Unit 6 : Data Mining Approaches and Methods

Types of Data Mining Models

Predictive Model

(a)Classification -Data is mapped into predefined groups or classes. Also termed as supervised learning as classes are established prior to examination of data.

(b) Regression- Mapping of data item into known type of functions. These may be linear, logistic functions etc.

(c) Time Series Analysis- Value of an attribute are examined at evenly spaced times, as it varies with time.

(d) Prediction- It means fore telling future data states based on past and current data.

Types of Data Mining Models

Descriptive Models

(a) Clustering- It is referred as unsupervised learning or segmentation/partitioning. In clustering groups are not pre-defined.

(b) Summarization- Data is mapped into subsets with simple descriptions . Also termed as Characterization or generalization.

(c) Sequence Discovery- Sequential analysis or sequence discovery utilized to find out sequential patterns in data. Similar to association but relationship is based on time.

(d) Association Rules- A model which identifies specific types of data associations.

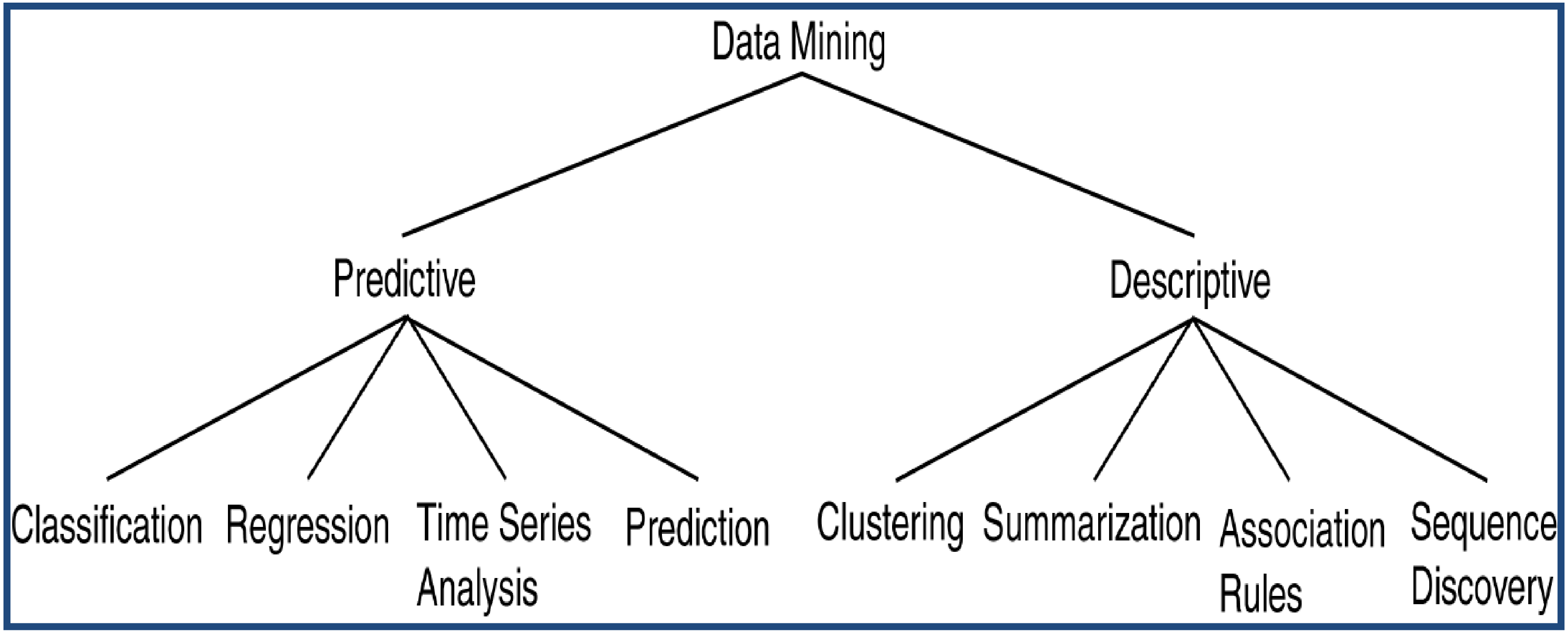
Descriptive vs. Predictive Data Mining

Descriptive Mining:

It describes concepts or task-relevant data sets in concise, summarative, informative, discriminative forms.

Predictive Mining:

It is based on data and analysis, constructs models for the database, and predicts the trend and properties of unknown data.



Supervised and Unsupervised learning

Supervised learning:

– The network answer to each input pattern is directly compared with the desired answer and a feedback is given to the network to correct possible errors

Unsupervised learning:

– The target answer is unknown. The network groups the input patterns of the training sets into clusters, based on correlation and similarities.

Supervised

- Bayesian Modeling
- Decision Trees
- Neural Networks

Type and number of classes are known in advance

Unsupervised

- One-way Clustering
- Two-way Clustering

Type and number of classes are **NOT** known in advance

Classification and Prediction

Classification and prediction are two forms of data analysis that can be used to extract models describing important data classes or to predict future data trends. Such analysis can help provide us with a better understanding of the data at large.

Whereas *classification* predicts categorical (discrete, unordered) labels, *prediction* models continuous valued functions.

For example, we can build a classification model to categorize bank loan applications as either safe or risky, or a prediction model to predict the expenditures in dollars of potential customers on computer equipment given their income and occupation.

Prediction

Prediction is viewed as the construction and use of a model to assess the class of an unlabeled sample or to assess the value ranges of an attribute that a given sample is likely to have.

It is a statement or claim that a particular event will occur in the future in more certain terms than a forecast . It is similar to classification. It constructs a model to predict unknown or missing values. Prediction is the most prevalent grade level expectation on reasoning in state mathematics standards.

Generally it predicts a continuous value rather than categorical label. Numeric prediction predicts the continuous value. The most widely used approach for numeric prediction is regression.

Regression analysis is used to model the relationship between one or more independent or predictor variables and a dependent or response variable. In the context of Data Mining, predictor variables are attributes of interest describing the tuple.

Linear Regression

Regression is a statistical methodology developed by Sir Frances Galton in 1822-1911. Straight line regression analysis involves a response variable y and a single predictor variable x.

The simplest form of regression is

y = a + bx

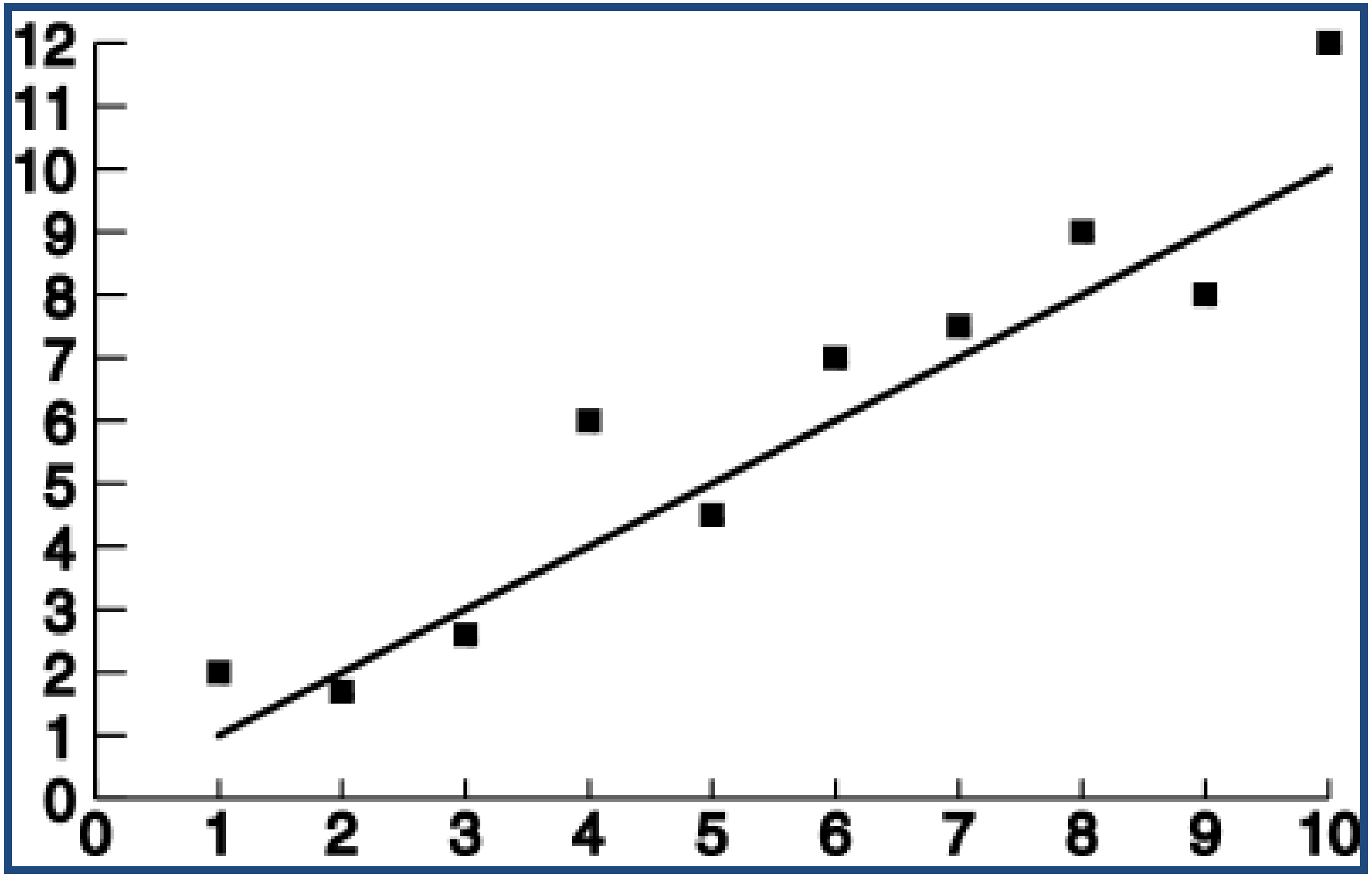
Where y is response variable and x is single predictor variable y is a linear function of x. a and b are regression coefficients.

As the regression coefficients are also considered as weights, we may write the above equation as:

y = w+w1x

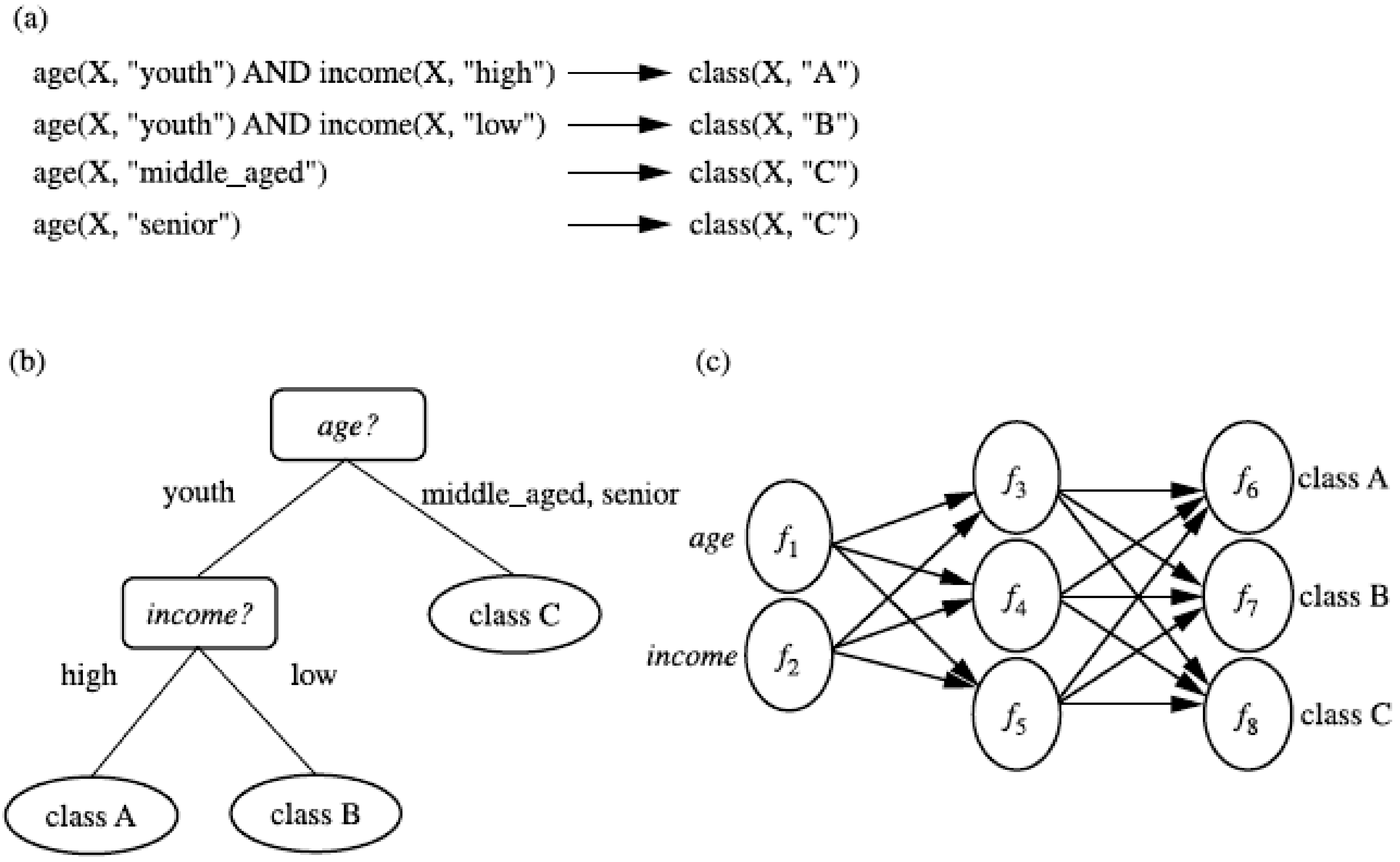
These coefficients are solved by the method of least squares, which estimates the best fitting straight line as the one that minimizes the error between the actual data and the estimate of the line.

Linear Regression



Classification is the process of finding a model (or function) that describes and distinguishes data classes or concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown. The derived model is based on the analysis of a set of training data (i.e., data objects whose class label is known).

"How is the derived model presented?" The derived model may be represented in various forms, such as *classification (IF-THEN) rules*, *decision trees*, *mathematical formulae*, or *neural networks*.



(a)

age(X, "youth") AND income(X, "high") → class(X, "A")

age(X, "youth") AND income(X, "low") → class(X, "B")

age(X, "middle_aged") → class(X, "C")

age(X, "senior") → class(X, "C")

Figure : A classification model can be represented in various forms, such as (a) IF-THEN rules, (b) a decision tree, or a (c) neural network.

Classification : Example of Grading

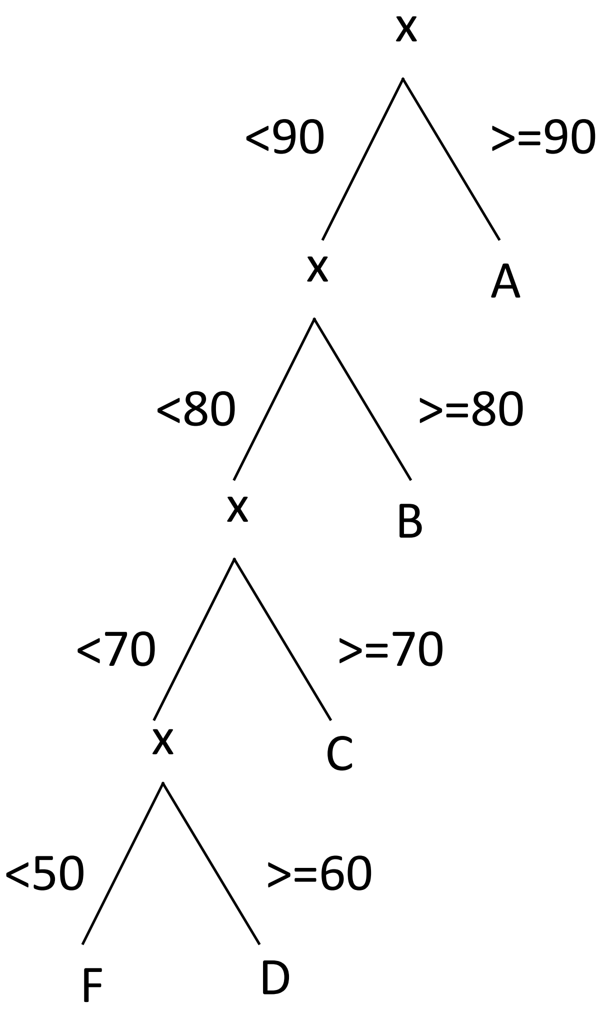
If x >= 90 then grade =A.

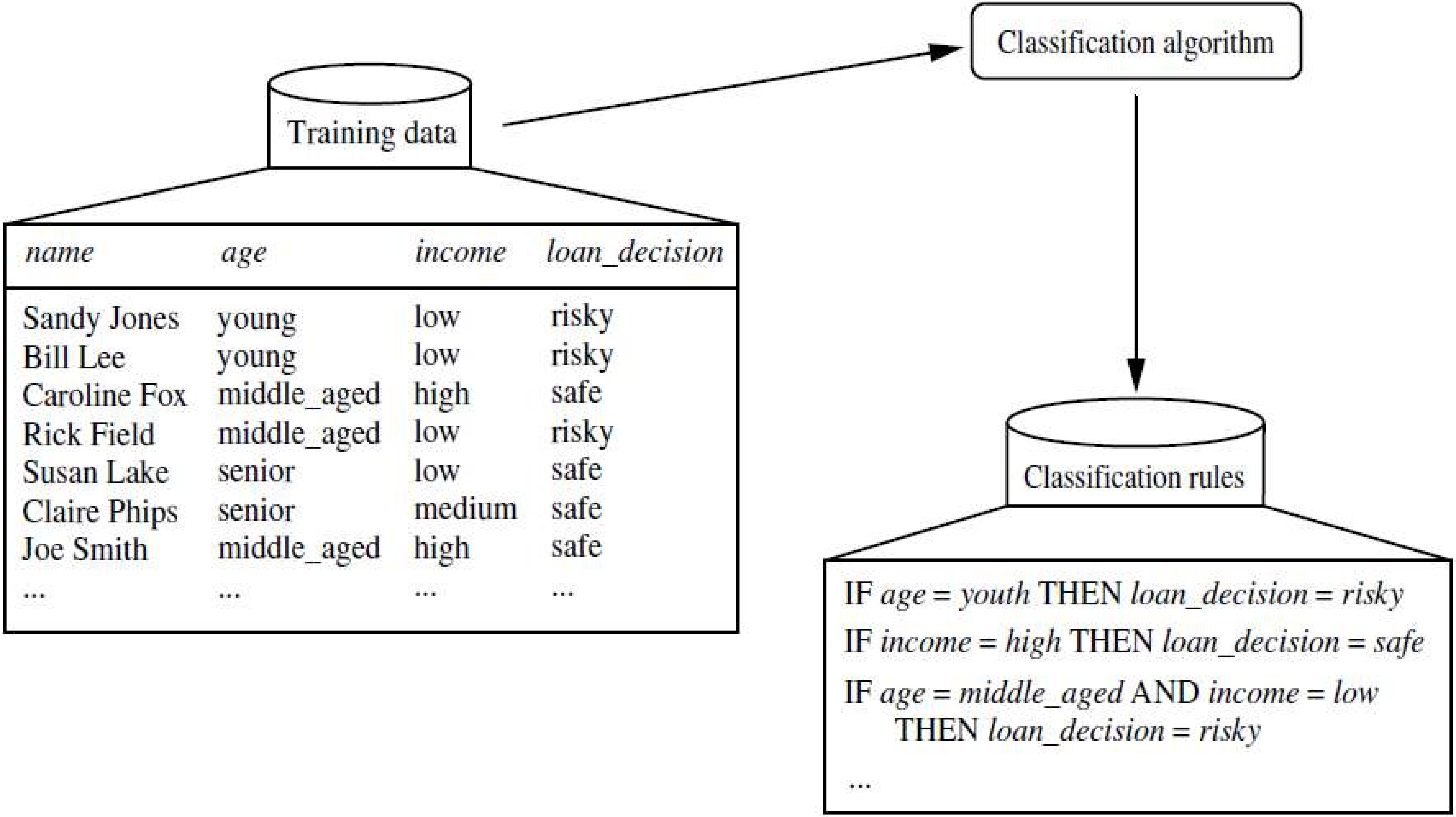
If 80<=x<90 then grade =B.

If 70<=x<80 then grade =C.

If 60<=x<70 then grade =D.

If x<50 then grade =F.





Training data

name	*age*	*income*	*loan_decision*
Sandy Jones	young	low	risky
Bill Lee	young	low	risky
Caroline Fox	middle_aged	high	safe
Rick Field	middle_aged	low	risky
Susan Lake	senior	low	safe
Claire Phips	senior	medium	safe
Joe Smith	middle_aged	high	safe
...	...	...	...

Classification algorithm

Classification rules

IF *age* = *youth* THEN *loan_decision* = *risky*

IF *income* = *high* THEN *loan_decision* = *safe*

IF *age* = *middle_aged* AND *income* = *low*
THEN *loan_decision* = *risky*

...

Figure: Learning

Here, the class label attribute is *loan decision*, and the learned model or classifier is represented in the form of classification rules.

Examples of Classification Algorithms

- Decision Trees
- Neural Networks
- Bayesian Networks

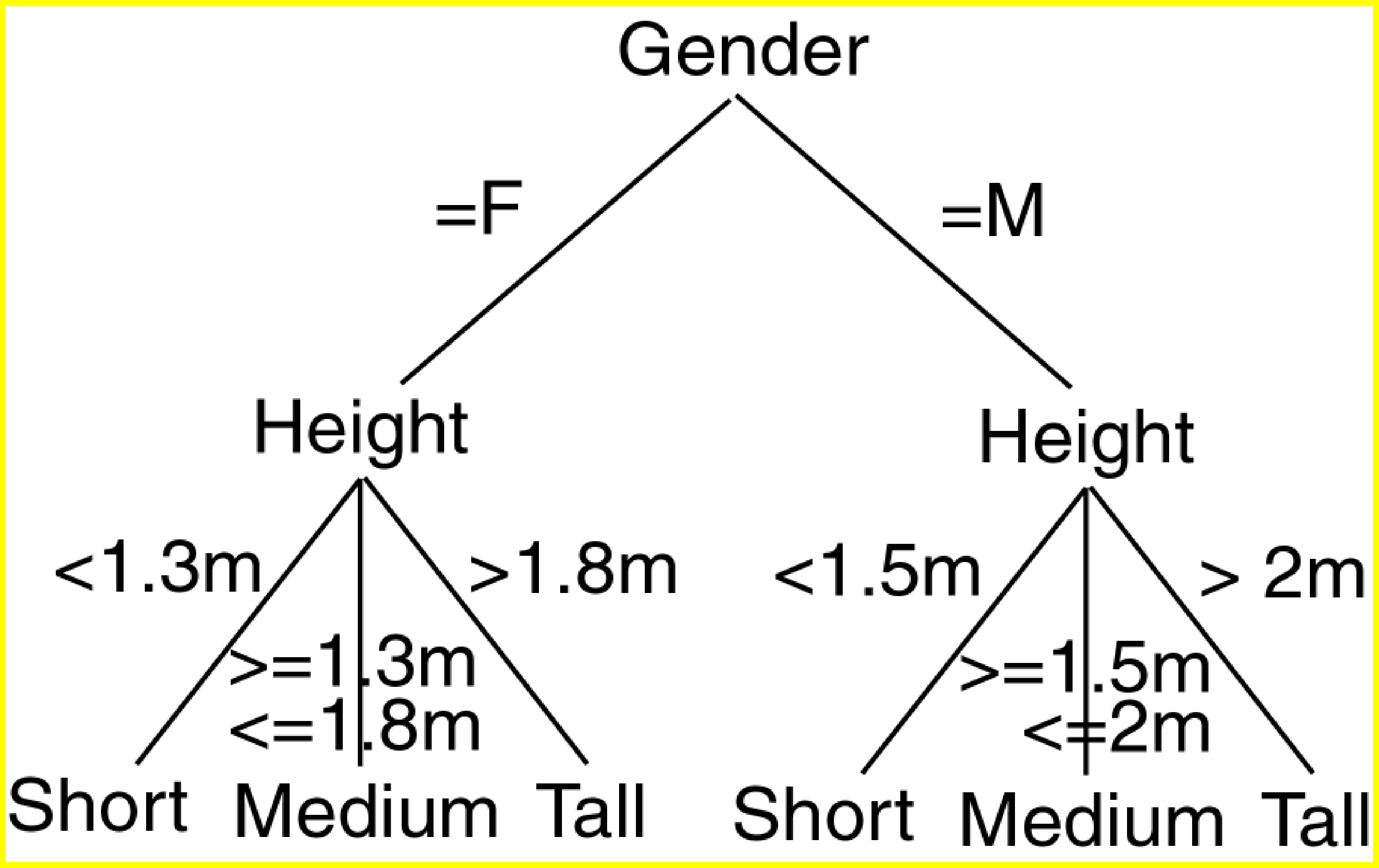
Decision Trees

A decision tree is a predictive model that as its name implies can be viewed as a tree. Specifically each branch of the tree is a classification question and the leaves are partitions of data set with their classification.

A decision tree makes a prediction on the basis of a series of decisions. The decision trees are being built on historical data and are a part of the supervised learning. The machine learning technique for inducting a decision tree from data is called decision tree learning.

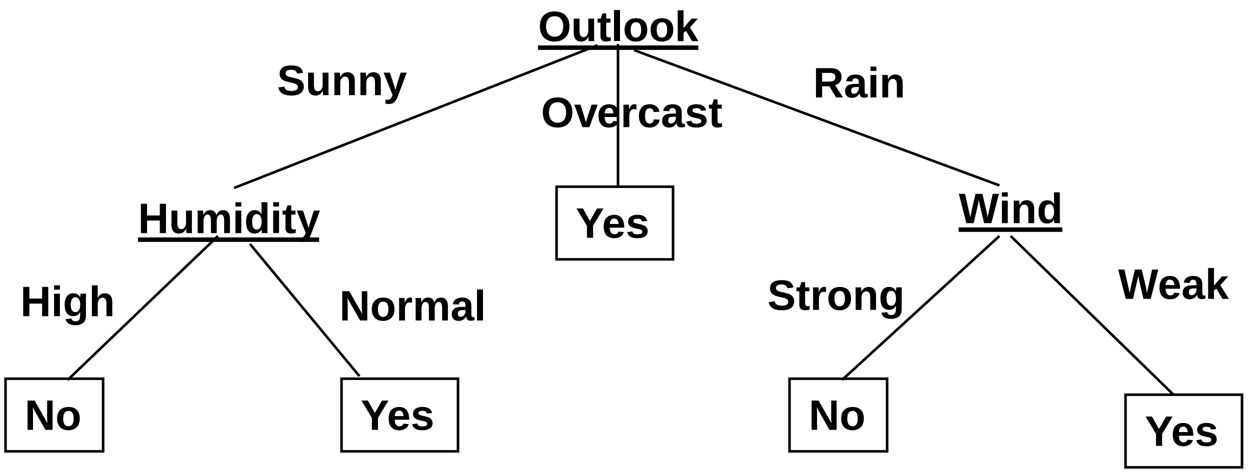
- Internal node denotes a test on an attribute
- Branch represents an outcome of the test
- Leaf nodes represent class labels or class distribution

Decision Tree Example



Decision Tree: Example

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



Attributes = {Outlook, Temperature, Humidity, Wind} Play Tennis = {yes, no}

Decision Tree Learning Algorithm - ID3

ID3 (Iterative Dichotomiser) is a simple decision tree learning algorithm developed by Ross Quinlan (1983). ID3 follow non-backtracking approach in which decision trees are constructed in a top-down recursive “divide and conquer” manner to test each attribute at every tree node. This approach starts with a training set of tuples and their associated class labels. Training set is recursively partitioned into smaller subsets as the tree is being built.

Pros and Cons of Decision Tree

Pros

- no distributional assumptions
- can handle real and nominal inputs
- speed and scalability
- robustness to outliers and missing values
- interpretability
- compactness of classification rules
- They are easy to use.
- Generated rules are easy to understand .
- Amenable to scaling and the database size.

Cons

- several tuning parameters to set with little guidance
- decision boundary is non-continuous
- Cannot handle continuous data.
- Incapable of handling many problems which cannot be divided into attribute domains.
- Can lead to over-fitting as the trees are constructed from training data.

Bayesian Classification

- Bayesian classifiers are statistical classifiers.
- They can predict class membership probabilities, such as the probability that a given tuple belongs to a particular class.
- Bayesian classification is based on Bayes’ theorem.
- Bayesian classifiers have also exhibited high accuracy and speed when applied to large databases.

Bayes’ Theorem

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}.$$

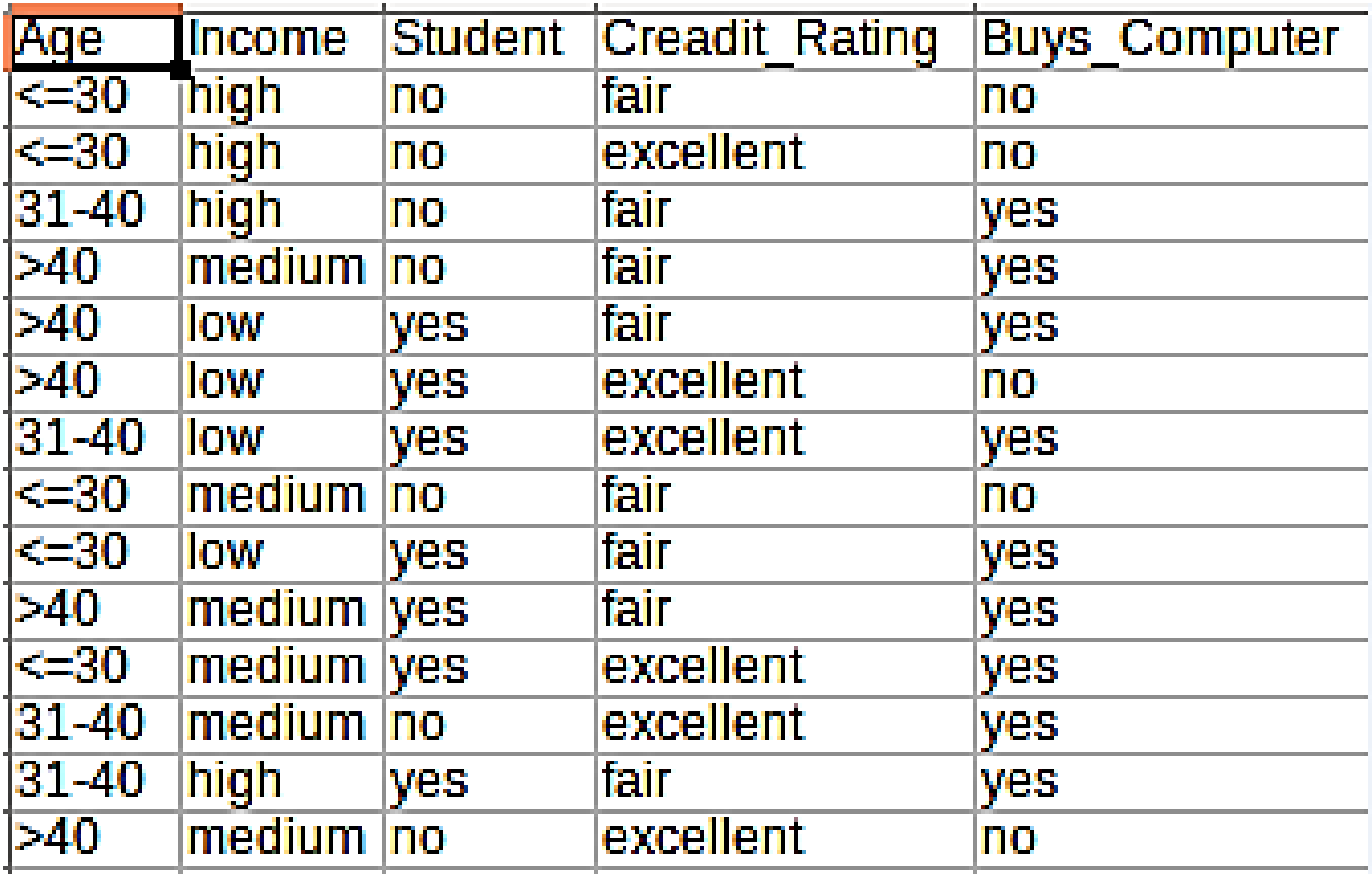
- *P*(*H*/***X***) is the posterior probability
- *P*(***X***|*H*) is the posterior probability of ***X*** conditioned on *H*.
- *P*(*H*) is the prior probability, or *a priori probability,* of *H*.
- *P*(***X***) is the prior probability of ***X***.

Naïve Bayesian Classification

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

- *P*(***X***) is constant for all classes, only *P*(***X***|*Ci*)*P*(*Ci*) need be maximized.

Example



Age	Income	Student	Creadit_Rating	Buys_Computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31-40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31-40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31-40	medium	no	excellent	yes
31-40	high	yes	fair	yes
>40	medium	no	excellent	no

RID	*age*	*income*	*student*	*credit_rating*	*Class: 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

- The data tuples are described by the attributes *age*, *income*, *student*, and *credit rating*.
- The class label attribute, *buys computer*, has two distinct values (namely, {*yes, no*}).
- Let *C*1 correspond to the class *buys computer = yes* and *C*2 correspond to *buys computer = no.*
- The tuple we wish to classify is
- ***X*** = (*age = youth, income = medium, student = yes, credit rating = fair*)
- We need to maximize *P*(***X***|*Ci*)*P*(*Ci*), for *i* = 1, 2.
- *P*(*Ci*), the prior probability of each class, can be computed based on the training tuples:
- *P*(*buys computer = yes*) = 9/14 = 0.643
- *P*(*buys computer = no*) = 5/14 = 0.357

- To compute *P(**X|**Ci*), for *i* = 1, 2, we compute the following conditional probabilities:
- *P*(*age = youth* | *buys computer = yes*) = 2/9 = 0.222
- *P*(*age = youth* | *buys computer = no*) = 3/5 = 0.600
- *P*(*income = medium* | *buys computer = yes*) = 4/9 = 0.444
- *P*(*income = medium* | *buys computer = no*) = 2/5 = 0.400
- *P*(*student = yes* | *buys computer = yes*) = 6/9 = 0.667
- *P*(*student = yes* | *buys computer = no*) = 1/5 = 0.200
- *P*(*credit rating = fair* | *buys computer = yes*) = 6/9 = 0.667
- *P*(*credit rating = fair* | *buys computer = no*) = 2/5 = 0.400

- Using the above probabilities, we obtain
- *P*(***X***|*buys computer = yes*) = *P*(*age = youth* | *buys computer = yes*) X *P*(*income = medium* | *buys computer = yes*) X *P*(*student = yes* | *buys computer = yes*) X *P*(*credit rating = fair* | *buys computer = yes*)

  = 0.222 x 0.444 x 0.667 x 0.667

  = 0.044.
- Similarly,
- *P*(***X***|*buys computer = no*)

  = 0.600 x 0.400 x 0.200 x 0.400 = 0.019.

- To find the class, *Ci*, that maximizes *P*(***X***|*Ci*)*P*(*Ci*), we compute
- *P*(***X***|*buys computer = yes*) X *P*(*buys computer = yes*)

  = 0.044 x 0.643

  = 0.028
- *P*(***X***|*buys computer = no*) X *P*(*buys computer = no*)

  = 0.019 x 0.357

  = 0.007
- Therefore, the naïve Bayesian classifier predicts "***buys_computer = yes***" for tuple ***X***.

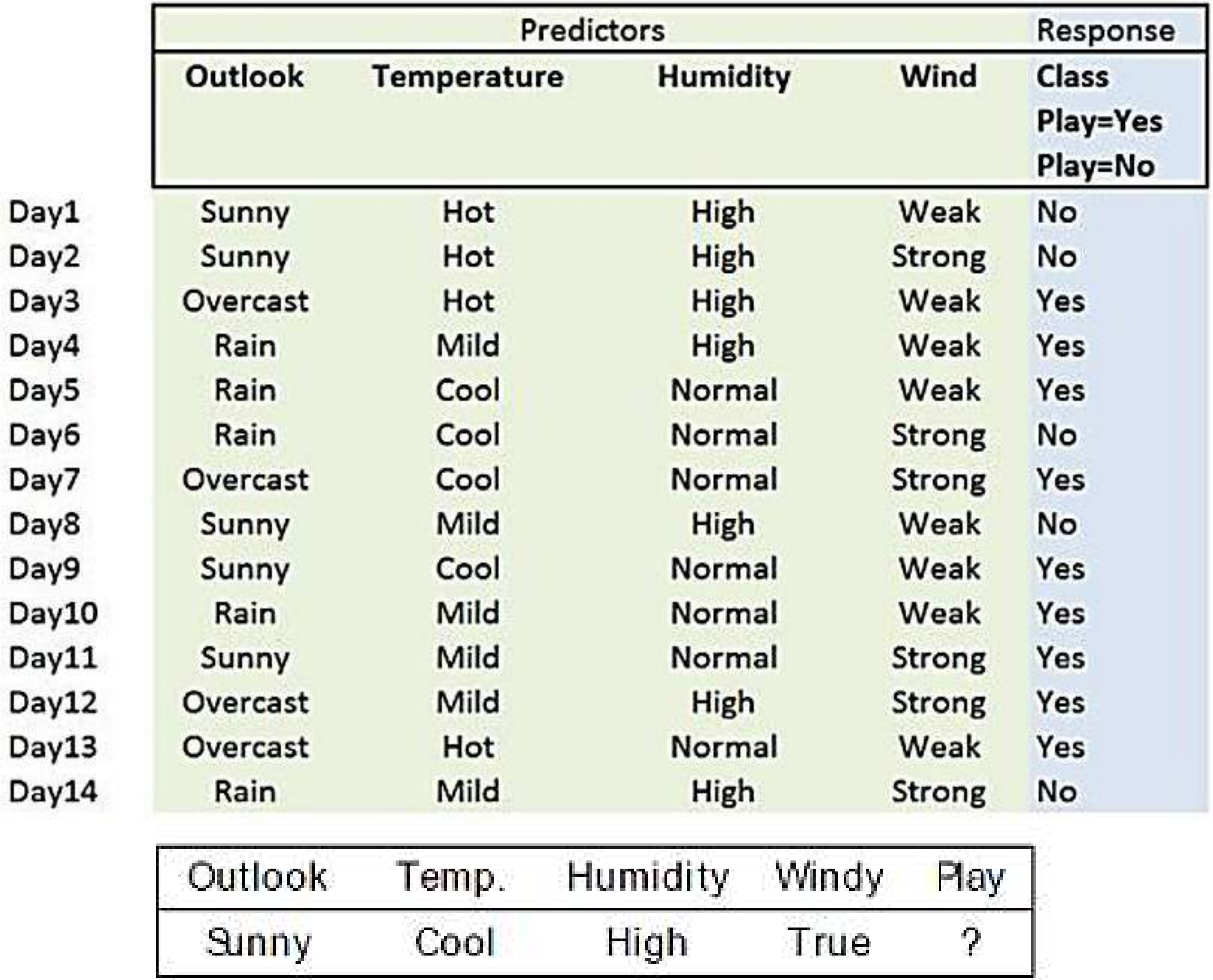
Example

Given all the previous patients I've seen (below are their symptoms and diagnosis)...

chills	runny nose	headache	fever	flu?
Y	N	Mild	Y	N
Y	Y	No	N	Y
Y	N	Strong	Y	Y
N	Y	Mild	Y	Y
N	N	No	N	N
N	Y	Strong	Y	Y
N	Y	Strong	N	N
Y	Y	Mild	Y	Y

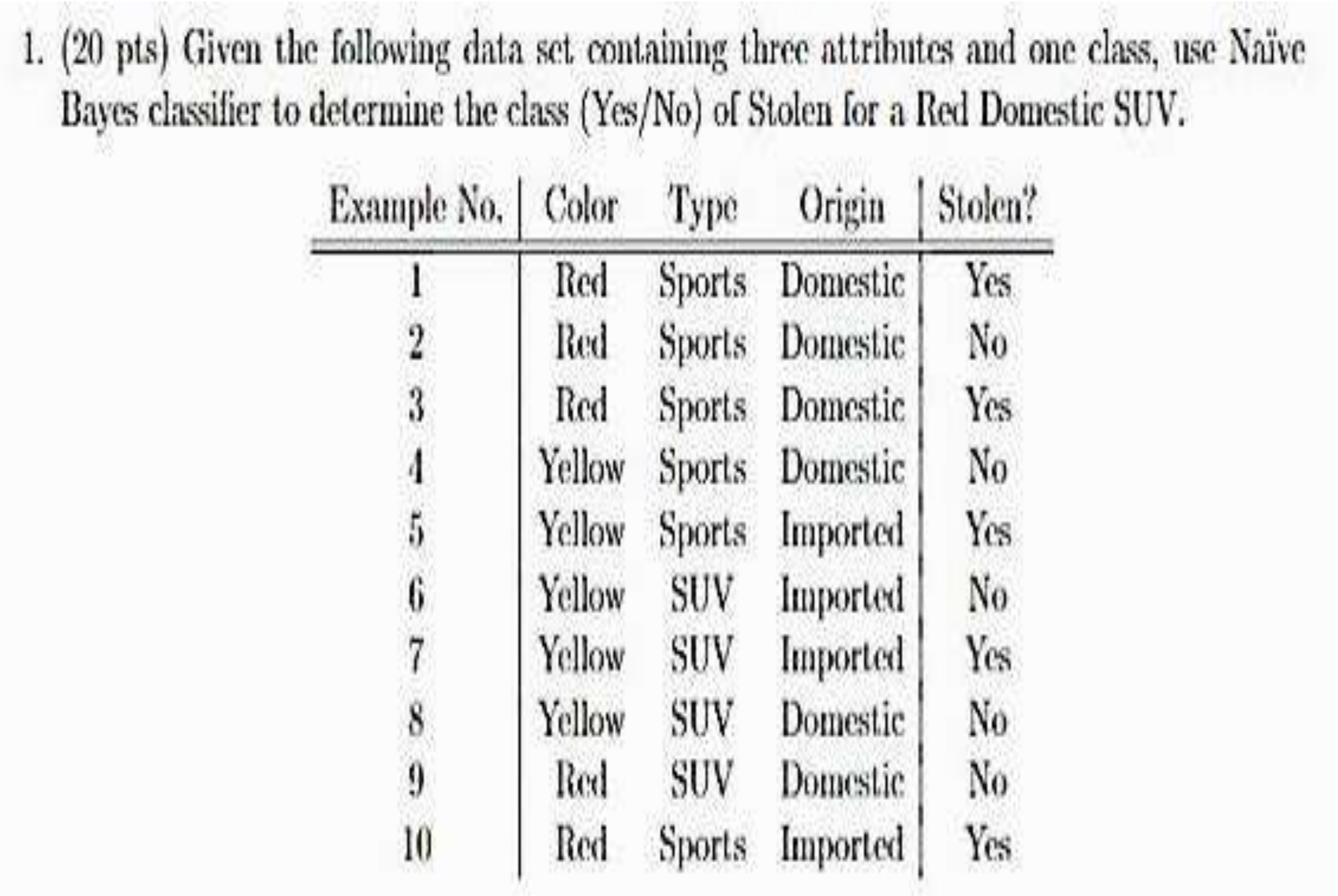
Do I believe that a patient with the following symptoms has the flu?

chills	runny nose	headache	fever	flu?
Y	N	Mild	Y	?



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

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?



1. (20 pts) Given the following data set containing three attributes and one class, use Naïve Bayes classifier to determine the class (Yes/No) of Stolen for a Red Domestic SUV.

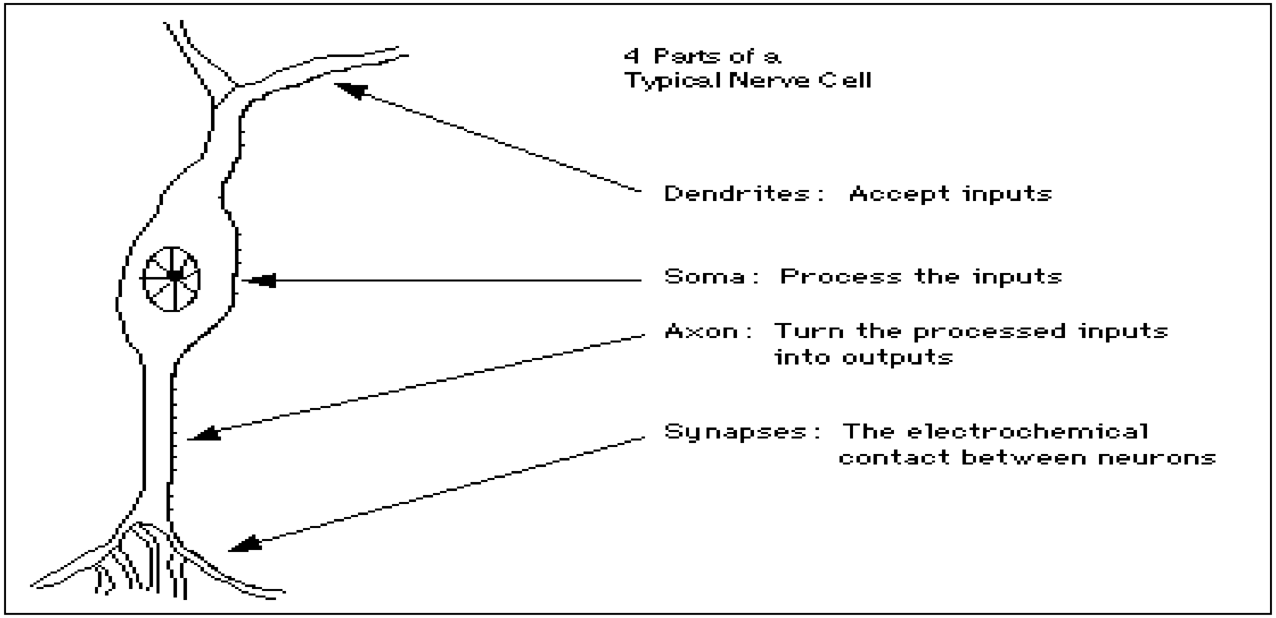
Example No.	Color	Type	Origin	Stolen?
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	Yes
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	Yes
8	Yellow	SUV	Domestic	No
9	Red	SUV	Domestic	No
10	Red	Sports	Imported	Yes

Neural Networks

Neural Network is a set of connected INPUT/OUTPUT UNITS, where each connection has a WEIGHT associated with it. It is a case of SUPERVISED, INDUCTIVE or CLASSIFICATION learning.

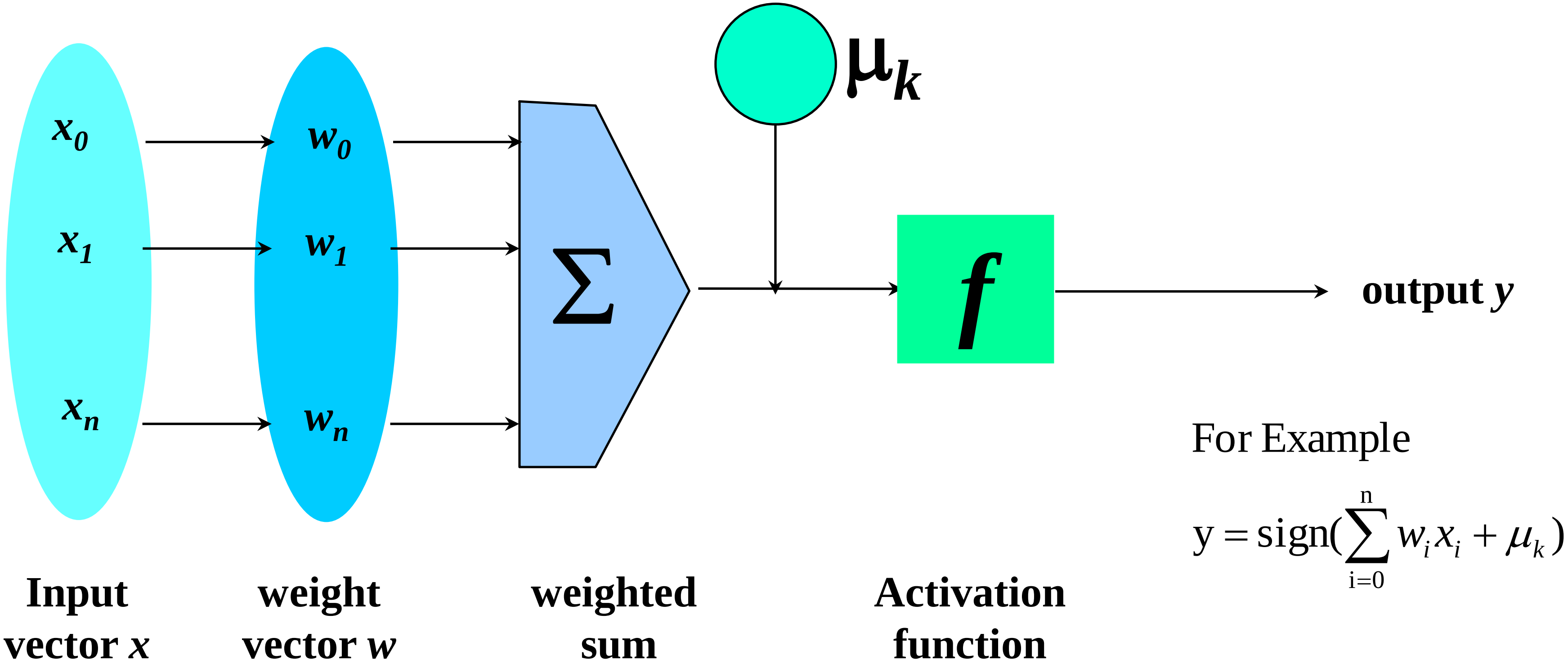
Neural Network learns by adjusting the weights so as to be able to correctly classify the training data and hence, after testing phase, to classify unknown data. Neural Network needs long time for training. Neural Network has a high tolerance to noisy and incomplete data.

Similarity with Biological Network


4 Parts of a
Typical Nerve Cell
Dendrites : Accept inputs
Soma : Process the inputs
Axon : Turn the processed inputs into outputs
Synapses : The electrochemical contact between neurons

- Fundamental processing element of a neural network is a neuron
- A human brain has 100 billion neurons
- An ant brain has 250,000 neurons

A Neuron (= a Perceptron)

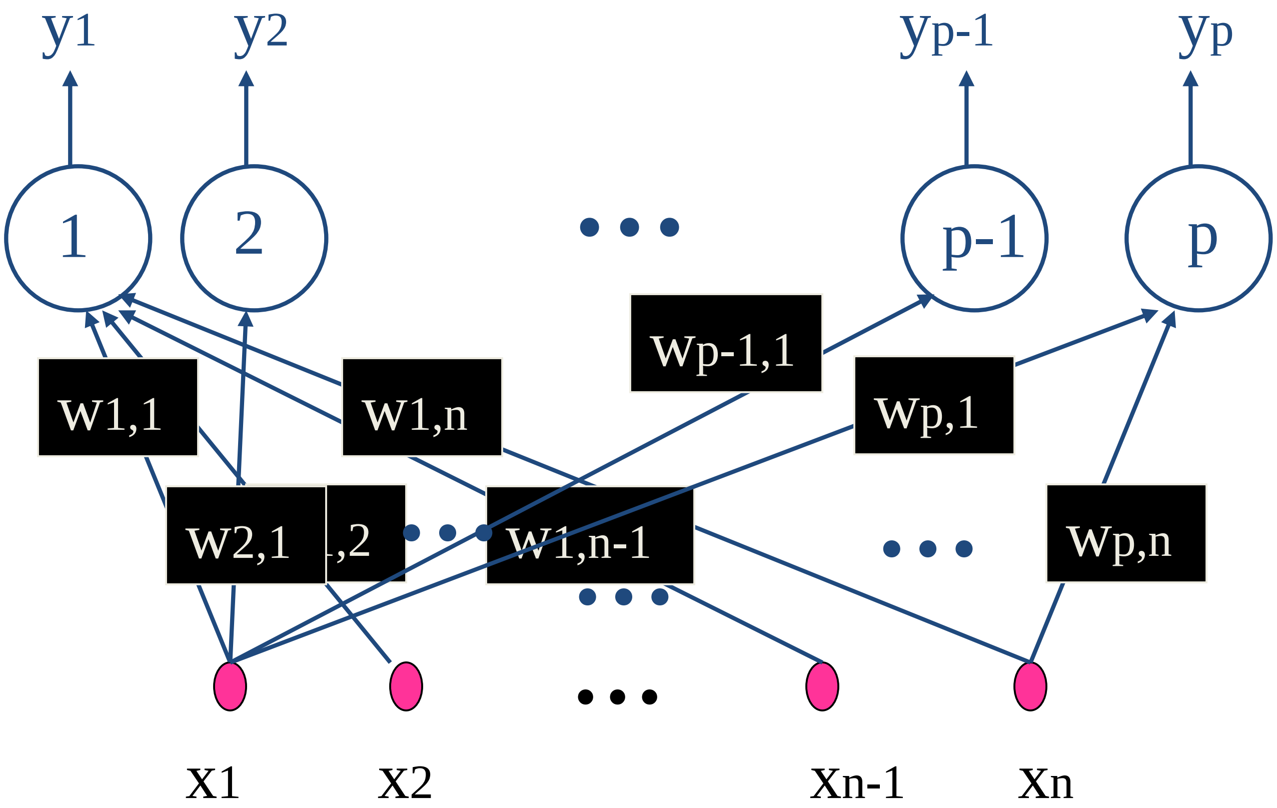


For Example

$$y = \mathrm{sign}\left(\sum_{i=0}^{n} w_i x_i + \mu_k\right)$$

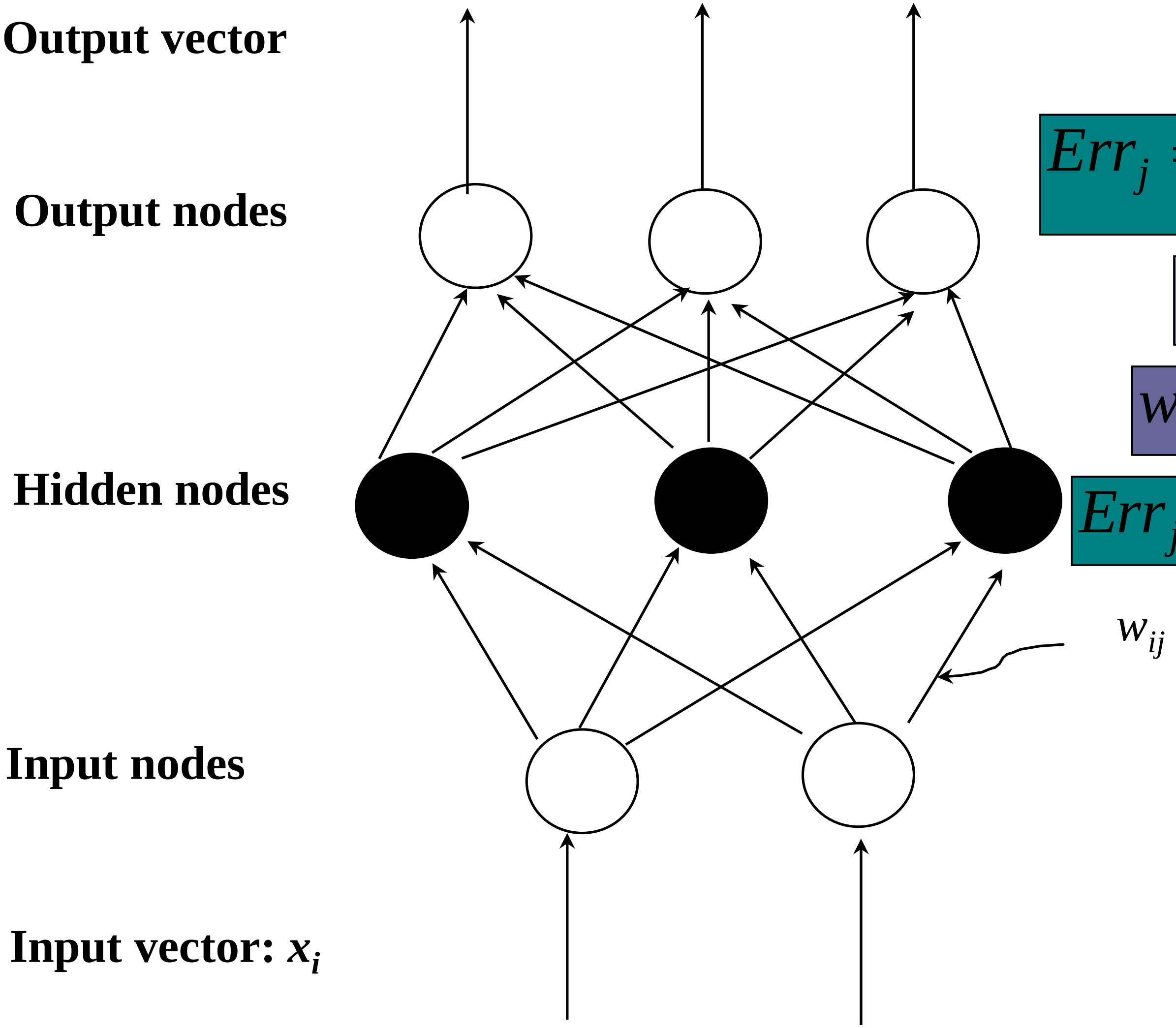
The *n*-dimensional input vector *x* is mapped into variable *y* by means of the scalar product and a nonlinear function mapping

Perceptron



$$y_i(t+1) = f\left(\sum_{k=0}^{n} w_{ik}\, x_k(t)\right) \qquad i = 1, 2, \ldots p$$

Multi-Layer Perceptron



$$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$$

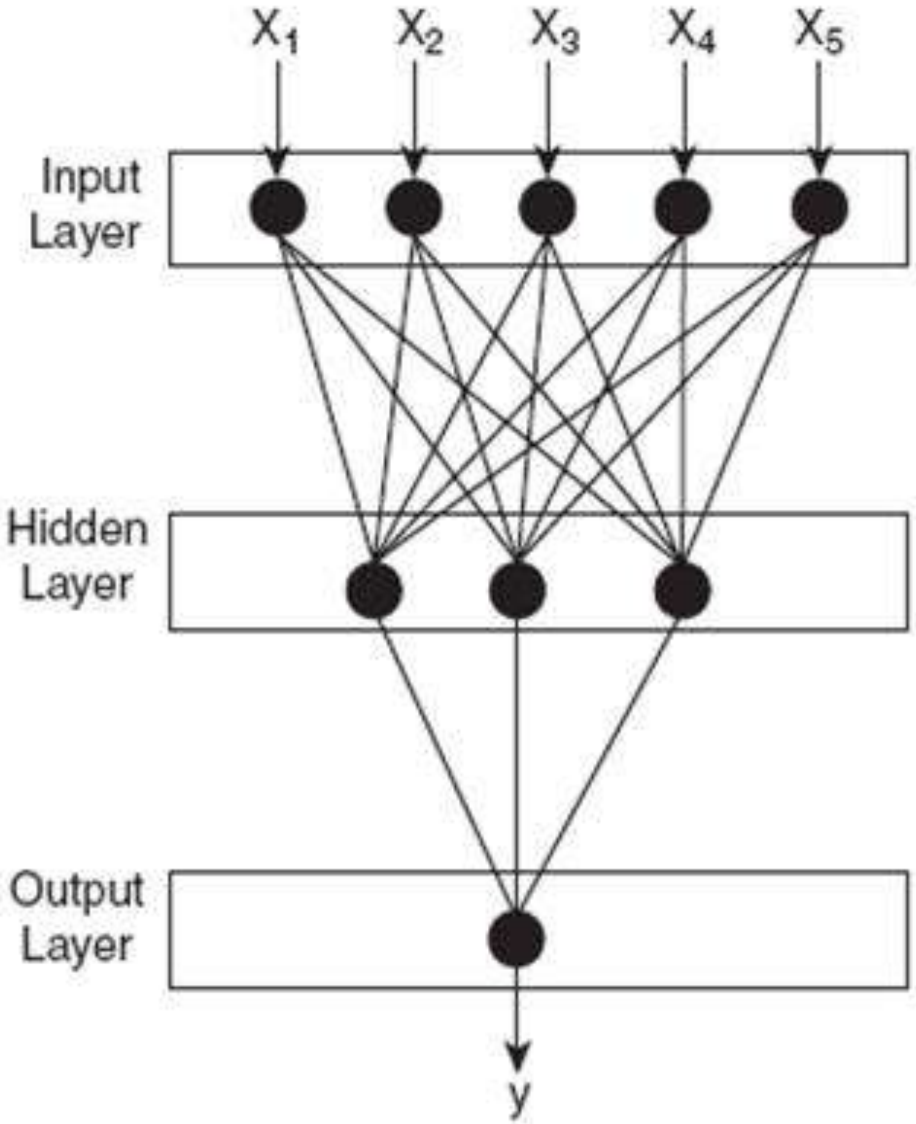
$$\theta_j = \theta_j + (l)Err_j$$

$$w_{ij} = w_{ij} + (l)Err_j O_i$$

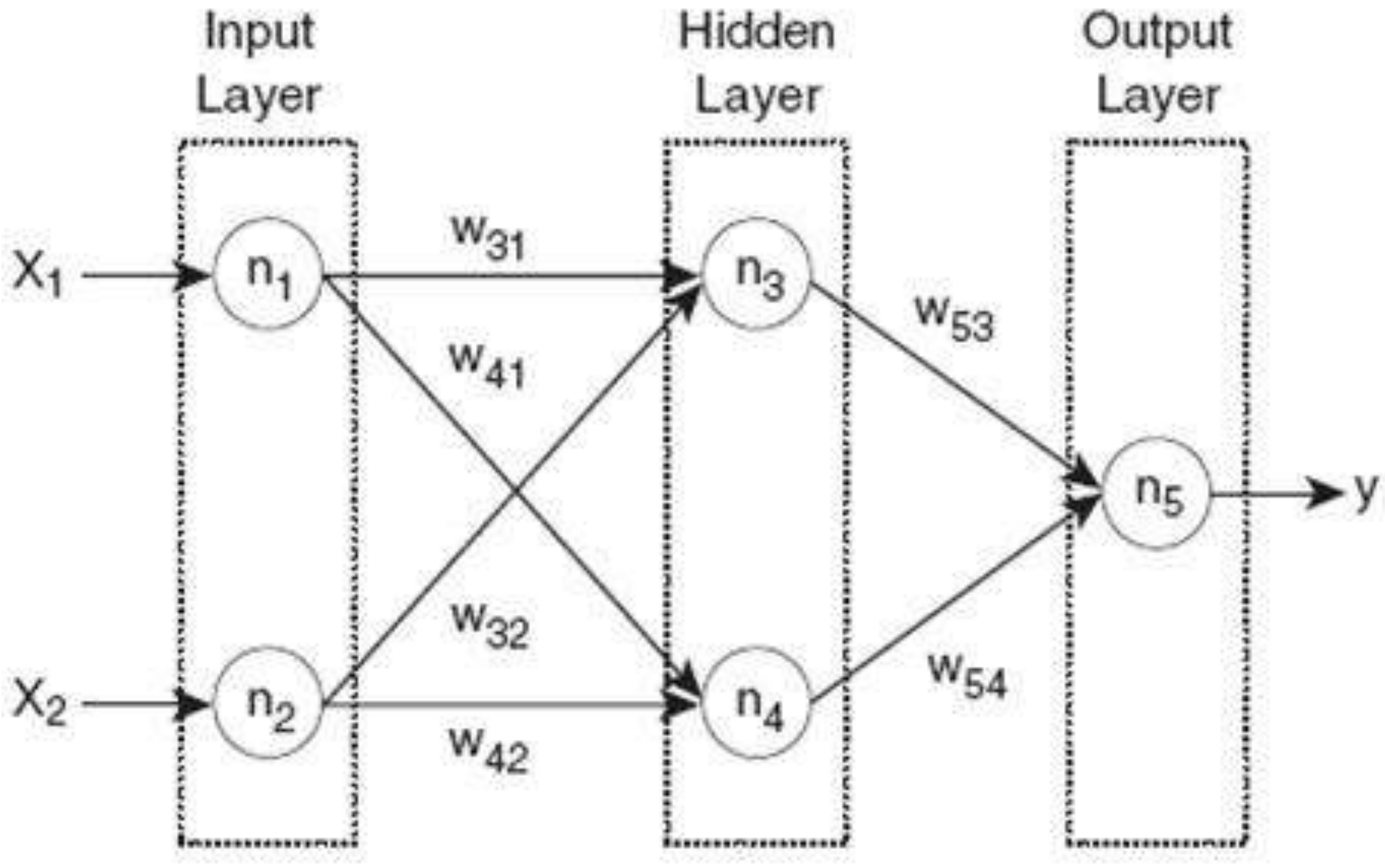
$$Err_j = O_j(1-O_j)(T_j - O_j)$$

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

$$I_j = \sum_i w_{ij} O_i + \theta_j$$



Example of a multilayer feed-forward artificial neural network (ANN).



A two-layer, feed-forward neural network

Advantages of Neural Network

- prediction accuracy is generally high
- robust, works when training examples contain errors
- output may be discrete, real-valued, or a vector of several discrete or real-valued attributes
- fast evaluation of the learned target function
- High tolerance to noisy data
- Ability to classify untrained patterns
- Well-suited for continuous-valued inputs and outputs
- Successful on a wide array of real-world data
- Algorithms are inherently parallel
- Techniques have recently been developed for the extraction of rules from trained neural networks

Disadvantages of Neural Network

- long training time
- difficult to understand the learned function (weights)
- not easy to incorporate domain knowledge
- 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

Association Rule

- Proposed by Agrawal et al in 1993.
- It is an important data mining model studied extensively by the database and data mining community.
- Assume all data are categorical.
- No good algorithm for numeric data.
- Initially used for Market Basket Analysis to find how items purchased by customers are related.
- Given a set of records each of which contain some number of items from a given collection;
  - Produce dependency rules which will predict occurrence of an item based on occurrences of other items.

TID	*Items*
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

Rules Discovered:

{Milk} --> {Coke}

{Diaper, Milk} --> {Beer}

Applications:

Basket data analysis, cross-marketing, catalog design, loss-leader analysis, clustering, classification, etc.

E.g., *98% of people who purchase tires and auto accessories also get automotive services done*

Concepts:

An *item*: an item/article in a basket
I: the set of all items sold in the store
A *transaction*: items purchased in a basket; it may have TID (transaction ID)
A *transactional dataset*: A set of transactions

The model: rules

A transaction *t* contains *X*, a set of items (itemset) in *I*, if $X \subseteq t$.
An association rule is an implication of the form:

$$X \rightarrow Y, \text{ where } X, Y \subset I, \textit{ and } X \cap Y = \emptyset$$

An itemset is a set of items.
E.g., X = {milk, bread, cereal} is an itemset.
A *k*-itemset is an itemset with *k* items.
E.g., {milk, bread, cereal} is a 3-itemset

Rule Strength Measures

Support:

The rule holds with support *sup* in *T* (the transaction data set) if sup% of transactions contain $X \cup Y$.

$sup = \Pr(X \cup Y)$

Confidence:

The rule holds in *T* with confidence *conf* if *conf*% of transactions that contain *X* also contain *Y*.

$conf = \Pr(Y \mid X)$

An association rule is a pattern that states when *X* occurs, *Y* occurs with certain probability.

Support and Confidence

- *support* of X in D is *count*(X)/|D|
- For an association rule $X \Rightarrow Y$, we can calculate

  support ($X \Rightarrow Y$) = support (XY)

  confidence ($X \Rightarrow Y$) = support (XY)/support (X)
- Relate Support (S) and Confidence (C) to Joint and Conditional probabilities
- There could be exponentially many A-rules
- Interesting association rules are (for now) those whose S and C are greater than minSup and minConf (some thresholds set by data miners)

Support and Confidence

Support count:

The support count of an itemset *X*, denoted by *X.count*, in a data set *T* is the number of transactions in *T* that contain *X*. Assume *T* has *n* transactions. Then,

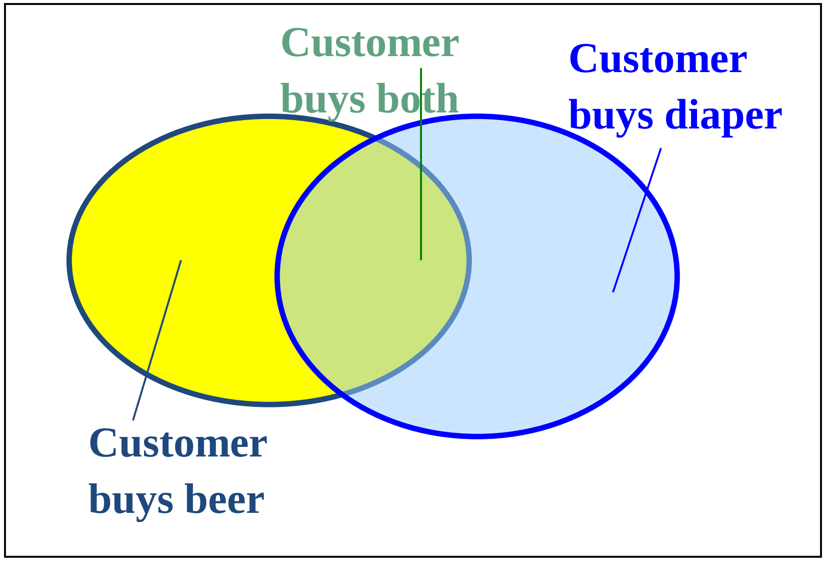
$$support = \frac{(X \cup Y).count}{n}$$

$$confidence = \frac{(X \cup Y).count}{X.count}$$

Basic Concepts: Association Rules

Transaction-id	Items bought
10	A, B, C
20	A, C
30	A, D
40	B, E, F

- Itemset X={x_1, …, x_k}
- Find all the rules *X→Y* with min confidence and support
  - support, *s*, probability that a transaction contains X∪Y
  - confidence, *c*, conditional probability that a transaction having X also contains *Y*.



Let minimum support 50%, and minimum confidence 50%, we have

$A \rightarrow C$ (50%, 66.7%)

$C \rightarrow A$ (50%, 100%)

Example

Data set *D*

TID	Itemsets
T100	1 3 4
T200	2 3 5
T300	1 2 3 5
T400	2 5

Count, Support, Confidence:

Count(13)=2

|D| = 4

Support(13)=0.5

Support(3→2)=0.5

Confidence(3→2)=0.67

Mining Association Rules: Example

Min. support 50%
Min. confidence 50%

Transaction-id	Items bought
10	A, B, C
20	A, C
30	A, D
40	B, E, F

Frequent pattern	Support
{A}	75%
{B}	50%
{C}	50%
{A, C}	50%

For rule $A \Rightarrow C$:

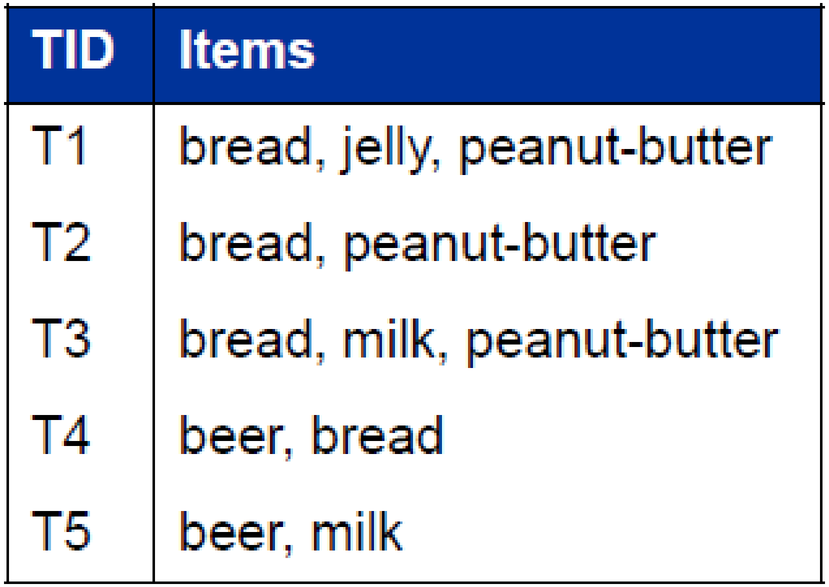
support = support($\{A\} \cup \{C\}$) = 50%

confidence = support($\{A\} \cup \{C\}$)/support($\{A\}$) = 66.6%

The Apriori principle:

Any subset of a frequent itemset must be frequent

Example of Association Rule



TID	Items
T1	bread, jelly, peanut-butter
T2	bread, peanut-butter
T3	bread, milk, peanut-butter
T4	beer, bread
T5	beer, milk

Examples:
bread ⇒ peanut-butter
beer ⇒ bread

Frequent itemsets: Items that frequently appear together

I = {bread, peanut-butter}

I = {beer, bread}

Support count (σ): Frequency of occurrence of and itemset

σ ({bread, peanut-butter}) = 3

σ ({ beer, bread}) = 1

Support: Fraction of transactions that contain an itemset

s ({bread,peanut-butter}) = 3/5

s ({beer, bread}) = 1/5

Frequent itemset: An itemset whose support is greater than or equal to a minimum support threshold (minsup)

What’s an Interesting Rule?

An association rule is an implication of two itemsets:

$$X \Rightarrow Y$$

Many measures of interest. The two most used are:

Support (s): The occurring frequency of the rule, i.e., number of transactions that contain both X and Y

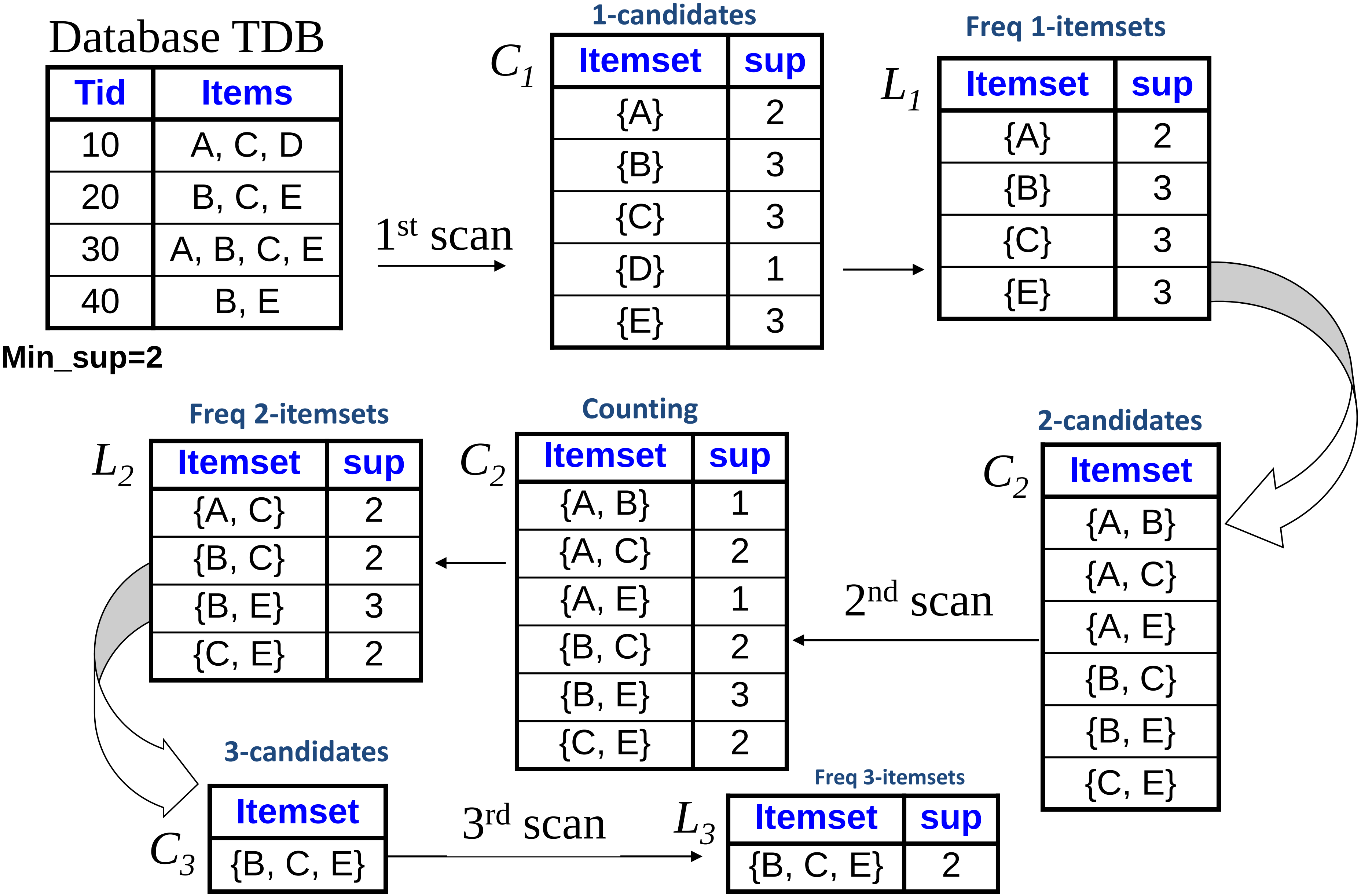
$$s = \frac{\sigma(X \cup Y)}{\#\text{ of trans.}}$$

Confidence (c): The strength of the association, i e measures of how often items (X)

$$c = \frac{\sigma(X \cup Y)}{\sigma(\mathrm{X})}$$

TID	s	c
bread ⇒ peanut-butter	0.60	0.75
peanut-butter ⇒ bread	0.60	1.00
beer ⇒ bread	0.20	0.50
peanut-butter ⇒ jelly	0.20	0.33
jelly ⇒ peanut-butter	0.20	1.00
jelly ⇒ milk	0.00	0.00

The Apriori Algorithm—An Example



Database TDB

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

Min_sup=2

1st scan

1-candidates

C_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

Freq 1-itemsets

L_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

2-candidates

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

2nd scan

Counting

C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

Freq 2-itemsets

L_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

3-candidates

C_3

Itemset
{B, C, E}

3rd scan

Freq 3-itemsets

L_3

Itemset	sup
{B, C, E}	2

Clustering and Cluster Analysis

A ***cluster*** is a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters.

Clustering is "the process of organizing objects into groups whose members are similar in some way".

*"**Cluster Analysis*** is a set of methods for constructing a (hopefully) sensible and informative classification of an initially unclassified set of data, using the variable values observed on each individual."

- B. S. Everitt (1998), "The Cambridge Dictionary of Statistics"

Applications of Cluster Analysis

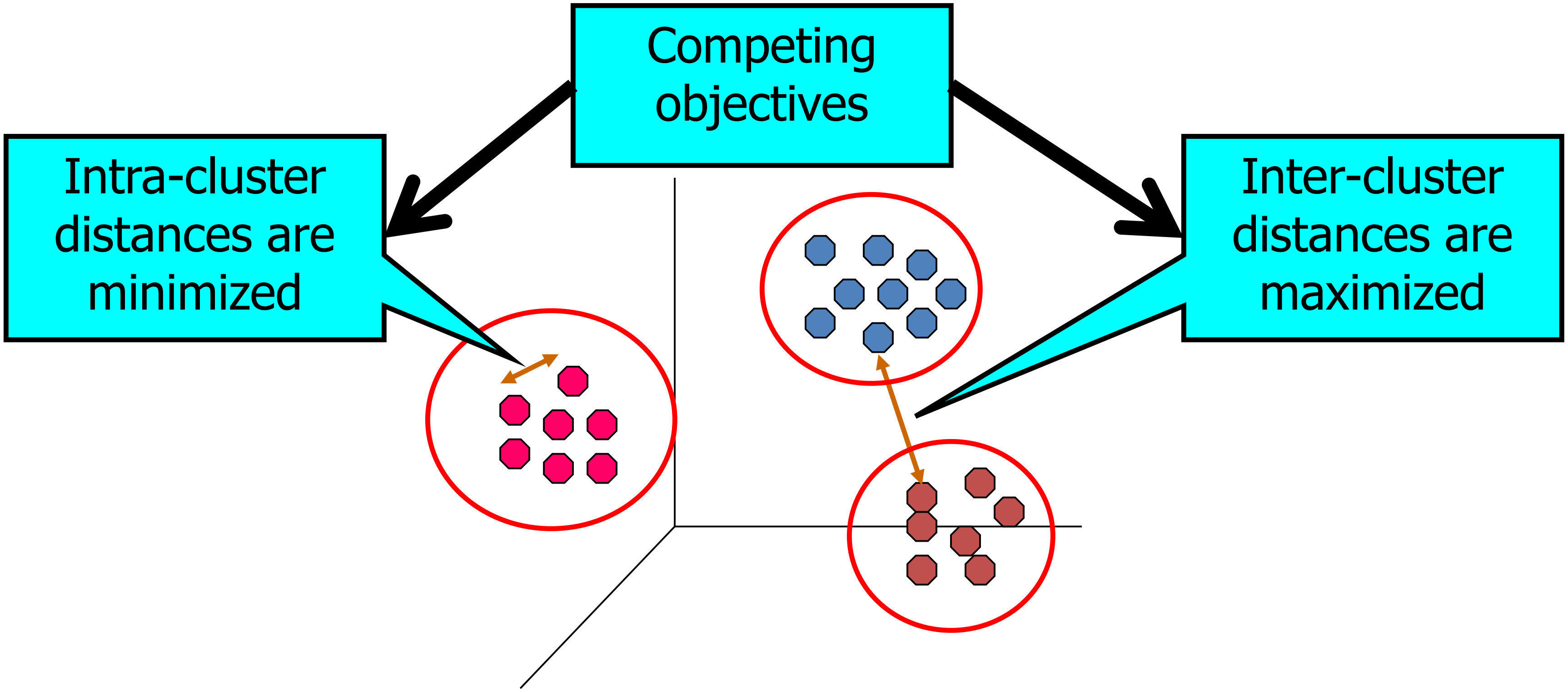
- Pattern Recognition
- Spatial Data Analysis
  - Create thematic maps in GIS by clustering feature spaces
  - Detect spatial clusters or for other spatial mining tasks
- Image Processing
- Economic Science (especially market research)
- WWW
  - Document classification
  - Cluster Weblog data to discover groups of similar access patterns

Applications of Cluster Analysis

- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults

Objectives of Cluster Analysis

Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups



Competing objectives

Intra-cluster distances are minimized

Inter-cluster distances are maximized

Types of Clusterings

❖ Partitioning Clustering

- A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
- Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
- Typical methods: k-means, k-medoids, CLARA (Clustering LARge Applications)

❖ Hierarchical clustering

- A set of nested clusters organized as a hierarchical tree
- Create a hierarchical decomposition of the set of data (or objects) using some criterion
- Typical methods: DiAna (Divisive Analysis), AgNes (Agglomerative Nesting), BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies), ROCK (RObust Clustering using linKs), CAMELEON

❖ Density-based Clustering

- Based on connectivity and density functions
- Typical methods: DBSACN (Density Based Spatial Clustering of Applications with Noise), OPTICS (Ordering Points To Identify the Clustering Structure), DenClue (DENsity-based CLUstEring)

❖ Grid-based Clustering

- based on a multiple-level granularity structure
- Typical methods: STING (STatistical INformation Grid), WaveCluster, CLIQUE (Clustering In QUEst)

❖ Model-based Clustering

- A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
- Typical methods: EM (Expectation Maximization), SOM (Self-Organizing Map), COBWEB

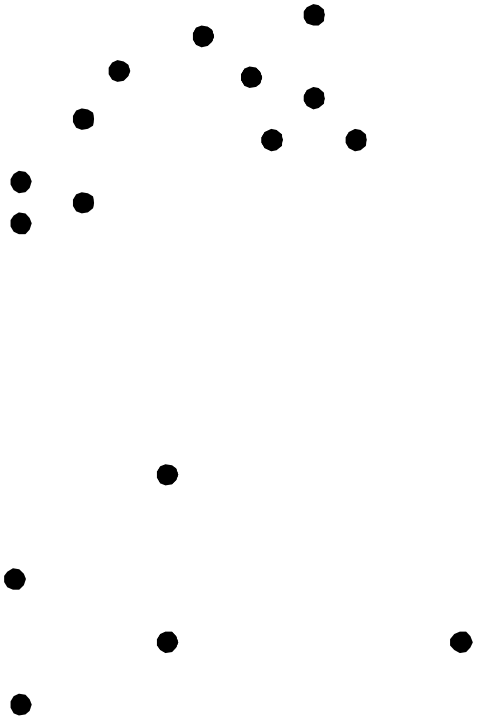
❖ Frequent pattern-based Clustering

- Based on the analysis of frequent patterns
- Typical methods: pCluster

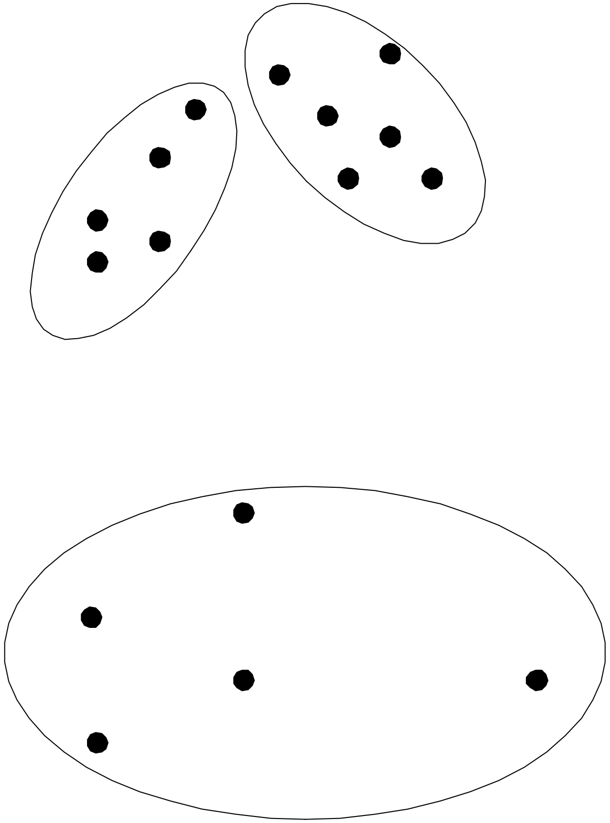
❖ User-guided or constraint-based Clustering

- Clustering by considering user-specified or application-specific constraints
- Typical methods: COD, constrained clustering

Partitioning Clustering

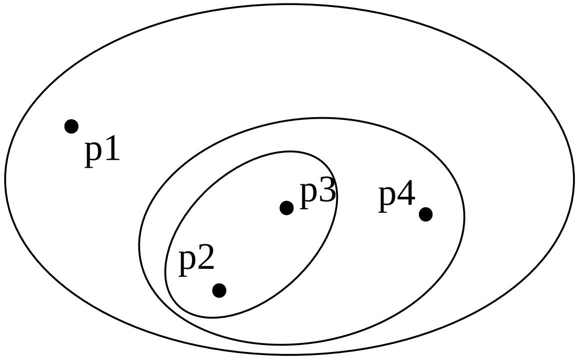


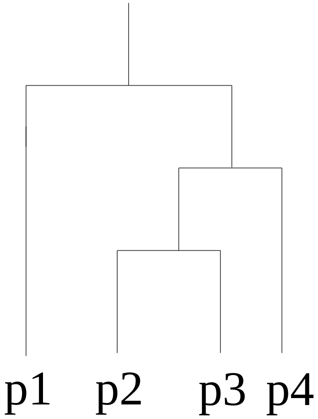
Original Points



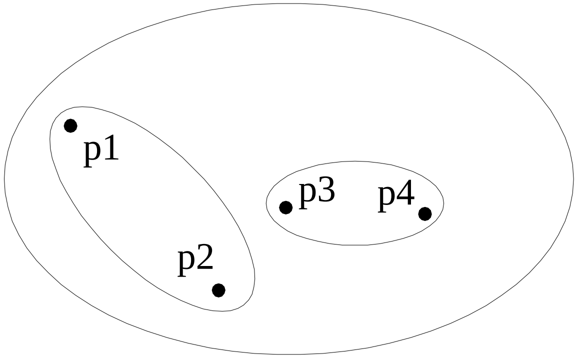
A Partitional Clustering

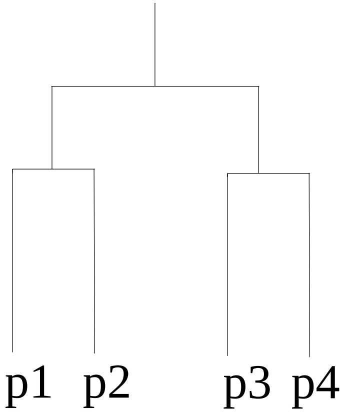
Hierarchical Clustering





Dendrogram 1





Dendrogram 2

Strengths of Hierarchical Clustering

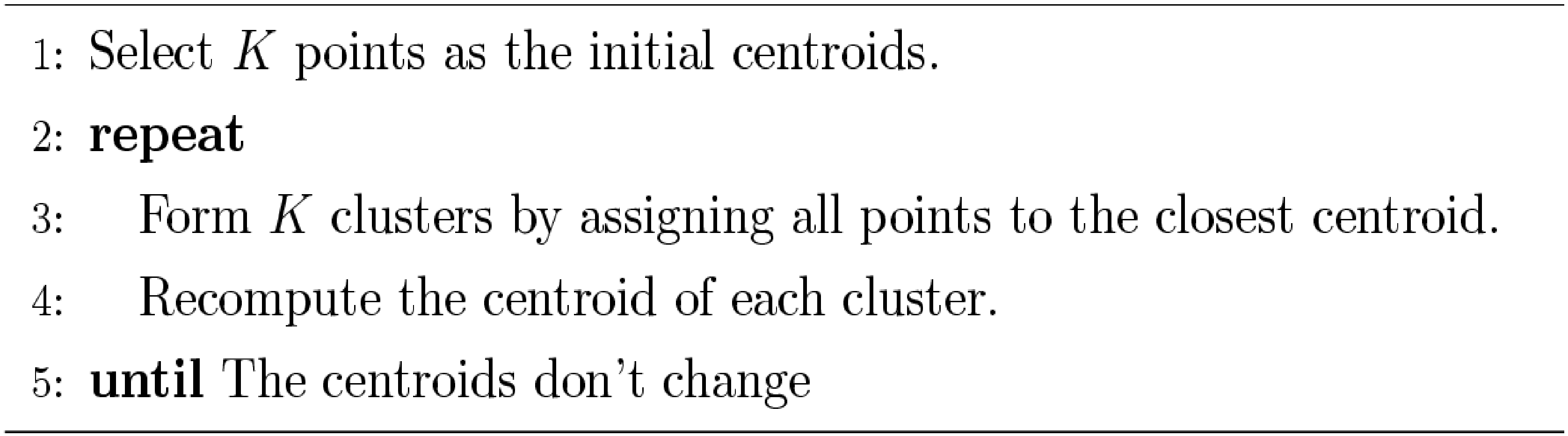
- ❖Do not have to assume any particular number of clusters
  - – Any desired number of clusters can be obtained by ‘cutting’ the dendogram at the proper level

- ❖They may correspond to meaningful taxonomies
  - – Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

K-means Algorithm

- Partitioning clustering approach
- Each cluster is associated with a centroid (center point or mean point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, K, must be specified

The basic algorithm is very simple:



```
Select $K$ points as the initial centroids.
repeat
    Form $K$ clusters by assigning all points to the closest centroid.
    Recompute the centroid of each cluster.
until The centroids don't change
```

The *k*-means partitioning algorithm.

Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:
k: the number of clusters,
D: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:
(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;
(2) repeat
(3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
(4) update the cluster means, i.e., calculate the mean value of the objects for each cluster;
(5) until no change;

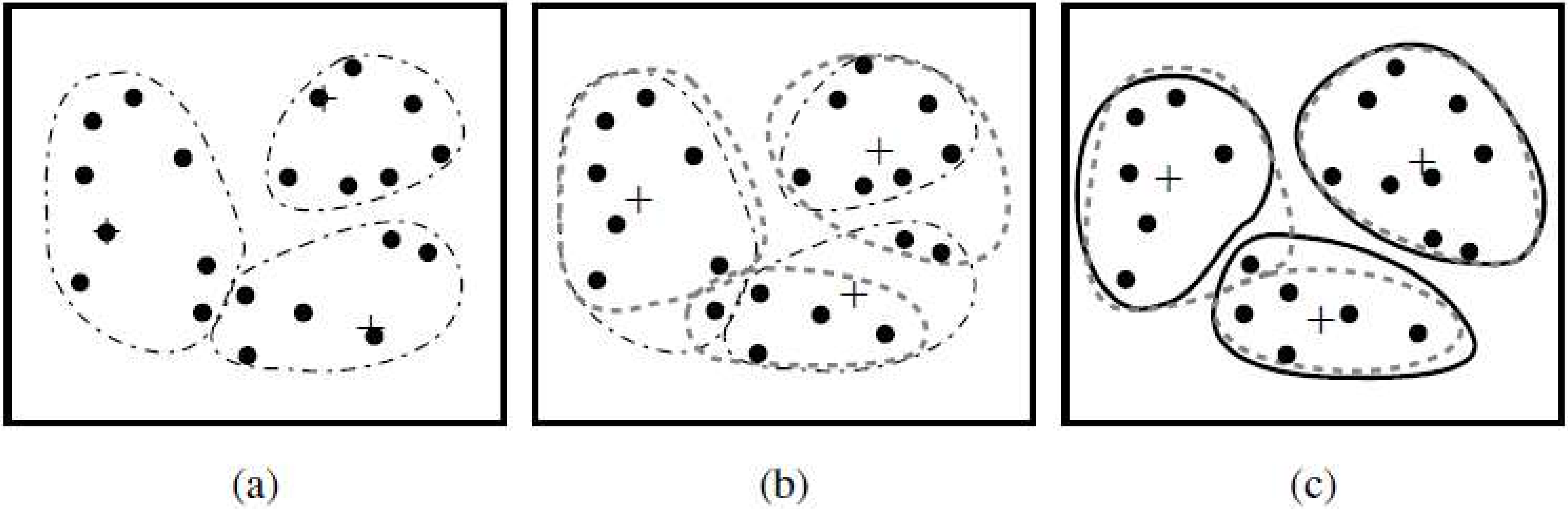
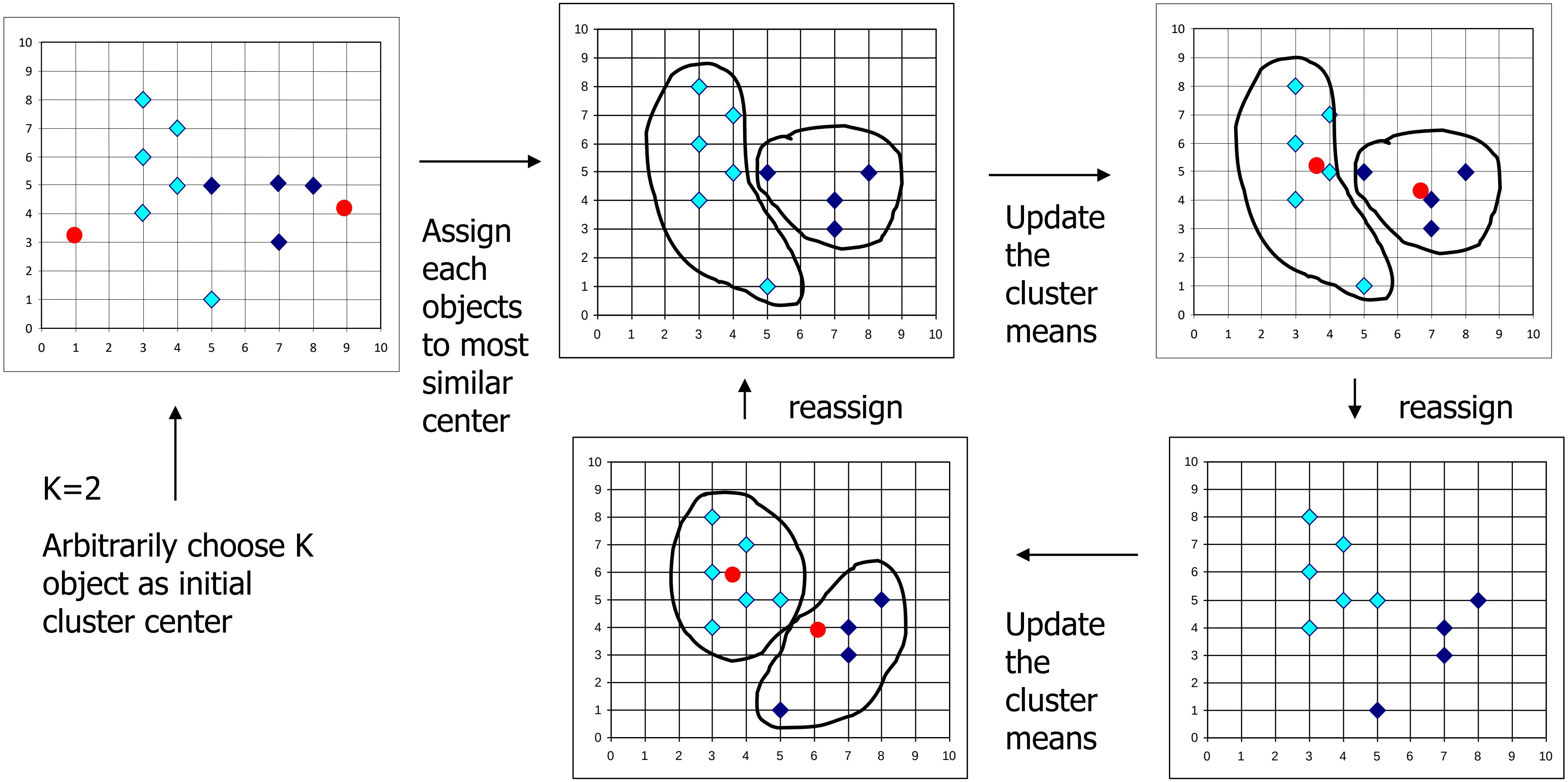


Figure: Clustering of a set of objects based on the *k*-means method. (The mean of each cluster is marked by a "+".)

Example



K=2

Arbitrarily choose K object as initial cluster center

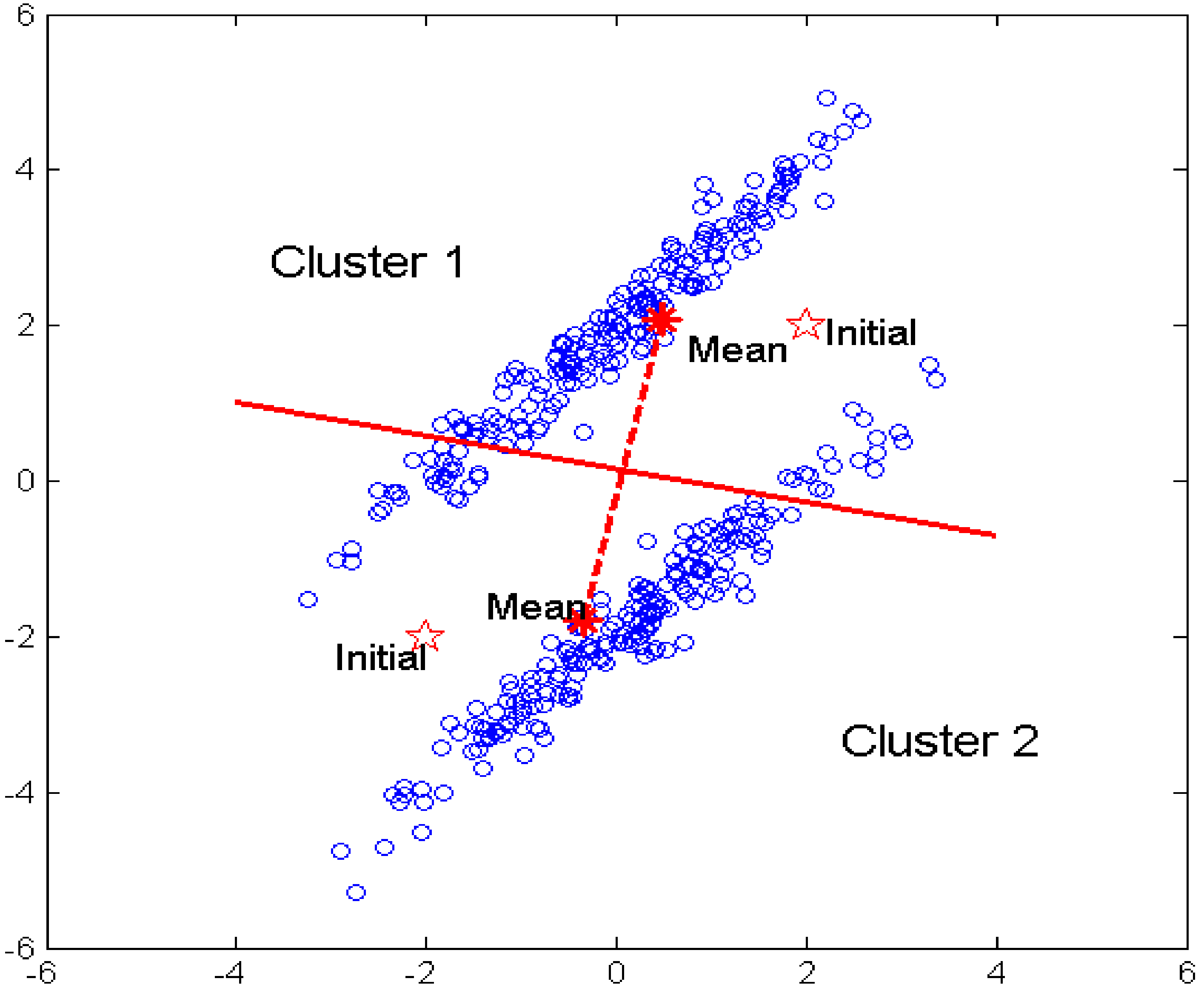
Assign each objects to most similar center

Update the cluster means

reassign

Update the cluster means

reassign


Cluster 1
Mean
Initial
Mean
Initial
Cluster 2
6
4
2
0
-2
-4
-6
-6
-4
-2
0
2
4
6

K-means Clustering – Details

- Initial centroids are often chosen randomly.
  - Clusters produced vary from one run to another.
- The centroid is (typically) the mean of the points in the cluster.
- ‘Closeness’ is measured mostly by Euclidean distance, cosine similarity, correlation, etc.

Typical choice

- K-means will converge for common similarity measures mentioned above.
- Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed to ‘Until relatively few points change clusters’
- Complexity is O(n * K * I * d)

  n = number of points, K = number of clusters,
  I = number of iterations, d = number of attributes

Issues and Limitations for K-means

- How to choose initial centers?
- How to choose K?
- How to handle Outliers?
- Clusters different in
    - Shape
    - Density
    - Size
- Assumes clusters are spherical in vector space
    - Sensitive to coordinate changes

K-means Algorithm

Pros

- Simple
- Fast for low dimensional data
- It can find pure sub clusters if large number of clusters is specified

Cons

- K-Means cannot handle non-globular data of different sizes and densities
- K-Means will not identify outliers
- K-Means is restricted to data which has the notion of a center (centroid)
- Applicable only when *mean* is defined, then what about categorical data?
- Need to specify *k,* the *number* of clusters, in advance
- Unable to handle noisy data and *outliers*
- Not suitable to discover clusters with *non-convex shapes*

Outliers

What are outliers?

The set of objects are considerably dissimilar from the remainder of the data

Example: Sports: Michael Jordon, Randy Orton, Sachin Tendulkar ...

Applications:

Credit card fraud detection

Telecom fraud detection

Customer segmentation

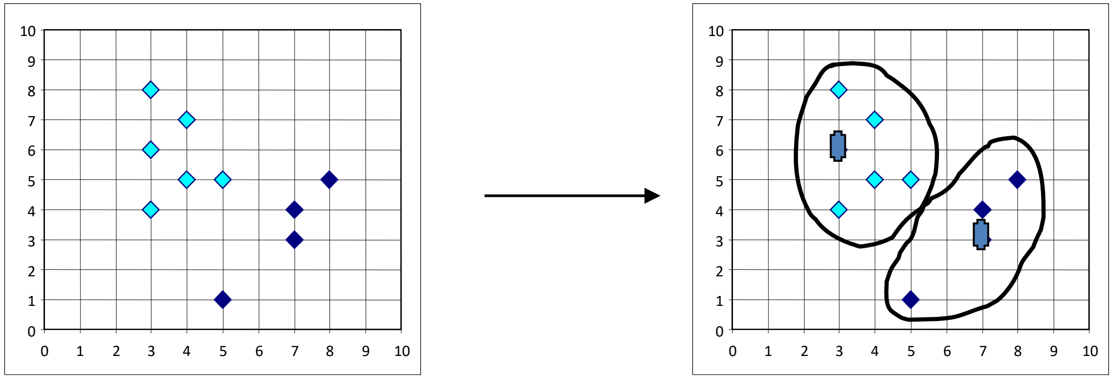
Medical analysis

Outlier detection and analysis are very useful for fraud detection, etc. and can be performed by statistical, distance-based or deviation-based approaches

How to handle Outliers?

- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data.

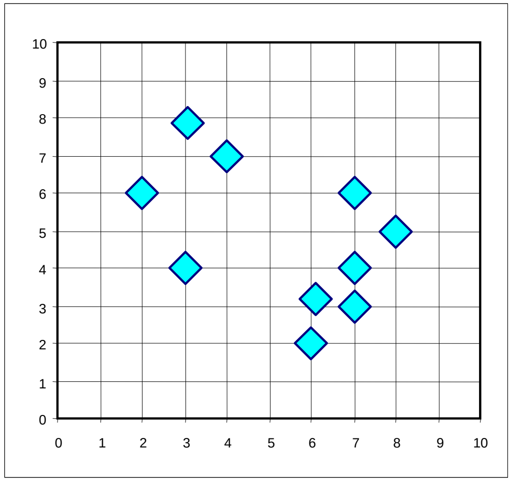
K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster.



Example:

Use in finding Fraudulent usage of credit cards. Outlier Analysis may uncover Fraudulent usage of credit cards by detecting purchases of extremely large amounts for a given account number in comparison to regular charges incurred by the same account. Outlier values may also be detected with respect to the location and type of purchase or the purchase frequency.

A Typical K-Medoids Algorithm (PAM)

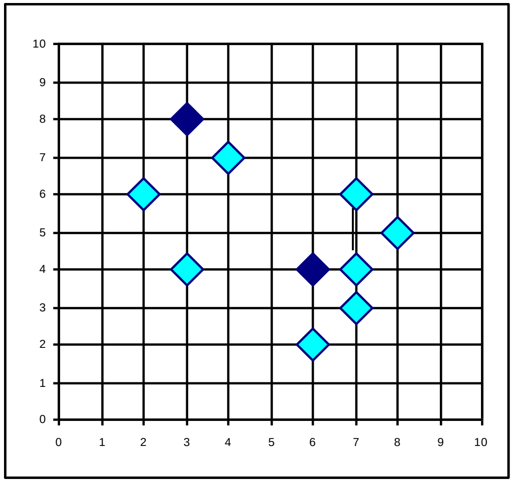


K=2

Do loop

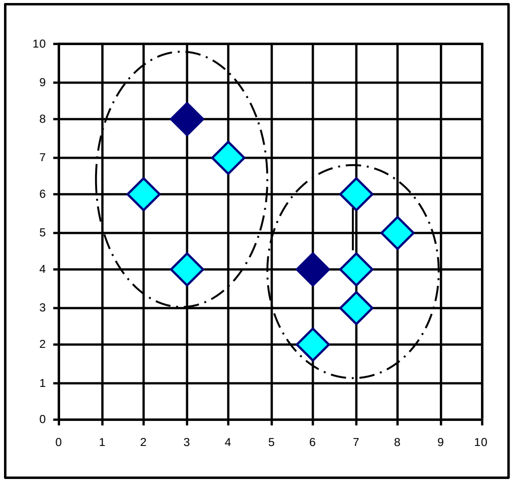
Until no change

Arbitrary choose k object as initial medoids

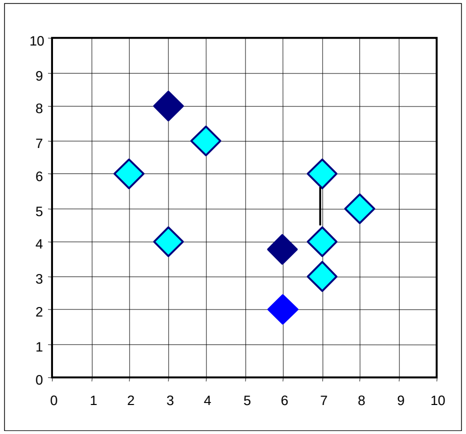


Assign each remaining object to nearest medoids

Total Cost = 20

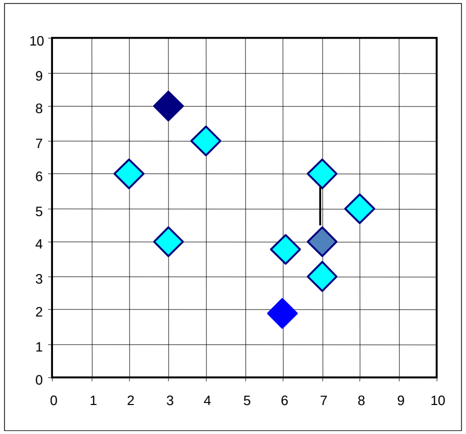


Randomly select a nonmedoid object,O_{ramdom}



Compute total cost of swapping

Total Cost = 26



Swapping O and O_{ramdom}

If quality is improved.

Example of K-medoids

Given the two medoids that are initially chosen are A and B. Based on the following table and randomly placing items when distances are identical to the two medoids, we obtain the clusters {A, C, D} and {B, E}. The three non-medoids {C, D, E} are examined to see which should be used to replace A or B. We have six costs to determine: TC_{AC} (the cost change by replacing medoid A with medoid C), TC_{AD}, TC_{AE}, TC_{BC}, TC_{BD} and TC_{BE}.

$TC_{AC}=C_{AAC}+C_{BAC}+C_{CAC}+C_{DAC}+C_{EAC} = 1 + 0 – 2 – 1 + 0 = -2$

Where C_{AAC} = the cost change of object A after replacing medoid A with medoid C

Comparison between K-means and K-medoids

The k-medoids method is more robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean. However, its processing is more costly than the k-means method. Both methods require the user to specify k, the number of clusters.

Thank you !!!