorithms**
- **Target Variable and attributes can be nominal or continues-valued**
- **ANN can be used for classification and prediction**

Artificial Neural Networks

Artificial Neural Networks (ANN) **Introduction**

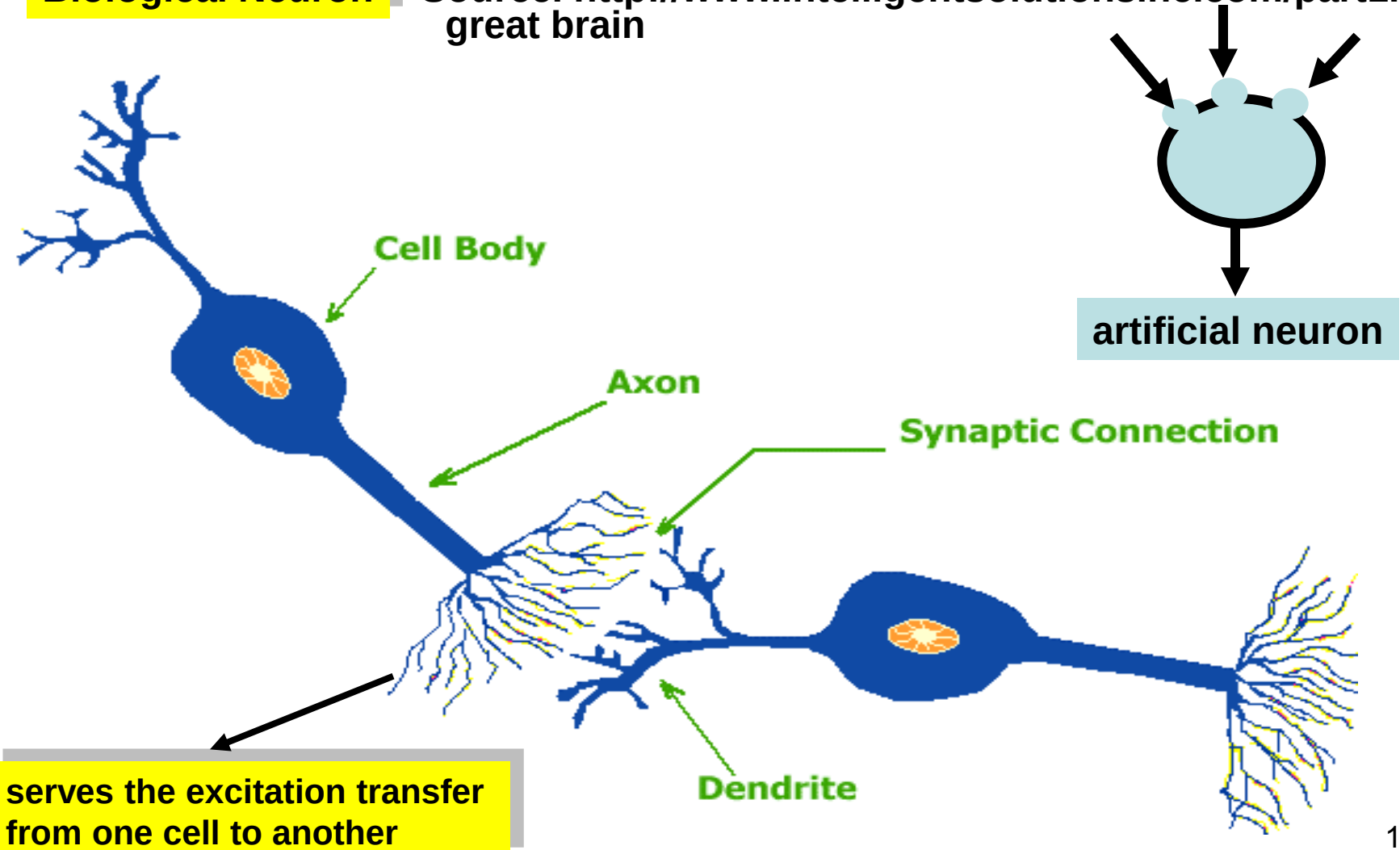
- Inspired by the way biological nervous systems e.g. the brain, process information
- An ANN consist of an input layer, an output layer and one or more hidden layer(s):
  - exchanges and process information
  - has learning capability
  - is an adaptive system that changes its structure during the learning phase



Artificial Neural Networks

Biological Neuron

Source. http://www.intelligentsolutionsinc.com/part1.htm
great brain



artificial neuron

serves the excitation transfer from one cell to another

Artificial Neural Networks

Rise and fall of Neural Networks

Fall

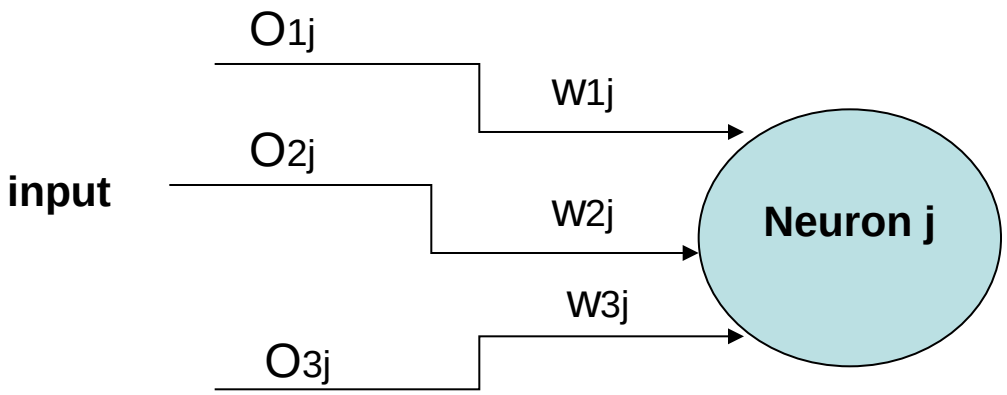
- In 1969 Minsky and Papert showed the limitations of single layer Perceptrons
- Their results caused that a lot of researchers loose their interest

Rise

- In the 70's and 80's, it was shown that multilevel perceptrons don't have These shortcomings
  - Paul J. Werbos invented 1974 the back-propagation having the ability to perform classification tasks beyond simple Perceptrons
  - Back-propagation was independently rediscovered in 1980s by David Rumelhat and David Parker

Artificial Neural Networks

Artificial Neural Networks | **Input function of neuron**



The input value is multiplied by the weight before entering the neuron
these weighted inputs are then added together and generate the :

$$\sum_{i=1}^{m} O_{ij}\, w_{ij} \longrightarrow \text{input function}$$

Oi : Input of the neuron i from the pervious layer

Artificial Neural Networks

Artificial Neural Networks | **Activation Function of neuron**

Selection between different activation functions is possible

Example of usual activation functions
Sigmoid (logistic) function:

$$O_j = \frac{1}{1 + e^{-aZj}}$$

O_j is the output of the unit j of the actuel layer.
The parameter **a** is a real number
and constant. Usually in ANN applications
a is selected between 0.5 and 2

Output of the unit i of the pervious layer

$$\text{with } Z_j = \sum_{i=1}^{m} O_{ij} W_{ij} + \Theta_j$$

Net input

where Θ_j is the bias

Artificial Neural Networks

Why nonlinear activation functions are needed ?

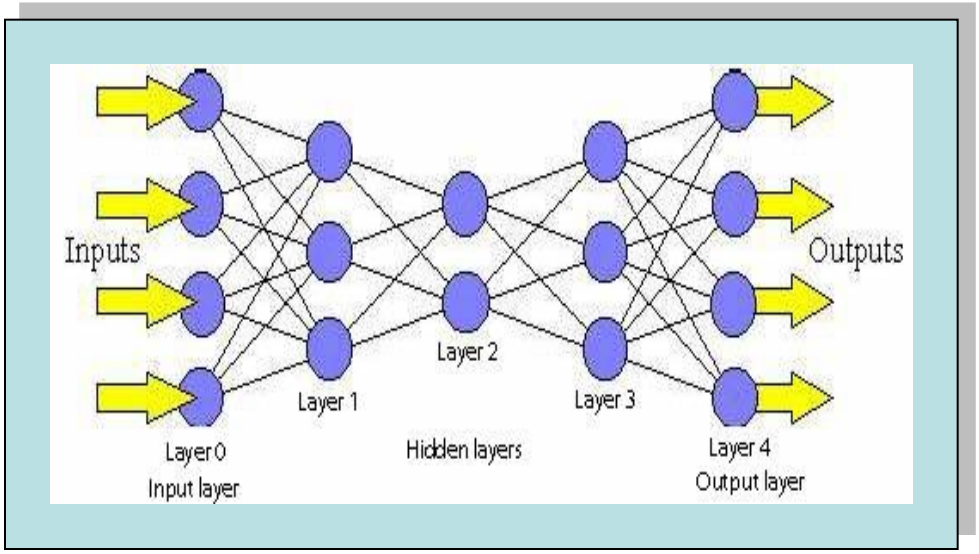
To catch the nonlinearity aspects in the data, otherwise the hidden units could not make the ANN more powerful than just Simple ***perceptrons*** without any hidden units.

The nonlinearity makes representing of nonlinear functions possible and is the most powerful adjective of multilayer ANN

Artificial Neural Networks

Artificial Neural Networks | **Architecture** | **Feed-Forward networks (FNN)**

- FNN (known also as multi-layer *perceptrons*) are widely used models specially in practical applications
- They were the first and simplest type of ANN developed



In a FFN

- the information moves in only one direction
- information at a later level never backpropagates to the previous levels
- there are no cycles or loops in the network

Artificial Neural Networks

Artificial Neural Networks | **Learning Process** | **Supervised Learning**

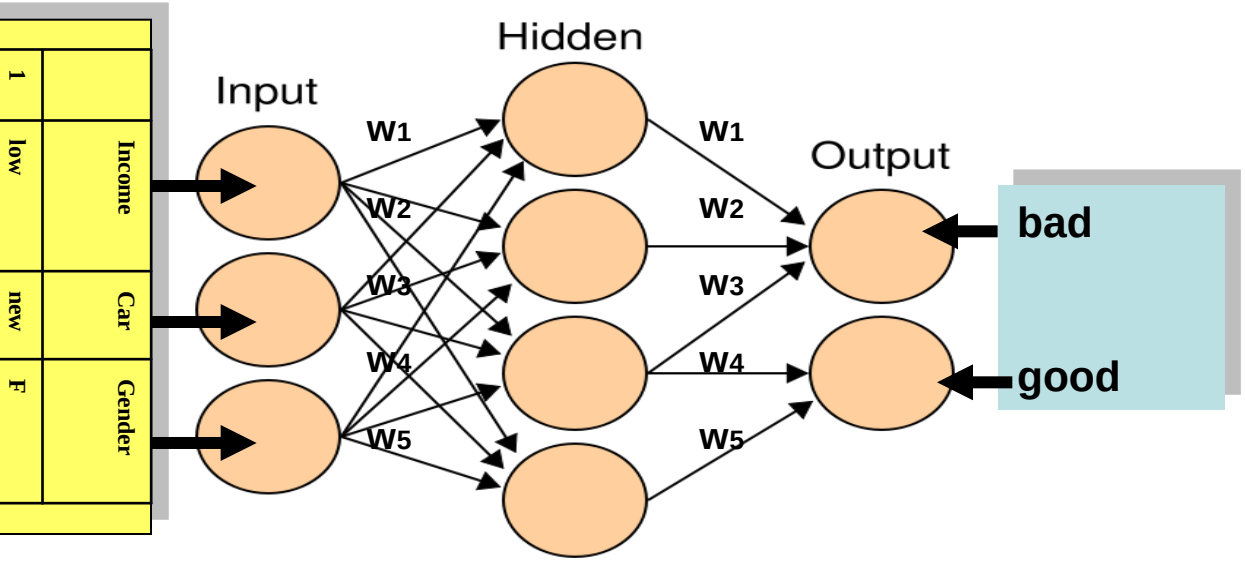
	Income	Car	Gender	Credit rate
1	low	new	F	bad
2	middle	old	F	bad
3	middle	new	M	good
4	low	new	M	bad
5	high	new	M	good
6	high	new	F	good
7	middle	new	F	good
8	high	old	F	good
9	middle	old	M	bad
10	low	old	F	bad

Artificial Neural Networks

Artificial Neural Networks

Learning Process

	Income	Car	Gender
1	low	new	F
2	middle	old	F
3	middle	new	M
4	low	new	M
5	high	new	M
6	high	new	F
7	middle	new	F
8	high	old	F
9	middle	old	M
10	low	old	F



Assumption: During the learning process, the net topology as well as input, activation and output functions remain constant. Only the weights change. Learning happens by weights adaptation.

The weights are adapted so long that the calculated net output is the correct desired output

Artificial Neural Networks

Training set:

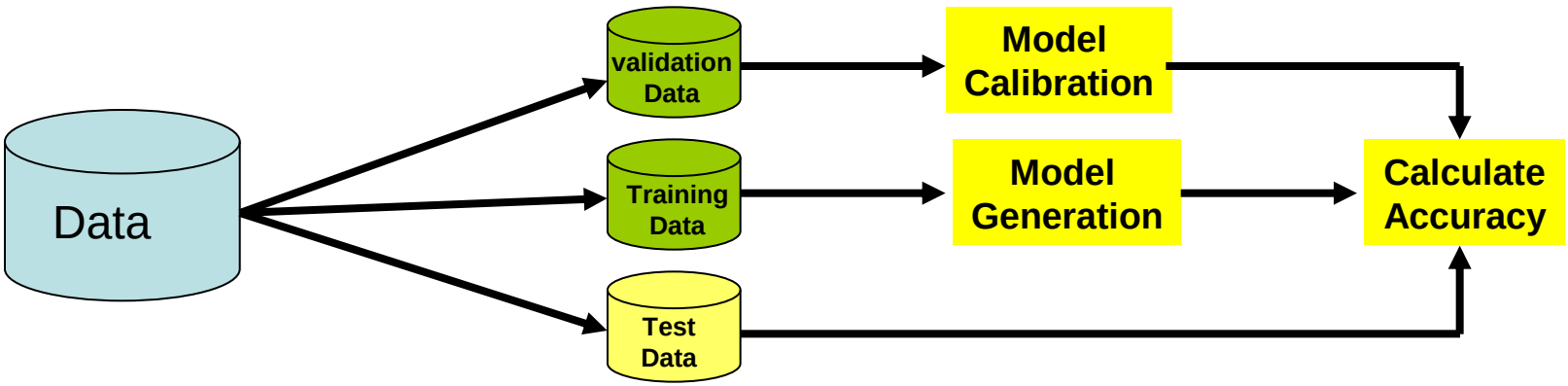
A set of examples used for learning, that is to fit the parameters [i.e., weights] of the classifier.

Validation set:

A set of examples used to tune the parameters [i.e., architecture, not weights] of a classifier, for example to choose the number of hidden units in a neural network.

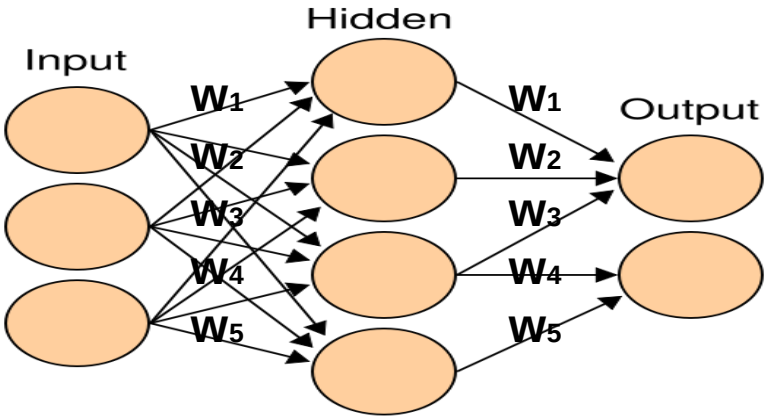
Test set:

A set of examples used only to assess the performance [generalization] of a fully-specified classifier.



Artificial Neural Networks

How should we calculate the weight changing ?



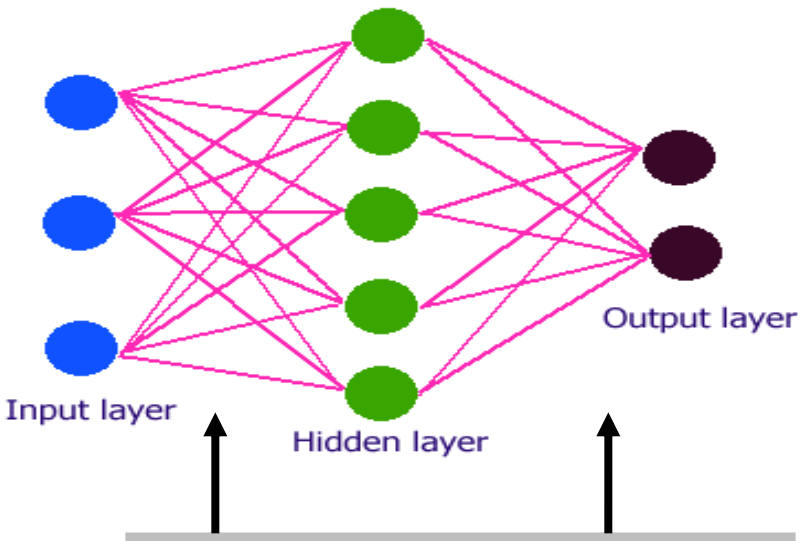
new weight = old weight + weight change

$w(t+1) = w(t) + \Delta w(t)$ ⟶ **Learning Rule**

In the last years several Learning Rules have been invented

Artificial Neural Networks

For each training Tuple:
The weights of the whole networks are modified in a manner to minimize the mean squared error (difference between the predicted and observed target value)



GD Learning Rule can be used to learn the weights of the outputs and hidden neurons

$$\text{Min: } E = \frac{1}{2} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

In most cases the output is a nonlinear function of weights and it is difficult to find the minimum of E analytically

↓

Gradient descent method

↓

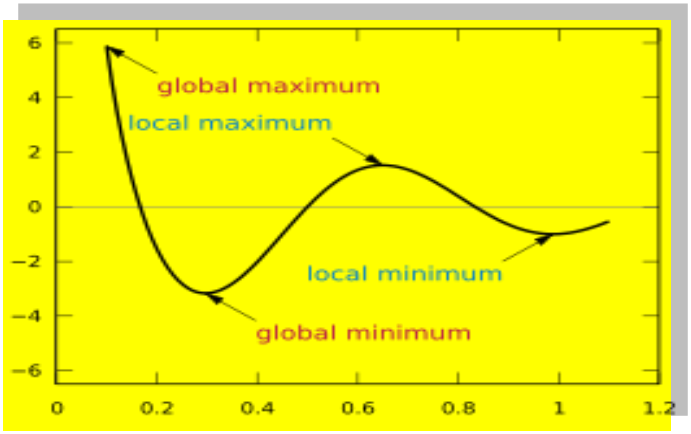
$$w_j \rightarrow w_j - \beta * \partial E(\mathbf{w}) / \partial w_j$$

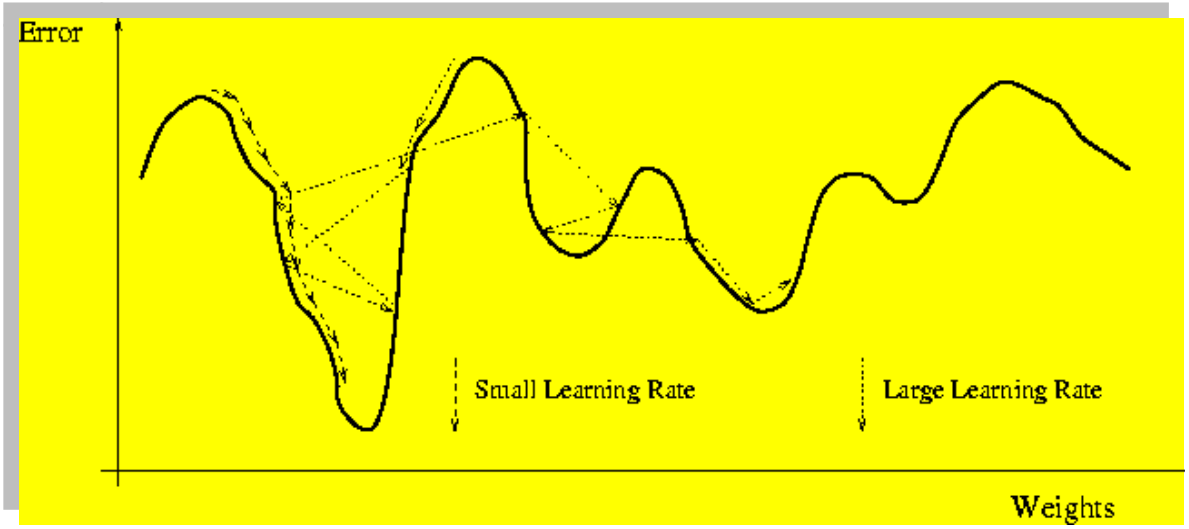
ß: Learning Rate between 0 and 1

Artificial Neural Networks

Disadvantages of Gradient descent method:

- Getting Stuck in a local optima
- Small gradient → small change (slow)
- Big gradient → big change (cavort)



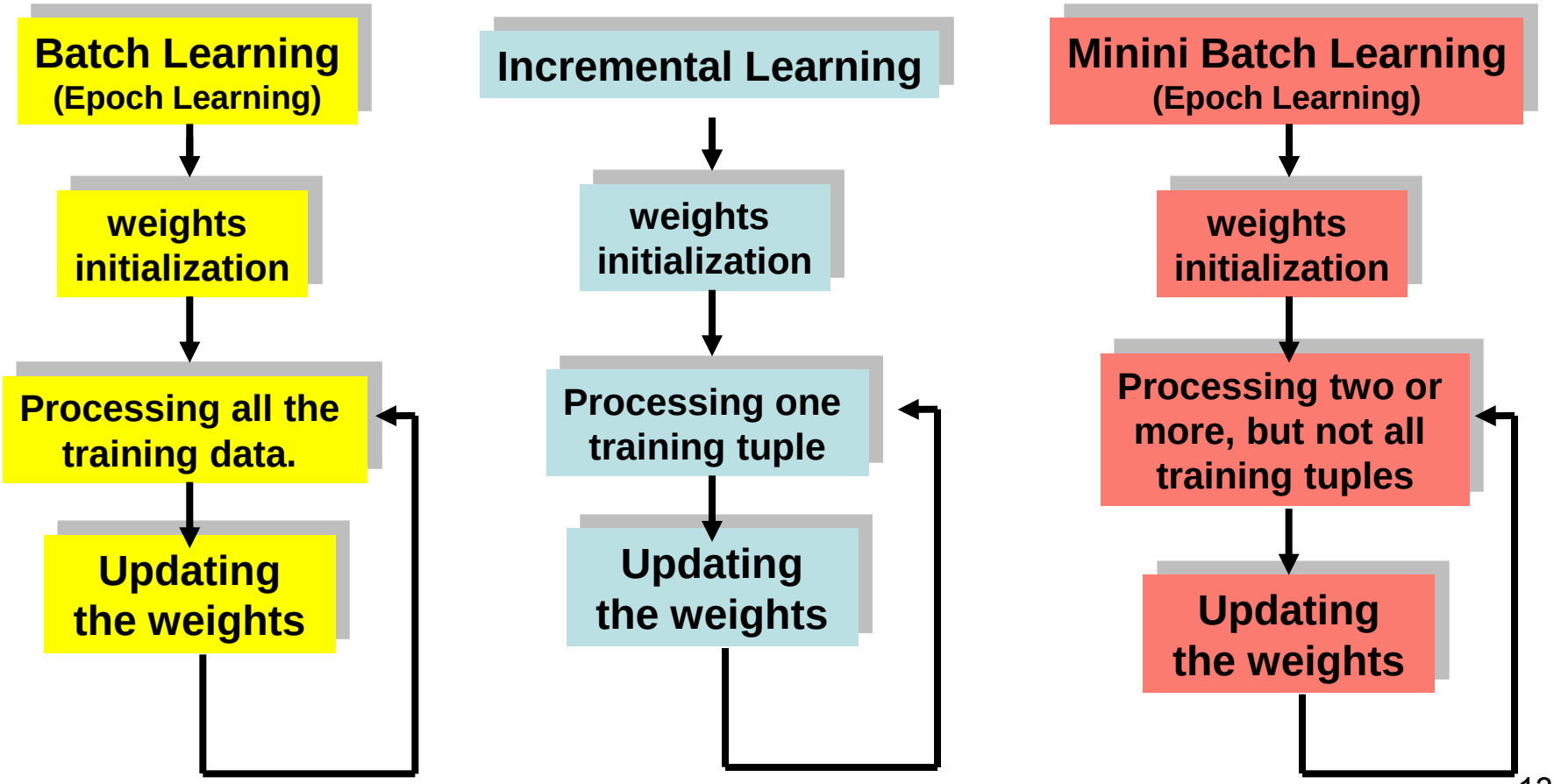


Source: http://www.teco.edu/~albrecht/neuro/html/node18.html#SECTION00630000000000000000

Artificial Neural Networks

Artificial Neural Networks | Learning Process | Learning Methods

Categorization of learning methods



Artificial Neural Networks

- Overfitting generates a relatively small error on the training dataset but larger error on new data
- To prevent Overfitting, it is sometime necessary to stop training by using a validation dataset

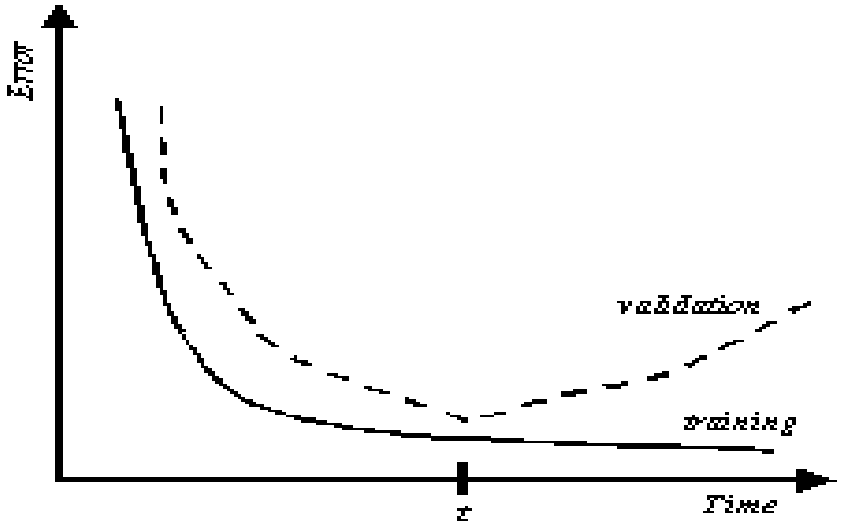
When stop training ?

- After a few epochs the ANN should be tested by using the validation dataset

Error on validation dataset increases higher than the last time error ?

↓

Training Stop



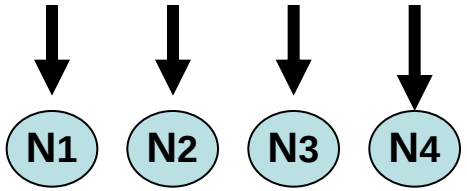
Artificial Neural Networks

there are two types of neurons:

- **Binary neurons** — **can only takes values from the set {0,1} or {1, -1}**
- **Real-valued neurons** — **can take values in the intervals [0,1] or [1, -1]**

Artificial Neural Networks

Marital status				
single	1	0	0	0
married	0	1	0	0
widowed	0	0	1	0
divorced	0	0	0	1

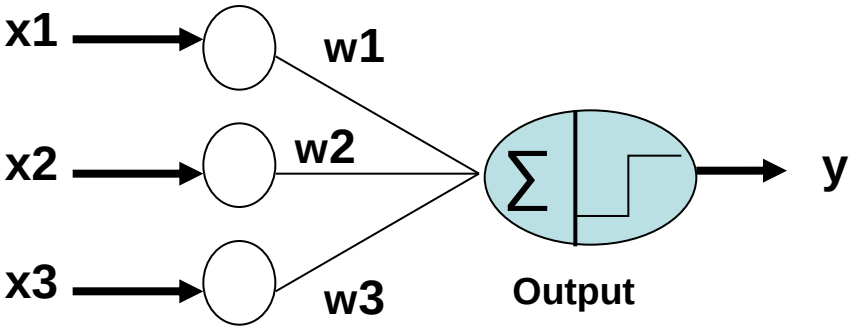


For each value of a nominal attribute an input neuron is necessary

Many nominal attributes with several values lead to a large number of input neurons

Artificial Neural Networks

Perceptron:
A simple ANN architecture consists only of input and output nodes (no hidden layer):



Computing the output y in Perceptron:

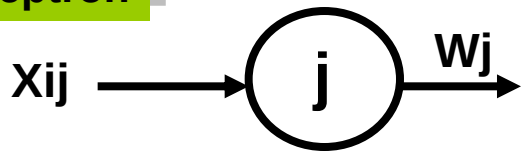
$$\hat{y} = \begin{cases} 1, & \text{if } w1x1 + w2\,x2 + w3x3 - t > 0 \\ -1, & \text{if } w1x1 + w2\,x2 + w3x3 - t =< 0 \end{cases}$$

t : Bias factor

Generally:

$$\hat{y} = \text{sign}\,(w1\,x1 + w2\,x2 + \ldots wj\,xj + \ldots wm\,xm - t)$$

Artificial Neural Networks



x11	x12 ….	x1j ….	x1m	y1
……				……
xi1	xi2 ….	xij ……	xim	yi
……				……
xn1	xn2 ….	xnj ….	xnm	yn

$$w_j^{(k+1)} = w_j^{(k)} + ß\ (y_i - \hat{y}_i^{(k)})\ x_{ij}$$

$w_j^{(k+1)}$: new weight in iteration k + 1 of neuron j

$w_j^{(k)}$: old weight in iteration k of neuron j

ß : learning rate

y_i : observed output of the tuple i

$\hat{y}_i^{(k)}$: calculated output of the tuple i in iteration k

x_{ij} : value of attribute j of the tuple i

The above Learning Rule is based on Gradient Descent method by minimizing

$$\text{Min: } E = \frac{1}{2} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Artificial Neural Networks

Min: $E = \frac{1}{2} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ for tuple i Min: $E = \frac{1}{2} (y_i - \hat{y}_i)^2$

$$\hat{y}i = \sum_{j=1}^{m} wij\, Xij \qquad \partial \hat{y}i \,/\, \partial\, wij = Xij$$

$$\partial E \,/\, \partial\, wj = -\,(yi - \hat{y}i) * \partial \hat{y}i \,/\, wij = -\,(yi - \hat{y}i)\, Xij \qquad (1)$$

$$Wj \longrightarrow Wj - ß * \partial\, E(\mathbf{W}) \,/\, \partial\, Wj \qquad (2)$$

(1) and (2) lead to

$$wj^{(k+1)} = wj^{(k)} + ß\ (yi - \hat{y}i^{(k)})\, xij$$

Artificial Neural Networks

The relation $w_1 x_1 + w_2 x_2 + \ldots w_j x_j + \ldots + w_m x_m$ is linear in w and x

For linearly separable classification problems, the learning algorithm

$$w_j^{(k+1)} = w_j^{(k)} + \beta \, (y_i - \hat{y}_i^{(k)}) \, x_{ij}$$

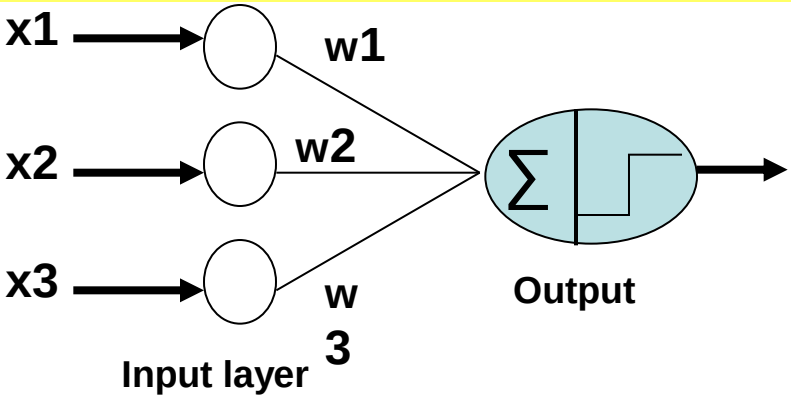
converges if the learning rate ß is sufficiently small

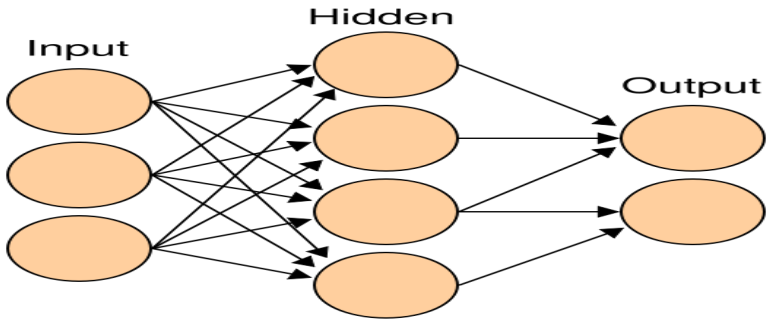
If the classes are not linearly separable, the above learning rule does not converge .

For such tasks ANN with hidden layers are necessary. Backpropagation is such an alternative.

Artificial Neural Networks

Backpropagation (backprop , BP) is the mostly used multilayer feed forward ANN, specially in the praxis . As we have seen, Perceptron can not handle the nonlinearly separable classes, but, BP can





The modification of the weights going to a hidden unit can not be performed by the method used in Perceptron because – in contrast to the output units – we have no information about the observed outputs of the hidden units. It means, it is not possible to determine the error term related to each hidden unit.

Solution: propagation of the errors of the output units backwards → Backpropagation

Artificial Neural Networks

Weight initialization : network weights are initialized randomly, usually from the interval [-1 , 1] or [- 0.5 , 0.5]

Calculation of the net input: As we have seen before the net Input of the neuron j is

$$z_j = \sum_{i=1}^{p} x_{ij} w_{ij} + \Theta_j \qquad (1)$$

Calculation of the output function: Using a logistic function we have got

$$O_j = \frac{1}{1 + e^{-a z_j}} \qquad (2)$$

This function is differentiable and nonlinear

Repeat (1) and (2) until we the output layer is reached and included

Artificial Neural Networks

Backpropagate the error

δj

E: Error of the unit j

$$\partial E / \partial w_{ij} = \partial E / \partial Z_j * \partial Z_j / \partial w_{ij} = X_{ij} * \partial E / \partial Z_j = X_{ij}\, \delta j$$

A- Neuron *j* belongs to the *Output Layer*

Contribution to **E** by j : $E = \frac{1}{2} (T_j - O_j)^2$ **Tj:** target Value of the neuron j

$\delta j = \partial / \partial Z_j \; E = -(T_j - O_j)\, \partial O_j / \partial Z_j$ and for **a =1**

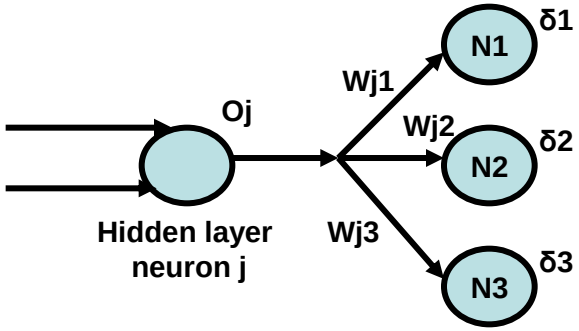
$$\delta j = -(T_j - O_j)(1 - O_j) O_j$$

Therefore : regarding the Gradient Descent Learning Rule : $W_{ij} \rightarrow W_{ij} + \beta\, \delta j\, X_{ij}$

Artificial Neural Networks

Backpropagate the error

Case B- Neuron *j* belongs to a *Hidden Layer*



$$E_j = \delta_j = O_j (1 - O_j) \sum_k \delta_k W_{jk}$$

Oj : Oput of the neuron j

δk : Error of the neuron k in the next layer

Wjk : weight of the connection from the neuron j to a neuron k in the next layer

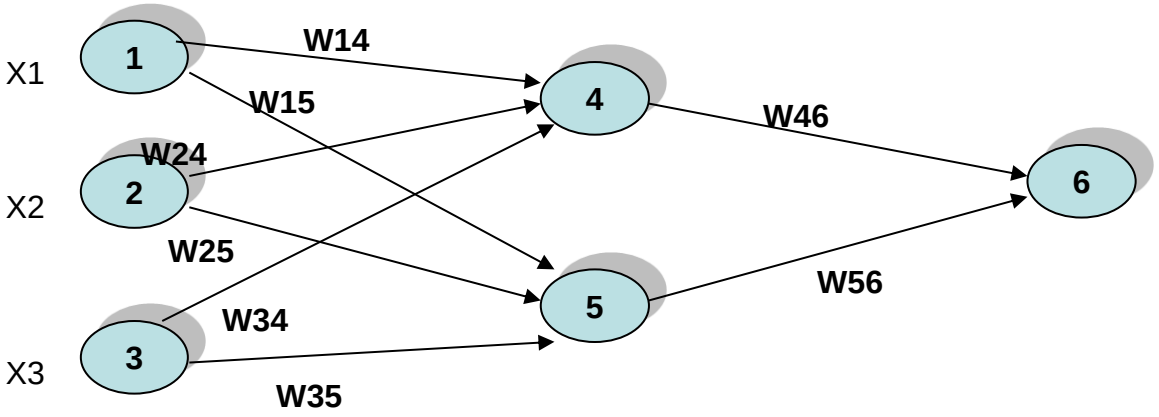
Gradient Descent Learning Rule : $W_{ij} \rightarrow W_{ij} + \beta \delta_j X_{ij}$

Bias updating for Neuron J : $\Theta_j = \Theta_j + \beta E_j$

Artificial Neural Networks

Artificial Neural Networks | Backpropagation Algorithms

Example
(source: Han et al (2006))



x1	x2	x3	w14	w15	w24	w25	w34	w35	w46	w56	Θ4	Θ5	Θ6
1	0	1	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0.1

Unit	Net Input, Zj	Output, Oj
4	0.2 + 0 -0.5 -0.4 = -0.7	$1/(1+e^{0.7}) = 0.332$
5	-0.3 + 0 +0.2 + 0.2 = 0.1	$1/(1+e^{-0.1}) = 0.525$
6	(-0.3) (0.332) – (02) (0.525) + 0.1 = -0.105	$1/(1+e^{0.105}) = 0.474$

Artificial Neural Networks

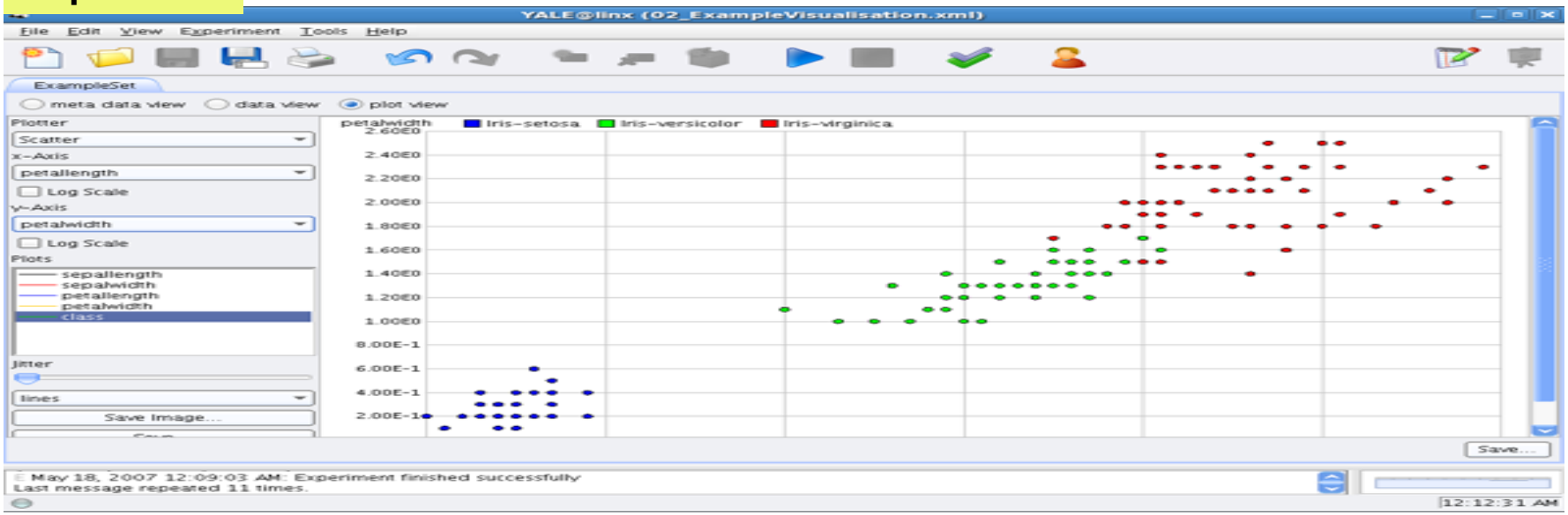
Example
(source: Han et al (2006))

Unit j	Error j
6	(0.474) (1- 0.474) (1- 0.474) = 0.1311
5	(0.525) (1-0.525) (0.1311) (-0.2) = - 0.0065
4	(0.332) (1- 0.332) (0.1311) (-0.3) = - 0.0087

Weight or bias	New value
W46	-0.3 + (0.9) (0.1311) (0.332) = - 0.261
W56	- 0.2 + (0.9) (0.1311) (0.525) = - 0.138
W14	0.2 + (0.9) (-0.0087) (1) = 0.192
W15	- 0.3 + (0.9) (-0.0065) (1) = -0.306
W24	0.4 + (0.9) (-0.0087) (0) = 0.4
W25	0.1 + (0.9) (-0.0065) (0) = 0.1
W34	-0.5 + (0.9) (-0.0087) (1) = - 0.508
W35	0.2 + (0.9) (-0.0065) (1) = 0.194
Θ6	0.1 + (0.9) (0.1311) = 0.218
Θ5	0.2 + (0.9) (-0.0065) = 0.194
Θ4	-0.4 + (0.9) (-0.0087) = - 0.408

Practical Work

RapidMiner



In Workspace:

Load German_CreditTr
Use Xvalidation and compare the performance of different algorithms to ANN

Weakness and Strength of ANN

Based on: http://www-sal.cs.uiuc.edu/~hanj/bk2/

- **Strength**
  - High tolerance to noisy data
  - Well-suited for continuous-valued inputs and outputs
  - Successful on a wide array of real-world data
  - Techniques have recently been developed for the extraction of rules from trained neural networks

- **Weakness**
  - Long training time, specially for the datasets with a large number of categorical attributes
  - Require a number of parameters typically best determined empirically, e.g., the network topology or ``structure."
  - Poor interpretability: Difficult to interpret the symbolic meaning behind the learned weights and of ``hidden units" in the network