

# Data preprocessing



Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis, Tania Cerquitelli  
*Politecnico di Torino*



# Outline

- Data types and properties
- Data preparation
- Data preparation for document data
- Similarity and dissimilarity
- Correlation

# Data types and properties



Data Base and Data Mining Group of Politecnico di Torino



# What is Data?

- Collection of ***data objects*** and their ***attributes***
- An ***attribute*** is a property or characteristic of an object
  - Examples: eye color of a person, temperature, etc.
  - Attribute is also known as variable, field, characteristic, dimension, or feature
- A collection of attributes describes an ***object***
  - Object is also known as record, point, case, sample, entity, or instance

Objects

Attributes

Tid	Refund	Marital Status	Taxable Income	Cheat
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



# Attribute Values

- ***Attribute values*** are numbers or symbols assigned to an attribute for a particular object
- Distinction between attributes and attribute values
  - Same attribute can be mapped to different attribute values
    - Example: height can be measured in feet or meters
  - Different attributes can be mapped to the same set of values
    - Example: Attribute values for ID and age are integers
    - But properties of attribute values can be different



# Attribute types

- There are different types of attributes
  - Nominal
    - Examples: ID numbers, eye color, zip codes
  - Ordinal
    - Examples: rankings (e.g., taste of potato chips on a scale from 1-10), grades, height in {tall, medium, short}
  - Interval
    - Examples: calendar dates
  - Ratio
    - Examples: temperature in Kelvin, length, time, counts



# Properties of Attribute Values

- The type of an attribute depends on which of the following properties it possesses:
  - Distinctness:                          $= \neq$
  - Order:                                   $< >$
  - Addition:                               $+ -$
  - Multiplication:                       $* /$
  
- Nominal attribute: distinctness
- Ordinal attribute: distinctness & order
- Interval attribute: distinctness, order & addition
- Ratio attribute: all 4 properties



# Discrete and Continuous Attributes

## ■ Discrete Attribute

- Has only a finite or countably infinite set of values
- Examples: zip codes, counts, or the set of words in a collection of documents
- Often represented as integer variables.
- Note: **binary attributes** are a special case of discrete attributes

## ■ Continuous Attribute

- Has real numbers as attribute values
- Examples: temperature, height, or weight.
- Practically, real values can only be measured and represented using a finite number of digits.
- Continuous attributes are typically represented as floating-point variables.



# More Complicated Examples

- ID numbers
  - Nominal, ordinal, or interval?
- Number of cylinders in an automobile engine
  - Nominal, ordinal, or ratio?



# Key Messages for Attribute Types

- The types of operations you choose should be “meaningful” for the type of data you have
  - Distinctness, order, meaningful intervals, and meaningful ratios are only four properties of data
  - The data type you see – often numbers or strings – may not capture all the properties or may suggest properties that are not there
  - Analysis may depend on these other properties of the data
    - Many statistical analyses depend only on the distribution
  - Many times what is meaningful is measured by statistical significance
  - But in the end, what is meaningful is measured by the domain



# Data set types

- Record
  - Tables
  - Document Data
  - Transaction Data
- Graph
  - World Wide Web
  - Molecular Structures
- Ordered
  - Spatial Data
  - Temporal Data
  - Sequential Data
  - Genetic Sequence Data



# Tabular Data

- A collection of records
  - Each record is characterized by a fixed set of attributes

<i>Tid</i>	Refund	Marital Status	Taxable Income	Cheat
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



# Document data

- It includes **textual data** that can be **semi-structured** or **unstructured**
  - Plain text can be organized in sentences, paragraphs, sections, documents
- Text acquired in different contexts may have a structure and/or a semantics
  - **Web pages** are enriched with tags
  - **Documents in digital libraries** are enriched with metadata
  - **E-learning documents** can be annotated or partly highlighted



# Transaction Data

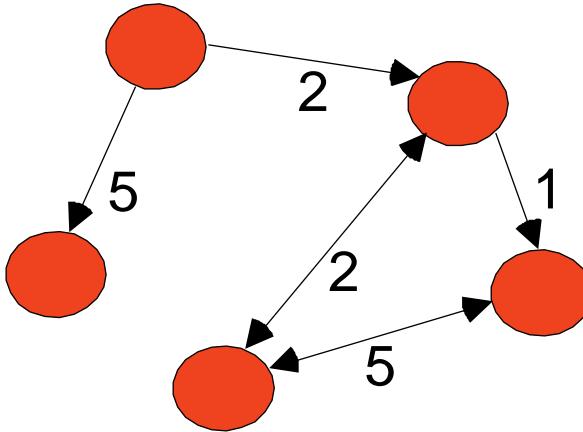
- A special type of record data, where
  - each record (transaction) involves a set of items.
  - For example, consider a grocery store. The set of products purchased by a customer during one shopping trip constitute a transaction, while the individual products that were purchased are the items.

<i>TID</i>	<i>Items</i>
1	<b>Bread, Coke, Milk</b>
2	<b>Beer, Bread</b>
3	<b>Beer, Coke, Diaper, Milk</b>
4	<b>Beer, Bread, Diaper, Milk</b>
5	<b>Coke, Diaper, Milk</b>



# Graph Data

- Examples: Generic graph, a molecule, and webpages



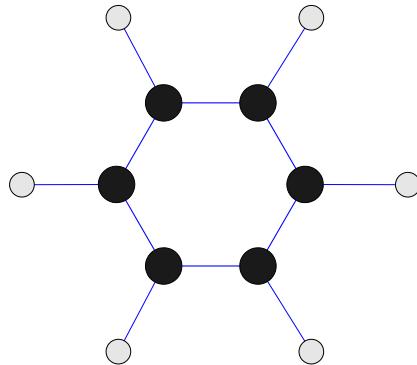
## Useful Links:

- [Bibliography](#)
- Other Useful Web sites
  - [ACM SIGKDD](#)
  - [KDnuggets](#)
  - [The Data Mine](#)

## Knowledge Discovery and Data Mining Bibliography

(Gets updated frequently, so visit often!)

- [Books](#)
- [General Data Mining](#)



Benzene Molecule: C<sub>6</sub>H<sub>6</sub>



# Ordered Data

- Sequences of transactions

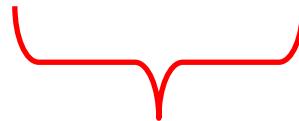
Items/Events



( A B) (D) (C E)

( B D) (C) (E)

( C D) (B) (A E)



An element of  
the sequence



# Ordered Data

- Genomic sequence data

```
GGTTCCGCCTTCAGCCCCGCGCC  
CGCAGGGCCCAGCCCCGCCGCCGTC  
GAGAAGGGCCCGCCTGGCGGGCG  
GGGGGAGGCAGGGGCCGCCGAGC  
CCAACCGAGTCCGACCAAGGTGCC  
CCCTCTGCTCGGCCTAGACCTGA  
GCTCATTAGGCAGCAGCGGACAG  
GCCAAGTAGAACACCGCGAAGCGC  
TGGGCTGCCTGCTGCGACCAGGG
```

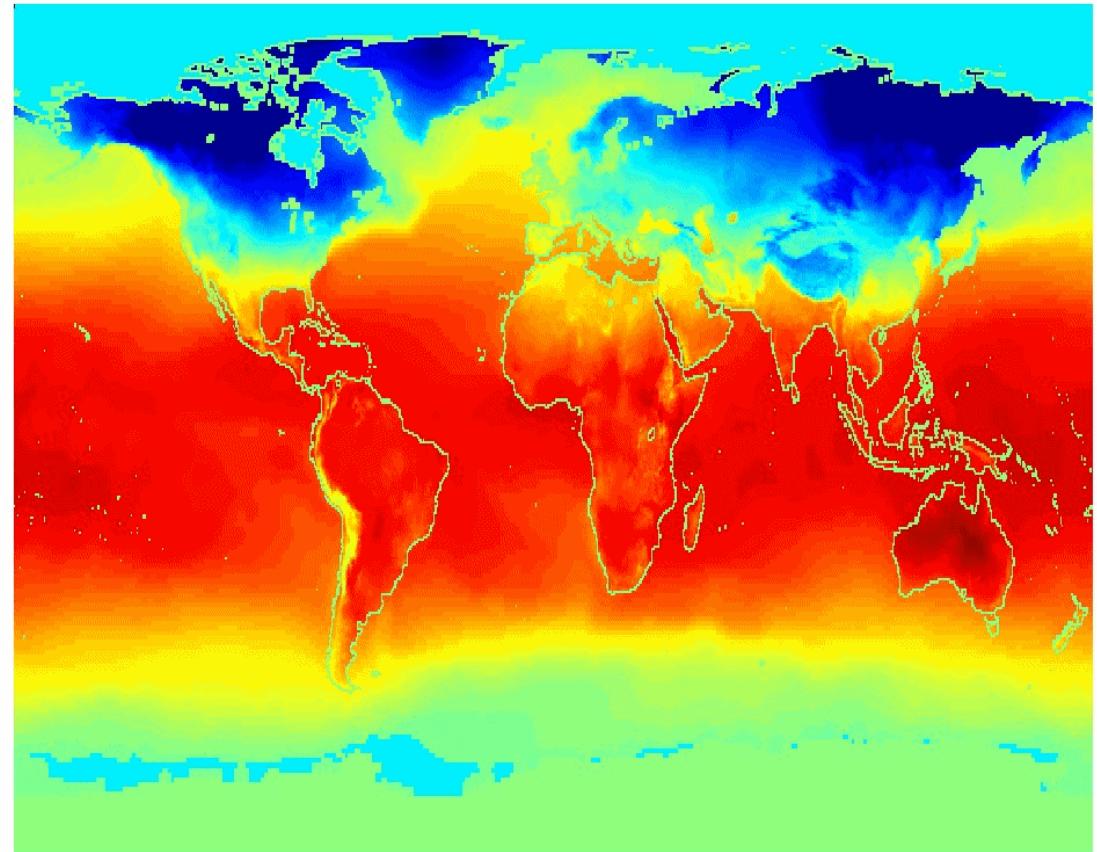


# Ordered Data

- Spatio-Temporal Data

Average Monthly Temperature of land and ocean

Jan





# Data Quality

- Poor data quality negatively affects many data processing efforts

"The most important point is that poor data quality is an unfolding disaster. Poor data quality costs the typical company at least ten percent (10%) of revenue; twenty percent (20%) is probably a better estimate."

Thomas C. Redman, DM Review, August 2004

- Data mining example: a classification model for detecting people who are loan risks is built using poor data
  - Some credit-worthy candidates are denied loans
  - More loans are given to individuals that default



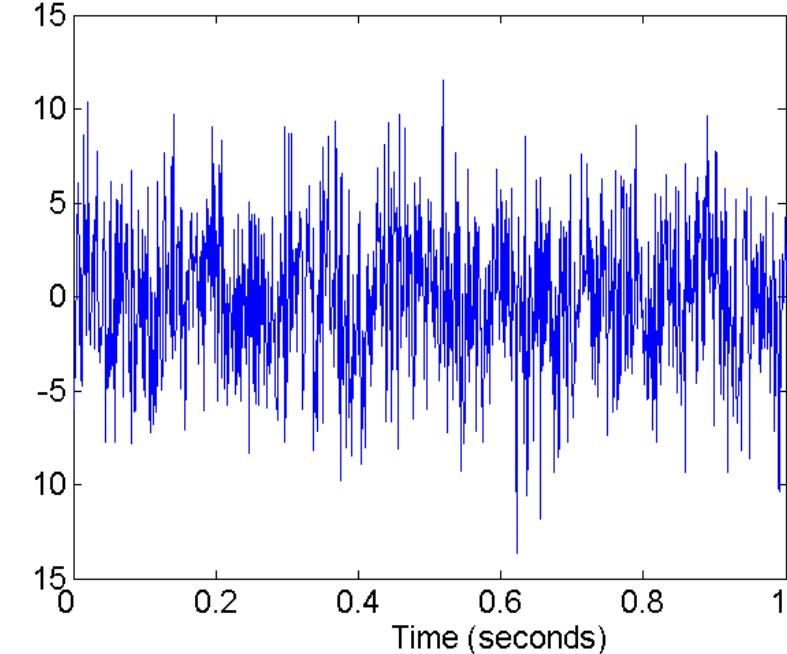
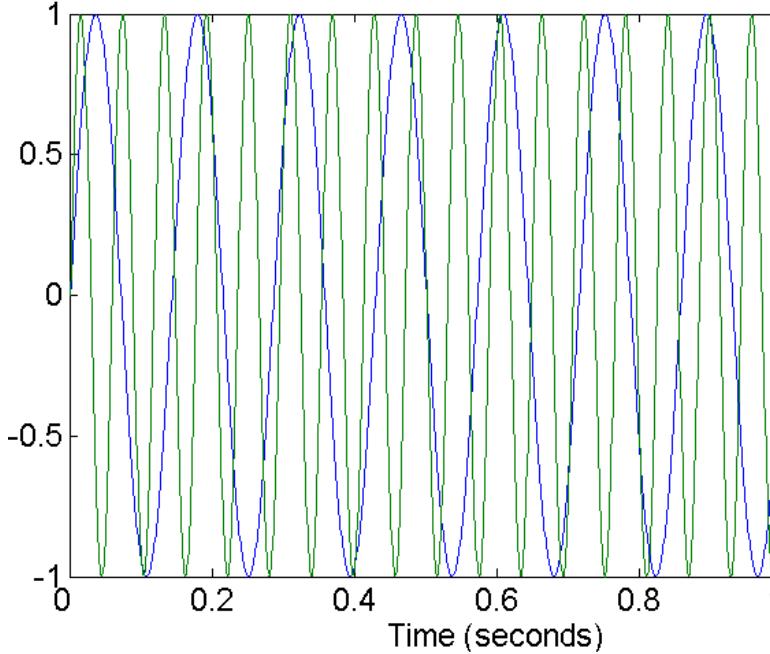
# Data Quality

- What kinds of data quality problems?
- How can we detect problems with the data?
- What can we do about these problems?
  
- Examples of data quality problems
  - Noise and outliers
  - Missing values
  - Duplicate data
  - Wrong data



# Noise

- Noise refers to modification of original values
  - Examples: distortion of a person's voice when talking on a poor phone and "snow" on television screen



Two Sine Waves

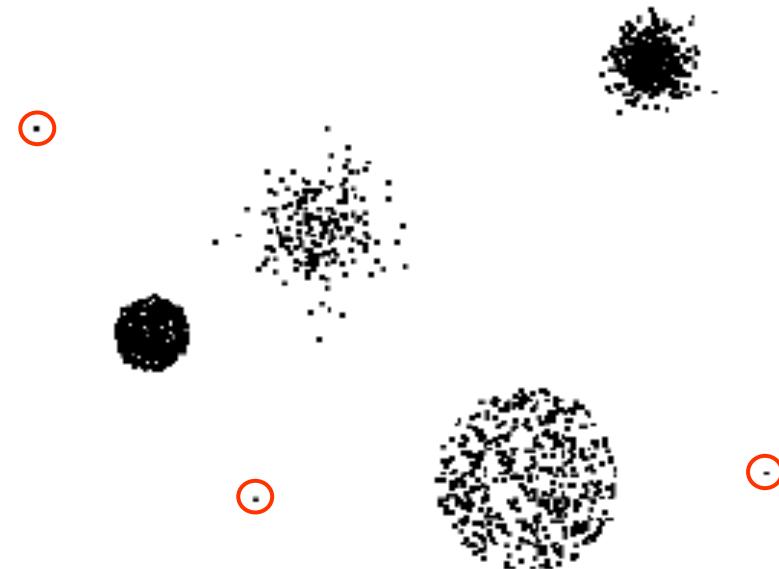
Two Sine Waves + Noise

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



# Outliers

- **Outliers** are data objects with characteristics that are considerably different than most of the other data objects in the data set
  - **Case 1:** Outliers are noise that interferes with data analysis
  - **Case 2:** Outliers are the goal of our analysis
    - Credit card fraud
    - Intrusion detection





# Missing Values

- Reasons for missing values
  - Information is not collected (e.g., people decline to give their age and weight)
  - Attributes may not be applicable to all cases (e.g., annual income is not applicable to children)
- Handling missing values
  - Eliminate data objects or variables
  - Estimate missing values
    - Example: time series of temperature
  - Ignore the missing value during analysis



# Duplicate Data

- Data set may include data objects that are duplicates, or almost duplicates of one another
  - Major issue when merging data from heterogeneous sources
- Examples
  - Different words/abbreviations for the same concept (e.g., Street, St.)
- Data cleaning
  - Process of dealing with duplicate data issues
- When should duplicate data not be removed?

# Data preparation



Data Base and Data Mining Group of Politecnico di Torino



# Data Preprocessing

- Aggregation
- Sampling
- Dimensionality Reduction
- Feature subset selection
- Feature creation
- Discretization and Binarization
- Attribute Transformation



# Aggregation

- Combining two or more attributes (or objects) into a single attribute (or object)
- Purpose
  - Data reduction
    - Reduce the number of attributes or objects
  - Change of scale
    - Cities aggregated into regions, states, countries, etc
  - More “stable” data
    - Aggregated data tends to have less variability



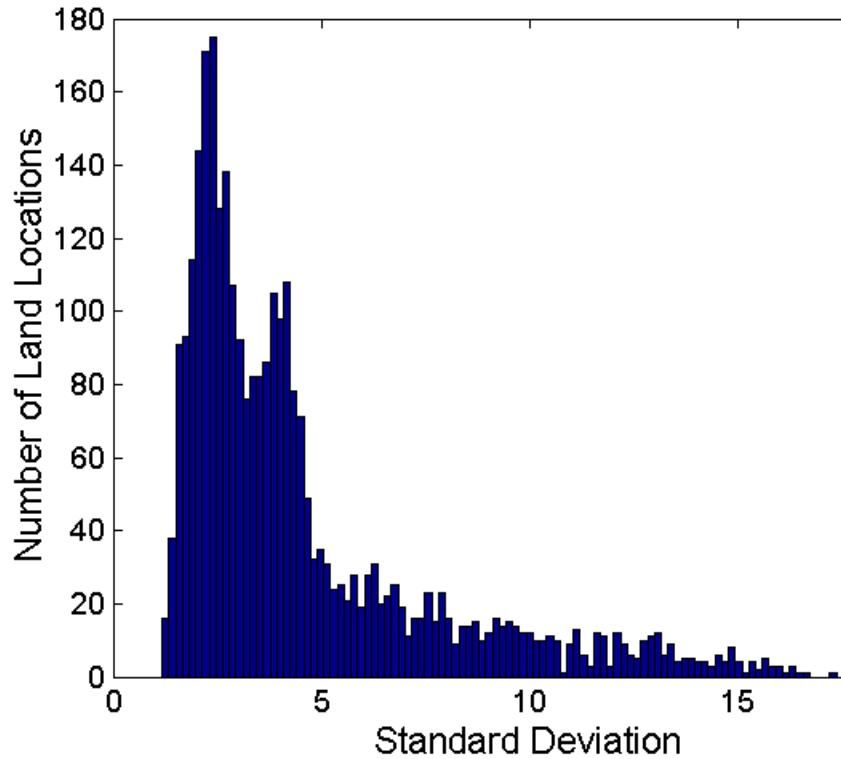
# Example: Precipitation in Australia

- The next slide shows precipitation in Australia from the period 1982 to 1993
  - A histogram for the standard deviation of average monthly precipitation for 3,030  $0.5^\circ$  by  $0.5^\circ$  grid cells in Australia
  - A histogram for the standard deviation of the average yearly precipitation for the same locations.
- The average yearly precipitation has less variability than the average monthly precipitation.
- All precipitation measurements (and their standard deviations) are in centimeters.

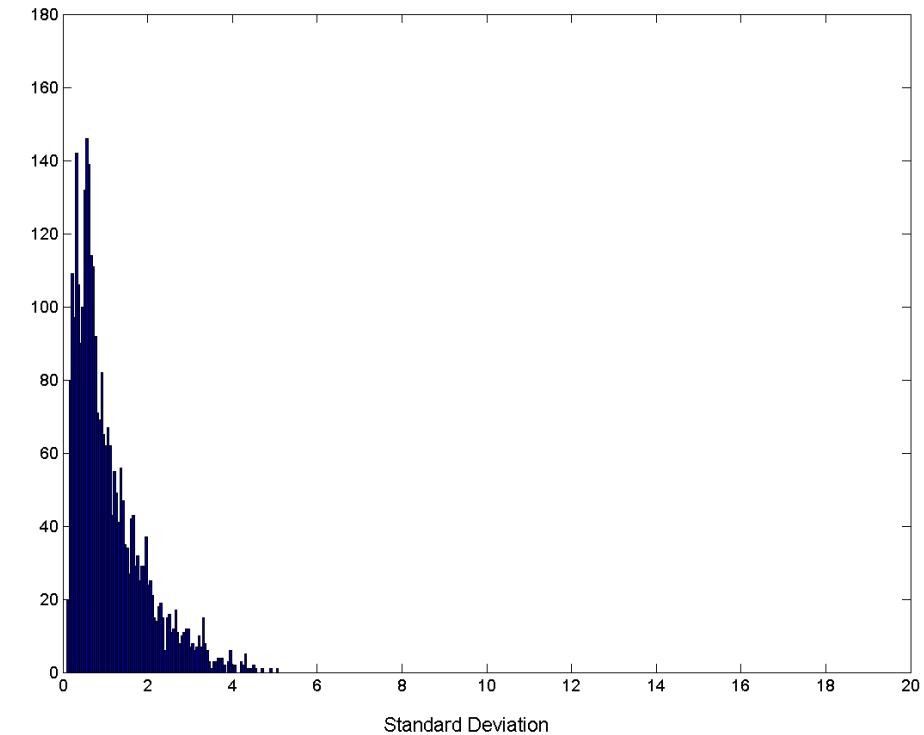


# Aggregation

## Variation of Precipitation in Australia



Standard Deviation of Average  
Monthly Precipitation



Standard Deviation of Average  
Yearly Precipitation



# Data reduction

- Generates a reduced representation of the dataset
- This representation is smaller in volume, but it can provide similar analytical results
  - sampling
    - reduces the cardinality of the set
  - feature selection
    - reduces the number of attributes
  - discretization
    - reduces the cardinality of the attribute domain



# Sampling

- Sampling is the main technique employed for data selection.
  - It is often used for both the preliminary investigation of the data and the final data analysis.
- Statisticians sample because **obtaining** the entire set of data of interest is too expensive or time consuming.
- Sampling is used in data mining because **processing** the entire set of data of interest is too expensive or time consuming.

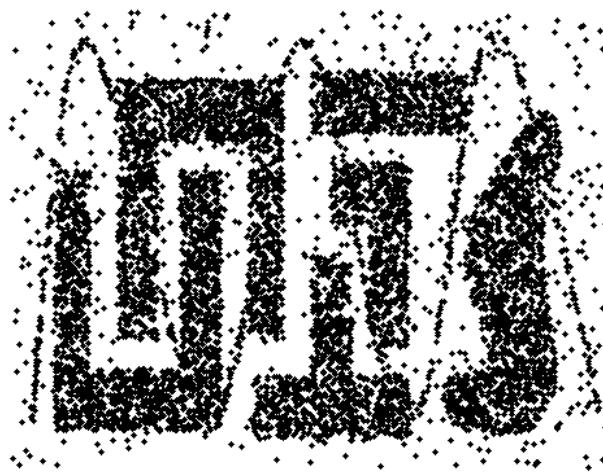


# Sampling ...

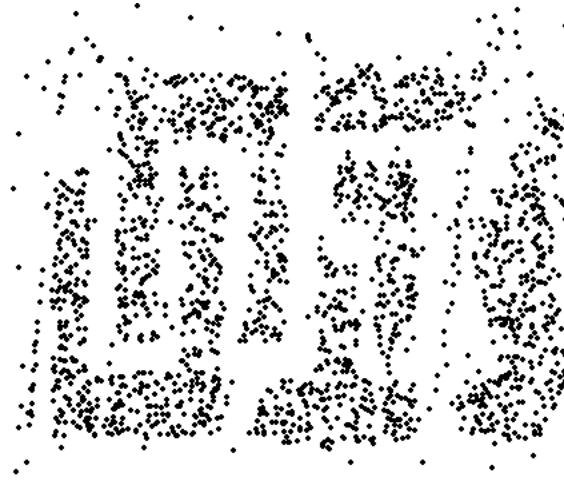
- The key principle for effective sampling is the following:
  - using a sample will work almost as well as using the entire data set, if the sample is representative
  - A sample is representative if it has approximately the same property (of interest) as the original set of data



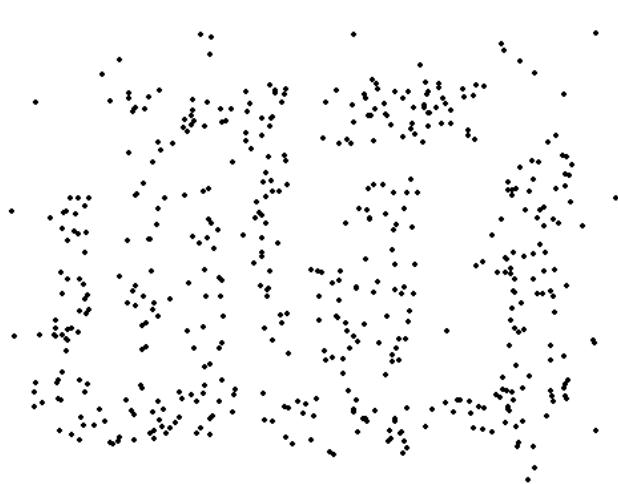
# Sample Size: examples



**8000 points**



**2000 Points**



**500 Points**



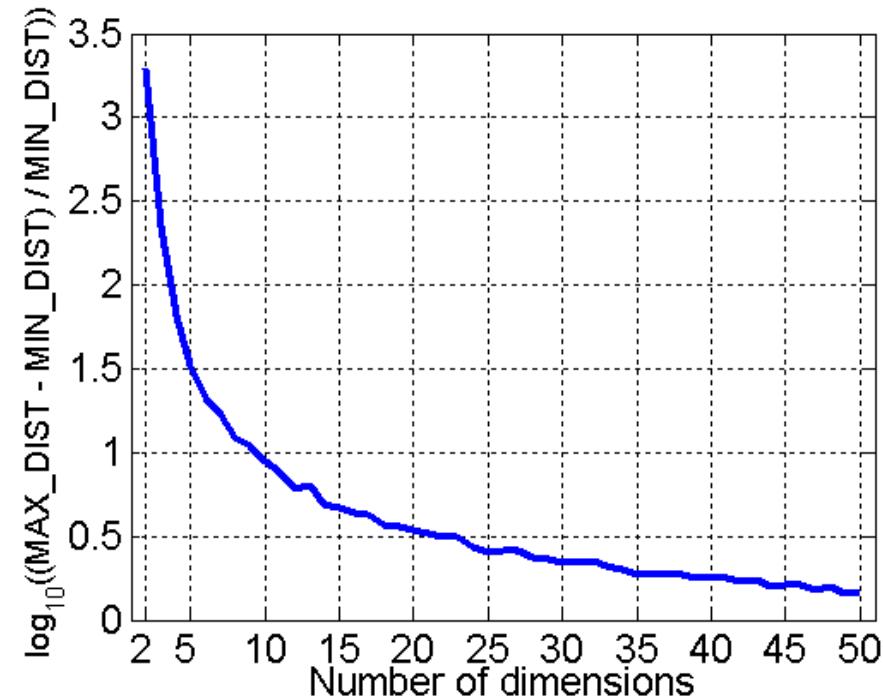
# Types of Sampling

- Simple Random Sampling
  - There is an equal probability of selecting any particular item
  - Sampling without replacement
    - As each item is selected, it is removed from the population
  - Sampling with replacement
    - Objects are not removed from the population as they are selected for the sample.
    - In sampling with replacement, the same object can be picked up more than once
- Stratified sampling
  - Split the data into several partitions; then draw random samples from each partition



# Curse of Dimensionality

- When dimensionality increases, data becomes increasingly sparse in the space that it occupies
- Definitions of density and distance between points, which is critical for clustering and outlier detection, become less meaningful



- Randomly generate 500 points
- Compute difference between max and min distance between any pair of points



# Dimensionality Reduction

## ■ Purpose

- Avoid curse of dimensionality
- Reduce amount of time and memory required by data mining algorithms
- Allow data to be more easily visualized
- May help to eliminate irrelevant features or reduce noise

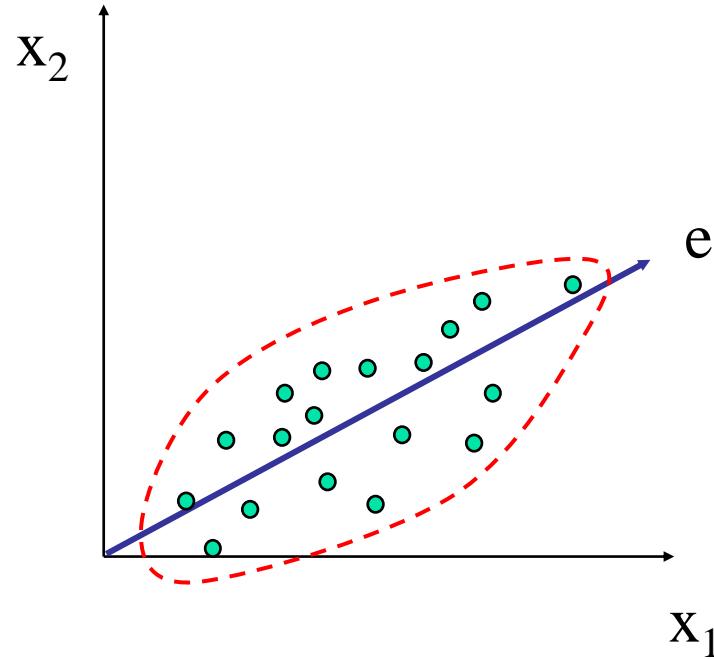
## ■ Techniques

- Principal Component Analysis (PCA)
- Singular Value Decomposition
- Others: supervised and non-linear techniques



# Dimensionality Reduction: PCA

- Goal is to find a projection that captures the largest amount of variation in data





# Dimensionality Reduction: PCA





# Feature Subset Selection

- Another way to reduce dimensionality of data
- Redundant features
  - duplicate much or all of the information contained in one or more other attributes
  - Example: purchase price of a product and the amount of sales tax paid
- Irrelevant features
  - contain no information that is useful for the data mining task at hand
  - Example: students' ID is irrelevant to the task of predicting students' GPA



# Feature Subset Selection

- Techniques
  - Brute-force approach
    - Try all possible feature subsets as input to data mining algorithm
  - Embedded approaches
    - Feature selection occurs naturally as part of the data mining algorithm
  - Filter approaches
    - Features are selected before data mining algorithm is run
  - Wrapper approaches
    - Use the data mining algorithm as a black-box to find best subset of attributes



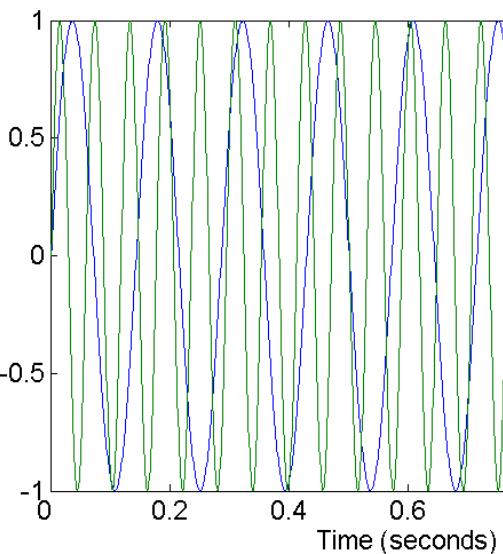
# Feature Creation

- Create new attributes that can capture the important information in a data set much more efficiently than the original attributes
- Three general methodologies
  - Feature Extraction
    - domain-specific
  - Mapping Data to New Space
  - Feature Construction
    - combining features

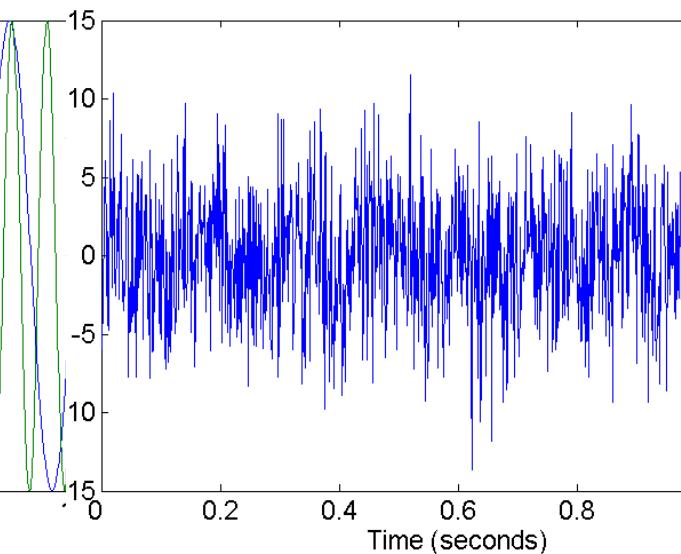


# Mapping Data to a New Space

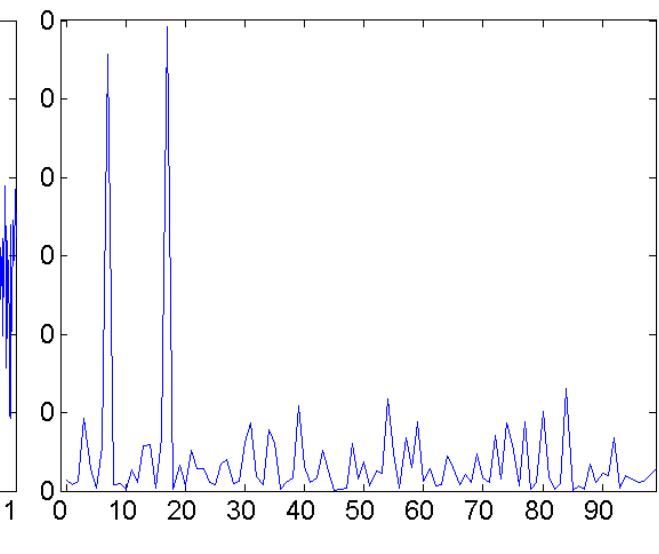
- Fourier transform
- Wavelet transform



Two Sine Waves



Two Sine Waves + Noise



Frequency



# Discretization

- **Discretization** is the process of converting a continuous attribute into an ordinal attribute
  - A potentially infinite number of values are mapped into a small number of categories
  - Discretization is commonly used in classification
    - Many classification algorithms work best if both the independent and dependent variables have only a few values



# Iris Sample Data Set

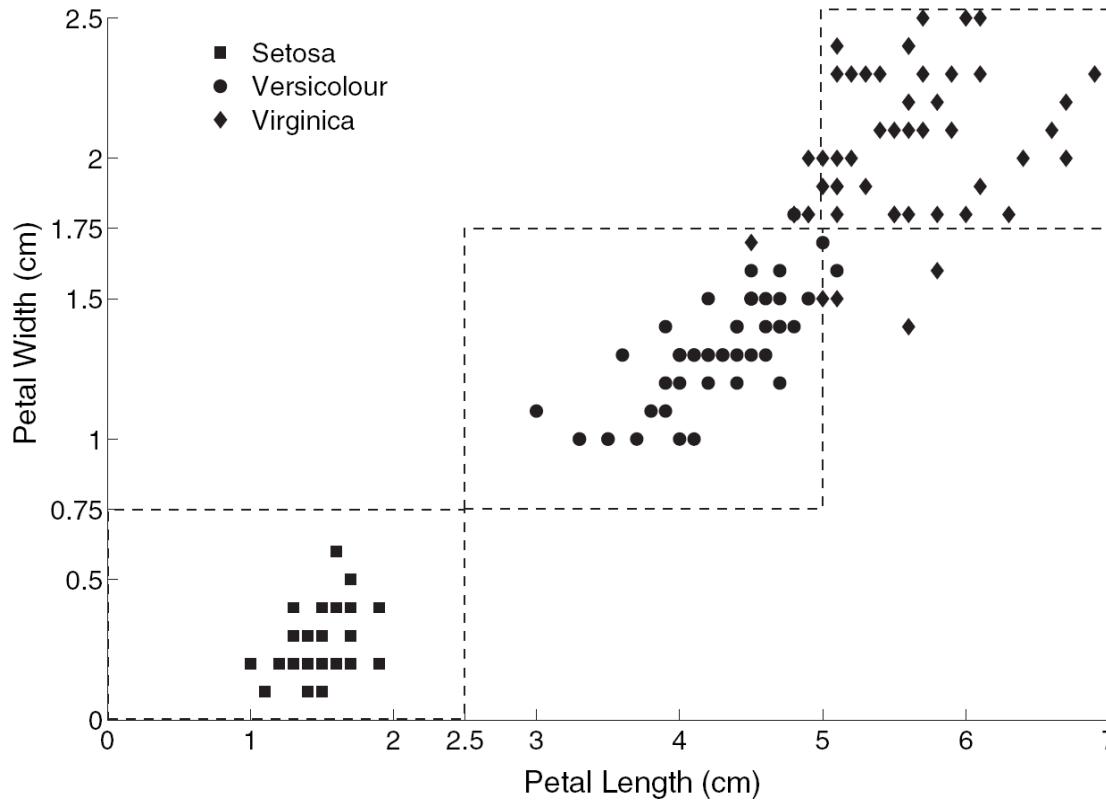
- Iris Plant data set
  - Can be obtained from the UCI Machine Learning Repository  
<http://www.ics.uci.edu/~mlearn/MLRepository.html>
  - From the statistician Douglas Fisher
  - Three flower types (classes)
    - Setosa
    - Versicolour
    - Virginica
  - Four (non-class) attributes
    - Sepal width and length
    - Petal width and length



Virginica. Robert H. Mohlenbrock. USDA NRCS. 1995. Northeast wetland flora: Field office guide to plant species. Northeast National Technical Center, Chester, PA. Courtesy of USDA NRCS Wetland Science Institute.



# Discretization: Iris Example



Petal width low or petal length low implies Setosa.

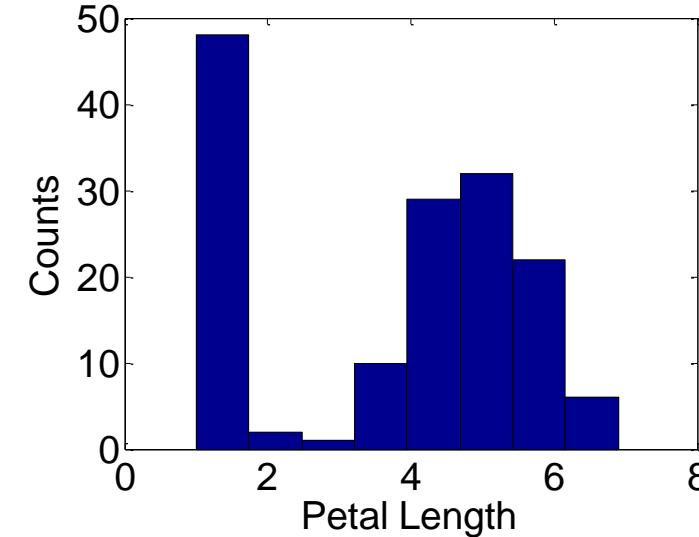
Petal width medium or petal length medium implies Versicolour.

Petal width high or petal length high implies Virginica.



# Discretization: Iris Example ...

- How can we tell what the best discretization is?
  - Unsupervised discretization: find breaks in the data values
    - Example:  
Petal Length



- Supervised discretization: Use class labels to find breaks

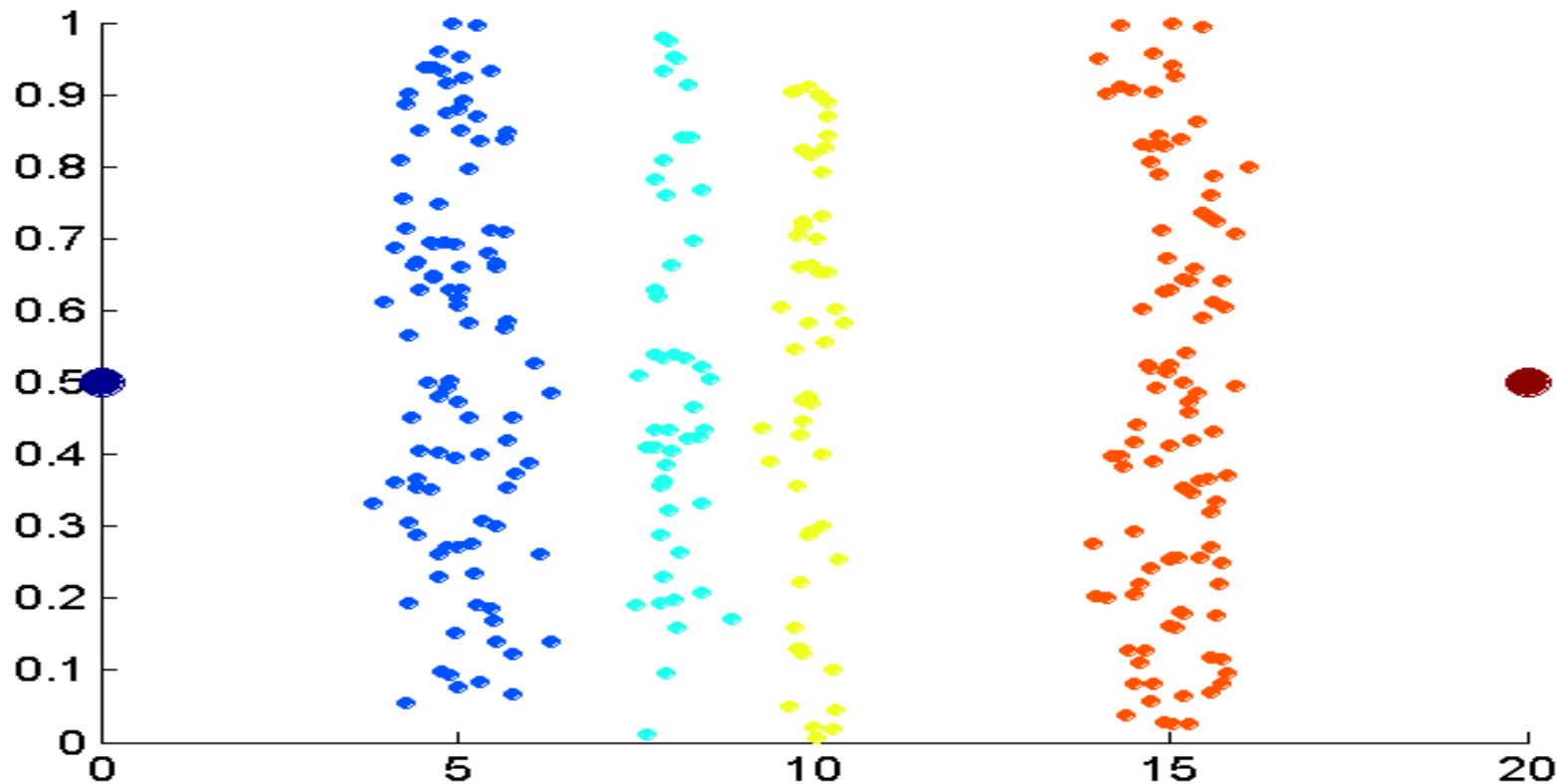


# Discretization

- Examples of **unsupervised discretization** techniques
  - N intervals with the same width  $W = (v_{\max} - v_{\min})/N$ 
    - Easy to implement
    - It can be badly affected by outliers and sparse data
    - Incremental approach
  - N intervals with (approximately) the same cardinality
    - It better fits sparse data and outliers
    - Non incremental approach
  - clustering
    - It fits well sparse data and outliers
  - analysis of data distribution
    - e.g., 4 intervals, one for each quartile



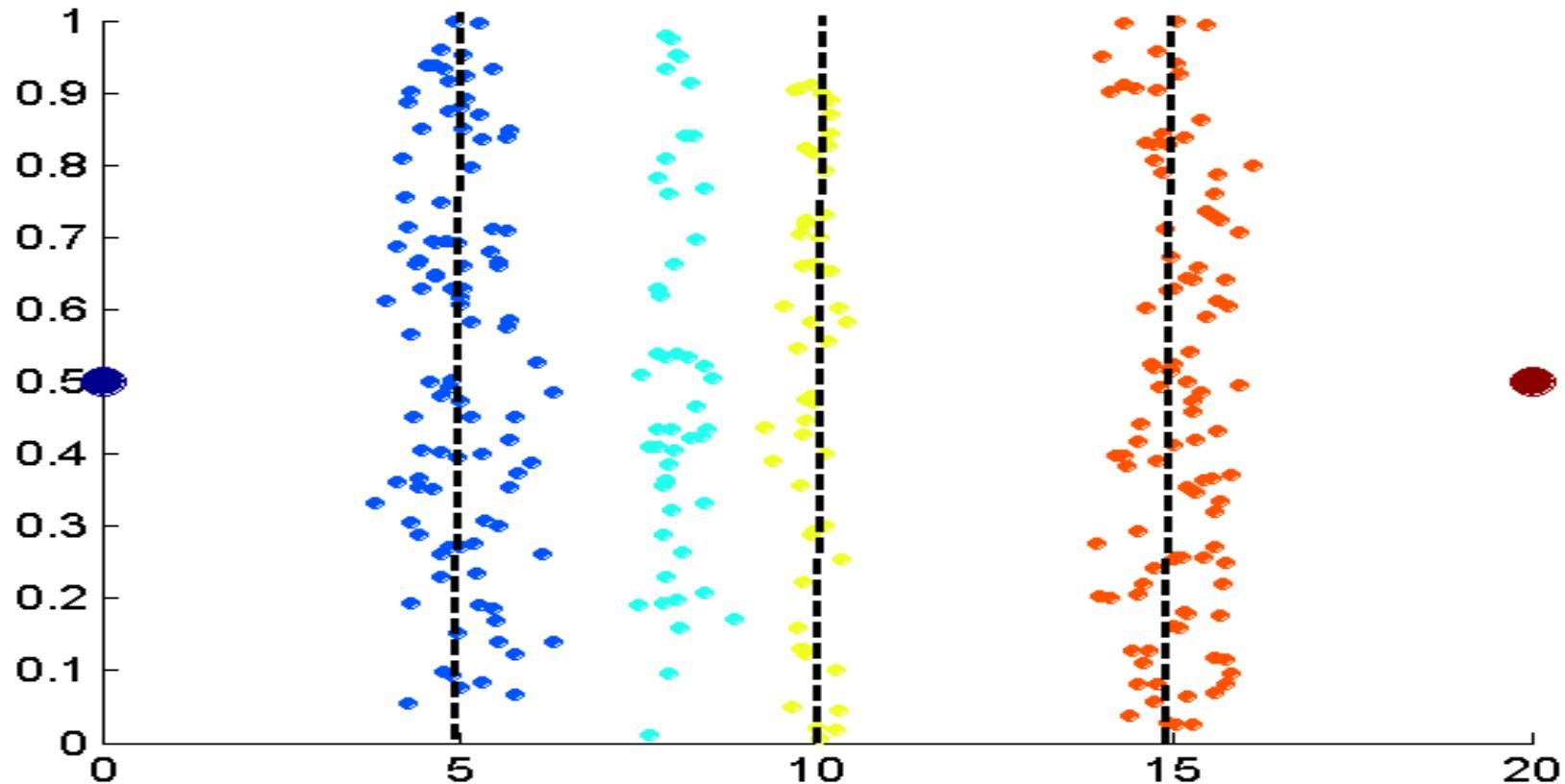
## Example: unsupervised discretization technique



**Data consists of four groups of points and two outliers. Data is one-dimensional, but a random y component is added to reduce overlap.**



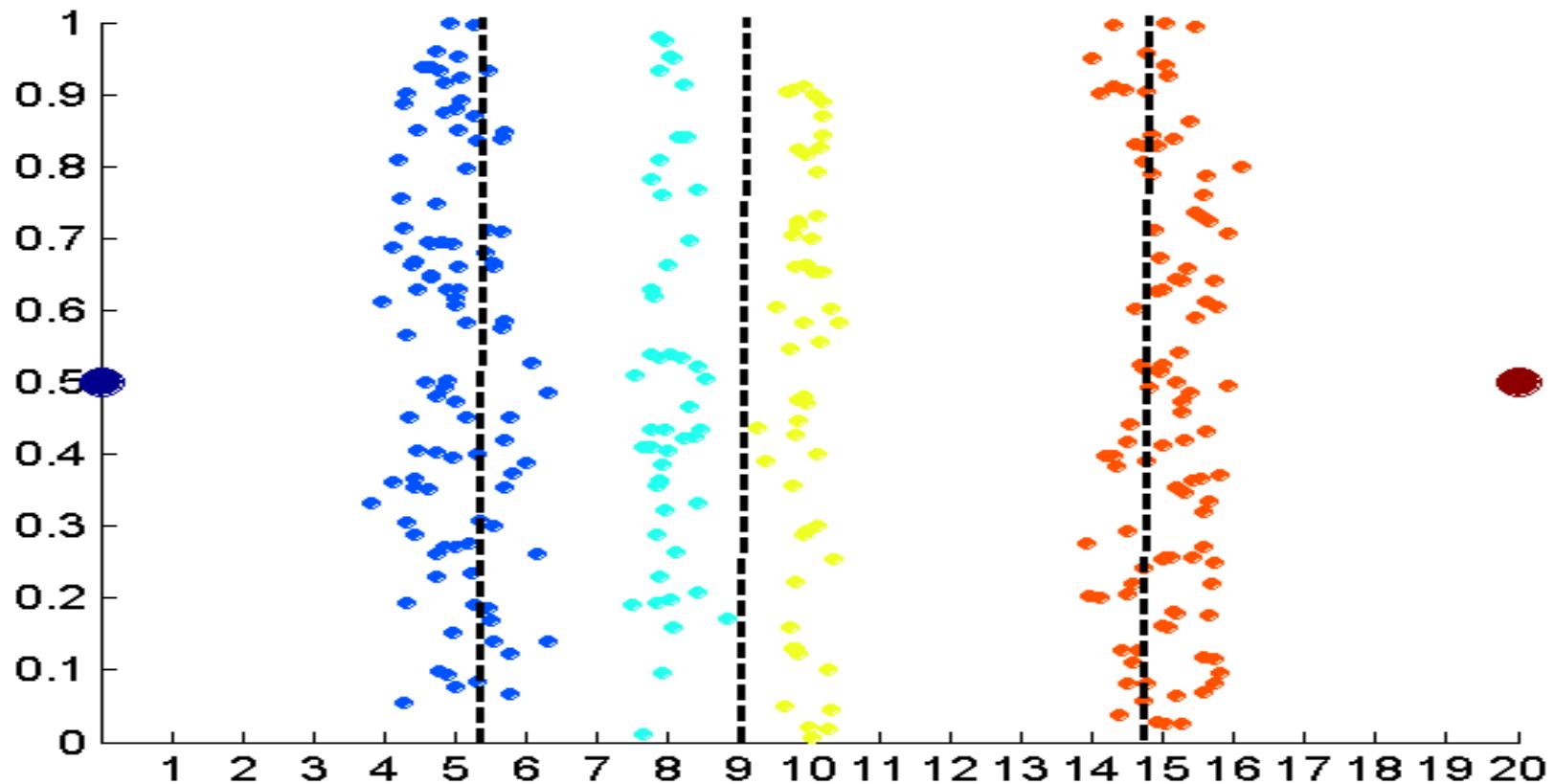
## Example: unsupervised discretization technique



**Equal interval width approach used to obtain 4 values.**



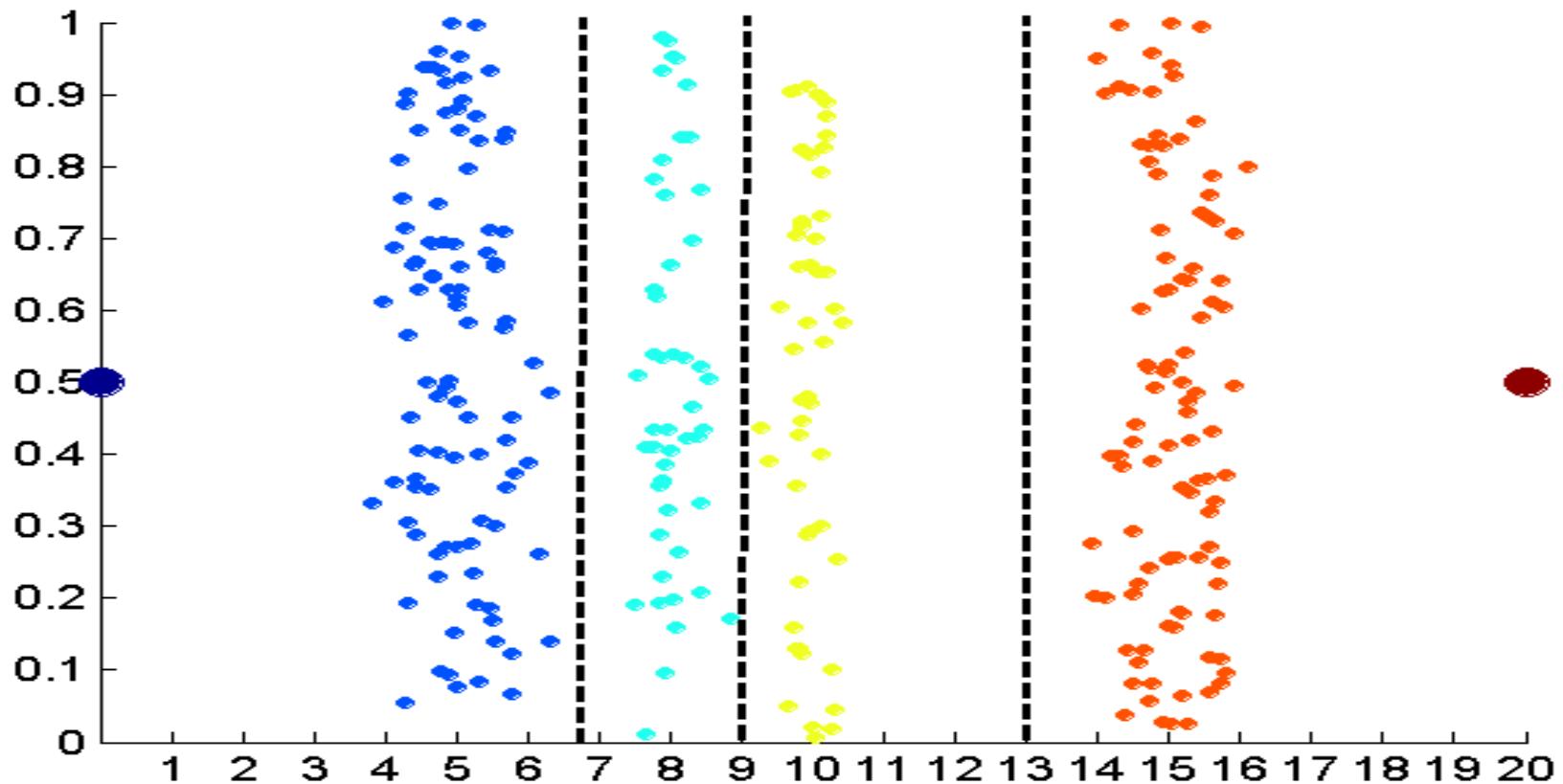
# Example: unsupervised discretization technique



**Equal frequency approach used to obtain 4 values.**



# Example: unsupervised discretization technique



**K-means approach to obtain 4 values.**



# Binarization

- Binarization maps an attribute into one or more binary variables
- **Continuous** attribute: first map the attribute to a categorical one
  - Example: height measured as {low, medium, high}
- **Categorical** attribute
  - Mapping to a set of binary attributes
  - Example: Low, medium, high as 1 0 0, 0 1 0, 0 0 1
  - **One-hot encoding**
    - Only 1 bit takes value 1
    - It represents the specific value taken by the attribute



# Attribute Transformation

- An **attribute transform** is a function that maps the entire set of values of a given attribute to a new set of replacement values such that each old value can be identified with one of the new values
  - Simple functions:  $x^k$ ,  $\log(x)$ ,  $e^x$ ,  $|x|$
- **Normalization**
  - Refers to various techniques to adjust to differences among attributes in terms of frequency of occurrence, mean, variance, range
  - Take out unwanted, common signal, e.g., seasonality
- In statistics, **standardization** refers to subtracting off the means and dividing by the standard deviation



# Normalization

- It is a type of data transformation
  - The values of an attribute are scaled so as to fall within a small specified range, typically [-1,+1] or [0,+1]
- Techniques
  - min-max normalization

$$v' = \frac{v - \min_A}{\max_A - \min_A} (new\_max_A - new\_min_A) + new\_min_A$$

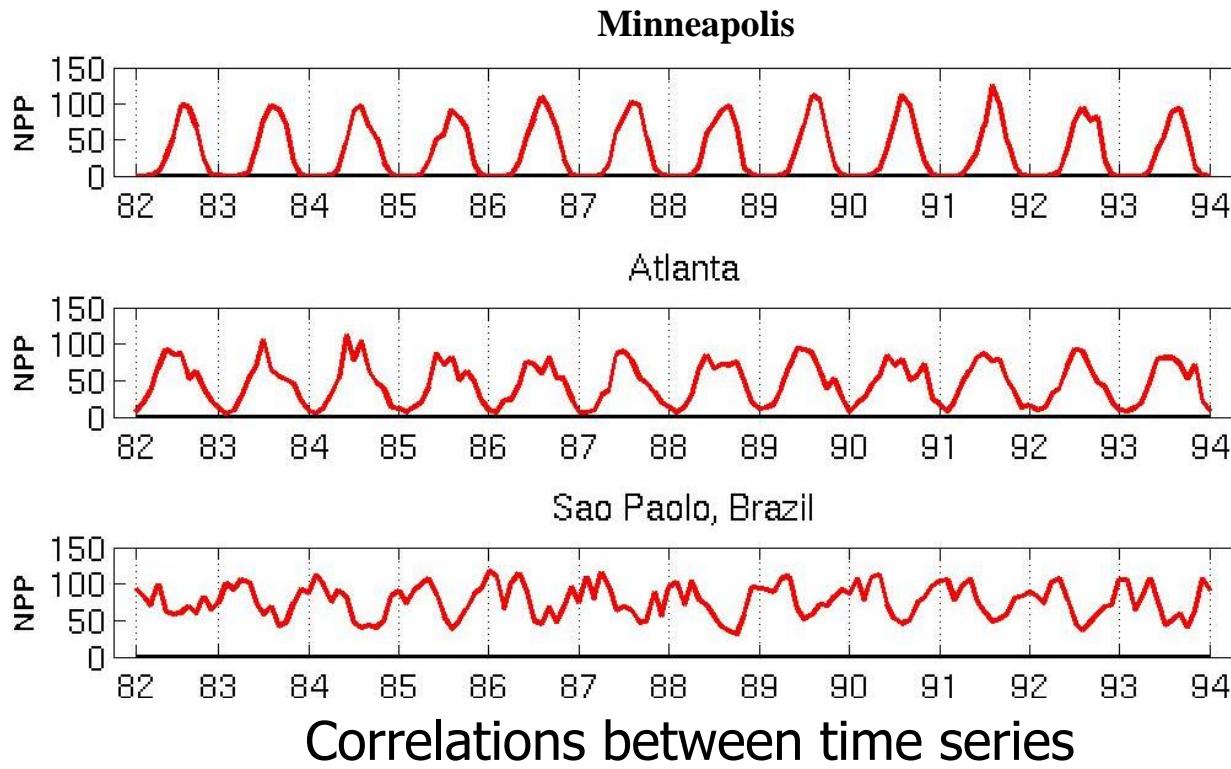
- z-score normalization
- decimal scaling

$$v' = \frac{v - \text{mean}_A}{\text{stand\_dev}_A}$$

$$v' = \frac{v}{10^j} \quad j \text{ is the smallest integer such that } \max(|v'|) < 1$$



# Example: Sample Time Series of Plant Growth

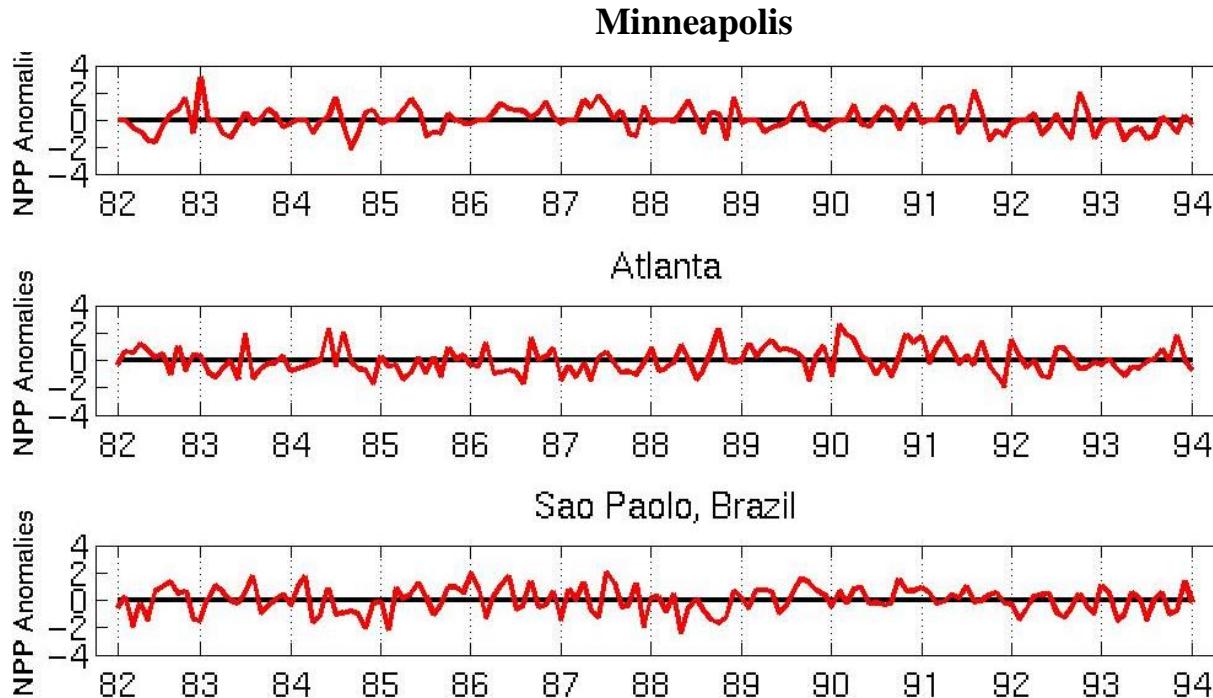


	Minneapolis	Atlanta	Sao Paolo
Minneapolis	1.0000	0.7591	-0.7581
Atlanta	0.7591	1.0000	-0.5739
Sao Paolo	-0.7581	-0.5739	1.0000

**Net Primary Production (NPP)** is a measure of plant growth used by ecosystem scientists.



# Example: Sample Time Series of Plant Growth



Normalized using monthly Z Score:  
Subtract off monthly mean and divide by monthly standard deviation

## Correlations between time series

	Minneapolis	Atlanta	Sao Paolo
Minneapolis	1.0000	0.0492	0.0906
Atlanta	0.0492	1.0000	-0.0154
Sao Paolo	0.0906	-0.0154	1.0000

From: Tan, Steinbach, Karpatne, Kumar, Introduction to Data Mining 2<sup>nd</sup> Ed., McGraw Hill 2018

# Data preparation for document data

D<sup>B</sup><sub>MG</sub>



Data Base and Data Mining Group of Politecnico di Torino



# Document representation

- A document might be modeled in different ways
  - The choice heavily affects the quality of the mining result
- The most common representation models a document as **a set of features**
  - Each feature might represent a set of characters, a word, a term, a concept



# Document processing

- It is the activity to generate a structured data representation of document data
- It includes five sequential steps
  - Document splitting
  - Tokenisation
  - Case normalisation
  - Stopword removal
  - Stemming



# Document splitting

- Based on the data analytics goal, documents can be split into
  - sentences, paragraphs, or analyzed in their entire content
- Short documents are typically not split
  - e.g., emails or social posts
- Long documents can be
  - broken up into sections or paragraphs
  - analyzed as a whole



# Tokenization

- It is the process of breaking text into sentences or text into tokens (i.e., words)
  - Identify sentence boundaries based on punctuation, capitalization
  - Separate words in sentences
  - Language-dependent



# Case normalization

- This step converts each token to completely upper-case or lower-case characters
  - Capitalisation helps human readers differentiate, for example, between nouns and proper nouns and can be useful for automated algorithms as well
  - However, an upper-case word at the beginning of the sentence should be treated no differently than the same word in lower case appearing elsewhere in a document



# Stemming

- Reduce a word to its root form (i.e., the **stem**)
  - It includes the identification and removal of prefixes, suffixes, and pluralisation
- It operates on a single word without knowledge of the context
  - It cannot discriminate between words which have different meanings depending on the part of speech
- Stemmers are
  - Easy to implement
  - Available for most spoken languages
  - Run significantly faster than lemmatization and POS tagging algorithms



# Stopword elimination

- “Stop words” refers to the most common words in a language
  - E.g., prepositions, articles, conjunctions in English
- Stop words are filtered out before or after processing of textual data
  - They are likely to have little semantic meaning



# Stopword elimination

- There is no single universal list of stop words used by all natural language processing tools
- Any group of words can be chosen as the stop words for a given purpose
  - different search engines use different stop word lists
  - Some of them remove lexical words, such as "want", from a query in order to improve performance
- Some tools specifically avoid removing these stop words to support phrase search

# Weighted document representation

D<sup>B</sup><sub>MG</sub>



Data Base and Data Mining Group of Politecnico di Torino



# Text representation: feature vectors

- Most data mining algorithms are unable to directly process textual data in their original form
  - documents are transformed into a more manageable representation
- Documents are represented by **feature vectors**
- A feature is simply an entity without internal structure
  - A dimension of the feature space
- A document is represented as a vector in this space
  - a collection of features and their weights



# Example

- Each document becomes a term vector
  - each term is a component (attribute) of the vector
  - the value of each component is the number of times the corresponding term occurs in the document

	team	coach	play	ball	score	game	winner	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2
Document 2	0	7	0	2	1	0	0	3	0	0
Document 3	0	1	0	0	1	2	2	0	3	0



# Bag-of-word representation

- All words in a document are considered as separate features
  - the dimension of the feature space is equal to the number of different words in the entire document collection
- The feature vector of a document consists of a set of weights, one for each distinct word
- The methods for giving weights to the features may vary



# Weighting schemes

- Binary
  - One, if the corresponding word is present in the document
  - Zero, otherwise
  - Occurrences of all words have the same importance
- Simple document frequency
  - The number of times in which the corresponding word occurs in the document
  - Most frequent words are not always representative of the document content



# Weighting schemes

- More complex weighting schemes are possible to take into account the frequency of the word
  - in the document
  - in the section/paragraph
  - in the category (for indexed documents)
  - in the collection of documents



# Weighting schemes

- Term frequency inverse document frequency (tf-idf)
  - Tf-idf of term  $t$  in document  $d$  of collection  $D$  (consisting of  $m$  documents)
$$\text{tf-idf}(t) = \text{freq}(t, d) * \log(m/\text{freq}(t, D))$$
  - Terms occurring frequently in a single document but rarely in the whole collection are preferred
- Suitable for
  - A single document consisting of many sections or subsections
  - A collection of *heterogeneous* documents



# Weighting schemes

- Term frequency document frequency (tf-df)
  - Tf-df of term t in document d of collection D
$$\text{tf-df}(t) = \text{freq}(t, d) * \log(\text{freq}(t, D))$$
  - Terms occurring frequently both in a single document and in the whole collection are preferred
- Suitable for
  - Single documents or parts of a document with *homogeneous* content
  - A collection of documents ranging over the same topic



# Tf-idf matrix example

major	malform	materi	matric	matrix	mean	measur	mechan	medicin	medium	medlin	method	methodolog	micro	microarch...	migrat	mo	model	molecular	morbid	moreov	mortal
0	0	0.153	0.051	0.021	0	0	0	0	0	0.051	0.069	0.072	0	0.020	0	0.034	0.072	0	0.072	0.063	
0.032	0.032	0.048	0.032	0.020	0.032	0.032	0.032	0.064	0.032	0.032	0.048	0.043	0.023	0.032	0.018	0.032	0.022	0.023	0.095	0.023	0.033
0	0	0	0	0.016	0	0.077	0.077	0	0	0	0.039	0.026	0	0.077	0.007	0.077	0	0	0	0	0.016
0.085	0.171	0	0	0	0	0	0	0	0.171	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0.153	0.051	0.021	0	0	0	0	0	0.051	0.069	0.072	0	0.020	0	0.034	0.072	0	0.072	0.063	
0	0	0	0.052	0	0.105	0	0	0.052	0	0.052	0	0.035	0	0	0.020	0	0.035	0	0	0	0.022
0.093	0	0	0	0.039	0	0	0	0.093	0	0.093	0	0	0	0	0.018	0	0	0	0	0	0
0.077	0	0.154	0	0.032	0	0	0	0.077	0	0.077	0	0	0	0	0.030	0	0.052	0	0	0	0.032



- Most common words (e.g., “model”) have low values
- Peculiar words (“e.g., “medlin”, “micro”, “methodolog”) have high values



# Weighting schemes

- Document-Term matrix  $X$ 
  - Local weight  $l_{ij}$
  - Global weight  $g_j$
- $X_{ij} = l_{ij} * g_j$

# Similarity and dissimilarity

D<sup>B</sup><sub>MG</sub>



Data Base and Data Mining Group of Politecnico di Torino



# Similarity and Dissimilarity

## ■ Similarity

- Numerical measure of how alike two data objects are
- Is higher when objects are more alike
- Often falls in the range [0,1]

## ■ Dissimilarity

- Numerical measure of how different are two data objects
- Lower when objects are more alike
- Minimum dissimilarity is often 0
- Upper limit varies

## ■ Proximity refers to a similarity or dissimilarity



# Similarity/Dissimilarity for Simple Attributes

The following table shows the similarity and dissimilarity between two objects,  $x$  and  $y$ , with respect to a single, simple attribute.

Attribute Type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{if } x \neq y \end{cases}$	$s = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases}$
Ordinal	$d =  x - y /(n - 1)$ (values mapped to integers 0 to $n-1$ , where $n$ is the number of values)	$s = 1 - d$
Interval or Ratio	$d =  x - y $	$s = -d, s = \frac{1}{1+d}, s = e^{-d},$ $s = 1 - \frac{d - \min_d}{\max_d - \min_d}$



# Euclidean Distance

- Euclidean Distance

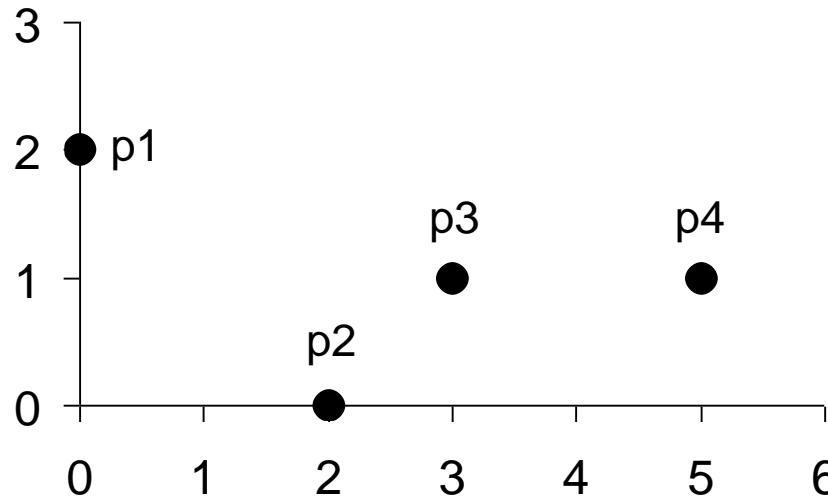
$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{k=1}^n (x_k - y_k)^2}$$

where  $n$  is the number of dimensions (attributes) and  $x_k$  and  $y_k$  are, respectively, the  $k^{th}$  attributes (components) of data objects  $\mathbf{x}$  and  $\mathbf{y}$ .

- Standardization is necessary, if scales differ.



# Euclidean Distance



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Distance Matrix



# Minkowski Distance

- Minkowski Distance is a generalization of Euclidean Distance

$$d(\mathbf{x}, \mathbf{y}) = \left( \sum_{k=1}^n |x_k - y_k|^r \right)^{1/r}$$

Where  $r$  is a parameter,  $n$  is the number of dimensions (attributes) and  $x_k$  and  $y_k$  are, respectively, the  $k^{\text{th}}$  attributes (components) or data objects  $x$  and  $y$ .



# Minkowski Distance: Examples

- $r = 1$ . City block (Manhattan, taxicab,  $L_1$  norm) distance
  - A common example of this is the Hamming distance, which is just the number of bits that are different between two binary vectors
- $r = 2$ . Euclidean distance
- $r \rightarrow \infty$ . “supremum” ( $L_{\max}$  norm,  $L_\infty$  norm) distance
  - This is the maximum difference between any component of the vectors
- Do not confuse  $r$  with  $n$ , i.e., all these distances are defined for all numbers of dimensions.



# Minkowski Distance

point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

L1	p1	p2	p3	p4
p1	0	4	4	6
p2	4	0	2	4
p3	4	2	0	2
p4	6	4	2	0

L2	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

$L_\infty$	p1	p2	p3	p4
p1	0	2	3	5
p2	2	0	1	3
p3	3	1	0	2
p4	5	3	2	0

## Distance Matrix



# Common Properties of a Distance

- Distances, such as the Euclidean distance, have some well-known properties.
  1.  $d(\mathbf{x}, \mathbf{y}) \geq 0$  for all  $x$  and  $y$  and  $d(\mathbf{x}, \mathbf{y}) = 0$  only if  $\mathbf{x} = \mathbf{y}$ . (Positive definiteness)
  2.  $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$  for all  $\mathbf{x}$  and  $\mathbf{y}$ . (Symmetry)
  3.  $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$  for all points  $\mathbf{x}$ ,  $\mathbf{y}$ , and  $\mathbf{z}$ . (Triangle Inequality)
- A distance that satisfies these properties is a **metric**



# Common Properties of a Similarity

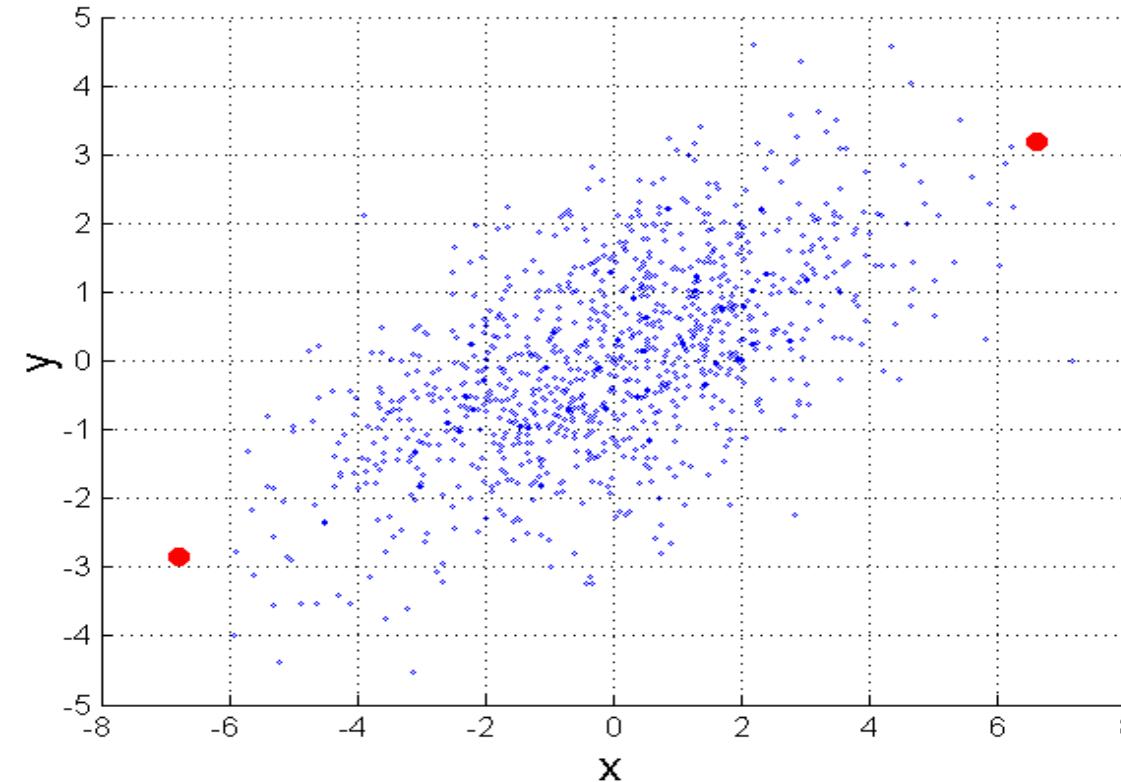
- Similarities also have some well known properties
  1.  $s(x, y) = 1$  (or maximum similarity) only if  $x = y$
  2.  $s(x, y) = s(y, x)$  for all  $x$  and  $y$  (symmetry)

where  $s(x, y)$  is the similarity between points (data objects)  $x$  and  $y$



# Mahalanobis Distance

$$\text{mahalanobis}(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T \Sigma^{-1} (\mathbf{x} - \mathbf{y})$$

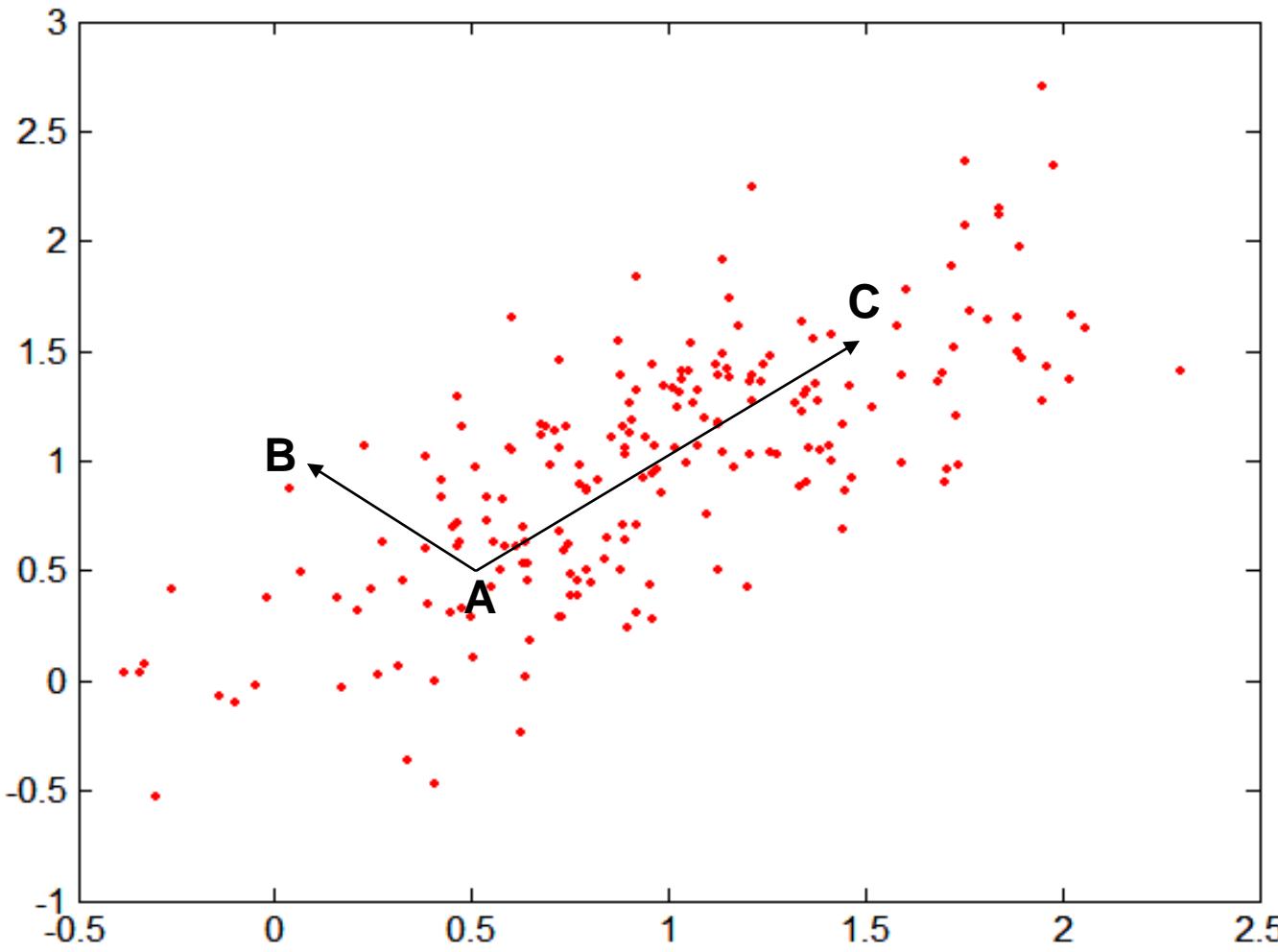


$\Sigma$  is the covariance matrix

For red points, the Euclidean distance is 14.7, Mahalanobis distance is 6.



# Mahalanobis Distance



**Covariance Matrix:**

$$\Sigma = \begin{bmatrix} 0.3 & 0.2 \\ 0.2 & 0.3 \end{bmatrix}$$

A: (0.5, 0.5)

B: (0, 1)

C: (1.5, 1.5)

**Mahal(A,B) = 5**

**Mahal(A,C) = 4**



# Similarity Between Binary Vectors

- Common situation is that objects  $p$  and  $q$  have only binary attributes
- Compute similarities using the following quantities

$M_{01}$  = the number of attributes where  $p$  was 0 and  $q$  was 1

$M_{10}$  = the number of attributes where  $p$  was 1 and  $q$  was 0

$M_{00}$  = the number of attributes where  $p$  was 0 and  $q$  was 0

$M_{11}$  = the number of attributes where  $p$  was 1 and  $q$  was 1

- Simple Matching and Jaccard Coefficients

$$\begin{aligned} \text{SMC} &= \text{number of matches} / \text{number of attributes} \\ &= (M_{11} + M_{00}) / (M_{01} + M_{10} + M_{11} + M_{00}) \end{aligned}$$

$$\begin{aligned} J &= \text{number of } 11 \text{ matches} / \text{number of not-both-zero attributes values} \\ &= (M_{11}) / (M_{01} + M_{10} + M_{11}) \end{aligned}$$



# SMC versus Jaccard: Example

$p = 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0$

$q = 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 1$

$M_{01} = 2$  (the number of attributes where p was 0 and q was 1)

$M_{10} = 1$  (the number of attributes where p was 1 and q was 0)

$M_{00} = 7$  (the number of attributes where p was 0 and q was 0)

$M_{11} = 0$  (the number of attributes where p was 1 and q was 1)

$$\text{SMC} = (M_{11} + M_{00}) / (M_{01} + M_{10} + M_{11} + M_{00}) = (0+7) / (2+1+0+7) = 0.7$$

$$J = (M_{11}) / (M_{01} + M_{10} + M_{11}) = 0 / (2 + 1 + 0) = 0$$



# Cosine Similarity

- If  $d_1$  and  $d_2$  are two document vectors, then

$$\cos(d_1, d_2) = (d_1 \bullet d_2) / \|d_1\| \|d_2\|,$$

where  $\bullet$  indicates vector dot product and  $\|d\|$  is the norm of vector  $d$

- Example:

$$d_1 = \begin{bmatrix} 3 & 2 & 0 & 5 & 0 & 0 & 0 & 2 & 0 & 0 \end{bmatrix}$$

$$d_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 \end{bmatrix}$$

$$d_1 \bullet d_2 = 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5$$

$$\|d_1\| = (\sqrt{3^2+2^2+0^2+5^2+0^2+0^2+0^2+2^2+0^2+0^2})^{0.5} = (42)^{0.5} = 6.481$$

$$\|d_2\| = (\sqrt{1^2+0^2+0^2+0^2+0^2+0^2+0^2+1^2+0^2+2^2})^{0.5} = (6)^{0.5} = 2.245$$

$$\cos(d_1, d_2) = .3150$$



# General Approach for Combining Similarities

- Sometimes attributes are of many different types, but an overall similarity is needed.
  - 1: For the  $k^{\text{th}}$  attribute, compute a similarity,  $s_k(\mathbf{x}, \mathbf{y})$ , in the range  $[0, 1]$ .
  - 2: Define an indicator variable,  $\delta_k$ , for the  $k^{\text{th}}$  attribute as follows:  
 $\delta_k = 0$  if the  $k^{\text{th}}$  attribute is an asymmetric attribute and both objects have a value of 0, or if one of the objects has a missing value for the  $k^{\text{th}}$  attribute  
 $\delta_k = 1$  otherwise

3. Compute  $\text{similarity}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{k=1}^n \delta_k s_k(\mathbf{x}, \mathbf{y})}{\sum_{k=1}^n \delta_k}$



# Using Weights to Combine Similarities

- May not want to treat all attributes the same.
  - Use non-negative weights  $\omega_k$

$$\text{■ } \text{similarity}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{k=1}^n \omega_k \delta_k s_k(\mathbf{x}, \mathbf{y})}{\sum_{k=1}^n \omega_k \delta_k}$$

- Can also define a weighted form of distance

$$d(\mathbf{x}, \mathbf{y}) = \left( \sum_{k=1}^n w_k |x_k - y_k|^r \right)^{1/r}$$

# Correlation

D<sup>B</sup><sub>MG</sub>



Data Base and Data Mining Group of Politecnico di Torino



# Data correlation

- Measure of the linear relationship between two data objects
  - having binary or continuous variables
- Useful during the data exploration phase
  - To be better aware of data properties
- Analysis of feature correlation
  - Correlated features should be removed
    - simplifying the next analytics steps
    - improving the performance of the data-driven algorithms



# Pearson's correlation

$$\text{corr}(\mathbf{x}, \mathbf{y}) = \frac{\text{covariance}(\mathbf{x}, \mathbf{y})}{\text{standard\_deviation}(\mathbf{x}) * \text{standard\_deviation}(\mathbf{y})} = \frac{s_{xy}}{s_x s_y},$$

where we are using the following standard statistical notation and definitions

$$\text{covariance}(\mathbf{x}, \mathbf{y}) = s_{xy} = \frac{1}{n - 1} \sum_{k=1}^n (x_k - \bar{x})(y_k - \bar{y})$$

$$\text{standard\_deviation}(\mathbf{x}) = s_x = \sqrt{\frac{1}{n - 1} \sum_{k=1}^n (x_k - \bar{x})^2}$$

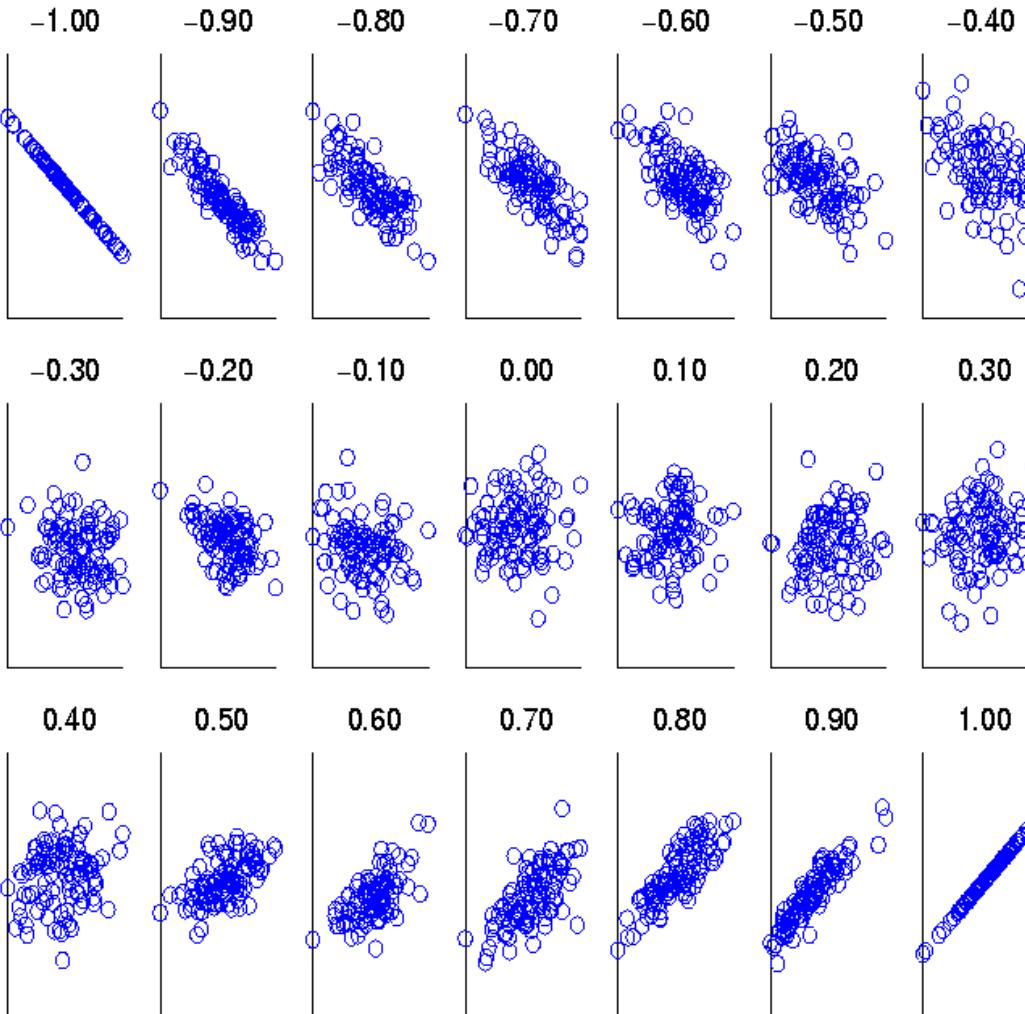
$$\text{standard\_deviation}(\mathbf{y}) = s_y = \sqrt{\frac{1}{n - 1} \sum_{k=1}^n (y_k - \bar{y})^2}$$

$$\bar{x} = \frac{1}{n} \sum_{k=1}^n x_k \text{ is the mean of } \mathbf{x}$$

$$\bar{y} = \frac{1}{n} \sum_{k=1}^n y_k \text{ is the mean of } \mathbf{y}$$



# Visually Evaluating Correlation



Scatter plots showing the similarity from –1 to 1.

Perfect linear correlation when value is 1 or -1



# Drawback of Correlation

- $\mathbf{x} = (-3, -2, -1, 0, 1, 2, 3)$
- $\mathbf{y} = (9, 4, 1, 0, 1, 4, 9)$

$$y_i = x_i^2$$

- $\text{mean}(\mathbf{x}) = 0, \text{mean}(\mathbf{y}) = 4$
- $\text{std}(\mathbf{x}) = 2.16, \text{std}(\mathbf{y}) = 3.74$

$$\begin{aligned}\text{corr} &= (-3)(5) + (-2)(0) + (-1)(-3) + (0)(-4) + (1)(-3) + (2)(0) + 3(5) / (6 * 2.16 * 3.74) \\ &= 0\end{aligned}$$

# Association Rules Fundamentals



Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis, Silvia Chiusano  
*Politecnico di Torino*



# Association rules

- Objective
  - extraction of frequent correlations or pattern from a transactional database

Tickets at a supermarket counter

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diapers, Milk
4	Beer, Bread, Diapers, Milk
5	Coke, Diapers, Milk
...	...

- Association rule

$\text{diapers} \Rightarrow \text{beer}$

- 2% of transactions contains both items
- 30% of transactions containing diapers also contains beer



# Association rule mining

- A collection of transactions is given
  - a transaction is a set of items
  - items in a transaction are  
*not ordered*
- Association rule
  - $A, B \Rightarrow C$
  - $A, B$  = items in the rule body
  - $C$  = item in the rule head
- The  $\Rightarrow$  means co-occurrence
  - *not* causality
- Example
  - coke, diapers  $\Rightarrow$  milk

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diapers, Milk
4	Beer, Bread, Diapers, Milk
5	Coke, Diapers, Milk
...	...



# Transactional formats

- Association rule extraction is an *exploratory technique* that can be applied to any data type
- A transaction can be any set of items
  - Market basket data
  - Textual data
  - Structured data
  - ...



# Transactional formats

- Textual data
  - A document is a transaction
  - Words in a document are items in the transaction
- Data example
  - Doc1: algorithm analysis customer data mining relationship
  - Doc2: customer data management relationship
  - Doc3: analysis customer data mining relationship social
- Rule example

customer, relationship  $\Rightarrow$  data, mining





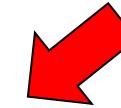
# Transactional formats

## ■ Structured data

- A table row is a transaction
- Pairs (attribute, value) are items in the transaction

## ■ Data example

Refund	Marital Status	Taxable Income	Cheat
No	Married	< 80K	No



Tid	Refund	Marital Status	Taxable Income	Cheat
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

- Transaction

Refund=no, MaritalStatus=married, TaxableIncome<80K, Cheat=No

## ■ Rule example

Refund=No, MaritalStatus=Married  $\Rightarrow$  Cheat = No



# Definitions

- **Itemset** is a set including one or more items
  - Example: {Beer, Diapers}
- **k-itemset** is an itemset that contains k items
- **Support count (#)** is the frequency of occurrence of an itemset
  - Example: #{{Beer,Diapers}} = 2
- **Support** is the fraction of transactions that contain an itemset
  - Example: sup({Beer, Diapers}) = 2/5
- **Frequent itemset** is an itemset whose support is greater than or equal to a *minsup* threshold

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



# Rule quality metrics

- Given the association rule

$$A \Rightarrow B$$

- A, B are itemsets
- Support** is the fraction of transactions containing both A and B

$$\frac{\#\{A,B\}}{|T|}$$

- |T| is the cardinality of the transactional database
- a priori probability of itemset AB
- rule frequency in the database
- Confidence** is the frequency of B in transactions containing A

$$\frac{\text{sup}(A,B)}{\text{sup}(A)}$$

- conditional probability of finding B having found A
- “strength” of the “ $\Rightarrow$ ”



# Rule quality metrics: example

- From itemset {Milk, Diapers} the following rules may be derived
- Rule: Milk  $\Rightarrow$  Diapers
  - support  
 $sup = \#\{\text{Milk, Diapers}\} / \#\text{trans.} = 3/5 = 60\%$
  - confidence  
 $conf = \#\{\text{Milk, Diapers}\} / \#\{\text{Milk}\} = 3/4 = 75\%$
- Rule: Diapers  $\Rightarrow$  Milk
  - same support  
 $s=60\%$
  - confidence  
 $conf = \#\{\text{Milk, Diapers}\} / \#\{\text{Diapers}\} = 3/3 = 100\%$

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



# Association rule extraction

- Given a set of transactions  $T$ , association rule mining is the extraction of the rules satisfying the constraints
  - support  $\geq \text{minsup}$  threshold
  - confidence  $\geq \text{minconf}$  threshold
- The result is
  - complete (*all* rules satisfying both constraints)
  - correct (*only* the rules satisfying both constraints)
- May add other more complex constraints



# Association rule extraction

- Brute-force approach
  - enumerate all possible permutations (i.e., association rules)
  - compute support and confidence for each rule
  - prune the rules that do not satisfy the *minsup* and *minconf* constraints
- Computationally *unfeasible*
- Given an itemset, the extraction process may be split
  - first generate frequent itemsets
  - next generate rules from each frequent itemset
- Example
  - Itemset  
 $\{\text{Milk, Diapers}\}$  sup=60%
  - Rules  
 $\text{Milk} \Rightarrow \text{Diapers}$  (conf=75%)  
 $\text{Diapers} \Rightarrow \text{Milk}$  (conf=100%)



# Association rule extraction

## (1) Extraction of frequent itemsets

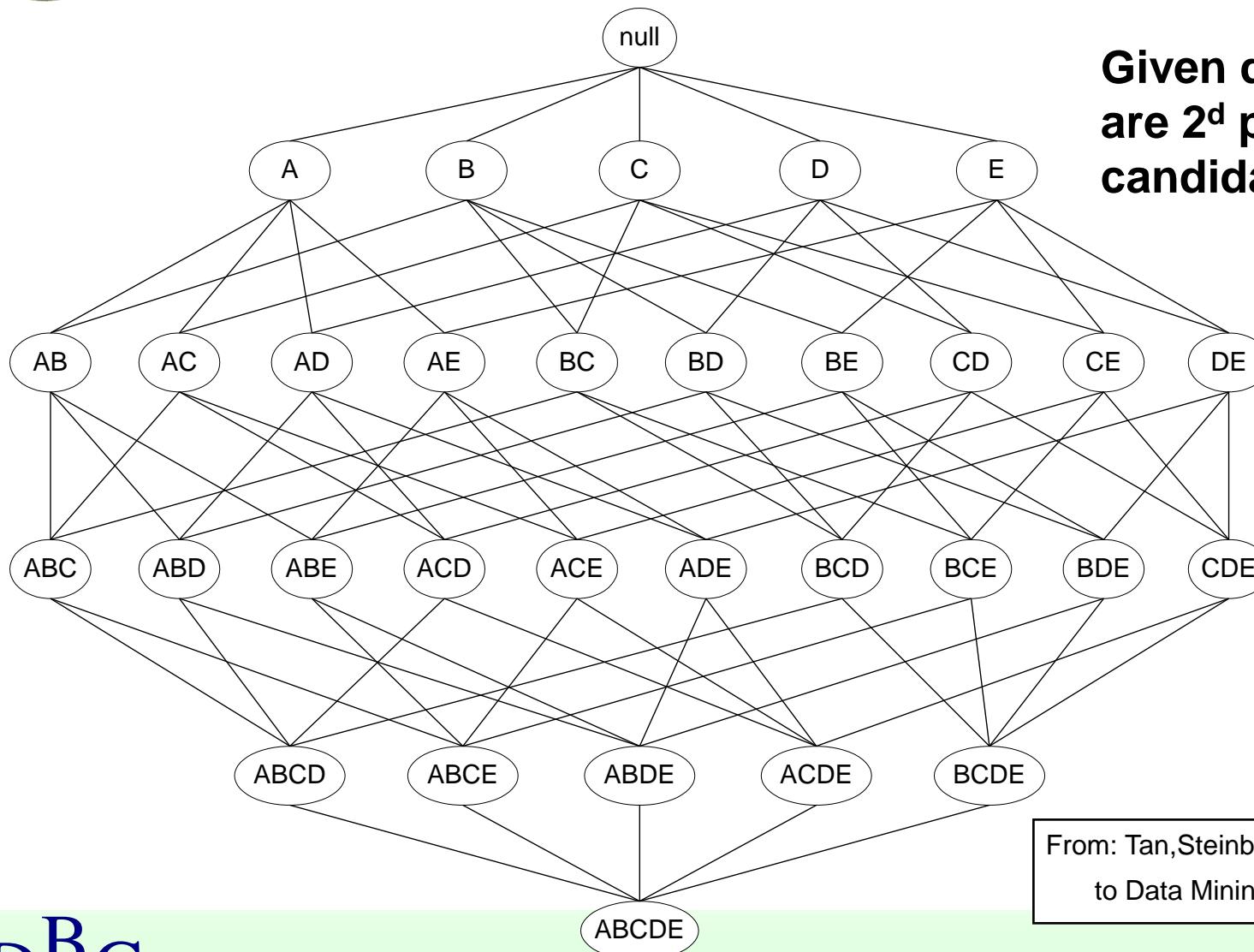
- many different techniques
  - level-wise approaches (Apriori, ...)
  - approaches without candidate generation (FP-growth, ...)
  - other approaches
- most computationally expensive step
  - limit extraction time by means of support threshold

## (2) Extraction of association rules

- generation of all possible binary partitioning of each frequent itemset
  - possibly enforcing a confidence threshold



# Frequent Itemset Generation



Given  $d$  items, there  
are  $2^d$  possible  
candidate itemsets

From: Tan, Steinbach, Kumar, Introduction  
to Data Mining, McGraw Hill 2006



# Frequent Itemset Generation

## ■ Brute-force approach

- each itemset in the lattice is a *candidate* frequent itemset
- scan the database to count the support of each candidate
  - match each transaction against every candidate
- Complexity  $\sim O(|T| 2^d w)$ 
  - $|T|$  is number of transactions
  - $d$  is number of items
  - $w$  is transaction length



# Improving Efficiency

- Reduce the number of candidates
  - Prune the search space
    - complete set of candidates is  $2^d$
- Reduce the number of transactions
  - Prune transactions as the size of itemsets increases
    - reduce  $|T|$
- Reduce the number of comparisons
  - Equal to  $|T| \cdot 2^d$
  - Use efficient data structures to store the candidates or transactions



# The Apriori Principle

*"If an itemset is frequent, then all of its subsets must also be frequent"*

- The support of an itemset can never exceed the support of any of its subsets
- It holds due to the antimonotone property of the support measure
  - Given two arbitrary itemsets A and B  
if  $A \subseteq B$  then  $\text{sup}(A) \geq \text{sup}(B)$
- It reduces the number of candidates

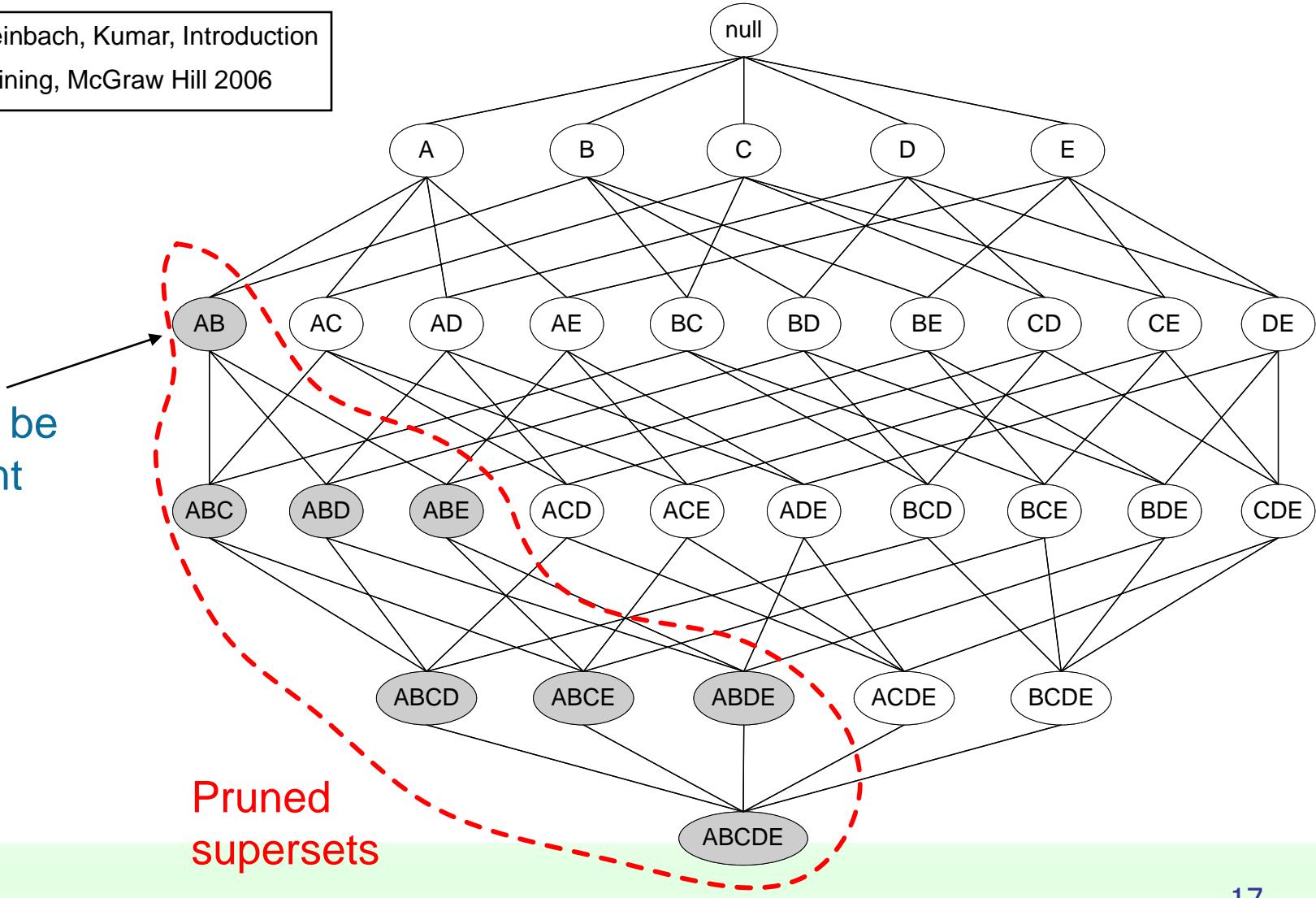


# The Apriori Principle

From: Tan, Steinbach, Kumar, Introduction  
to Data Mining, McGraw Hill 2006

Found to be  
Infrequent

Pruned  
supersets





# Apriori Algorithm [Agr94]

- Level-based approach
  - at each iteration extracts itemsets of a given length k
- Two main steps for each level
  - (1) Candidate generation
    - Join Step
      - generate candidates of length  $k+1$  by joining frequent itemsets of length k
    - Prune Step
      - apply Apriori principle: prune length  $k+1$  candidate itemsets that contain at least one k-itemset that is not frequent
  - (2) Frequent itemset generation
    - scan DB to count support for  $k+1$  candidates
    - prune candidates below minsup



# Apriori Algorithm [Agr94]

## ■ Pseudo-code

$C_k$ : Candidate itemset of size k

$L_k$  : frequent itemset of size k

$L_1 = \{\text{frequent items}\};$

**for** ( $k = 1$ ;  $L_k \neq \emptyset$ ;  $k++$ ) **do**

**begin**

$C_{k+1}$  = candidates generated from  $L_k$ ;

**for each** transaction  $t$  in database **do**

increment the count of all candidates in  $C_{k+1}$

that are contained in  $t$

$L_{k+1}$  = candidates in  $C_{k+1}$  satisfying  $\text{minsup}$

**end**

**return**  $\cup_k L_k$ ;



# Generating Candidates

- Sort  $L_k$  candidates in lexicographical order
- For each candidate of length  $k$ 
  - Self-join with each candidate sharing same  $L_{k-1}$  prefix
  - Prune candidates by applying Apriori principle
- Example: given  $L_3 = \{abc, abd, acd, ace, bcd\}$ 
  - Self-join
    - $abcd$  from  $abc$  and  $abd$
    - $acde$  from  $acd$  and  $ace$
  - Prune by applying Apriori principle
    - $acde$  is removed because  $ade, cde$  are not in  $L_3$
    - $C_4 = \{abcd\}$



# Apriori Algorithm: Example

Example DB

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

$\text{minsup} > 1$



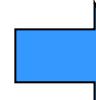
# Generate candidate 1-itemsets

Example DB

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

$C_1$

1<sup>st</sup> DB  
scan



itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

$\text{minsup} > 1$

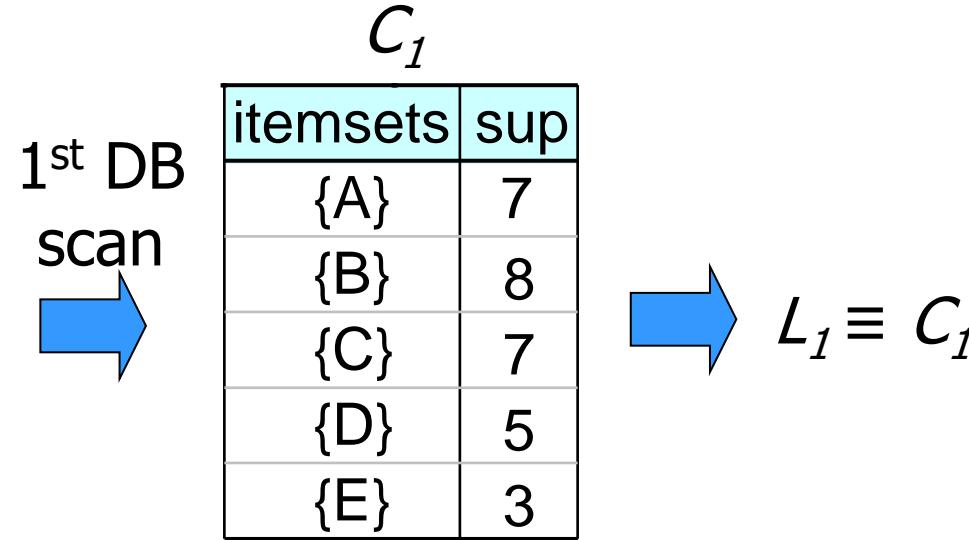


# Prune infrequent candidates in $C_1$

Example DB

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

$\text{minsup} > 1$



- All itemsets in set  $C_1$  are frequent according to  $\text{minsup} > 1$



# Generate candidates from $L_1$

$L_1$	
itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

$C_2$
itemsets
{A,B}
{A,C}
{A,D}
{A,E}
{B,C}
{B,D}
{B,E}
{C,D}
{C,E}
{D,E}



# Count support for candidates in $C_2$

$L_1$	
itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

$C_2$	
itemsets	
{A,B}	
{A,C}	
{A,D}	
{A,E}	
{B,C}	
{B,D}	
{B,E}	
{C,D}	
{C,E}	
{D,E}	

2<sup>nd</sup>  
DB  
scan

$C_2$	
itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{B,E}	1
{C,D}	3
{C,E}	2
{D,E}	2



# Prune infrequent candidates in $C_2$

$L_1$	
itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

$C_2$	
itemsets	
{A,B}	
{A,C}	
{A,D}	
{A,E}	
{B,C}	
{B,D}	
{B,E}	
{C,D}	
{C,E}	
{D,E}	

2<sup>nd</sup>  
DB  
scan

$C_2$	
itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{B,E}	1
{C,D}	3
{C,E}	2
{D,E}	2

$L_2$	
itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{C,D}	3
{C,E}	2
{D,E}	2



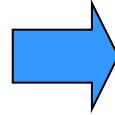
# Generate candidates from $L_2$

$L_2$

itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{C,D}	3
{C,E}	2
{D,E}	2

$C_3$

itemsets
{A,B,C}
{A,B,D}
{A,B,E}
{A,C,D}
{A,C,E}
{A,D,E}
{B,C,D}
{C,D,E}





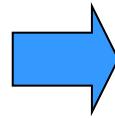
# Apply Apriori principle on $C_3$

$L_2$

itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{C,D}	3
{C,E}	2
{D,E}	2

$C_3$

itemsets
{A,B,C}
{A,B,D}
<b>{A,B,E}</b>
{A,C,D}
{A,C,E}
{A,D,E}
{B,C,D}
{C,D,E}



- Prune {A,B,E}
  - Its subset {B,E} is infrequent ({B,E} is not in  $L_2$ )



# Count support for candidates in $C_3$

$L_2$

itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{C,D}	3
{C,E}	2
{D,E}	2

$C_3$

itemsets
{A,B,C}
{A,B,D}
<b>{A,B,E}</b>
{A,C,D}
{A,C,E}
{A,D,E}
{B,C,D}
{C,D,E}

3<sup>rd</sup>  
DB  
scan

$C_3$

itemsets	sup
{A,B,C}	3
{A,B,D}	2
{A,C,D}	2
{A,C,E}	1
{A,D,E}	2
{B,C,D}	2
{C,D,E}	1



# Prune infrequent candidates in $C_3$

$L_2$

itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{C,D}	3
{C,E}	2
{D,E}	2

$C_3$

itemsets
{A,B,C}
{A,B,D}
{A,B,E}
{A,C,D}
{A,C,E}
{A,D,E}
{B,C,D}
{C,D,E}

3<sup>rd</sup>  
DB  
scan

$C_3$

itemsets	sup
{A,B,C}	3
{A,B,D}	2
{A,C,D}	2
{A,C,E}	1
{A,D,E}	2
{B,C,D}	2
{C,D,E}	1

$L_3$

itemsets	sup
{A,B,C}	3
{A,B,D}	2
{A,C,D}	2
{A,D,E}	2
{B,C,D}	2

- {A,C,E} and {C,D,E} are actually infrequent
  - They are discarded from  $C_3$



# Generate candidates from $L_3$

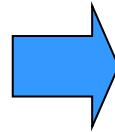
$L_3$

itemsets	sup
{A,B,C}	3
{A,B,D}	2
{A,C,D}	2
{A,D,E}	2
{B,C,D}	2

$C_4$

itemsets

{A,B,C,D}
-----------



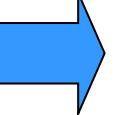


# Apply Apriori principle on $C_4$

itemsets	sup	$L_3$
{A,B,C}	3	
{A,B,D}	2	
{A,C,D}	2	
{A,D,E}	2	
{B,C,D}	2	

$C_4$

itemsets  
  
{A,B,C,D}



- Check if  $\{A,C,D\}$  and  $\{B,C,D\}$  belong to  $L_3$ 
  - $L_3$  contains all 3-itemset subsets of  $\{A,B,C,D\}$
  - $\{A,B,C,D\}$  is potentially frequent



# Count support for candidates in $C_4$

$L_3$

itemsets	sup
{A,B,C}	3
{A,B,D}	2
{A,C,D}	2
{A,D,E}	2
{B,C,D}	2

$C_4$

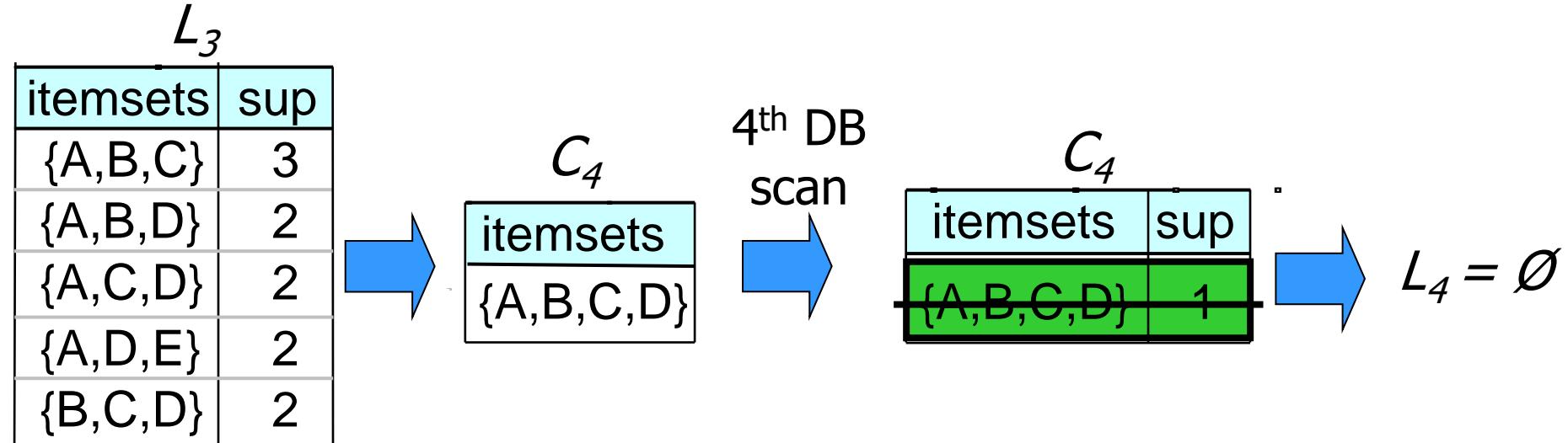
itemsets
{A,B,C,D}

4<sup>th</sup> DB  
scan

itemsets	sup
{A,B,C,D}	1



# Prune infrequent candidates in $C_4$



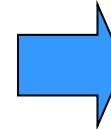
- {A,B,C,D} is actually infrequent
  - {A,B,C,D} is discarded from  $C_4$



# Final set of frequent itemsets

Example DB

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}



$L_1$

itemsets	sup
{A}	7
{B}	8
{C}	7
{D}	5
{E}	3

$L_2$

itemsets	sup
{A,B}	5
{A,C}	4
{A,D}	4
{A,E}	2
{B,C}	6
{B,D}	3
{C,D}	3
{C,E}	2
{D,E}	2

$L_3$

itemsets	sup
{A,B,C}	3
{A,B,D}	2
{A,C,D}	2
{A,D,E}	2
{B,C,D}	2

minsup>1

D<sup>B</sup><sub>MG</sub>



# Counting Support of Candidates

- Scan transaction database to count support of each itemset
  - total number of candidates may be large
  - one transaction may contain many candidates
- Approach [Agr94]
  - candidate itemsets are stored in a *hash-tree*
    - *leaf* node of hash-tree contains a list of itemsets and counts
    - *interior* node contains a hash table
  - subset function finds all candidates contained in a transaction
    - match transaction subsets to candidates in hash tree



# Performance Issues in Apriori

- Candidate generation
  - Candidate sets may be huge
    - 2-itemset candidate generation is the most critical step
    - extracting long frequent itemsets requires generating all frequent subsets
- Multiple database scans
  - $n + 1$  scans when longest frequent pattern length is  $n$



# Factors Affecting Performance

- Minimum support threshold
  - lower support threshold increases number of frequent itemsets
    - larger number of candidates
    - larger (max) length of frequent itemsets
- Dimensionality (number of items) of the data set
  - more space is needed to store support count of each item
  - if number of frequent items also increases, both computation and I/O costs may also increase
- Size of database
  - since Apriori makes multiple passes, run time of algorithm may increase with number of transactions
- Average transaction width
  - transaction width increases in dense data sets
  - may increase max length of frequent itemsets and traversals of hash tree
    - number of subsets in a transaction increases with its width



# Improving Apriori Efficiency

- Hash-based itemset counting [Yu95]
  - A  $k$ -itemset whose corresponding hashing bucket count is below the threshold cannot be frequent
- Transaction reduction [Yu95]
  - A transaction that does not contain any frequent  $k$ -itemset is useless in subsequent scans
- Partitioning [Sav96]
  - Any itemset that is potentially frequent in DB must be frequent in at least one of the partitions of DB



# Improving Apriori Efficiency

- Sampling [Toi96]
  - mining on a subset of given data, lower support threshold + a method to determine the completeness
- Dynamic Itemset Counting [Motw98]
  - add new candidate itemsets only when all of their subsets are estimated to be frequent



# FP-growth Algorithm [Han00]

- Exploits a main memory compressed representation of the database, the FP-tree
  - high compression for dense data distributions
    - less so for sparse data distributions
  - complete representation for frequent pattern mining
    - enforces support constraint
- Frequent pattern mining by means of FP-growth
  - recursive visit of FP-tree
  - applies divide-and-conquer approach
    - decomposes mining task into smaller subtasks
- Only two database scans
  - count item supports + build FP-tree



# FP-tree construction

## Example DB

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

$\text{minsup} > 1$

- (1) Count item support and prune items below minsup threshold
- (2) Build Header Table by sorting items in decreasing support order

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3



# FP-tree construction

Example DB

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

$\text{minsup} > 1$

- (1) Count item support and prune items below minsup threshold
- (2) Build Header Table by sorting items in decreasing support order
- (3) Create FP-tree

For each transaction  $t$  in DB

- order transaction  $t$  items in decreasing support order
  - same order as Header Table
- insert transaction  $t$  in FP-tree
  - use existing path for common prefix
  - create new branch when path becomes different

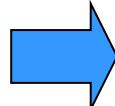


# FP-tree construction

Transaction

TID	Items
1	{A,B}

Sorted transaction

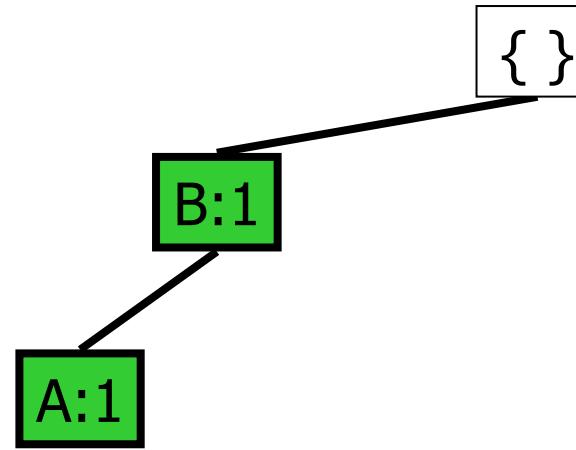


TID	Items
1	{B,A}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree



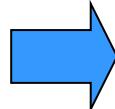


# FP-tree construction

Transaction

TID	Items
2	{B,C,D}

Sorted transaction

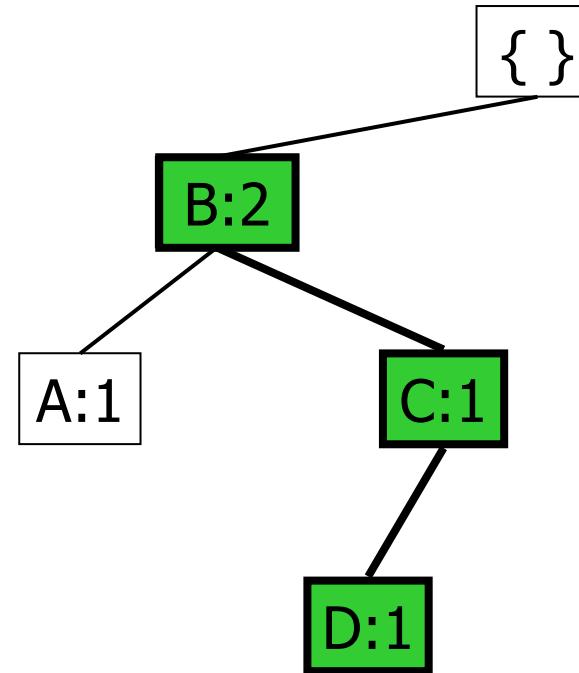


TID	Items
2	{B,C,D}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree





# FP-tree construction

Transaction

TID	Items
3	{A,C,D,E}

Sorted transaction

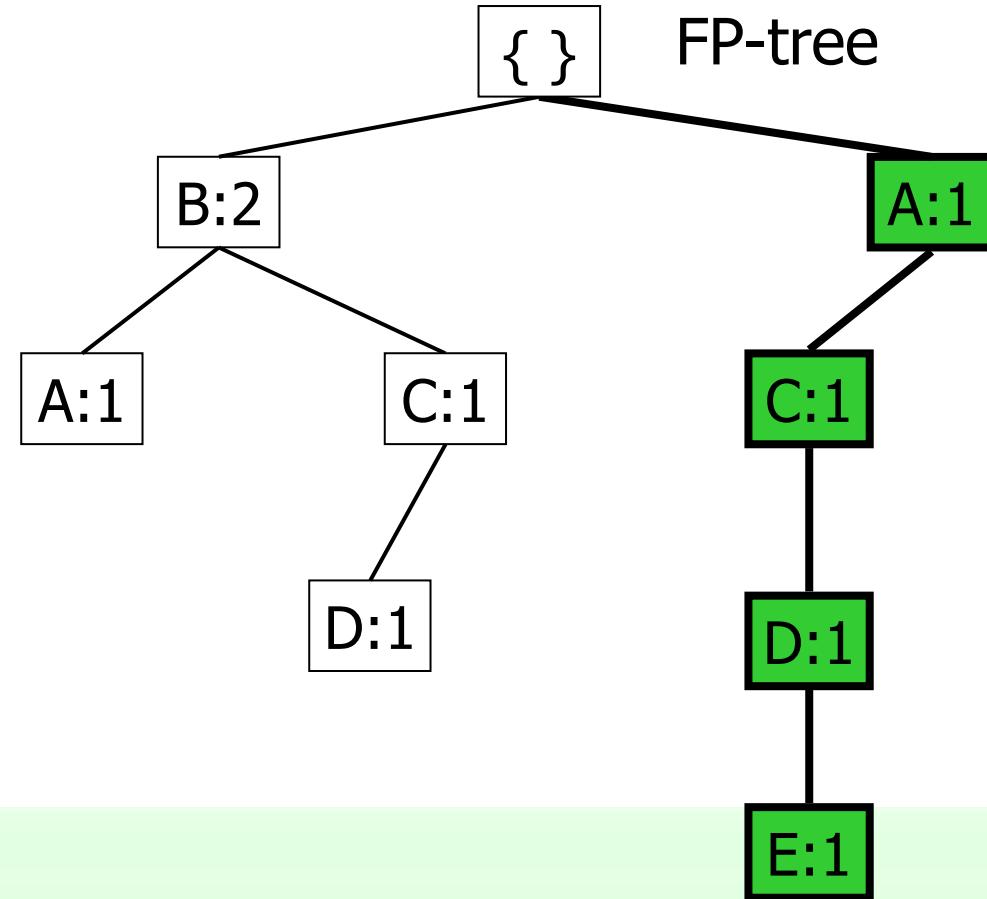


TID	Items
3	{A,C,D,E}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree



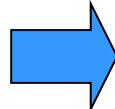


# FP-tree construction

Transaction

TID	Items
4	{A,D,E}

Sorted transaction

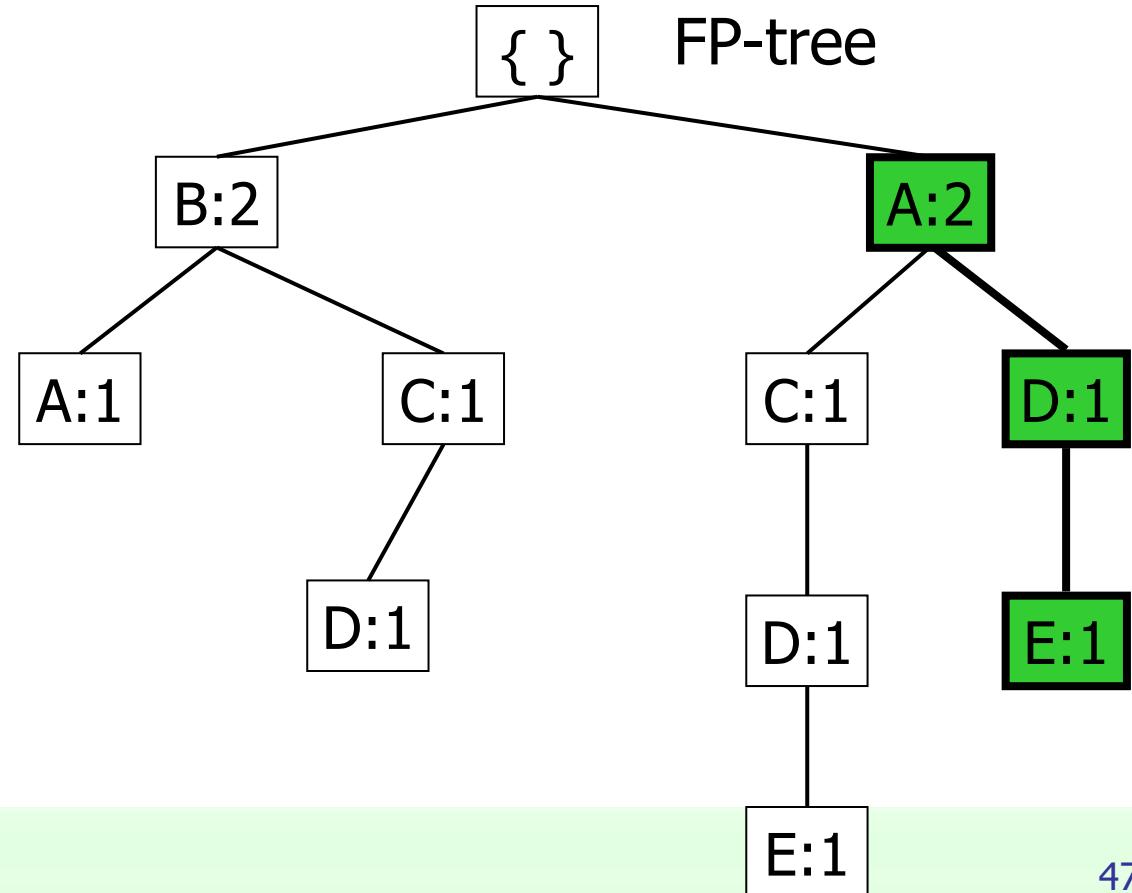


TID	Items
4	{A,D,E}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree



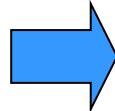


# FP-tree construction

Transaction

TID	Items
5	{A,B,C}

Sorted transaction

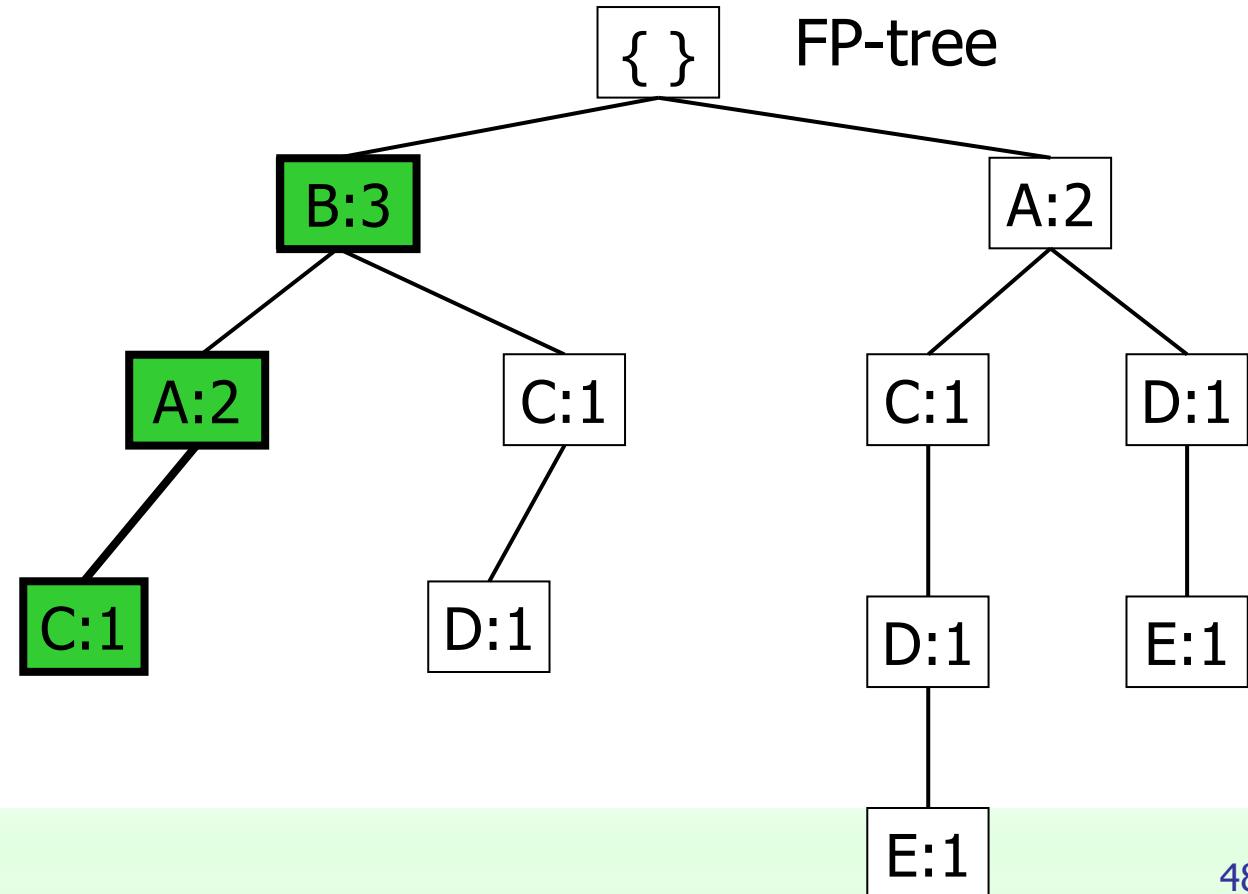


TID	Items
5	{B,A,C}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree



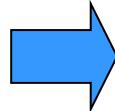


# FP-tree construction

Transaction

TID	Items
6	{A,B,C,D}

Sorted transaction

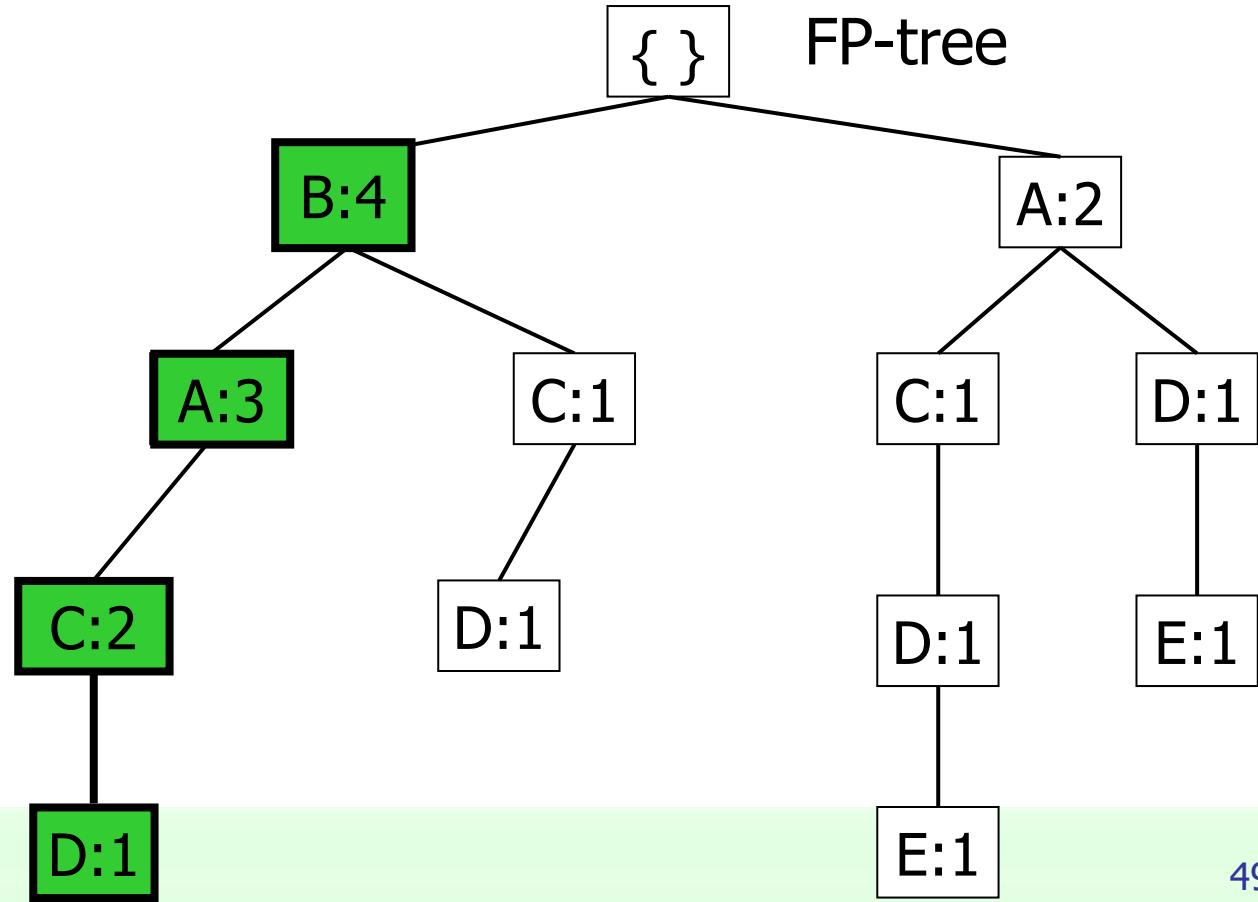


TID	Items
6	{B,A,C,D}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree



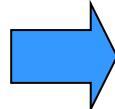


# FP-tree construction

Transaction

TID	Items
7	{B,C}

Sorted transaction

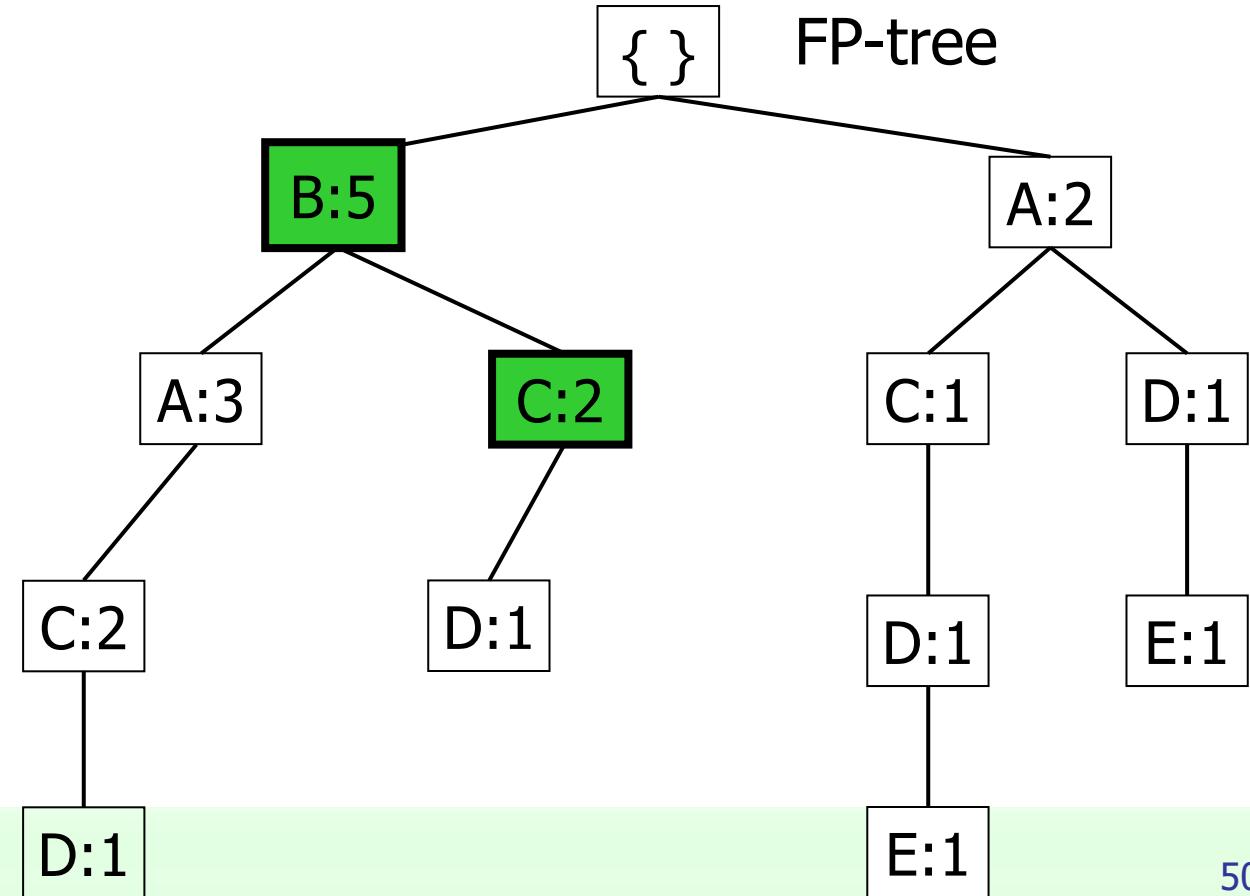


TID	Items
7	{B,C}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree



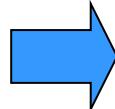


# FP-tree construction

Transaction

TID	Items
8	{A,B,C}

Sorted transaction

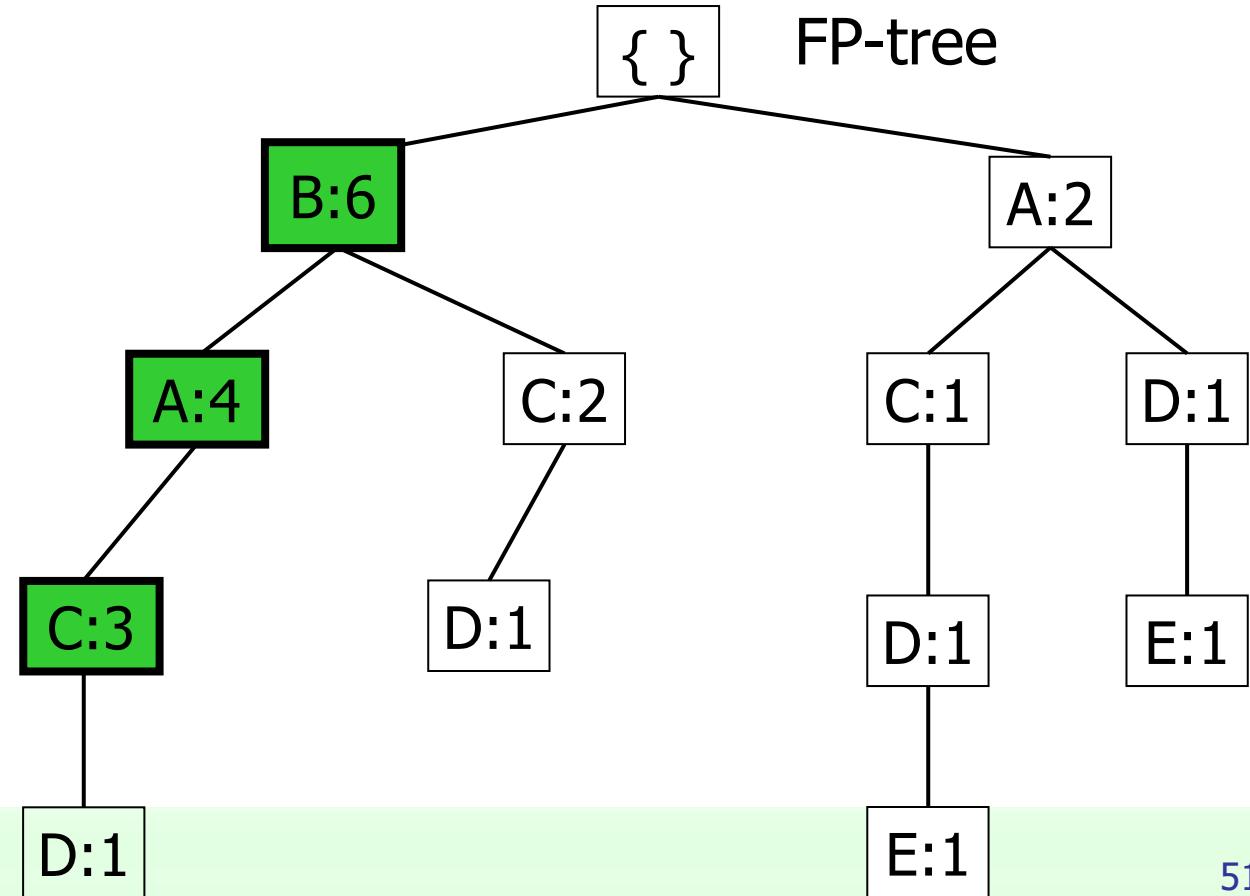


TID	Items
8	{B,A,C}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree



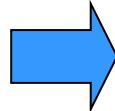


# FP-tree construction

Transaction

TID	Items
9	{A,B,D}

Sorted transaction

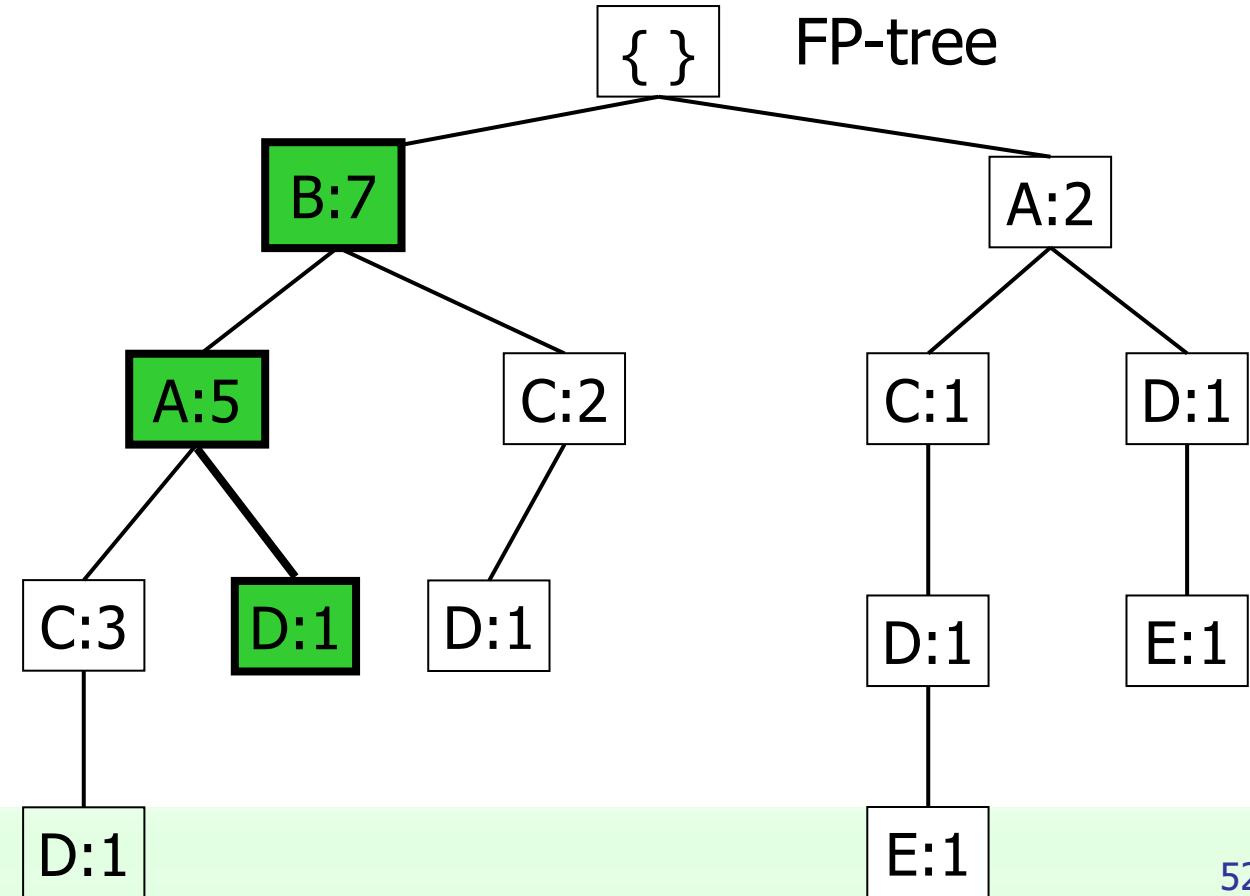


TID	Items
9	{B,A,D}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree



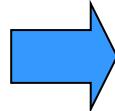


# FP-tree construction

Transaction

TID	Items
10	{B,C,E}

Sorted transaction

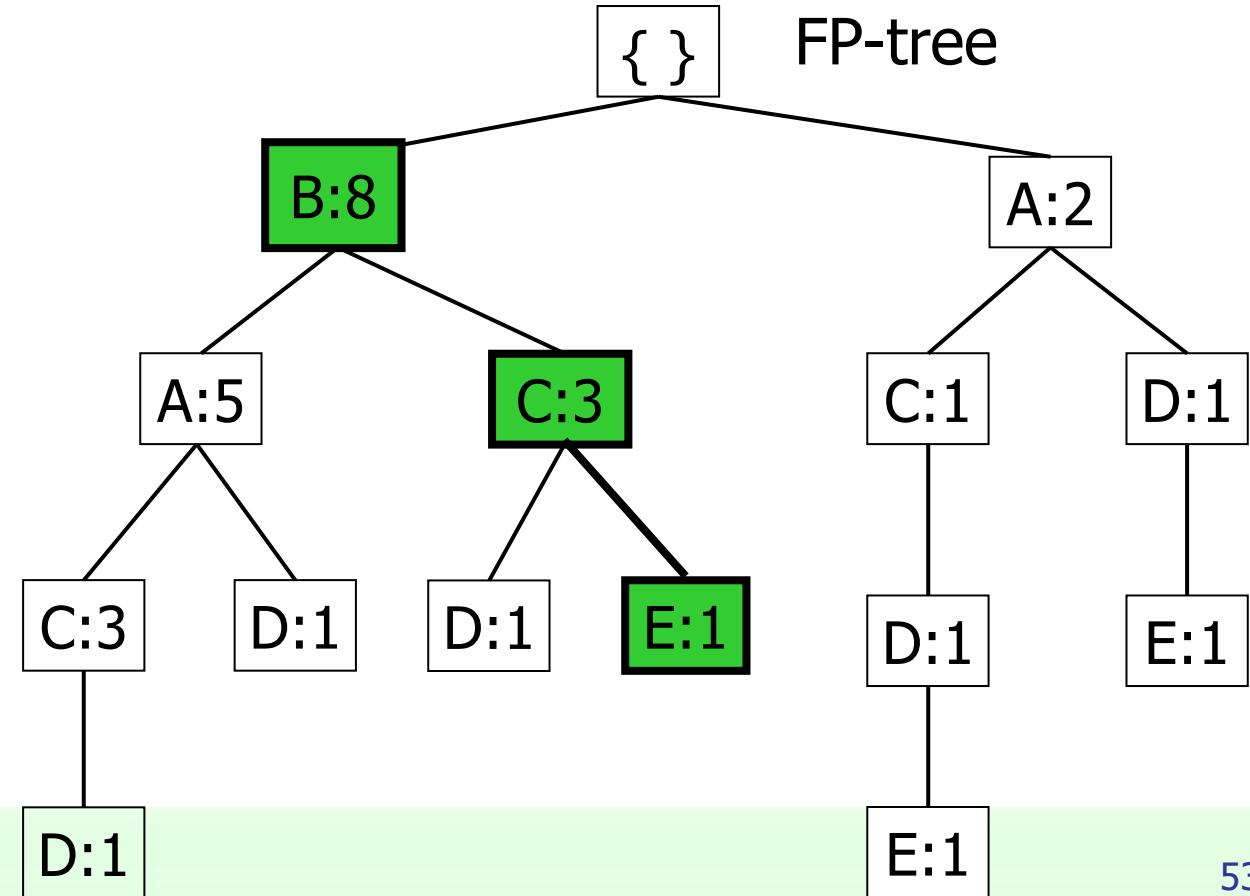


TID	Items
10	{B,C,E}

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

FP-tree

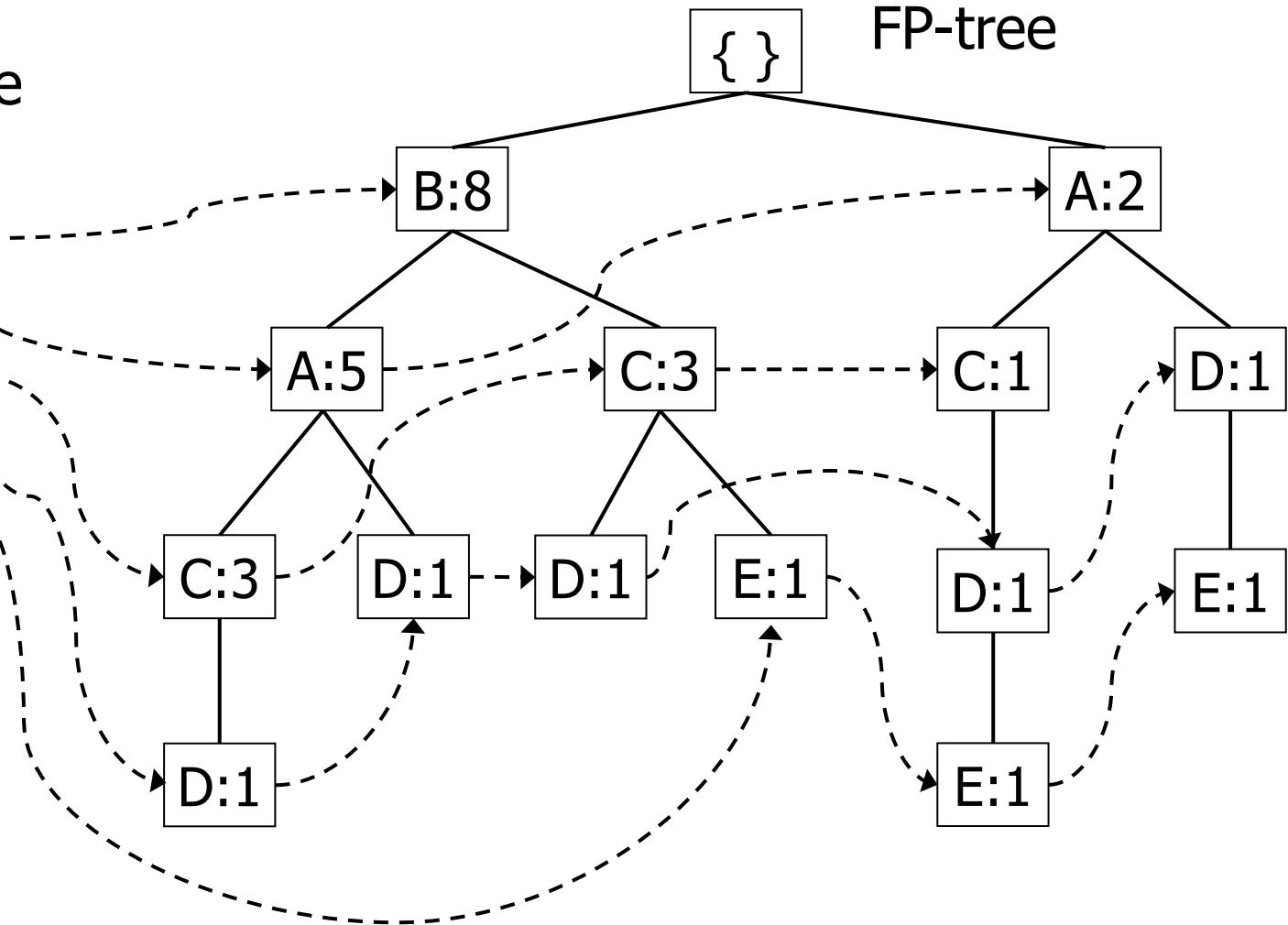




# Final FP-tree

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3



Item pointers are used to assist frequent itemset generation



# FP-growth Algorithm

- Scan Header Table from lowest support item up
- For each item  $i$  in Header Table extract frequent itemsets including item  $i$  and items preceding it in Header Table
  - (1) build Conditional Pattern Base for item  $i$  ( $i$ -CPB)
    - Select prefix-paths of item  $i$  from FP-tree
  - (2) recursive invocation of FP-growth on  $i$ -CPB

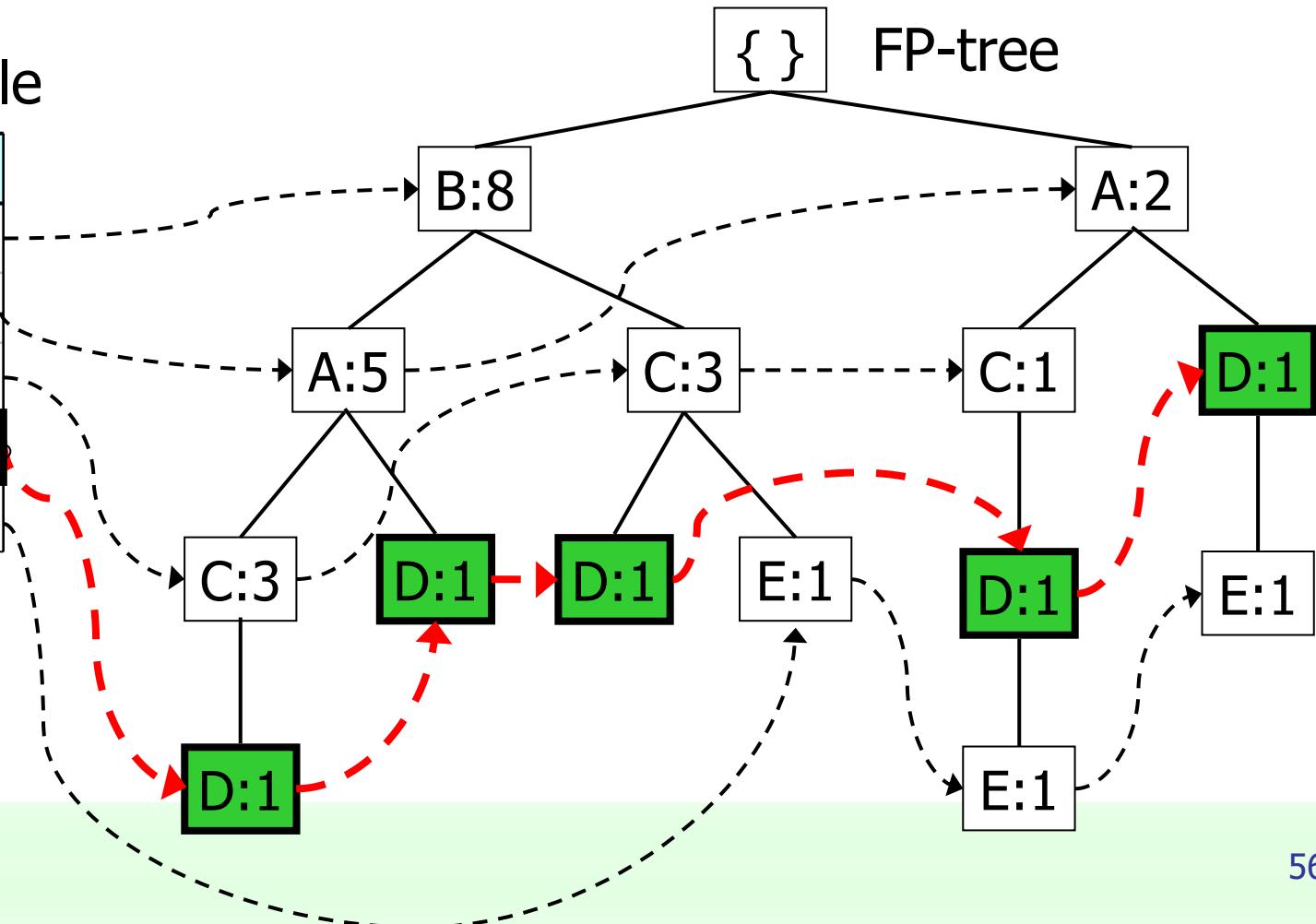


# Example

- Consider item D and extract frequent itemsets including
  - D and supported combinations of items A, B, C

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
<b>{D}</b>	<b>5</b>
{E}	3





# Conditional Pattern Base of D

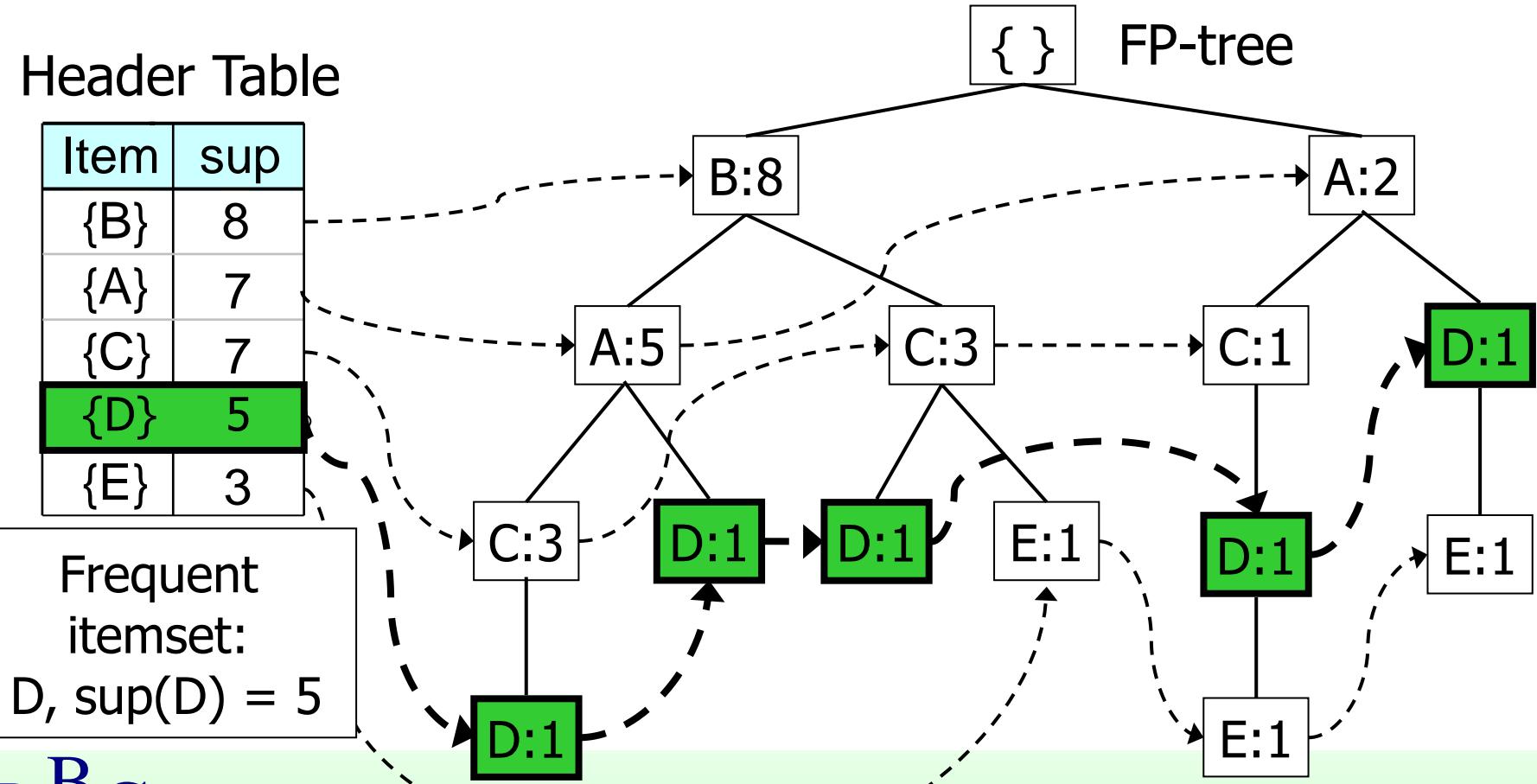
- (1) Build D-CPB
  - Select prefix-paths of item D from FP-tree

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
<b>{D}</b>	<b>5</b>
{E}	3

Frequent itemset:  
 $D$ ,  $\text{sup}(D) = 5$

$D^B_M G$





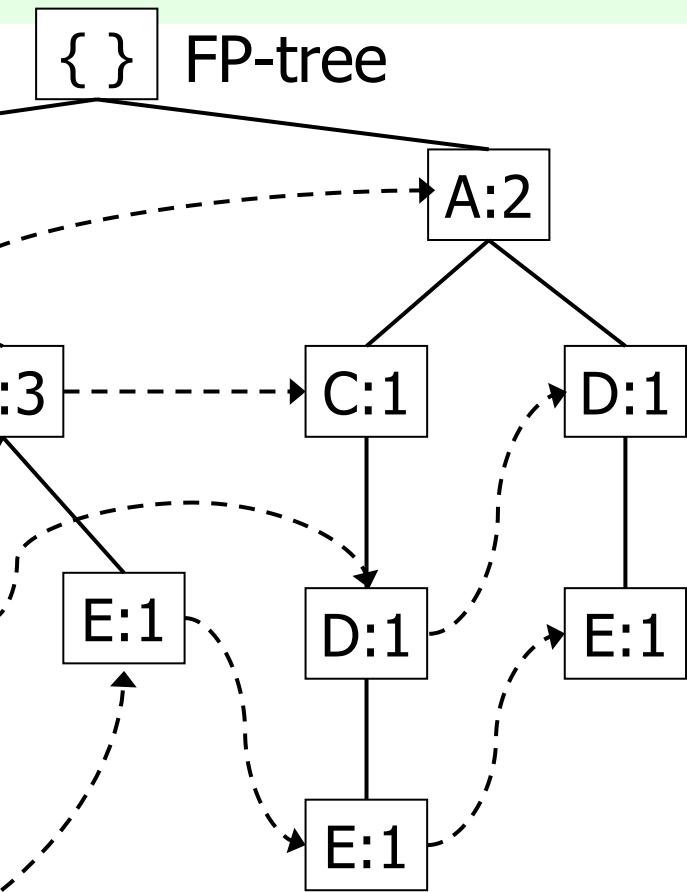
# Conditional Pattern Base of D

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

D-CPB

Items	sup
{B,A,C}	1





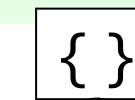
# Conditional Pattern Base of D

Header Table

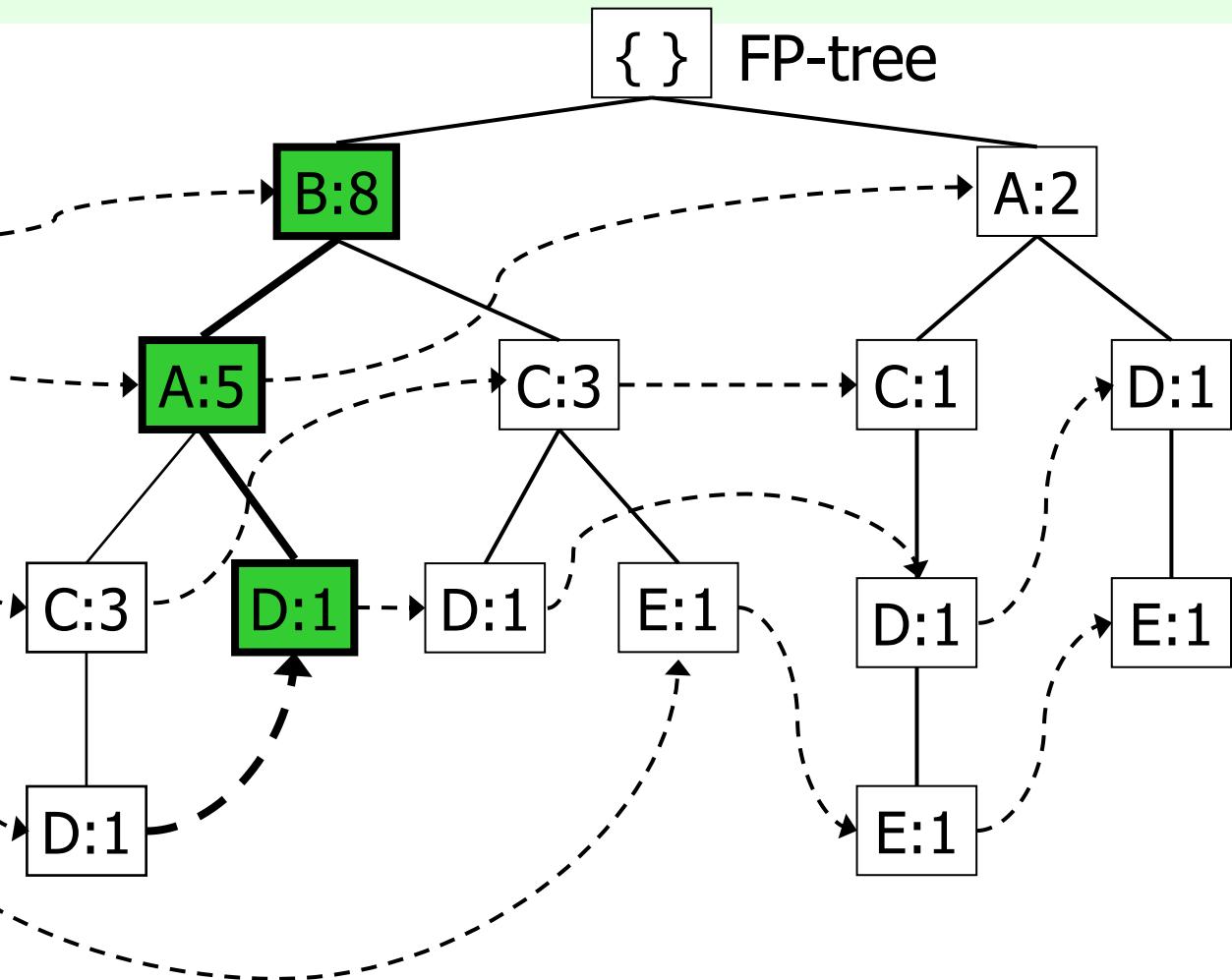
Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

D-CPB

Items	sup
{B,A,C}	1
{B,A}	1



FP-tree





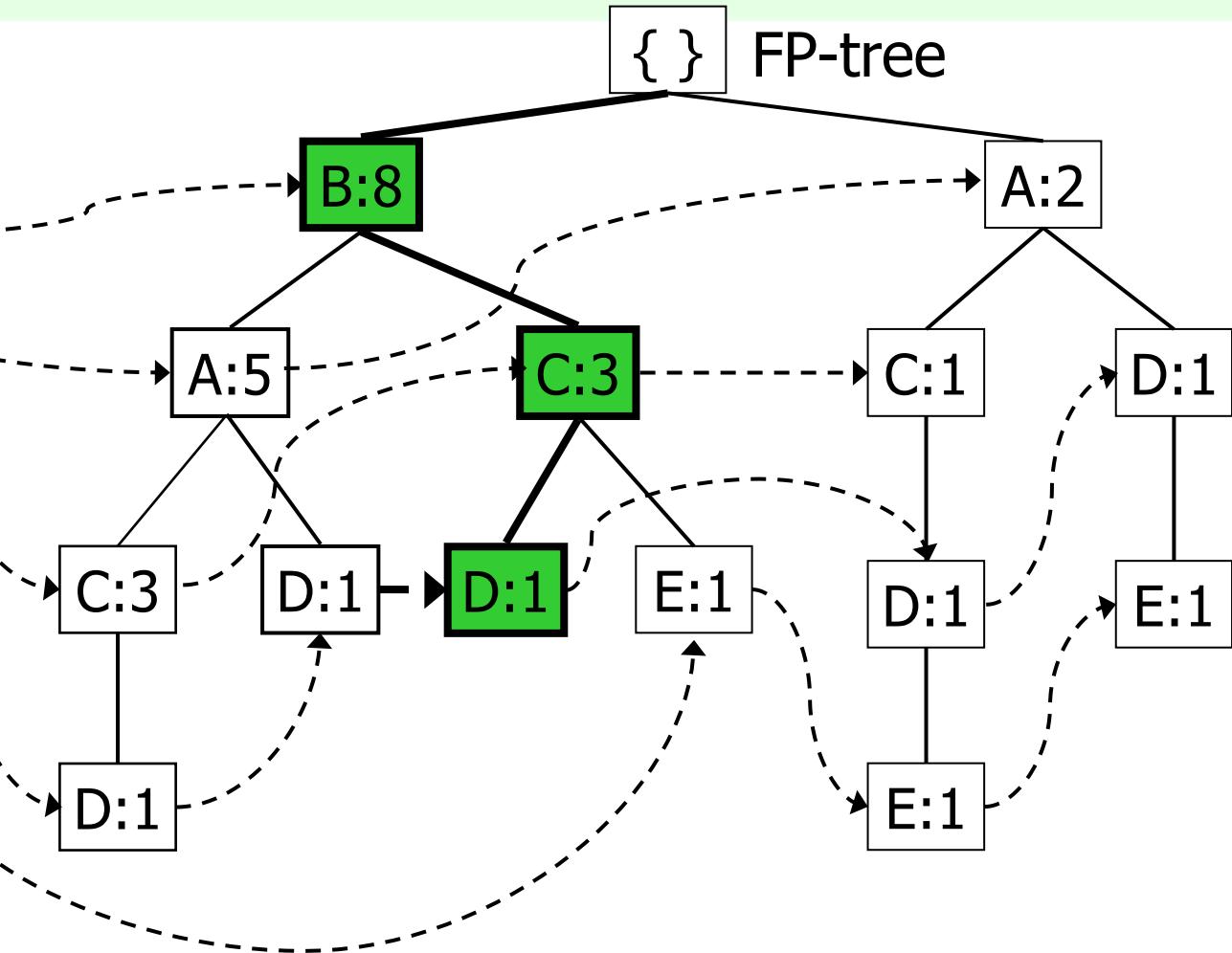
# Conditional Pattern Base of D

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
<b>{D}</b>	<b>5</b>
{E}	3

D-CPB

Items	sup
{B,A,C}	1
{B,A}	1
<b>{B,C}</b>	<b>1</b>





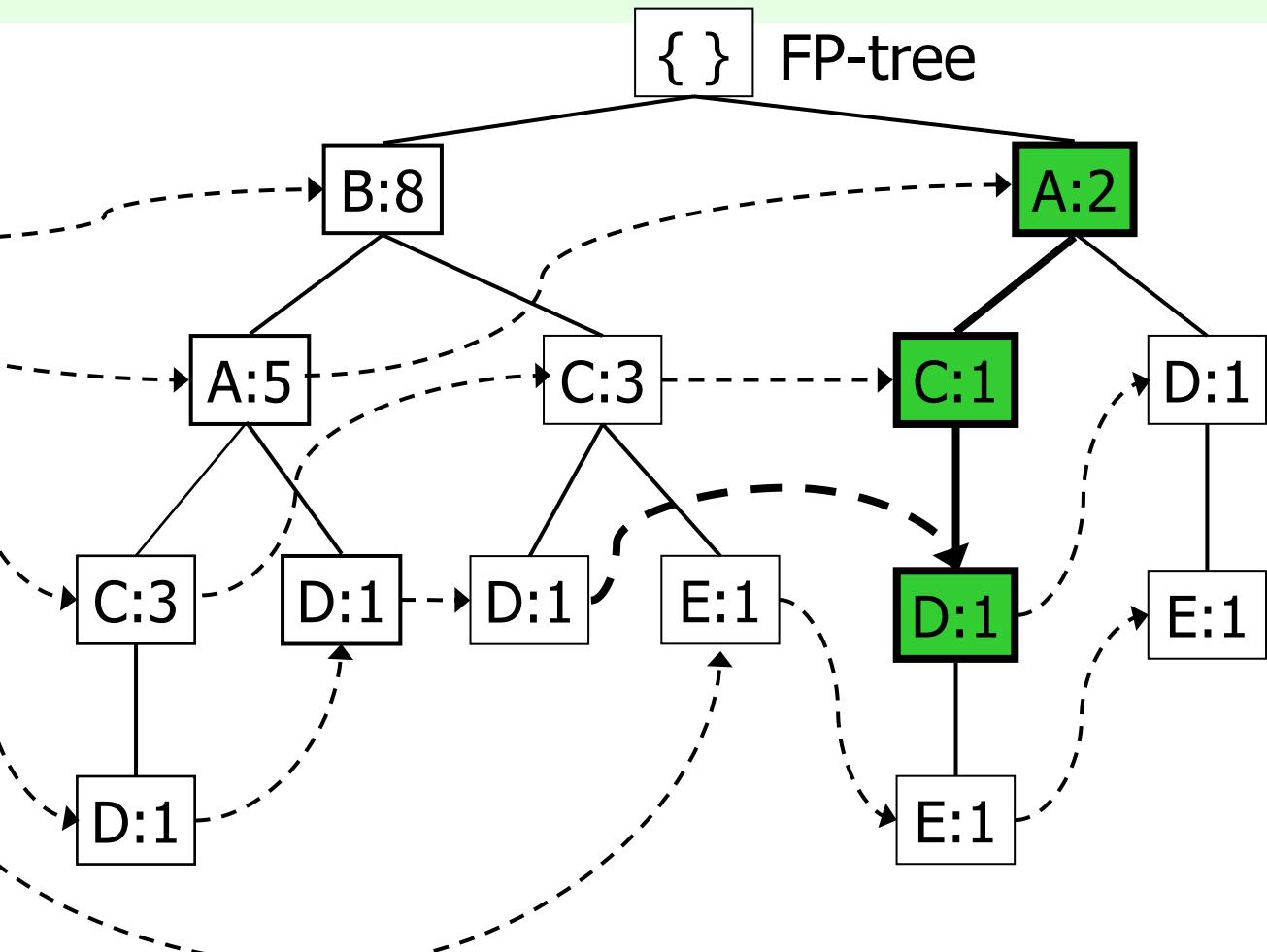
# Conditional Pattern Base of D

Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

D-CPB

Items	sup
{B,A,C}	1
{B,A}	1
{B,C}	1
{A,C}	1





# Conditional Pattern Base of D

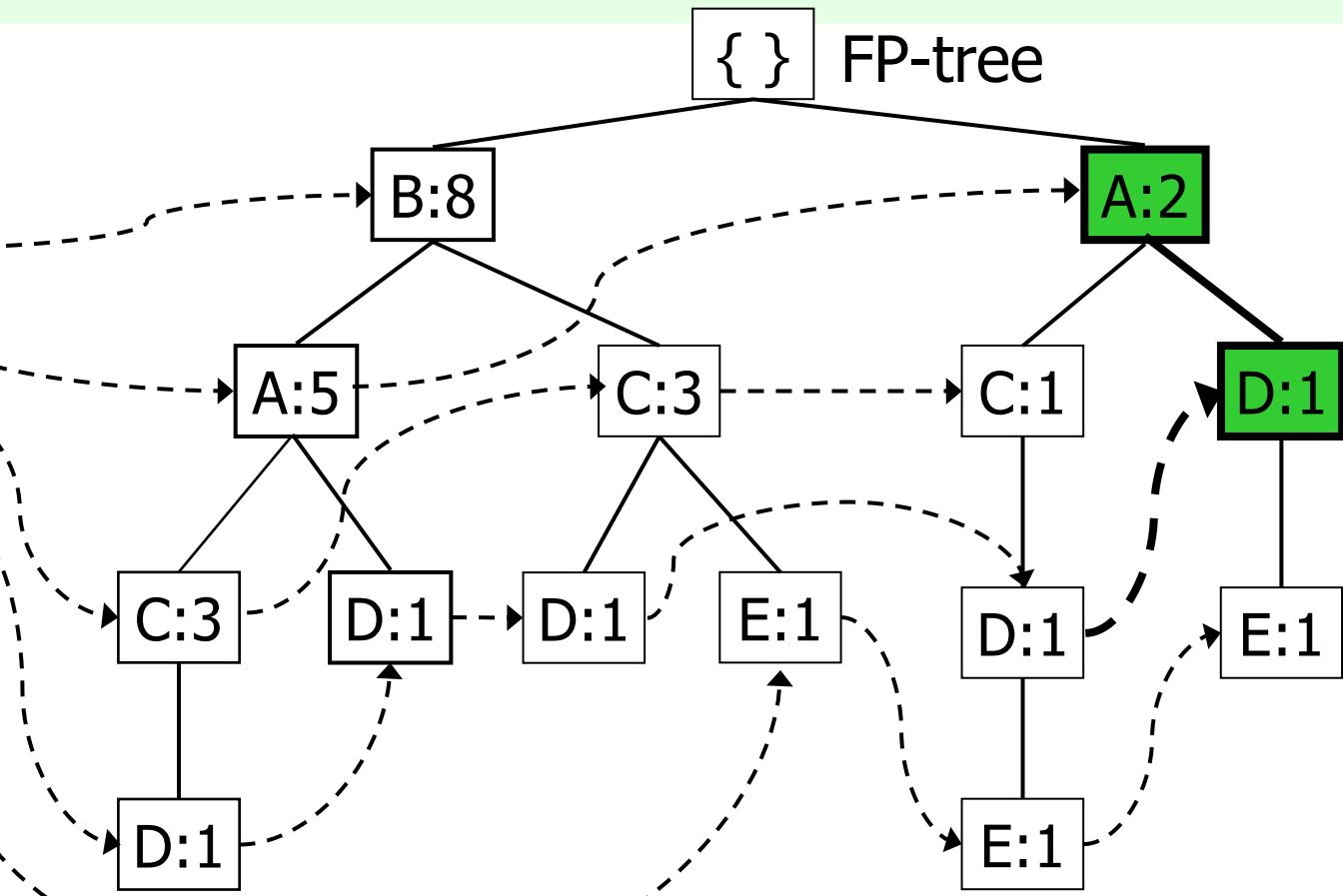
Header Table

Item	sup
{B}	8
{A}	7
{C}	7
{D}	5
{E}	3

D-CPB

Items	sup
{B,A,C}	1
{B,A}	1
{B,C}	1
{A,C}	1
{A}	1

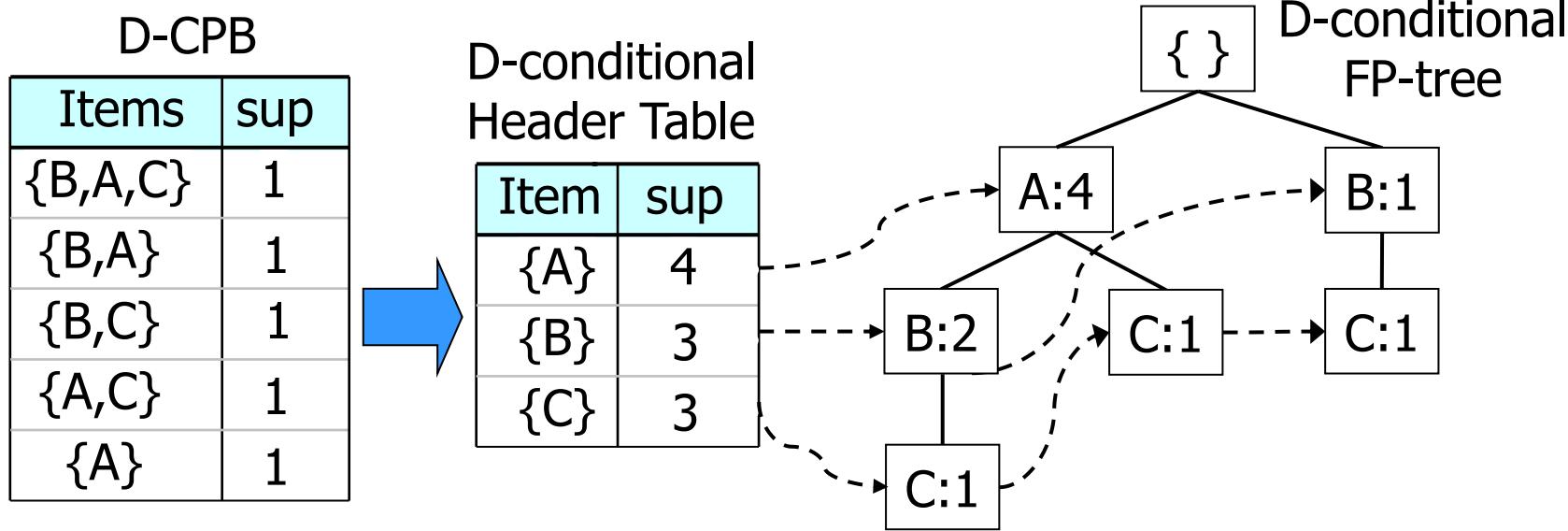
D<sup>B</sup><sub>MG</sub>





# Conditional Pattern Base of D

- (1) Build D-CPB
  - Select prefix-paths of item D from FP-tree



- (2) Recursive invocation of FP-growth on D-CPB



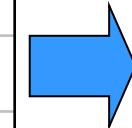
# Conditional Pattern Base of DC

- (1) Build DC-CPB
  - Select prefix-paths of item C from D-conditional FP-tree

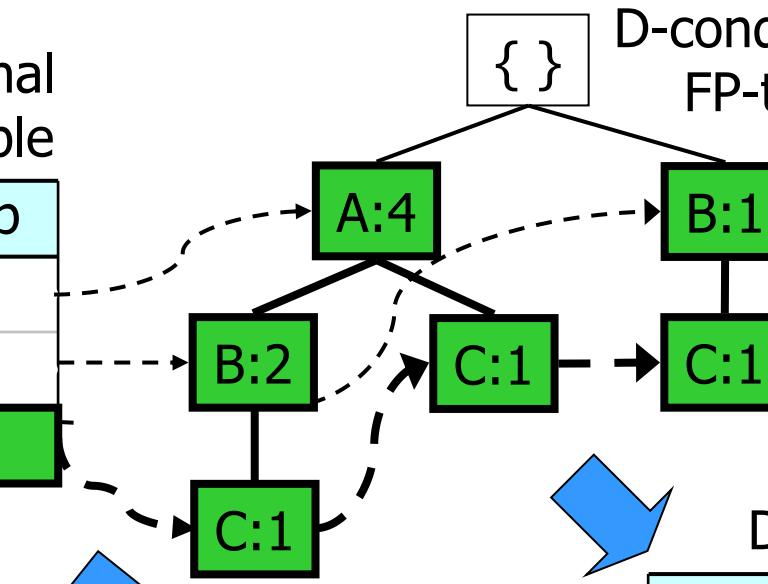
Items	sup
{B,A,C}	1
{B,A}	1
{B,C}	1
{A,C}	1
{A}	1

D-conditional Header Table

Item	sup
{A}	4
{B}	3
{C}	3



D-conditional FP-tree



DC-CPB

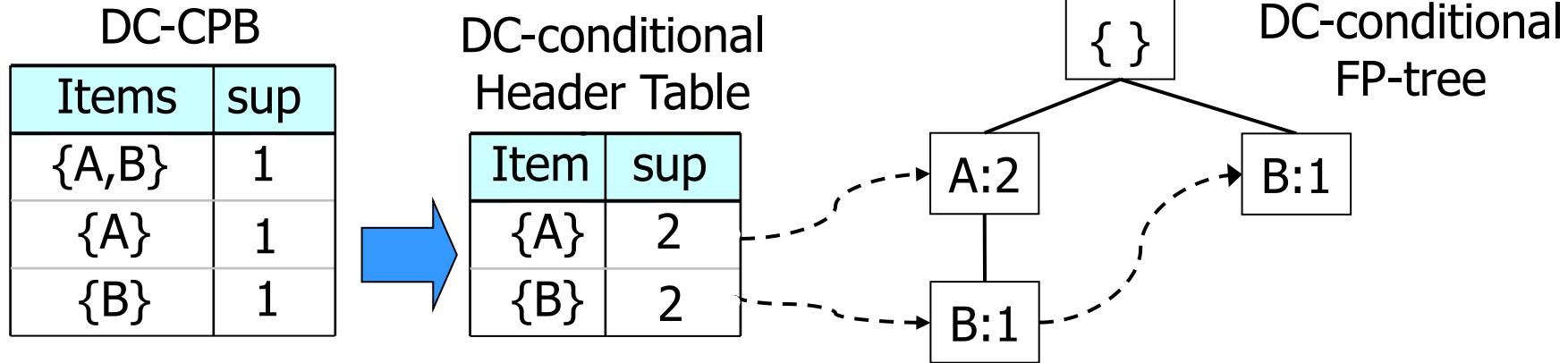
Items	sup
{A,B}	1
{A}	1
{B}	1

Frequent itemset:  
DC, sup(DC) = 3



# Conditional Pattern Base of DC

- (1) Build DC-CPB
  - Select prefix-paths of item C from D-conditional FP-tree

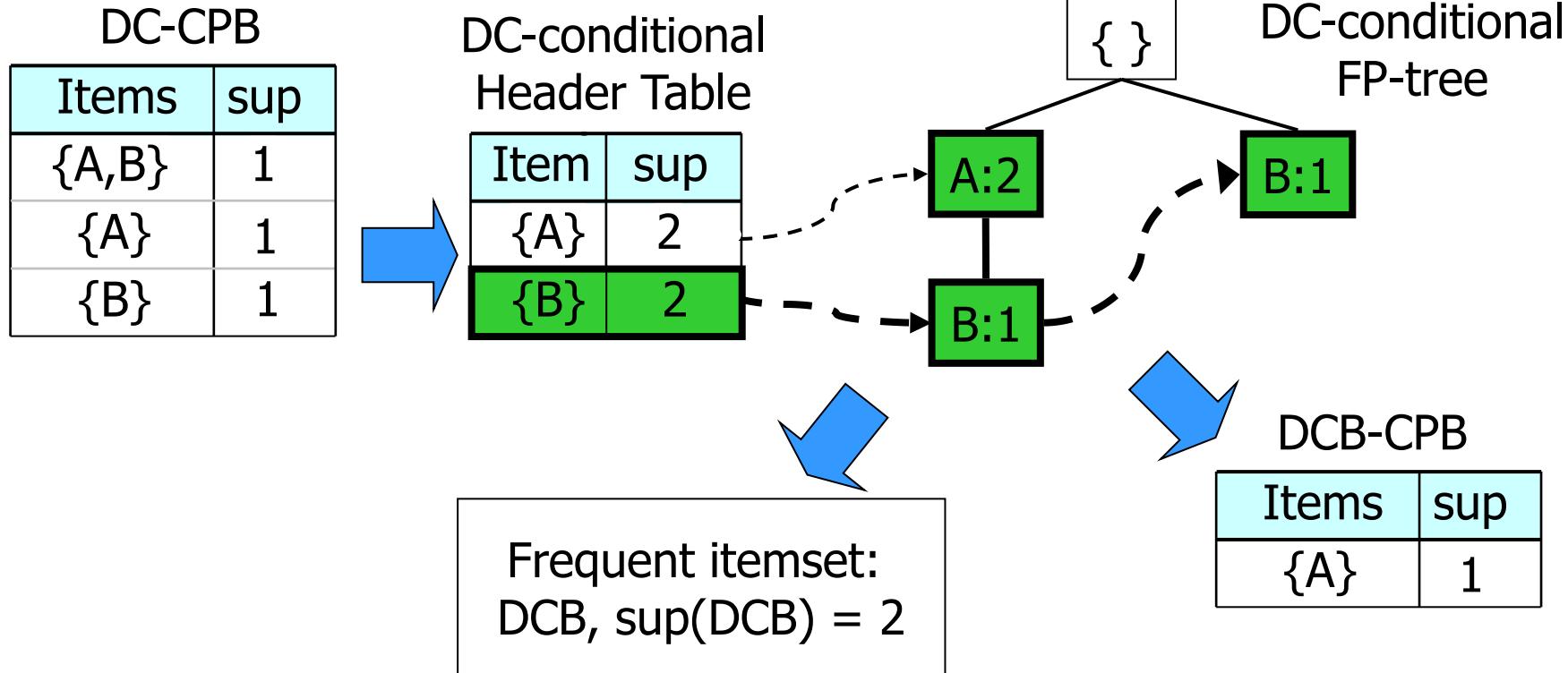


- (2) Recursive invocation of FP-growth on DC-CPB



# Conditional Pattern Base of DCB

- (1) Build DCB-CPB
  - Select prefix-paths of item B from DC-conditional FP-tree



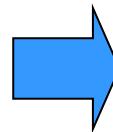


# Conditional Pattern Base of DCB

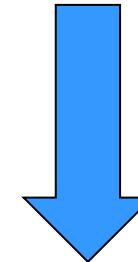
- (1) Build DCB-CPB
  - Select prefix-paths of item B from DC-conditional FP-tree

DCB-CPB

Items	sup
{A}	1



- Item A is infrequent in DCB-CPB
  - A is removed from DCB-CPB
  - DCB-CPB is empty

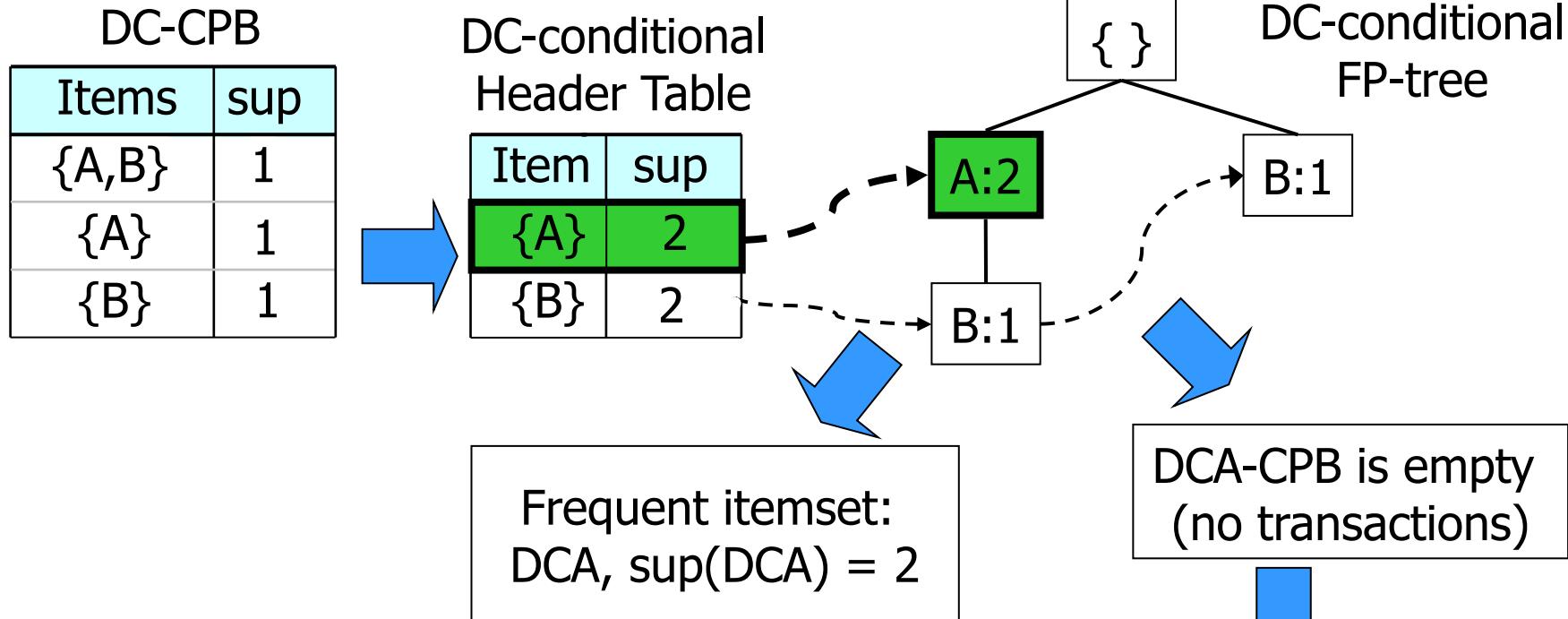


- (2) The search backtracks to DC-CPB



# Conditional Pattern Base of DCA

- (1) Build DCA-CPB
  - Select prefix-paths of item A from DC-conditional FP-tree

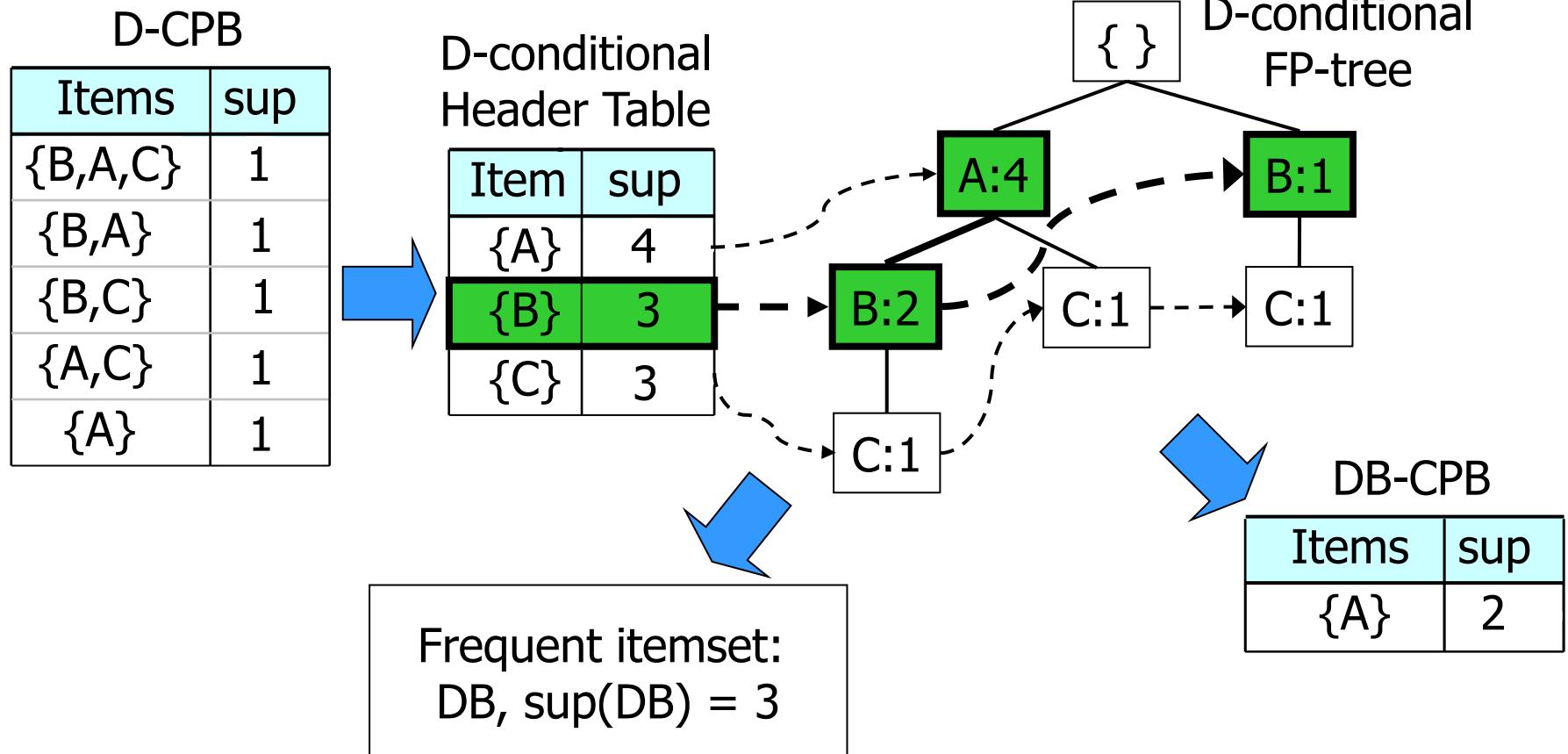


- (2) The search backtracks to D-CBP



# Conditional Pattern Base of DB

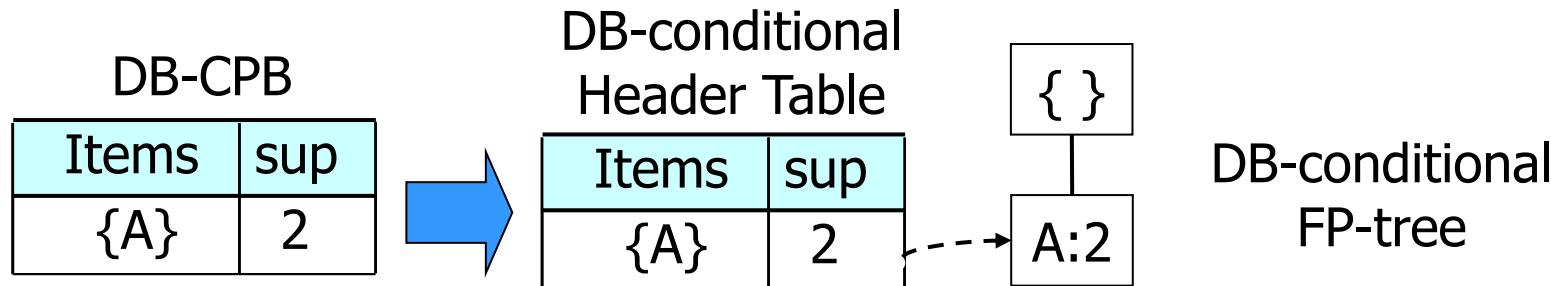
- (1) Build DB-CPB
  - Select prefix-paths of item B from D-conditional FP-tree





# Conditional Pattern Base of DB

- (1) Build DB-CPB
  - Select prefix-paths of item B from D-conditional FP-tree

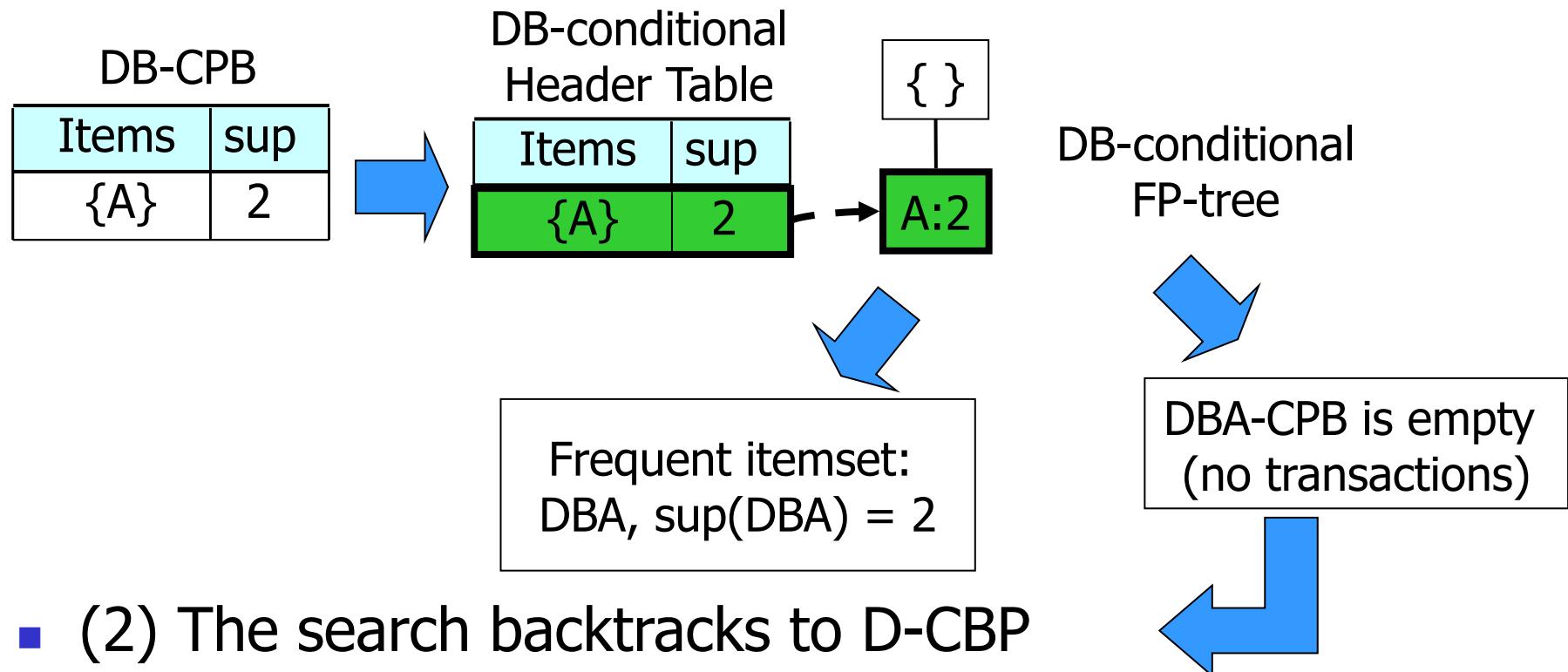


- (2) Recursive invocation of FP-growth on DB-CPB



# Conditional Pattern Base of DBA

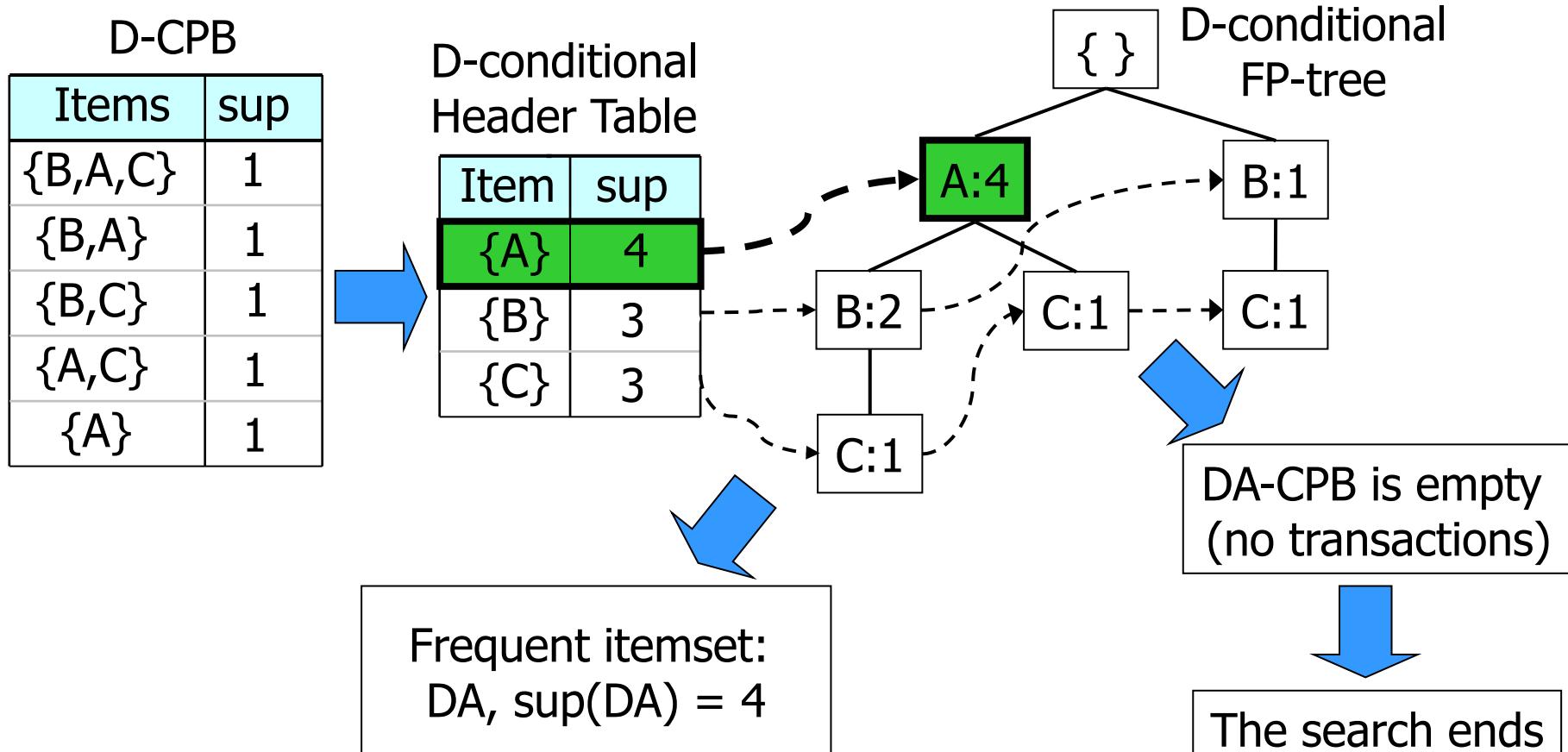
- (1) Build DBA-CPB
  - Select prefix-paths of item A from DB-conditional FP-tree





# Conditional Pattern Base of DA

- (1) Build DA-CPB
  - Select prefix-paths of item A from D-conditional FP-tree



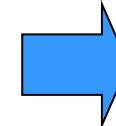


# Frequent itemsets with prefix D

- Frequent itemsets including D and supported combinations of items B,A,C

Example DB

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}



itemsets	sup
{D}	5
{A,D}	4
{B,D}	3
{C,D}	3
{A,B,D}	2
{A,C,D}	2
{B,C,D}	2

$\text{minsup} > 1$



# Other approaches

- Many other approaches to frequent itemset extraction
- May exploit a different database representation
  - represent the tidset of each item [Zak00]

Horizontal  
Data Layout

TID	Items
1	A,B,E
2	B,C,D
3	C,E
4	A,C,D
5	A,B,C,D
6	A,E
7	A,B
8	A,B,C
9	A,C,D
10	B

Vertical Data Layout

A	B	C	D	E
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				



# Compact Representations

- Some itemsets are redundant because they have identical support as their supersets

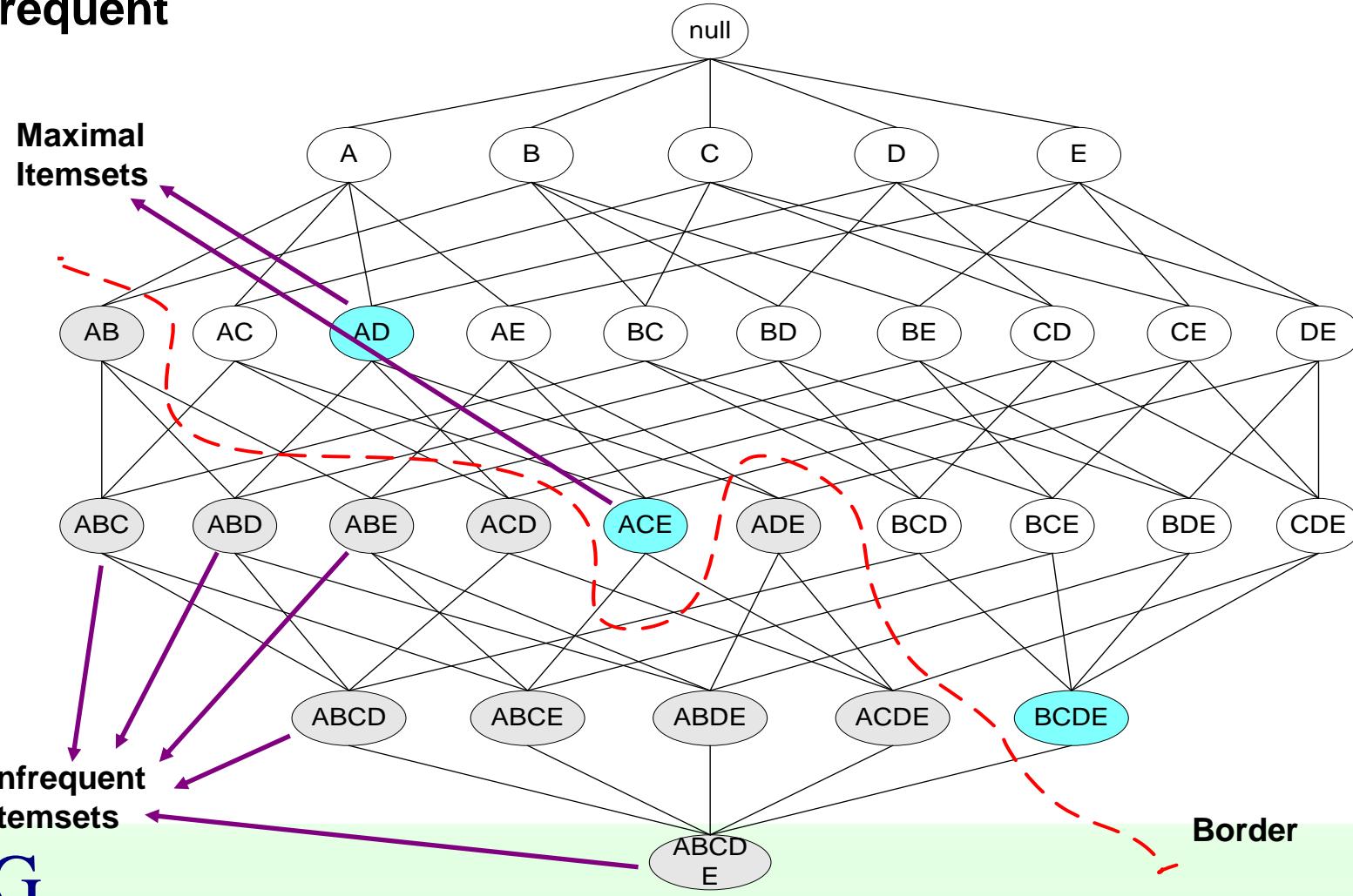
TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

- Number of frequent itemsets =  $3 \times \sum_{k=1}^{10} \binom{10}{k}$
- A compact representation is needed



# Maximal Frequent Itemset

An itemset is frequent maximal if none of its immediate supersets is frequent





# Closed Itemset

- An itemset is closed if none of its immediate supersets has the same support as the itemset

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

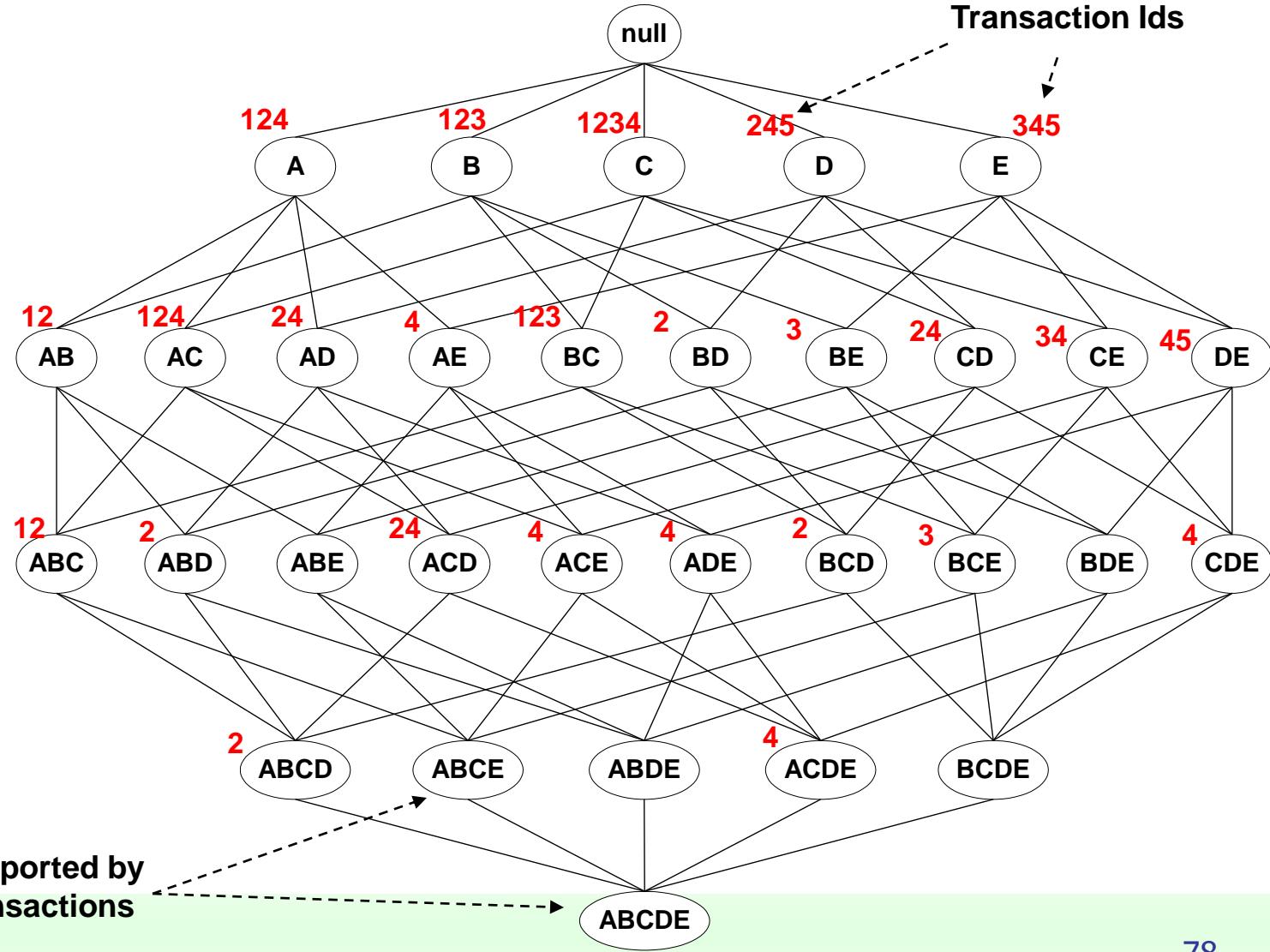
itemset	sup
{A}	4
<b>{B}</b>	<b>5</b>
{C}	3
{D}	4
<b>{A,B}</b>	<b>4</b>
{A,C}	2
{A,D}	3
{B,C}	3
<b>{B,D}</b>	<b>4</b>
{C,D}	3

itemset	sup
{A,B,C}	2
<b>{A,B,D}</b>	<b>3</b>
{A,C,D}	2
<b>{B,C,D}</b>	<b>3</b>
<b>{A,B,C,D}</b>	<b>2</b>



# Maximal vs Closed Itemsets

TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE

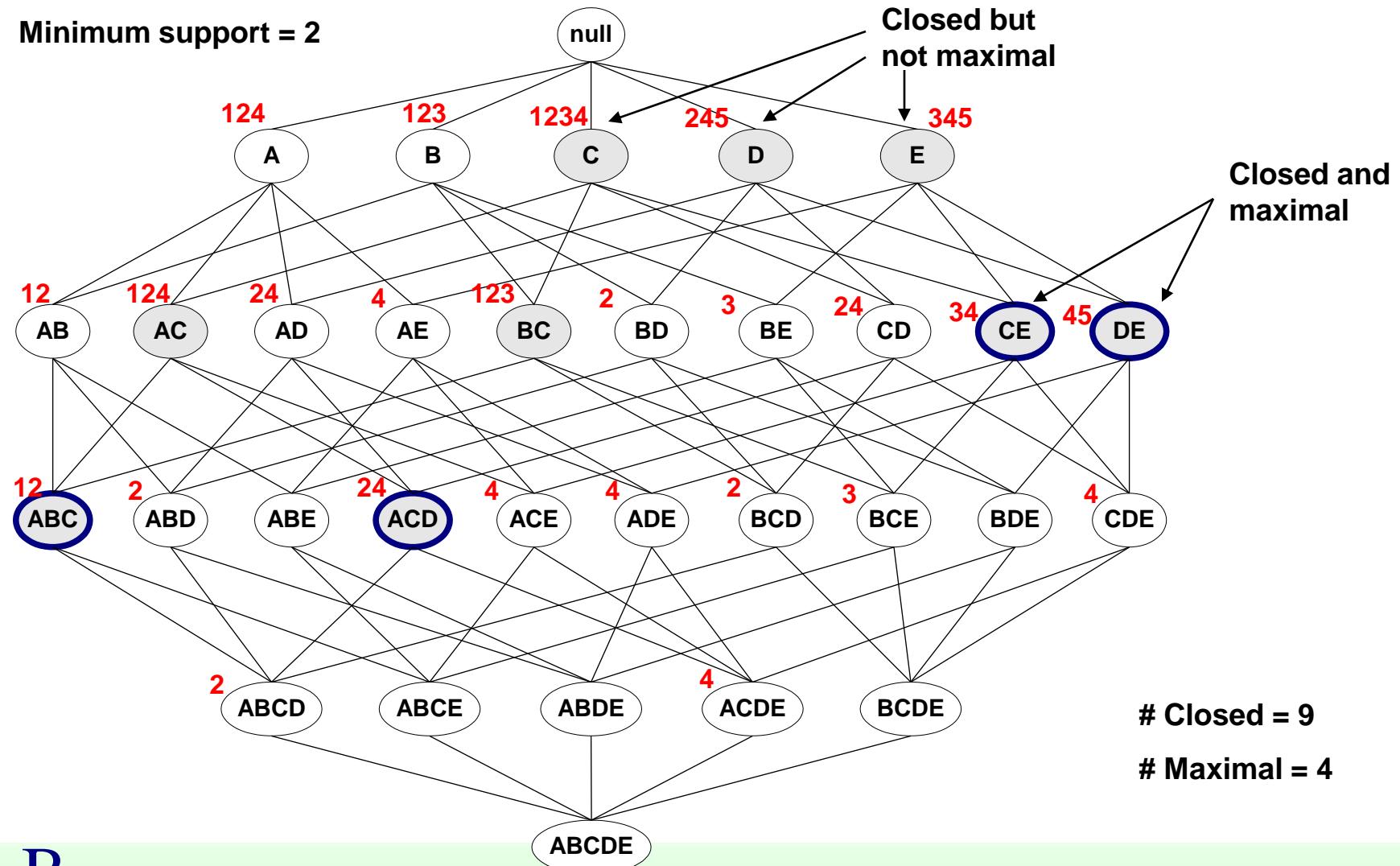


From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



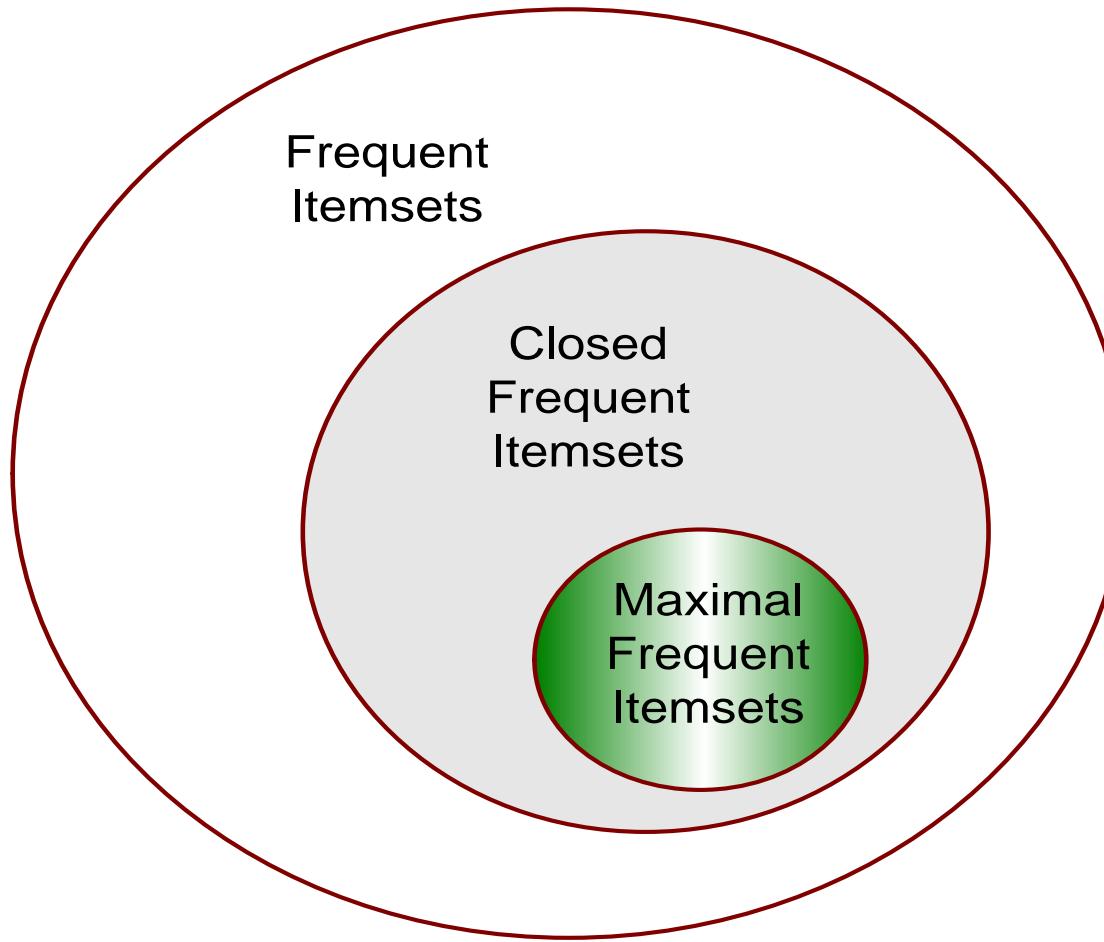
# Maximal vs Closed Frequent Itemsets

Minimum support = 2





# Maximal vs Closed Itemsets



From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



# Effect of Support Threshold

- Selection of the appropriate *minsup* threshold is not obvious
  - If *minsup* is too high
    - itemsets including rare but interesting items may be lost
      - example: pieces of jewellery (or other expensive products)
  - If *minsup* is too low
    - it may become computationally *very expensive*
    - the number of frequent itemsets becomes *very large*



# Interestingness Measures

- A large number of patterns may be extracted
  - rank patterns by their interestingness
- Objective measures
  - rank patterns based on statistics computed from data
  - initial framework [Agr94] only considered support and confidence
    - other statistical measures available
- Subjective measures
  - rank patterns according to user interpretation [Silb98]
    - interesting if it contradicts the expectation of a user
    - interesting if it is actionable



# Confidence measure: always reliable?

- 5000 high school students are given
  - 3750 eat cereals
  - 3000 play basket
  - 2000 eat cereals and play basket
- Rule

play basket  $\Rightarrow$  eat cereals  
sup = 40%, conf = 66,7%

is misleading because eat cereals has sup 75% ( $>66,7\%$ )

- Problem caused by high frequency of rule head
  - negative correlation

	basket	not basket	total
cereals	2000	1750	3750
not cereals	1000	250	1250
total	3000	2000	5000



# Correlation or lift

$r: A \Rightarrow B$

$$\text{Correlation} = \frac{P(A, B)}{P(A)P(B)} = \frac{\text{conf}(r)}{\text{sup}(B)}$$

- Statistical independence
  - Correlation = 1
- Positive correlation
  - Correlation > 1
- Negative correlation
  - Correlation < 1



# Example

- Association rule

play basket  $\Rightarrow$  eat cereals

has corr = 0.89

- negative correlation

- but rule

play basket  $\Rightarrow$  not (eat cereals)

has corr = 1,34



#	Measure	Formula
1	$\phi$ -coefficient	$\frac{P(A,B) - P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	Goodman-Kruskal's ( $\lambda$ )	$\frac{\sum_j \max_k P(A_j, B_k) + \sum_k \max_j P(A_j, B_k) - \max_j P(A_j) - \max_k P(B_k)}{2 - \max_j P(A_j) - \max_k P(B_k)}$
3	Odds ratio ( $\alpha$ )	$\frac{P(A,B)P(\bar{A},\bar{B})}{P(\bar{A},B)P(\bar{A},\bar{B})}$
4	Yule's $Q$	$\frac{P(A,B)P(\bar{A}\bar{B}) - P(A,\bar{B})P(\bar{A},B)}{P(A,B)P(\bar{A}\bar{B}) + P(A,\bar{B})P(\bar{A},B)} = \frac{\alpha-1}{\alpha+1}$
5	Yule's $Y$	$\frac{\sqrt{P(A,B)P(\bar{A}\bar{B})} - \sqrt{P(A,\bar{B})P(\bar{A},B)}}{\sqrt{P(A,B)P(\bar{A}\bar{B})} + \sqrt{P(A,\bar{B})P(\bar{A},B)}} = \frac{\sqrt{\alpha}-1}{\sqrt{\alpha}+1}$
6	Kappa ( $\kappa$ )	$\frac{P(A,B) + P(\bar{A},\bar{B}) - P(A)P(B) - P(\bar{A})P(\bar{B})}{1 - P(A)P(B) - P(\bar{A})P(\bar{B})}$ $\frac{\sum_i \sum_j P(A_i, B_j) \log \frac{P(A_i, B_j)}{P(A_i)P(B_j)}}{\min(-\sum_i P(A_i) \log P(A_i), -\sum_j P(B_j) \log P(B_j))}$
7	Mutual Information ( $M$ )	
8	J-Measure ( $J$ )	$\max \left( P(A,B) \log \left( \frac{P(B A)}{P(B)} \right) + P(\bar{A}\bar{B}) \log \left( \frac{P(\bar{B} \bar{A})}{P(\bar{B})} \right), P(A,B) \log \left( \frac{P(A B)}{P(A)} \right) + P(\bar{A}\bar{B}) \log \left( \frac{P(\bar{A} B)}{P(\bar{A})} \right) \right)$
9	Gini index ( $G$ )	$\max \left( P(A)[P(B A)^2 + P(\bar{B} A)^2] + P(\bar{A})[P(B \bar{A})^2 + P(\bar{B} \bar{A})^2] - P(B)^2 - P(\bar{B})^2, P(B)[P(A B)^2 + P(\bar{A} B)^2] + P(\bar{B})[P(A \bar{B})^2 + P(\bar{A} \bar{B})^2] - P(A)^2 - P(\bar{A})^2 \right)$
10	Support ( $s$ )	$P(A,B)$
11	Confidence ( $c$ )	$\max(P(B A), P(A B))$
12	Laplace ( $L$ )	$\max \left( \frac{NP(A,B)+1}{NP(A)+2}, \frac{NP(A,B)+1}{NP(B)+2} \right)$
13	Conviction ( $V$ )	$\max \left( \frac{P(A)P(\bar{B})}{P(A\bar{B})}, \frac{P(B)P(\bar{A})}{P(B\bar{A})} \right)$
14	Interest ( $I$ )	$\frac{P(A,B)}{P(A)P(B)}$
15	cosine ( $IS$ )	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
16	Piatetsky-Shapiro's ( $PS$ )	$P(A,B) - P(A)P(B)$
17	Certainty factor ( $F$ )	$\max \left( \frac{P(B A)-P(B)}{1-P(B)}, \frac{P(A B)-P(A)}{1-P(A)} \right)$
18	Added Value ( $AV$ )	$\max(P(B A) - P(B), P(A B) - P(A))$
19	Collective strength ( $S$ )	$\frac{P(A,B) + P(\bar{A}\bar{B})}{P(A)P(B) + P(\bar{A})P(\bar{B})} \times \frac{1 - P(A)P(B) - P(\bar{A})P(\bar{B})}{1 - P(A,B) - P(\bar{A}\bar{B})}$ $\frac{P(A,B)}{P(A) + P(B) - P(A,B)}$
20	Jaccard ( $\zeta$ )	
21	Klosgen ( $K$ )	$\sqrt{P(A,B)} \max(P(B A) - P(B), P(A B) - P(A))$



# Considering weight

- Items may be characterized by different importance within a transaction
  - Example: product quantity or price in transactions
- Transactions may be weighted
  - Example: discount on entire market basket
- Weighted dataset
  - Each item is assigned a weight measuring its relevance in the corresponding transaction



# Weighted association rules

- Consider item/transaction weights during association rule extraction
- Extend rule quality measures
  - E.g., weighted support, weighted confidence
- Apply ad-hoc weight aggregation functions
  - E.g., min, max, avg



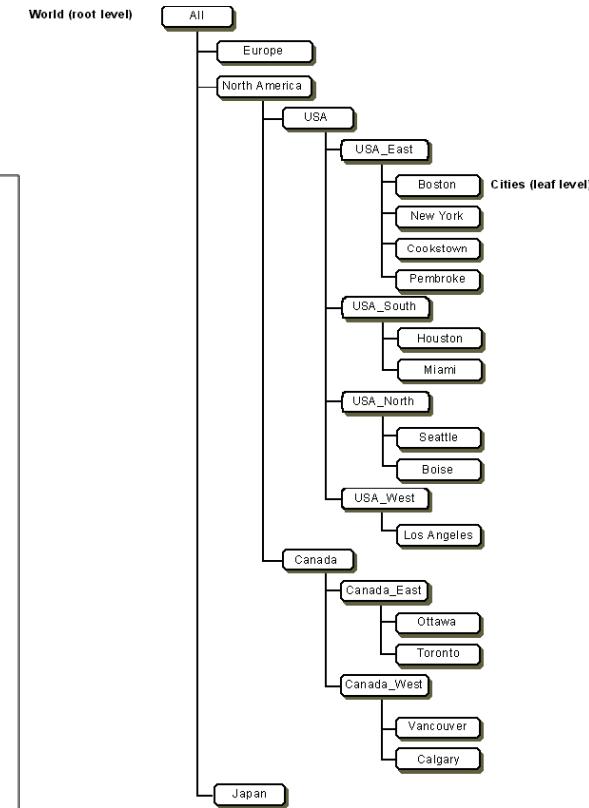
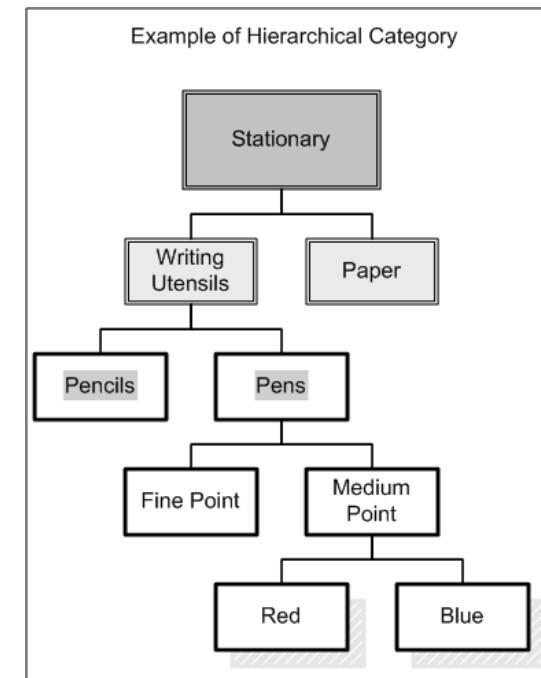
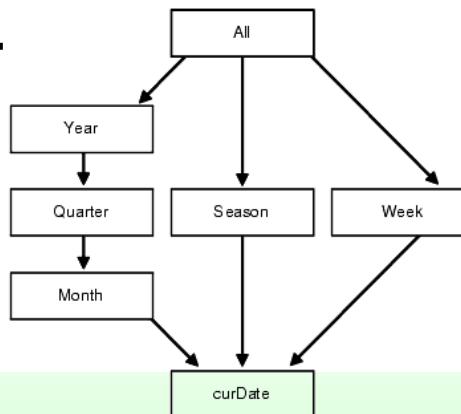
# Considering hierarchies

## ■ Generalization hierarchies

- Aggregation over attributes in a dataset
- Typically user provided

## ■ Examples

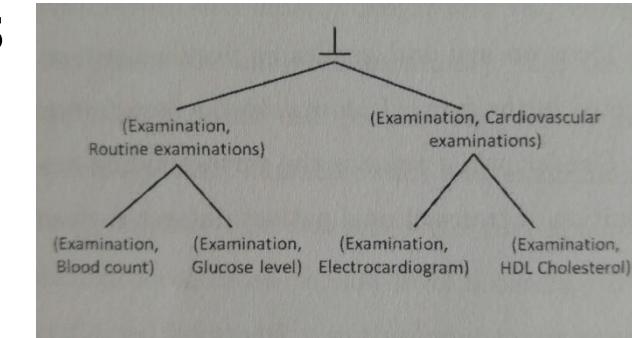
- Time hierarchy
- Product category
- Location hierarchy





# Taxonomy

- A taxonomy is a set of is-a hierarchies that aggregate data items into higher-level concepts
- Data item
  - Instance in the (transactional) dataset
  - Represents detailed concepts
- Generalized item
  - Aggregation in higher-level concepts
  - Represents abstractions on instances





# Generalized itemsets

- Sets of items at different generalization levels
  - May contain data items together with generalized items defined in the taxonomy
  - Summarize knowledge represented by a set of lower-level descendants
    - Both frequent and infrequent
- A generalized itemset covers a transaction when all
  - its generalized items are ancestors of items included in the transaction
  - its data items are included in the transaction
- Generalized itemset support
  - ratio between number of covered transactions and dataset cardinality



# Context-aware data analysis

- Context data provided by different, possibly heterogeneous, sources
  - Mobile devices provide information on
    - the user context (e.g., GPS coordinates)
    - the supplied services
      - temporal information
      - service description
      - duration
  - Additional information available
    - demographics of the user requesting the service



# Generalized itemset example

***user: John, time: 6.05 p.m., service: Weather***  
**(s = 0.005%)**

- A very low support
  - The itemset may be discarded
- By generalizing
  - the time attribute on a time period
  - the user on a user category

***user: employee, time: 6 p.m. to 7 p.m., service: Weather***  
**(s = 0.2%)**

- May discover interesting properties generalizing *infrequent* items



# Generalized association rules

- Extension of “classical” association rules

$$X \rightarrow Y$$

- X and Y are either generalized or not generalized itemsets
  - Extract associations among data items at multiple abstraction levels
  - Support, confidence and lift are defined accordingly



# Patient data analysis

- Analysis of multiple level correlations on patient treatment historical data
  - Dataset collected by an Italian Local Health Center
    - Diabetes complications at various severity levels
    - 95K records, 3.5K patients
  - Features
    - Prescribed examinations (26 examinations, 7 categories)
    - Prescribed drugs (200 drugs, 14 categories)
    - Census patient data (gender, age discretized in age groups)
- Sparse dataset
  - Difficult setting of support threshold
    - Low: generates too many rules
    - High: interesting information at lower levels of abstraction may remain hidden



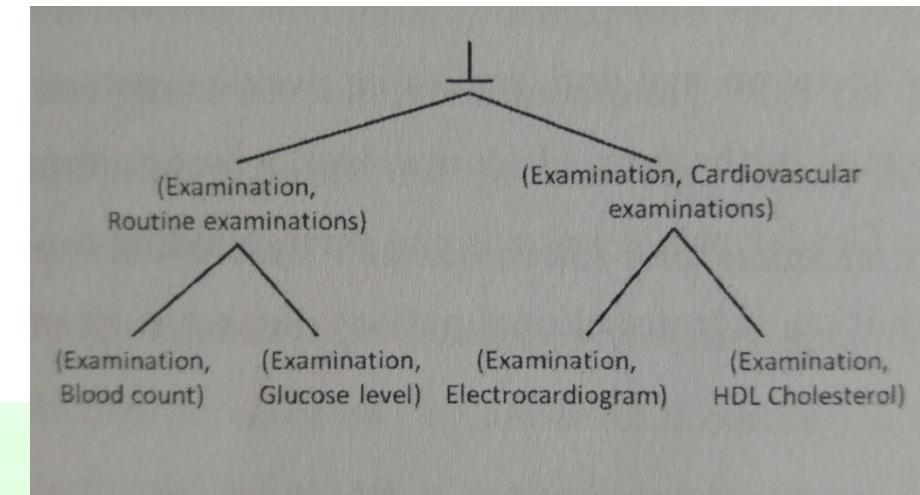
# High level rules

- Only generalized itemsets
  - Represent general knowledge
    - May be too high level to perform targeted analyses

- Example

(Examination, Liver) -> (Examination, Kidney)

- Frequently prescribed together
- May be used for examination scheduling





# Cross level rules

- Different abstraction levels (generalized items and data items)
  - Combine detailed and general information
- Example
  - (Examination, Liver) -> (Examination, Uric acid)
  - Insight into specific kidney examinations correlated with liver examinations
    - Confidence: 74.8%



# Low-level rules

- Only not generalized itemsets (only data items)
  - Very detailed knowledge
    - Covered by high and cross-level rules
  - Large rule set
    - Challenging exploration task
  - Drill down exploration based on formerly extracted high and cross-level rules



POLITECNICO  
DI TORINO

# Data Science Lab

## Data Exploration

DataBase and Data Mining Group

Tania Cerquitelli and Elena Baralis

- It includes a set of preliminary analyses that allows exploring and characterizing the data
- It is a very important step that aims to better design all the steps of the KDD pipeline
  - Quality of data has an impact on the quality of extracted knowledge
  - Understanding the input data allows the data scientist to make better decision on further and deeper analysis
  - Time saving

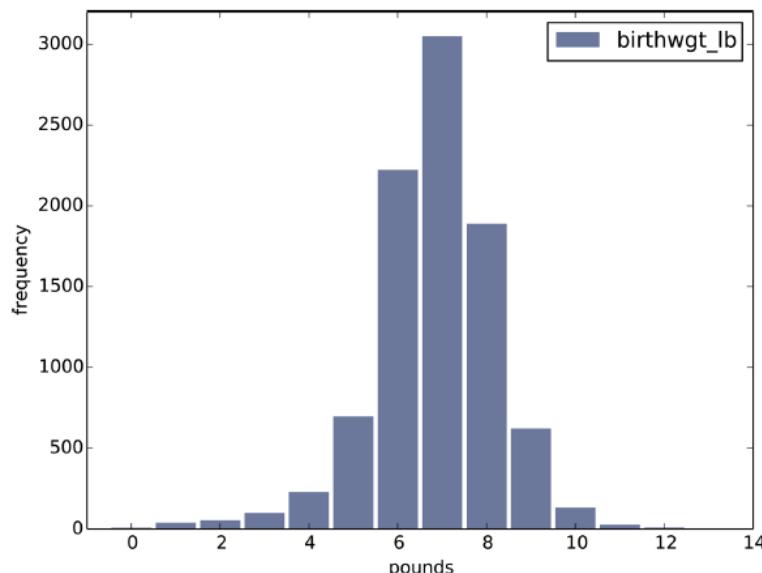
- A dataset is a collection of data
  - e.g., a tabular representation of data includes rows and columns
    - **Rows** correspond to objects, records, point, case, sample, entity, or instance
    - **columns** are the attributes
- The size of the dataset has an impact on the choice of the analyses
  - Some algorithms require considerable hardware resources when applied to large datasets, in some cases it is not possible to execute them at all.
  - There are solutions to reduce the size of the dataset preserving the completeness of the data
    - data sampling can reduce the dataset size in terms of number of rows
    - feature selection can reduce the number of attributes

- Each column of the dataset represents one attribute/feature
  - Data exploration can be performed in a univariate or multivariate fashions
- For further analysis please consider the following basic information for each attribute
  - Unit of measurement
  - Attribute Type
    - Categorical (not numerical or fixed number of possible values)
    - Continuous (numerical)
  - Attribute Domain
    - It is a good practice to verify if the attribute values satisfy the domain-driven constraints

# Univariate analysis: Distribution

## Distribution of the attribute

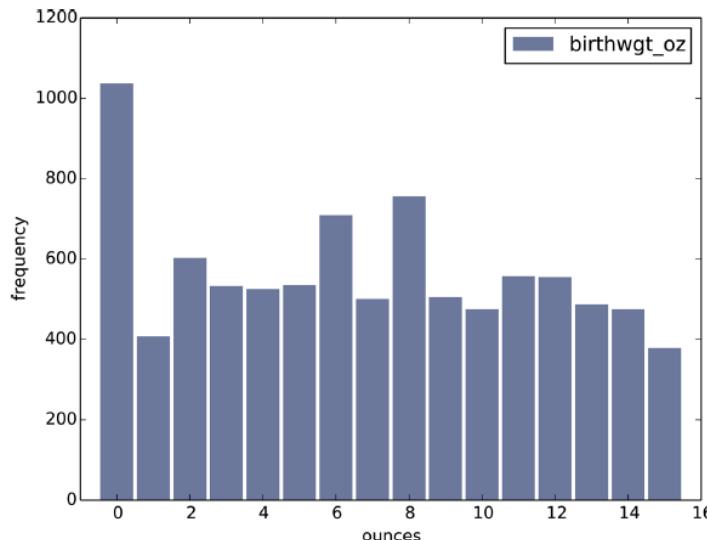
- Attribute description through the plot that shows for each attribute value how many times it appears in the dataset
- The most common representation of a distribution is a **histogram**
  - A graph that shows the **frequency** of each value.



- Example of a histogram that shows the distribution of the pound part of birth weight
  - The distribution is approximately bell-shaped, which is the shape of the **normal** distribution

# Univariate analysis: Distribution

- Distribution of the attribute
  - Attribute description through the plot that shows for each value how many times it appears in the dataset.
  - The most common representation of a distribution is a **histogram**
    - A graph that shows the **frequency** of each value.



- Example of a histogram that shows the distribution of the ounces part of birth weight
  - This distribution is not uniform
    - 0 is more common than the other values,
    - 1 and 15 are less common, probably because respondents round off birth weights that are close to an integer value.

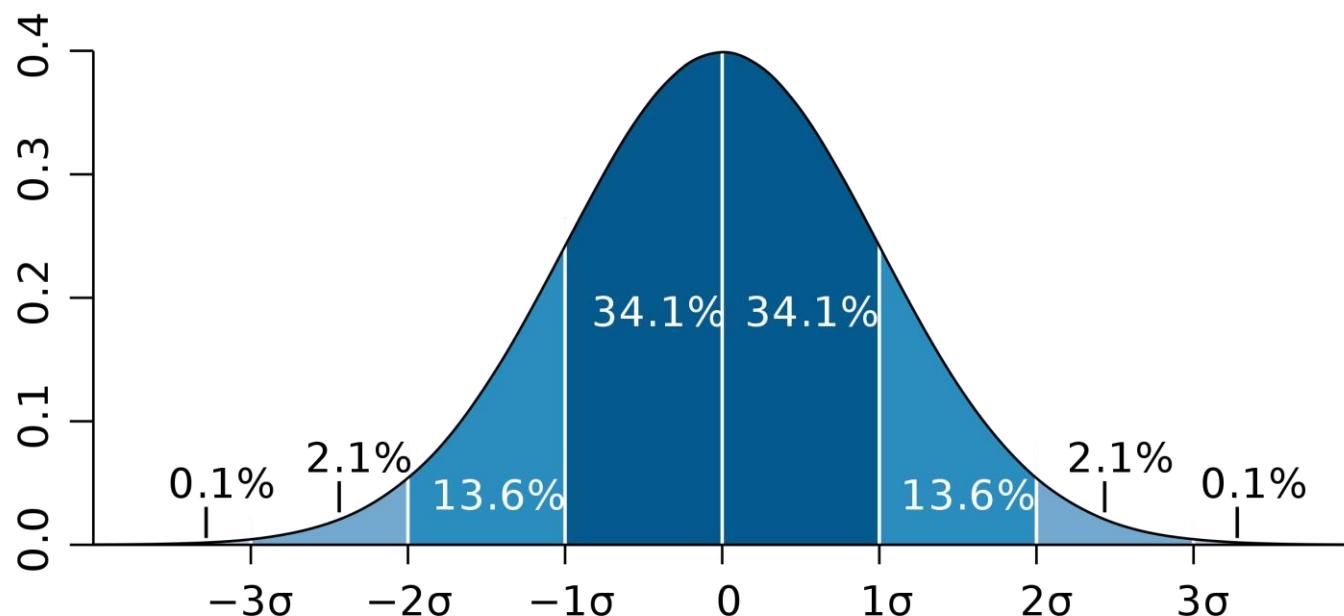
# Characterizing Distributions

- Given a distribution
  - Minimum value
  - Maximum value
  - Mean value
  - Number of samples
  - Standard deviation
  - Mathematical functions
    - A **probability distribution** that provides the probabilities of occurrence of different possible values in a dataset
      - Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean
      - Kurtoisis is a measure of the "tailedness" of the probability distribution of a real-valued random variable
    - ...

- Probability distributions are usually classified as
  - **Continuous probability distribution**
    - The set of possible outcomes can assume values in a continuous range
    - It is typically described by a probability density functions (modelling)
  - **Discrete probability distribution**
    - Characterized by a discrete list of the probabilities of the outcomes
    - It is typically described by a probability mass function (description)

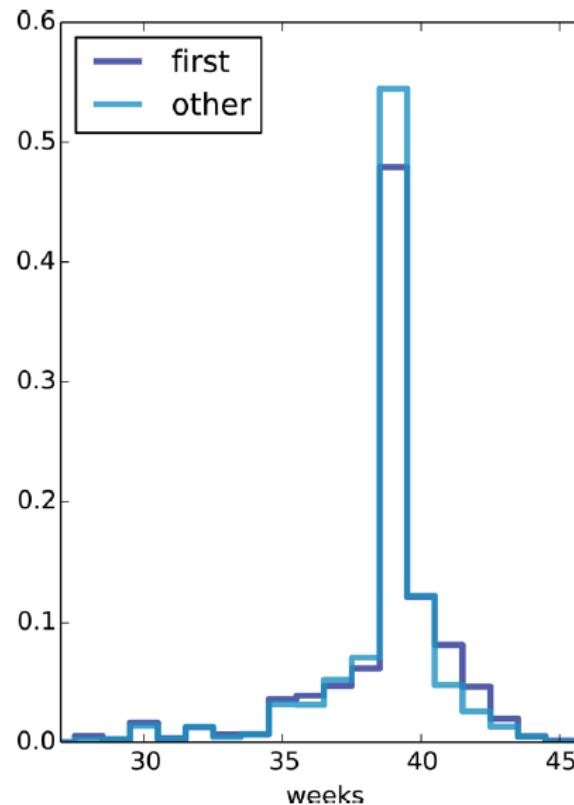
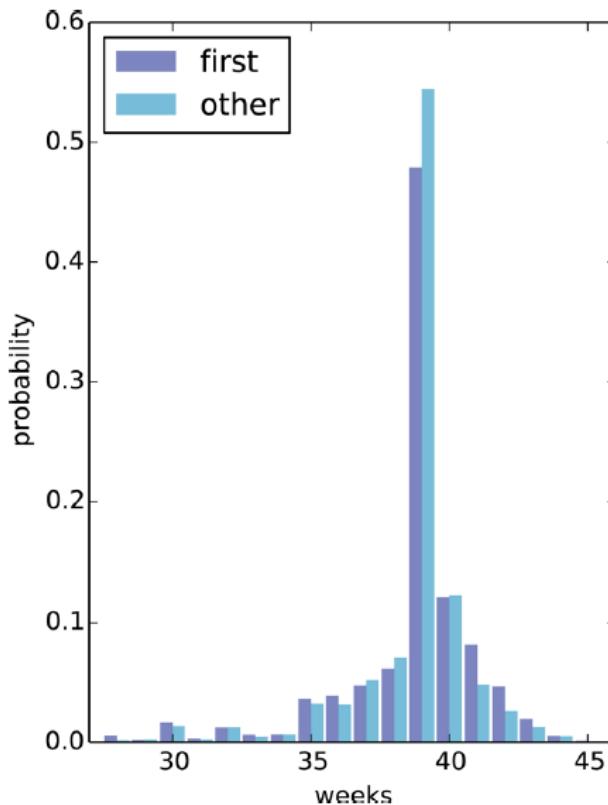
# Continuous probability

- Example of the **probability density function** (PDF) of the normal distribution



# Discrete probability

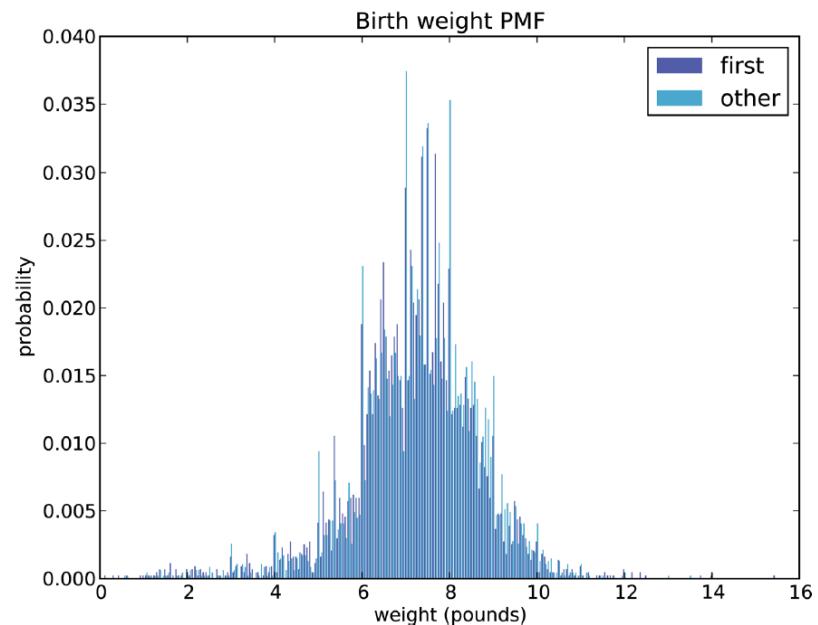
- Example of the **probability mass function (PMF)** of a **discrete** probability distribution



PMF of pregnancy lengths for first babies and others, using bar graphs (left) and step functions (right)

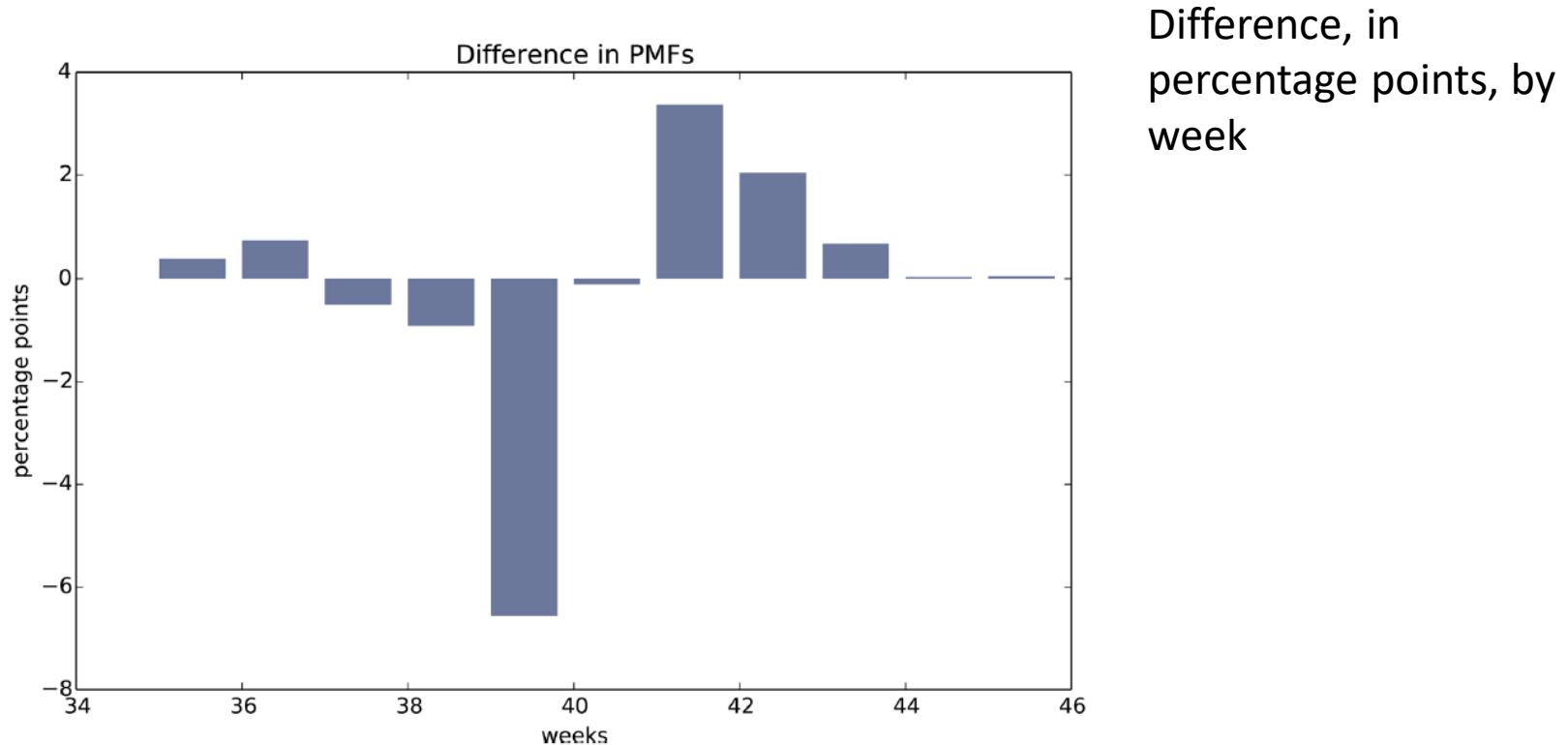
# PMF limitations

- In cases of a lot of samples to show it is very hard to read information from a PMF plot
- Possible solutions include
  - calculating the cumulative distribution function (CDF)
  - Showing the difference of distributions



# Difference of distributions

- Example of difference between two PMFs (probability mass functions )



# Cumulative Distribution Function (CDF)

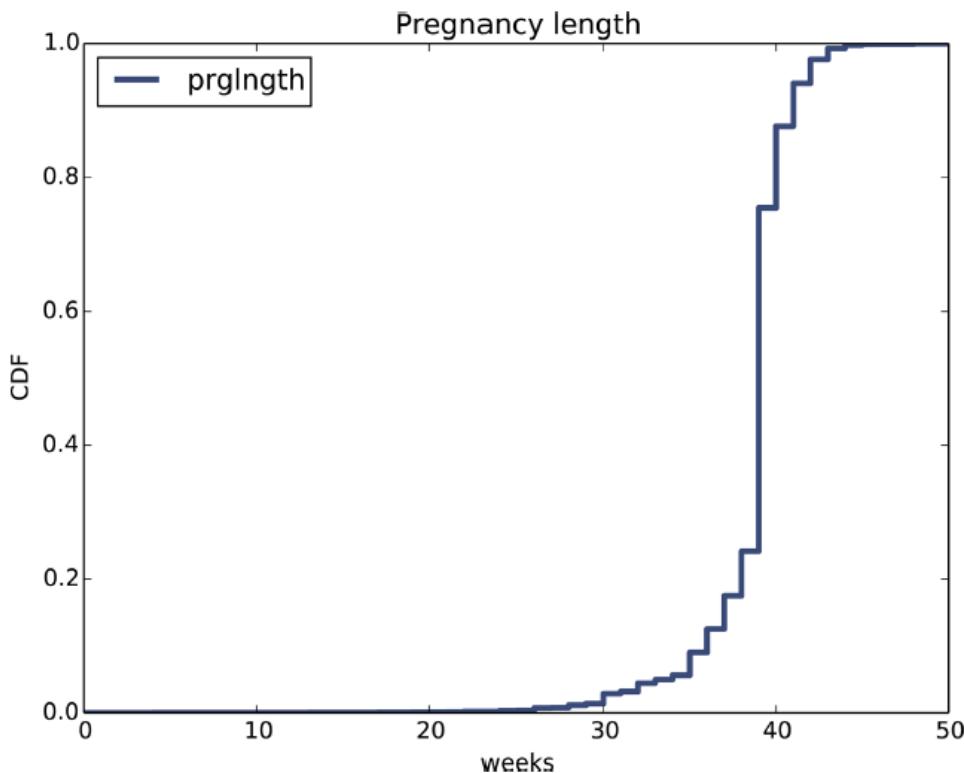
- The CDF is the function that maps a value to its percentile rank.
  - The CDF is a function of  $x$ , where  $x$  is any value that might appear in the distribution.
- The CDF also provides a visual representation of the shape of the distribution
  - Common values appear as steep or vertical sections of the CDF
- To evaluate  $\text{CDF}(x)$  for a particular value of  $x$ , we compute the fraction of values in the distribution less than or equal to  $x$ .

$$F_x(x) = P(X \leq x)$$

$$P(a < X \leq b) = F_x(b) - F_x(a)$$

# Cumulative Distribution Function

## Example of CDF



E.g. about 10% of pregnancies are shorter than 36 weeks, and about 90% are shorter than 41 weeks.

The CDF also provides a visual representation of the shape of the distribution. Common values appear as steep or vertical sections of the CDF; in this example, the mode at 39 weeks is apparent.

There are few values below 30 weeks, so the CDF in this range is flat.



POLITECNICO  
DI TORINO

# Data Science Lab

Data Exploration: Outlier detection

DataBase and Data Mining Group

Tania Cerquitelli and Elena Baralis

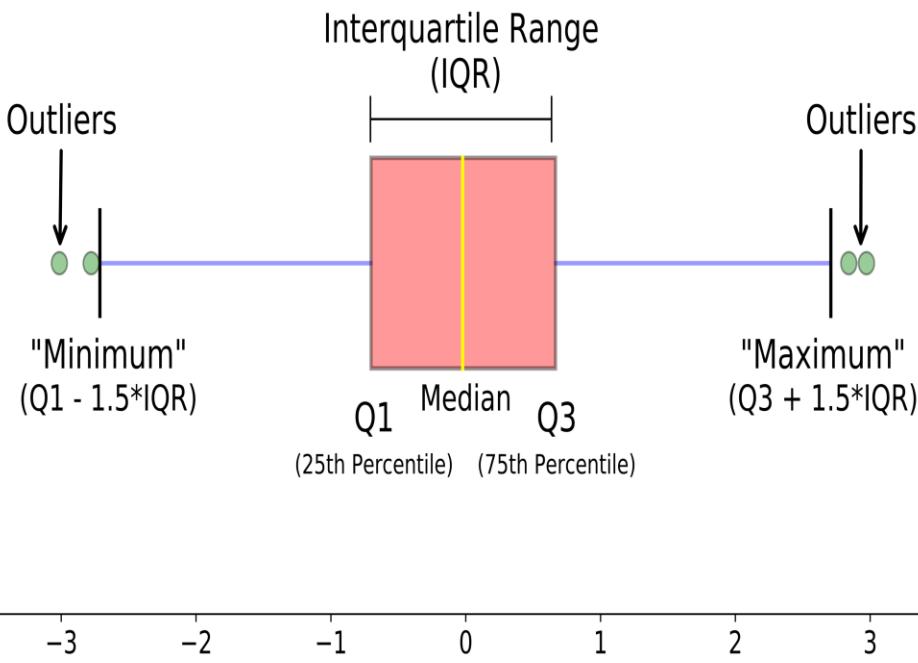
- **Outliers** are extreme values that might be
  - errors in measurement and recording
  - accurate reports of rare events
- The best way to handle outliers depends on  
**“domain knowledge”**
  - Information about where the data come from and what they mean
  - it depends on what analysis you are planning to address

- Outliers can be detected through
  - Univariate analysis
    - Boxplot
    - Percentiles
    - Histograms
    - GESD
    - ...
  - Multivariate analysis
    - DBSCAN
    - ...
  - More specific techniques

# Outliers Detection

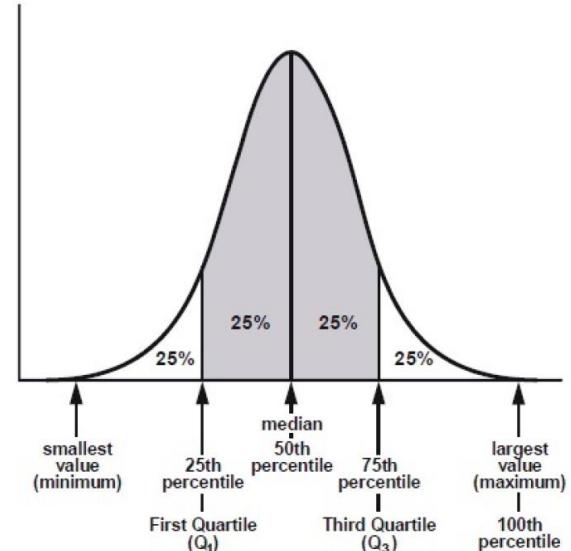
## Boxplot

**Boxplots** are a standardized way of displaying the **distribution** of data based on a **five number summary** (“minimum”, first quartile (Q1), median, third quartile (Q3), and “maximum”).



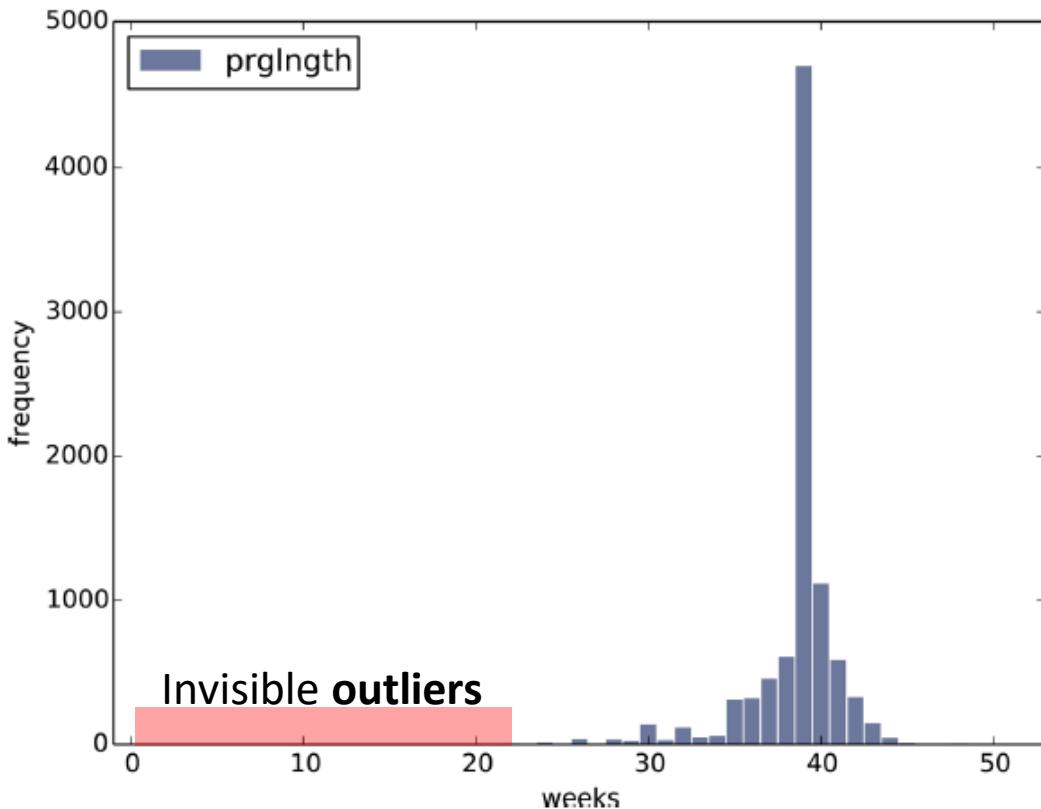
- **median** (Q2/50th Percentile): the middle value of the attribute.
- **first quartile** (Q1/25th Percentile): the middle number between the smallest number (not the “minimum”) and the median of the attribute.
- **third quartile** (Q3/75th Percentile): the middle value between the median and the highest value (not the “maximum”) of the attribute.
- **interquartile range (IQR)**: 25th to the 75th percentile.
- **whiskers** (shown in blue)
- **outliers** (shown as green circles)

- **Percentile** indicates the value below which a given percentage of observations in a group of observations falls
- Representing a feature/attribute/variable through percentiles allow representing the entire distribution
  - Selecting the four percentiles
  - Selecting the ten percentiles
    - selected **10 percentiles**: 10, 20, 30, 40, 50, 60, 70, 80, 90, 99
- Outliers
  - e.g., values in the **first** and **last** percentile of the distribution



# Histogram

- Example of a histogram that shows the distribution of the of pregnancy length in weeks



- The most common value of pregnancy length is 39 weeks. The left tail is longer than the right:
  - early babies are common, but need to establish the threshold where the value is a rare case or when it is an error
  - pregnancies seldom go past 43 weeks, and doctors often intervene if they do.

## Generalized Extreme Studentized Deviate (GESD)

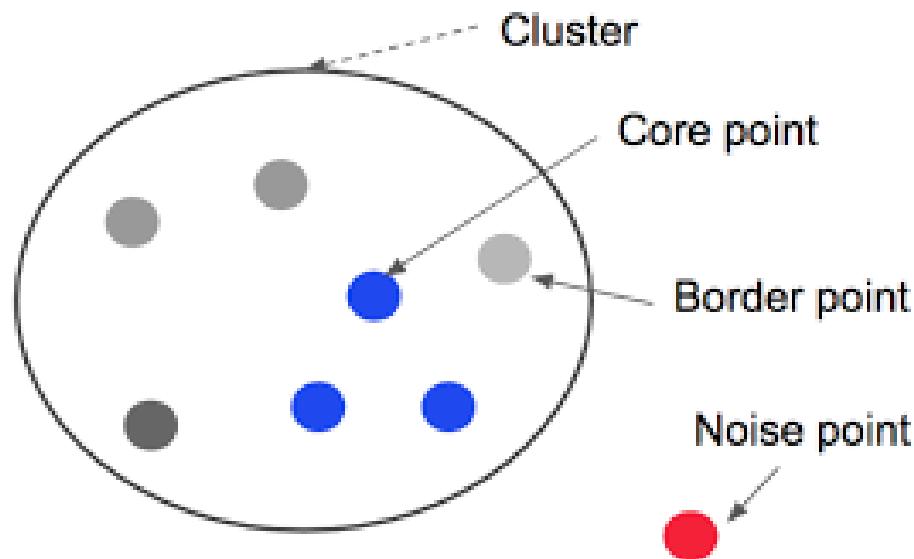
- It is used to detect one or more **outliers** in a **univariate** data set that follows an approximately normal distribution

Given the upper bound,  $r$ , the generalized ESD test essentially performs  $r$  separate tests: a test for one outlier, a test for two outlier, and so on up to  $r$  outliers.

$$R_i = \frac{\max_i |x_i - \bar{x}|}{s}$$

With  $\bar{x}$  and  $s$  denoting the sample mean and sample standard deviation, respectively

DBSCAN is a **density-based clustering** non-parametric algorithm: given a set of points in some space, it **groups together** points that are closely packed together (points with many nearby neighbors), marking as **outliers** points that lie alone in low-density regions (whose nearest neighbors are too far away)





POLITECNICO  
DI TORINO

# Data Science Lab

Data Exploration: Correlation analysis

DataBase and Data Mining Group

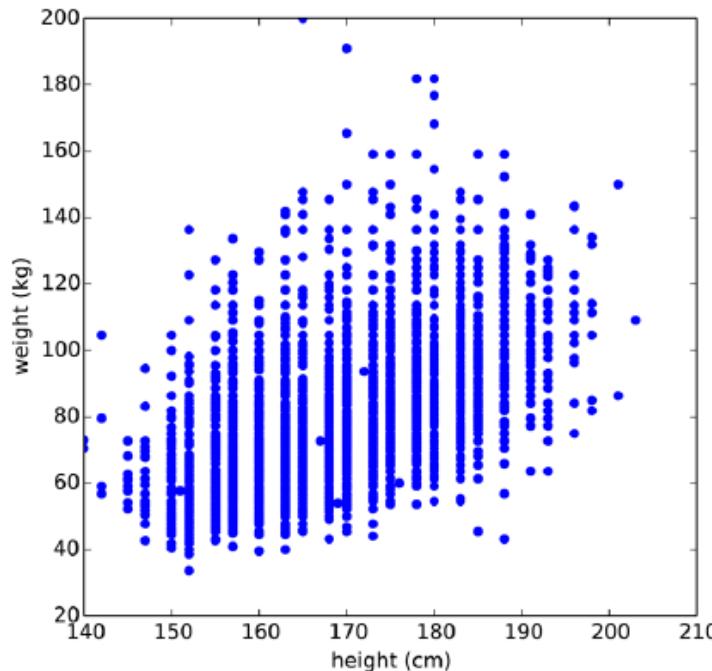
Tania Cerquitelli and Elena Baralis

# Characterizing multivariate dataset

- A dataset usually includes different features/attributes
  - The description of the main relations between attributes assumes a key role
- Statistical descriptions includes
  - Scatter plot
  - Scatter plot percentiles
  - Correlation analysis
  - ...

# Scatter Plot

- The simplest way to check for a relationship between two variables is a **scatter plot**
  - e.g. plot the correlation between height and weight



In real case people who are taller tend to be heavier

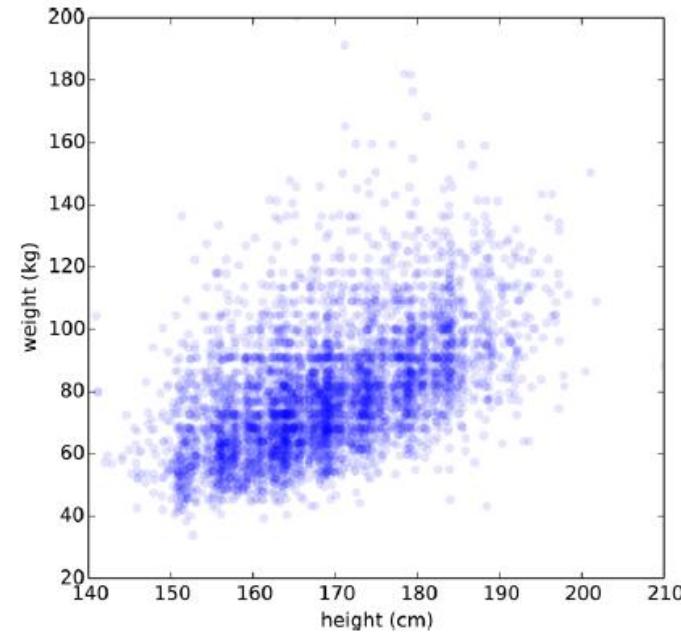
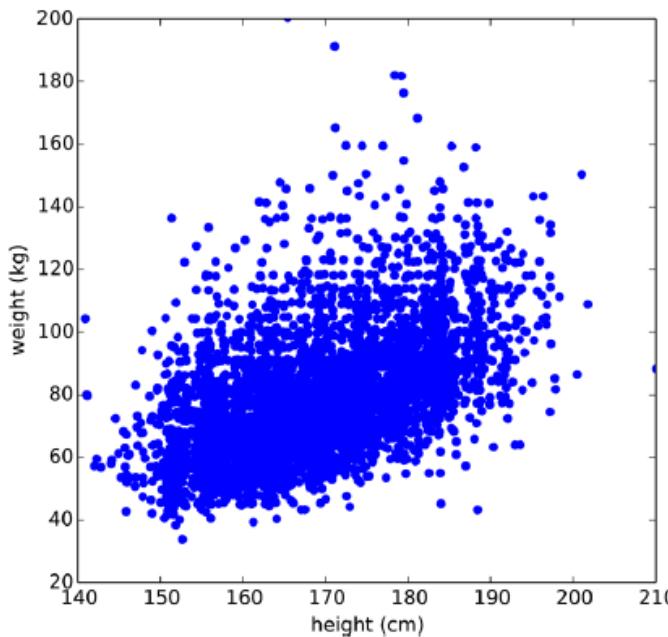
- the coordinates do not faithfully represent reality due to rounding and subsequent conversion done in this example dataset (inch to cm)

# Scatter Plot

A possible solution is to **jittering** the data, which means adding random noise to reverse the effect of rounding off

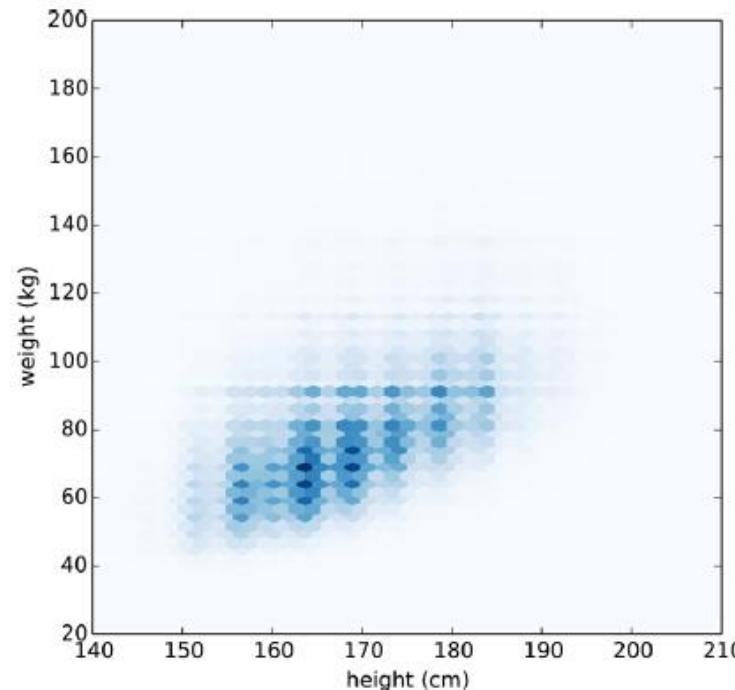
Then adding **alpha** parameter to each point in order to retrieve density information

Darker zones corresponds to higher density zones



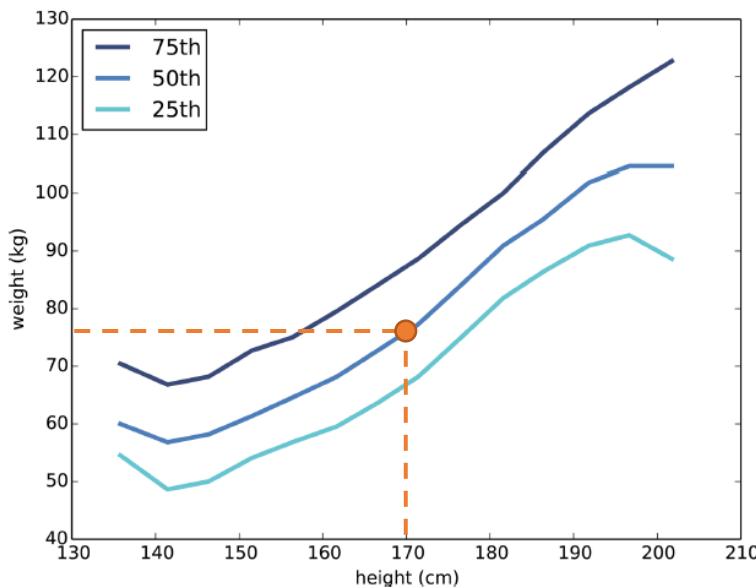
# Scatter Plot

- Scatter plot is converted into **hexbin plot**
  - The hexbin plot uses **hexagonal bins** that are colored according to how many data points fall in it
- The main issue of the scatter plot is the limitation of representing huge quantity of points



# Scatter Plot Percentiles

- This technique includes to bin one variable and plot percentiles of the other
  - It is an alternative to scatter plot



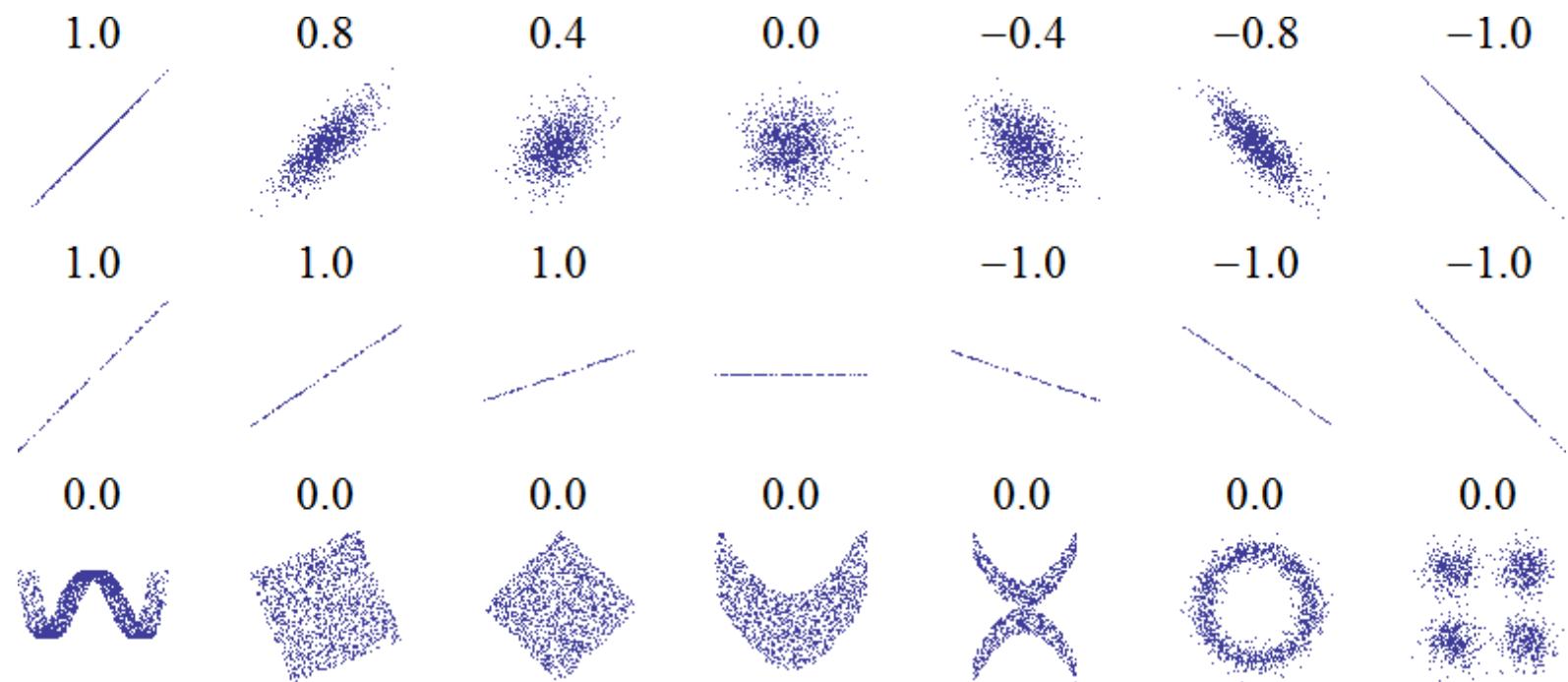
Line plot where the 25th, 50th and 75th percentiles are shown

e.g. orange intersection means that 50% of people 170 cm tall weigh less than 75kg

- A **correlation** is a statistic intended to quantify the strength of the relationship between two variables.
- Possible way to compute correlations are:
  - Covariance
    - In order to compare two variables, they must have the same unit of measurement
    - alternatively, they must be **normalised**
  - Pearson
    - Solves the problem of normalization
    - Only detect linear correlation
  - Spearman

# Nonlinear Correlation

- There are some types of nonlinear correlations that Pearson's correlation can't detect. Here is an example

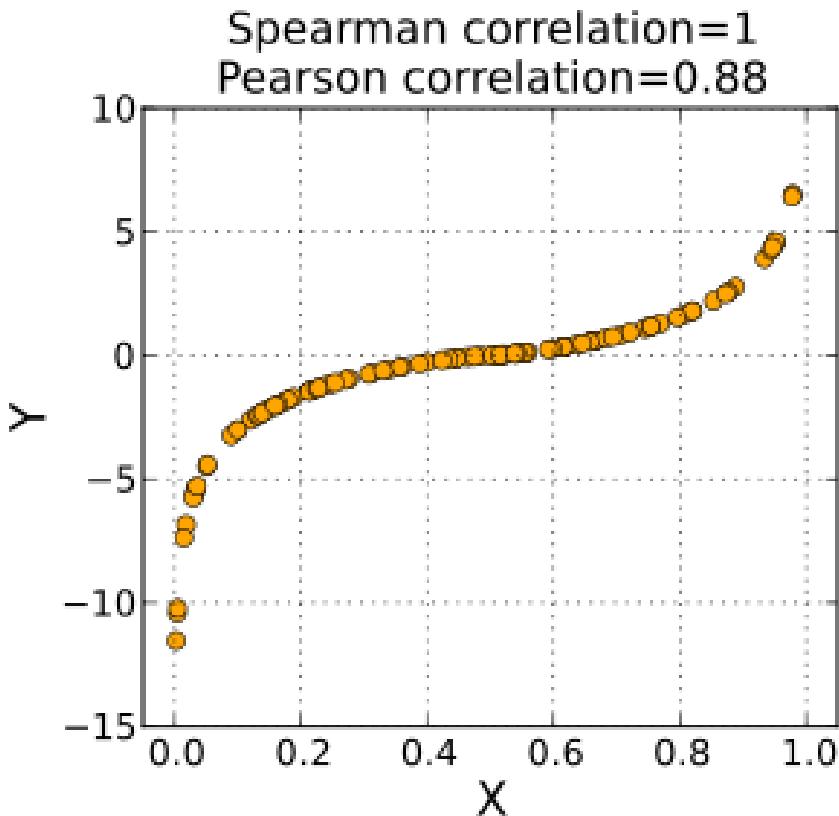


# Spearman's Rank Correlation

- In some cases the Spearman index allows to find a correlation when the Pearson index returns a value close to 0
- The Spearman index or Spearman's rank uses the variables rank instead of Pearson that uses the variables themselves
- Can find monotonic function correlation between two variables

# Spearman's Rank Correlation

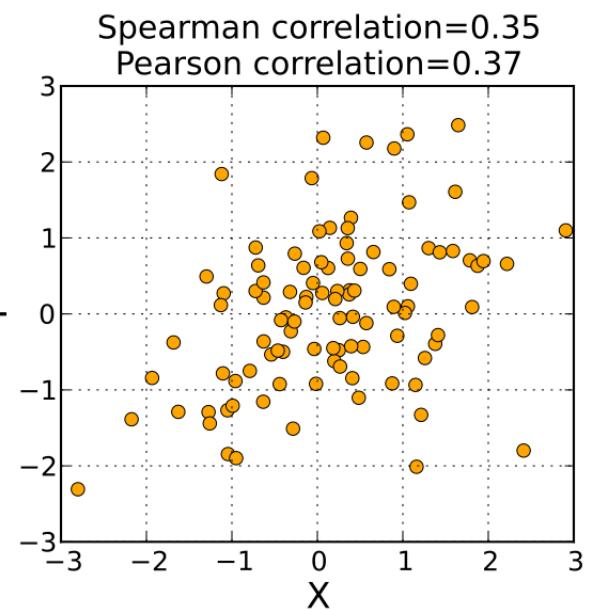
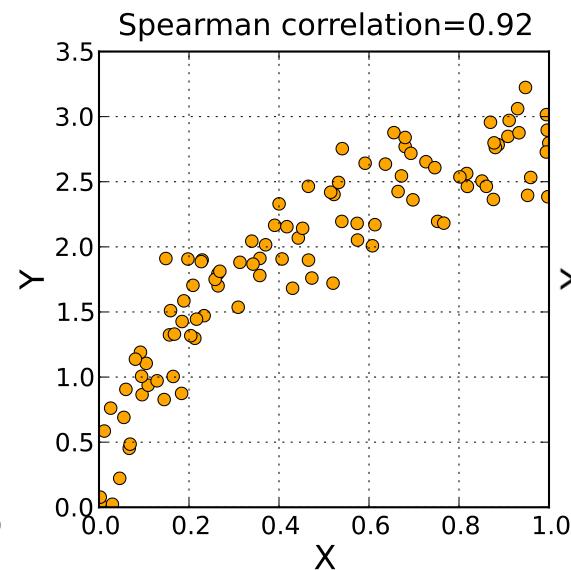
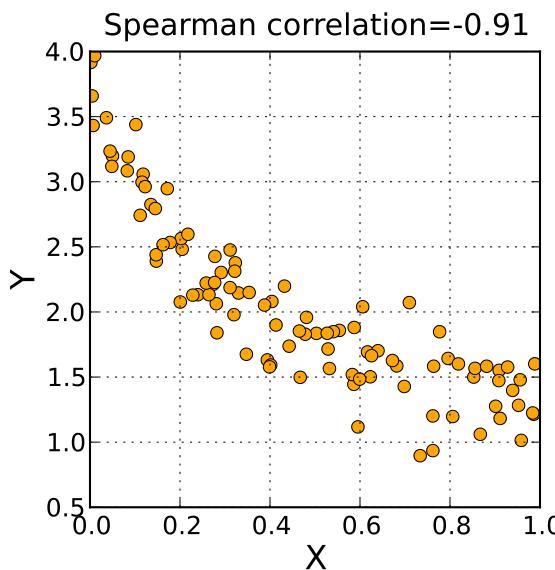
- Example of a monotonic correlation, Spearman index is 1 while Pearson is only 0.88 because the correlation is non linear



assesses monotonic relationships  
(whether linear or not)

# Spearman's Rank Correlation

- Spearman's rank is
  - 1 when two variables are correlated by an increasing monotonic function
  - -1 if the function is decreasing monotonic
  - 0 if there isn't a monotonic function correlation





POLITECNICO  
DI TORINO

# Data Science Lab

## Feature Engineering

DataBase and Data Mining Group

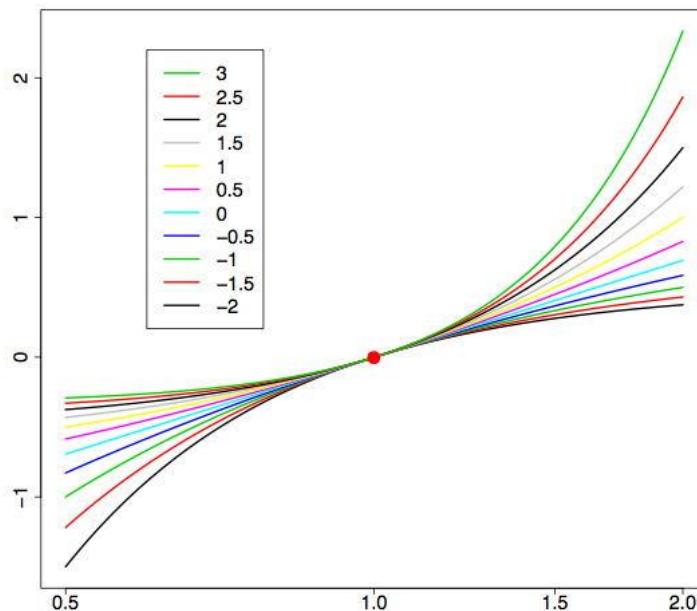
Tania Cerquitelli and Elena Baralis

- *Feature engineering* is the act of extracting features from raw data and transforming them into formats that are suitable for the machine learning model
- A *feature* is a numeric representation of an aspect of raw data

- Accordingly to the type of data under analysis different feature engineering techniques are needed
  - Structured
    - Numerical data, Categorical data
  - Unstructured
    - Text, Images, Signals
  - Mixed
- Basic types of feature engineering techniques include
  - Normalization
  - Discretization
  - Binarization
  - Data transformation

- Data transformation is the process of converting data from one format to another
- Why transforming data
  - Non numerical data is difficult to analyze if not transformed into numerical
  - To fit simpler models (e.g. normal distribution)
  - To better visualize the data (e.g. transform linear scale to logarithmic scale in audio context)
  - ...

- **Power transform** try to fit the distribution to the Normal distribution in order to achieve better results in further analysis
  - Some state-of-art techniques better perform with specific data distributions
- The Box-cox transformation changes the distribution of the variable so that the variance is no longer dependent on the mean



**Box-cox Formula:**

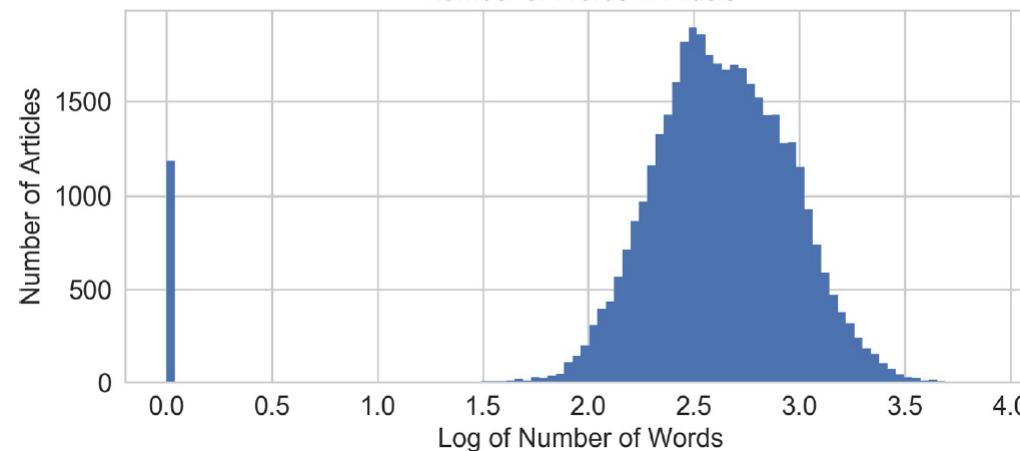
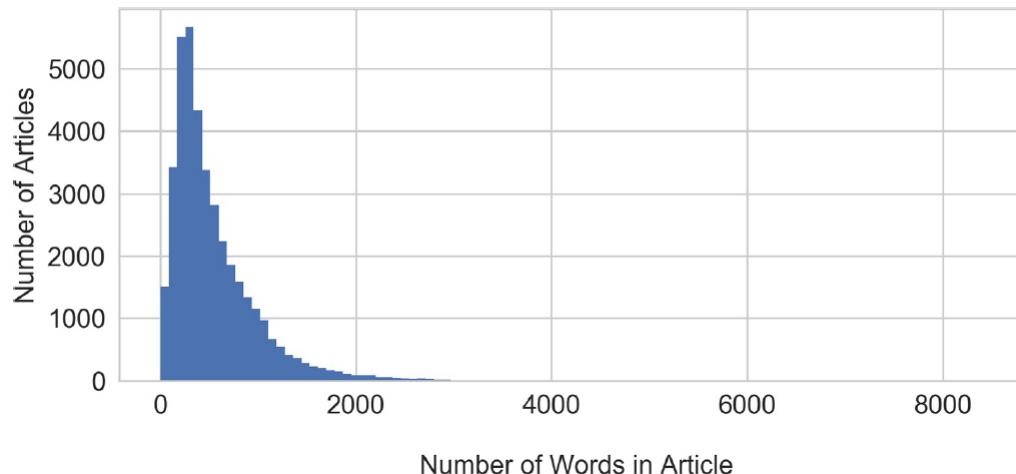
$$\tilde{x} = \begin{cases} \frac{x^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0, \\ \ln(x) & \text{if } \lambda = 0. \end{cases}$$

e.g.

$\lambda = 0$  corresponds to  
Log Transformation

# Power Transform: Log Transformation

## Example of Log Transformation



Original data distribution. All values are close to 0

Data is transformed with Log Transformation in order to fit the Normal Distribution

- A *categorical variable*, as the name suggests, is used to represent categories or labels.
  - e.g., cities, season, etc...
- Some encoding methods are required to use categorial variables with some data analytics algorithms:
  - One-Hot Encoding
  - Dummy Coding
  - Effect Coding

# One-Hot Encoding

- One-Hot Encoding use a group of bits.
- Each bit represents a possible category.
- If the variable cannot belong to multiple categories at once, then only one bit in the group can be “on.”
  - Example: the attribute city assumes only 3 values
  - The one-hot encoding representation is reported below

	e1	e2	e3
San Francisco	1	0	0
New York	0	1	0
Seattle	0	0	1

- The problem with one-hot encoding is that it allows for  $k$  degrees of freedom, but the variable itself needs only  $k-1$ .
- Dummy Coding encodes the effect of each category relative to the reference category encoded with zeroes (Seattle)
- Example of a dummy coding

	e1	e2
San Francisco	1	0
New York	0	1
Seattle	0	0

- It is similar to dummy coding, with the difference that the reference category is now represented by the vector of all -1's
- Example of an effect coding

	e1	e2
San Francisco	1	0
New York	0	1
Seattle	-1	-1

# Pro-Cons

	PRO	CONS
One Hot	<ul style="list-style-type: none"><li>each feature clearly corresponds to a category</li><li>missing data can be encoded as the all zeros Vector</li><li>output should be the overall mean of the target variable</li></ul>	<ul style="list-style-type: none"><li>Redundant</li></ul>
Dummy	<ul style="list-style-type: none"><li>Not Redundant</li></ul>	<ul style="list-style-type: none"><li>cannot easily handle missing data, since the all-zeros vector is already mapped to the reference category.</li></ul>
Effect	<ul style="list-style-type: none"><li>using a different code for the reference Category( -1)</li></ul>	<ul style="list-style-type: none"><li>the vector of all -1's is a dense vector, which is expensive for both storage and computation</li></ul>

# Feature engineering vs feature reduction

- Feature engineering also means feature reduction
  - Why reduce the number of features?
    - Reduce overfitting
    - Better performance on reduced data
    - Improve the generalization of models.
    - Gain a better understanding of the features and their relationship to the response variables.
- Feature reduction
  - **Dimensionality reduction using Singular Value Decomposition**
    - it can work with sparse matrices efficiently
  - **Principal component analysis (PCA)**
    - subtract the mean sample from each row of the data matrix
    - perform SVD on the resulting matrix.
  - **Linear Discriminant Analysis (LDA)**

- Feature engineering might also require the feature selection step
- Traditional approaches can be used
  - Recursive feature elimination
  - Analysis of variance (ANOVA)
  - Exploiting feature importance of interpretable models
  - Automatic feature selection
  - ...

## ■ Recursive feature elimination

- assigns weights to features
  - e.g., the coefficients of a linear model
- selects features by recursively considering smaller and smaller sets of features
- First, the estimator is trained on the initial set of features and the importance of each feature is obtained
- The least important features are pruned from current set of features
- That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached

- **Analysis of variance (ANOVA)**
  - It is a statistical technique that allows to compare two or more groups of data by comparing the *variance* within these groups with the *variance* between groups
    - e.g. **Fisher-Snedecor** analysis is used to compare the variance between two variable
  - Assigns ranks to the features that help to select the most important ones

- Exploiting feature importance of interpretable models
  - Some models give the information about feature importance
    - e.g. Decision tree, Linear Regression
- Automatic feature selection
  - The components of this decomposition techniques allow to identify which are the most important features/components in the data
    - e.g. PCA, SVD



POLITECNICO  
DI TORINO

# Data Science Lab

## Data Visualization

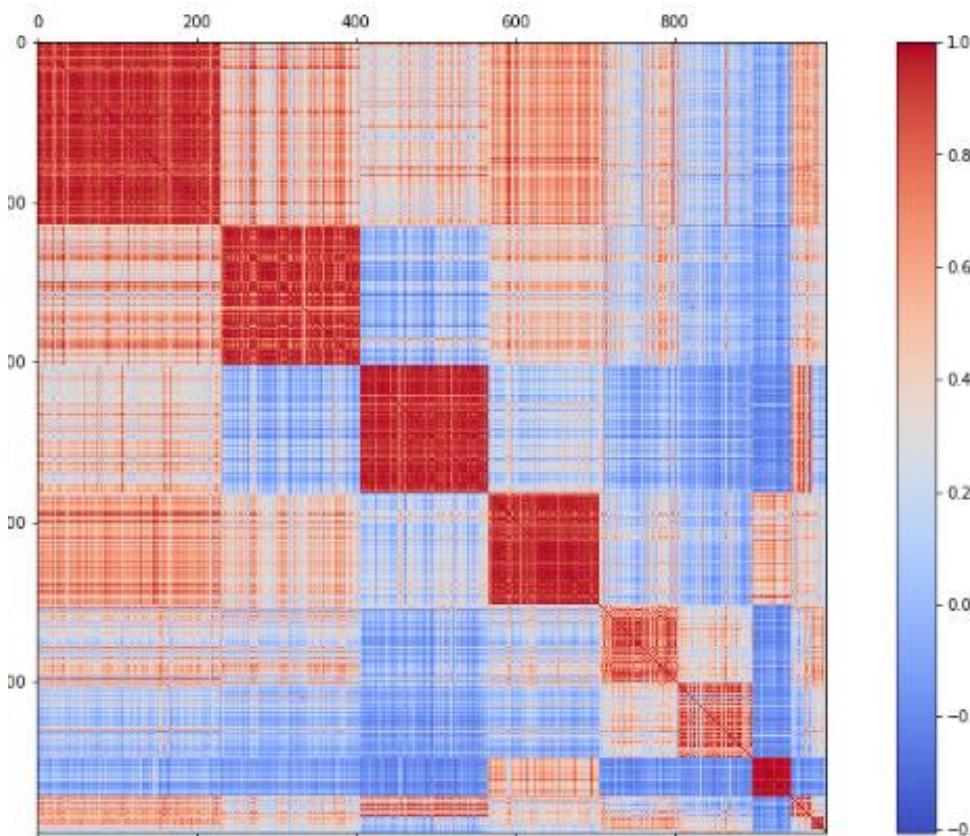
DataBase and Data Mining Group

Tania Cerquitelli and Elena Baralis

- It is important to visualize your data when possible
  - To explore the raw input data
  - To analyze your output
- Choosing the correct visualization method is not trivial
  - Different kind of analytics tasks require proper visualization techniques

# Visualization: heatmap

- Correlation matrix can be visualized through **heatmaps** to represents the correlation between two variables



e.g. correlation between documents.

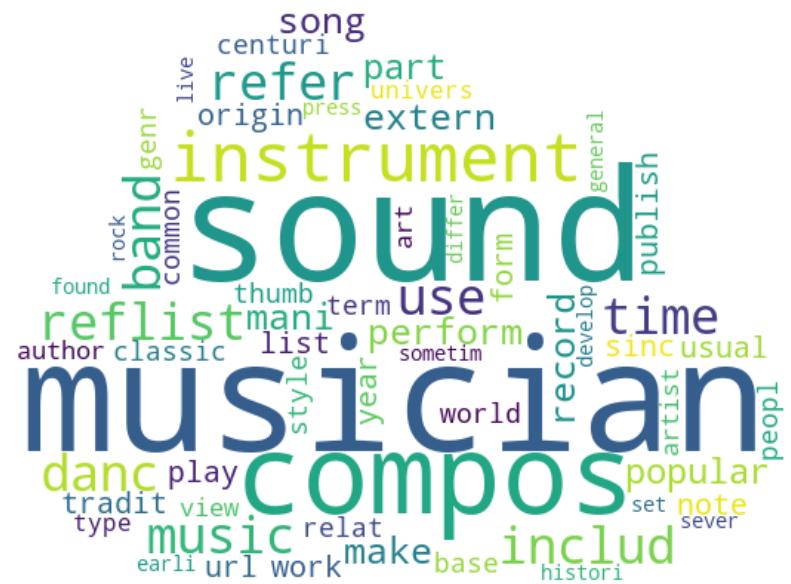
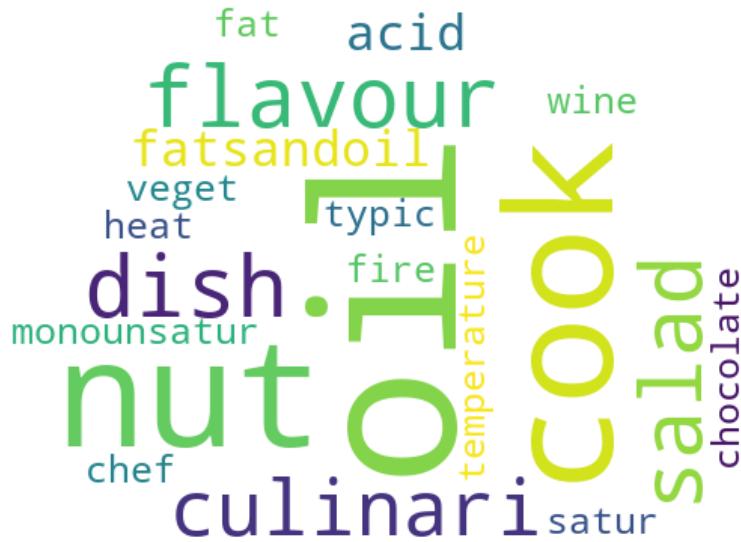
x and y axes represents the documents

documents were previously clustered by topics

Red dots represent high correlation, while blue dots low correlation.

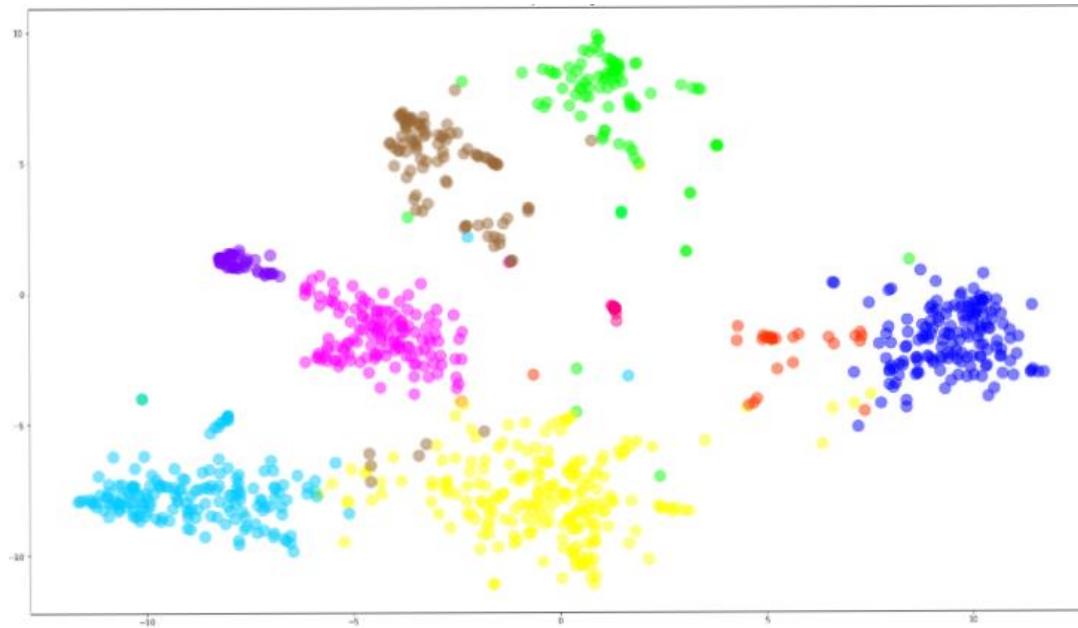
# Visualization: word cloud

- In the context of text mining the **word cloud** can easily represents the topics of a group of similar documents
- Each word cloud contains the most important words characterizing a topic



# Visualization: t-sne

- In some cases it is useful to reduce the size of attributes to show information in plots
- e.g **t-distributed stochastic neighbor embedding (t-SNE)**

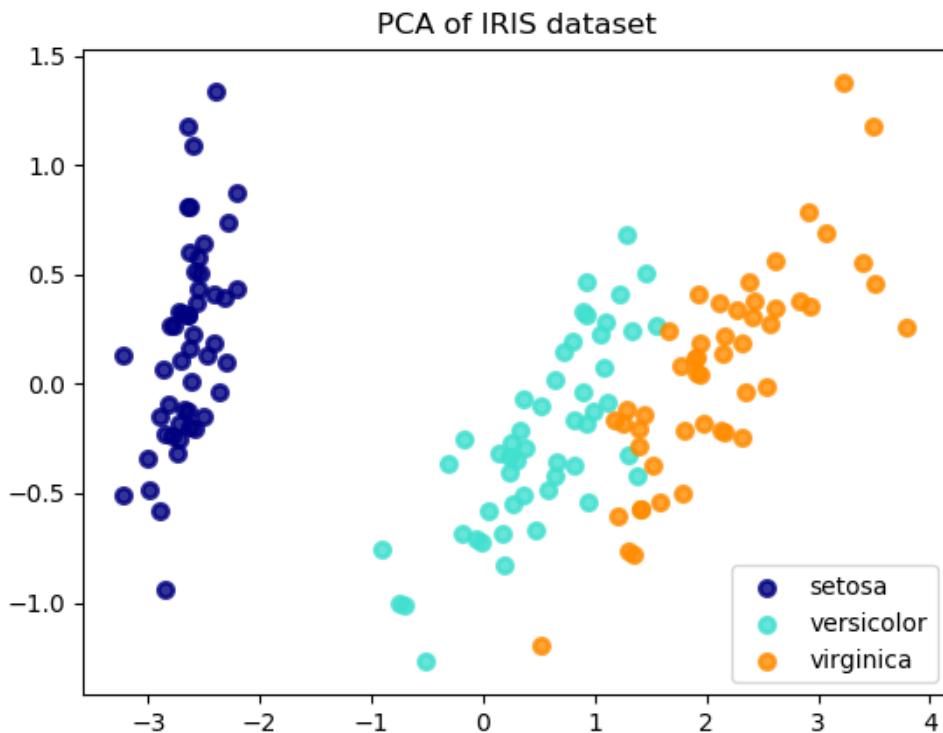


In the example T-sne shows text document in bidimensional space. Each color corresponds to a cluster label

WARNING: using the t-sne as dimensionality reduction technique in ML pipelines is not suggested since it does not preserve the information of your data.

# Visualization: PCA

- In some cases it is useful to reduce the size of attributes to show information in plots
- e.g **Principal component analysis (PCA)**



In the example, the **Iris** dataset was reduced with **PCA** in two features and represented in scatter plot. Each color corresponds to the original labels. See how the 3 categories are separated in bidimensional space

- Think Stats, Allen B. Downey - Feature  
Engineering for Machine Learning: Principles and  
Techniques for Data Scientists

# Classification fundamentals



Data Base and Data Mining Group of Politecnico di Torino

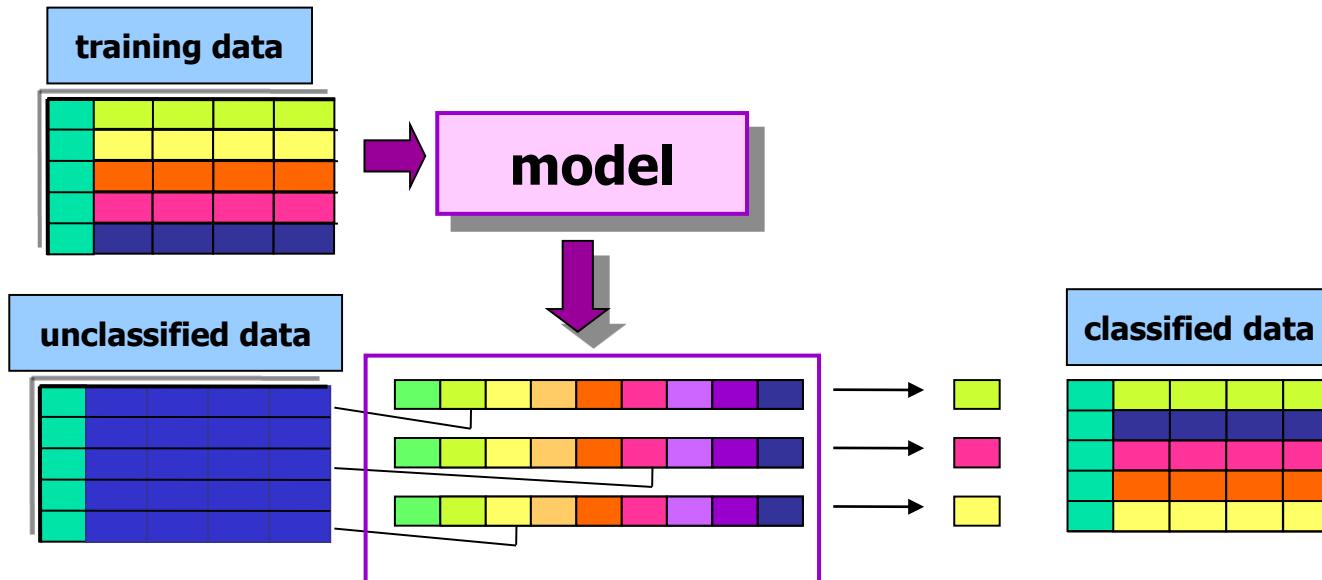
Elena Baralis  
*Politecnico di Torino*



# Classification

## ■ Objectives

- prediction of a class label
- definition of an interpretable model of a given phenomenon

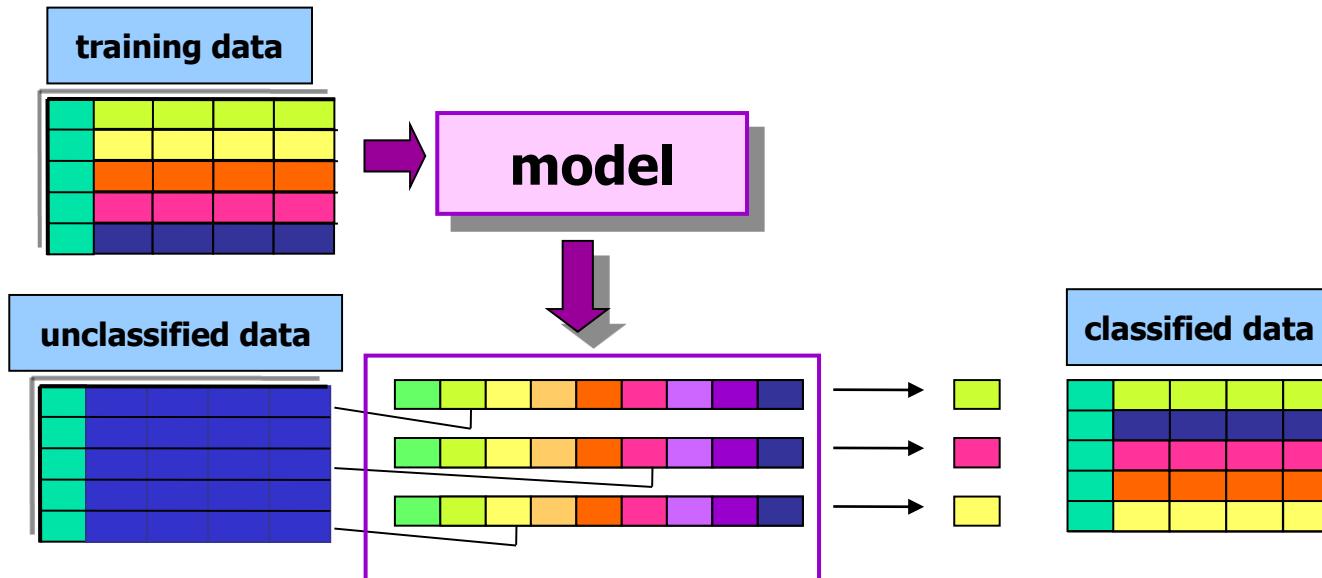




# Classification

## ■ Applications

- detection of customer propensity to leave a company (churn or attrition)
- fraud detection
- classification of different pathology types
- ...





# Classification: definition

- Given
  - a collection of class labels
  - a collection of data objects labelled with a class label
- Find a descriptive profile of each class, which will allow the assignment of unlabeled objects to the appropriate class



# Definitions

- Training set
  - Collection of labeled data objects used to learn the classification model
- Test set
  - Collection of labeled data objects used to validate the classification model



# Classification techniques

- Decision trees
- Classification rules
- Association rules
- Neural Networks
- Naïve Bayes and Bayesian Networks
- k-Nearest Neighbours (k-NN)
- Support Vector Machines (SVM)
- ...



# Evaluation of classification techniques

- Accuracy
  - quality of the prediction
- Interpretability
  - model interpretability
  - model compactness
- Incrementality
  - model update in presence of newly labelled record
- Efficiency
  - model building time
  - classification time
- Scalability
  - training set size
  - attribute number
- Robustness
  - noise, missing data

# Decision trees



Data Base and Data Mining Group of Politecnico di Torino

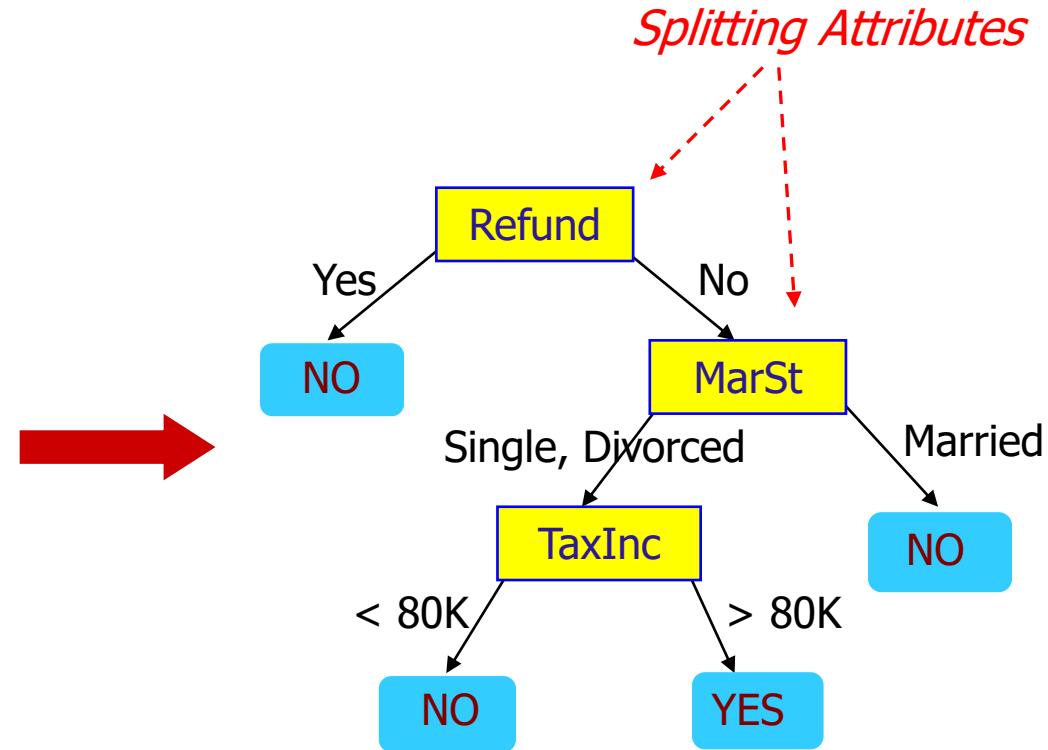
Elena Baralis  
*Politecnico di Torino*



# Example of decision tree

Categorical  
categorical  
continuous  
class

Tid	Refund	Marital Status	Taxable Income	Cheat
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



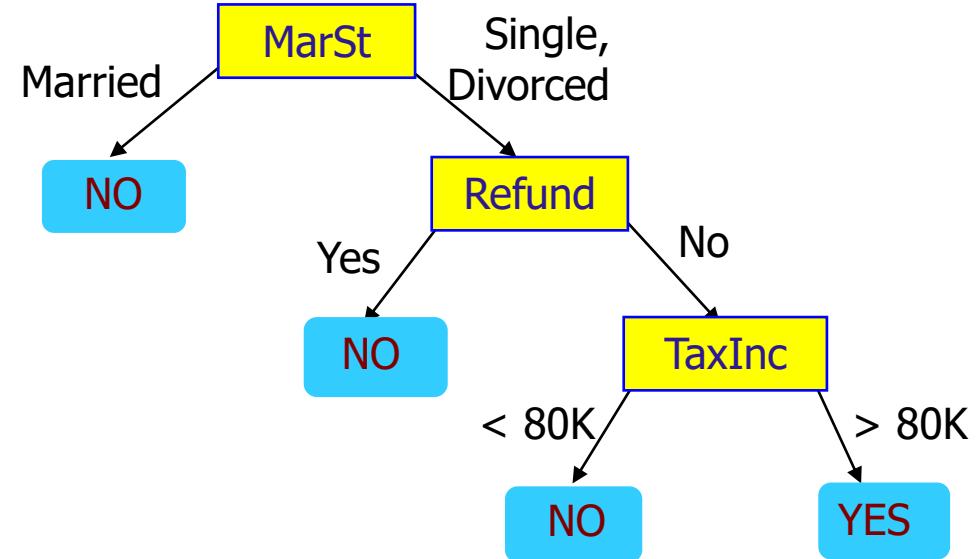
Training Data

Model: Decision Tree



# Another example of decision tree

Tid	Refund	Marital Status	Taxable Income	Cheat	categorical	categorical	continuous	class
					1	2	3	4
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				

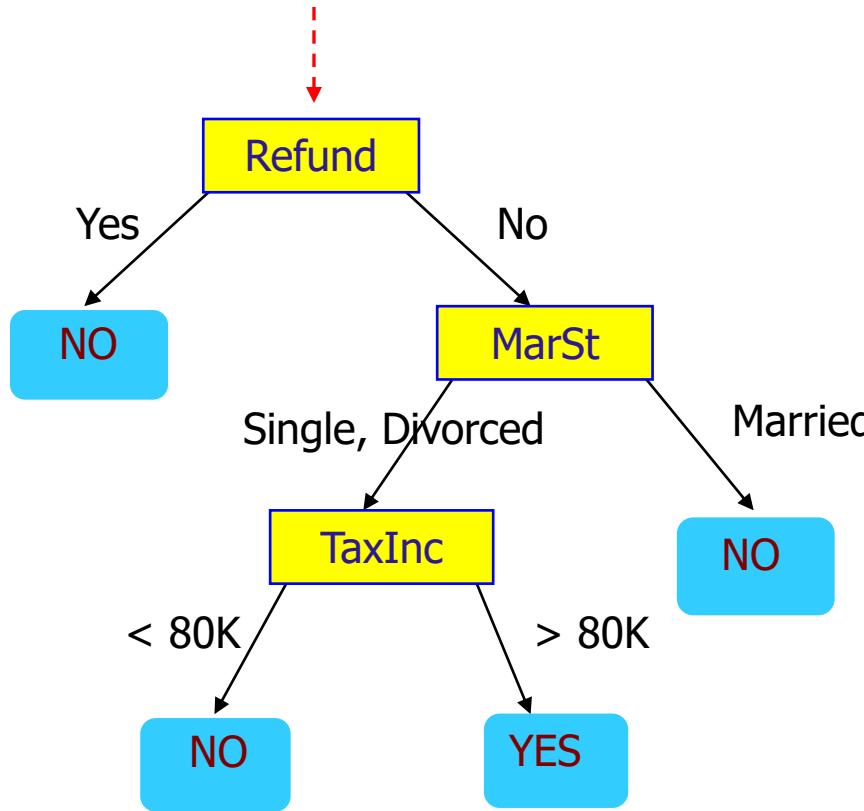


There could be more than one tree that fits the same data!



# Apply Model to Test Data

Start from the root of tree.

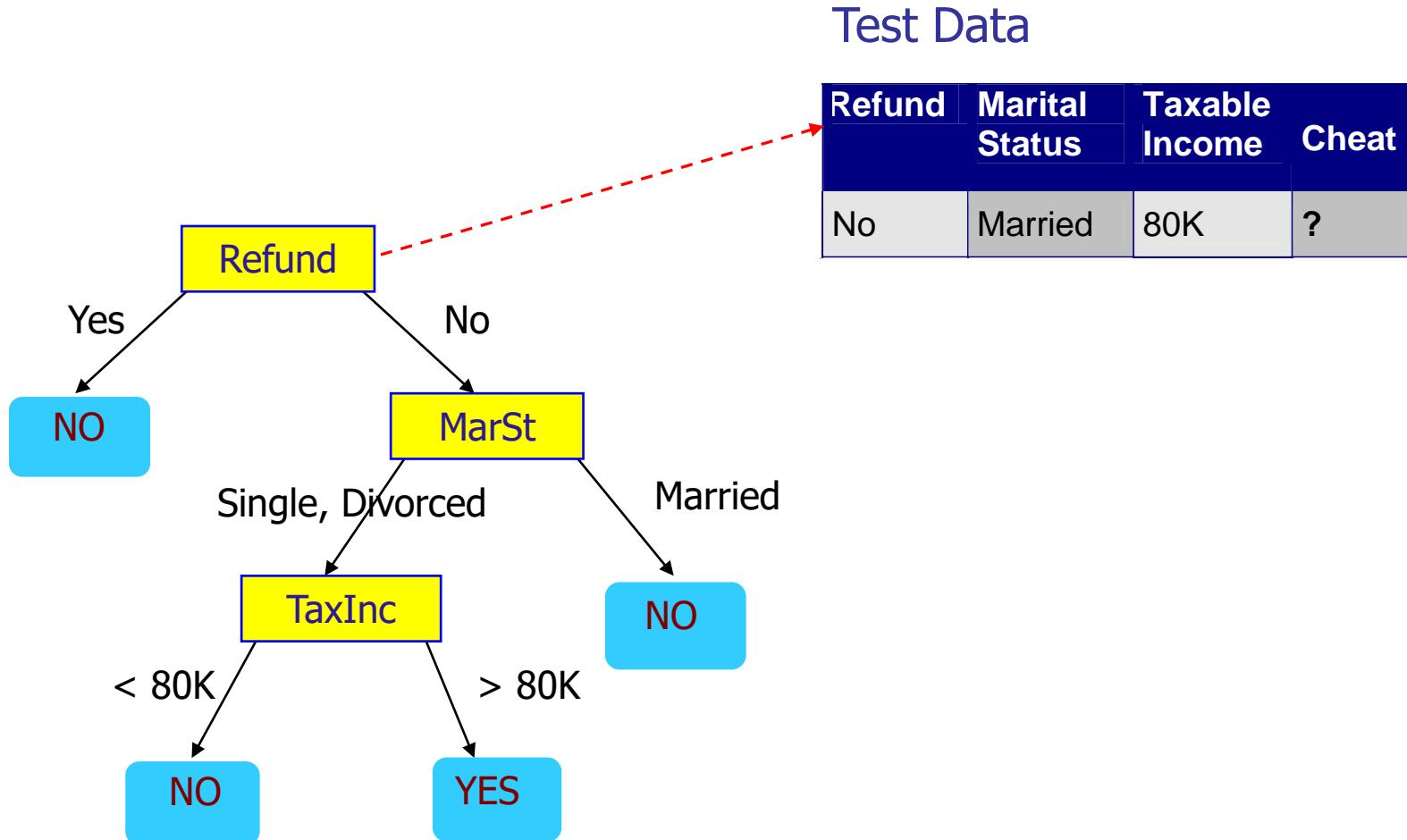


Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



# Apply Model to Test Data

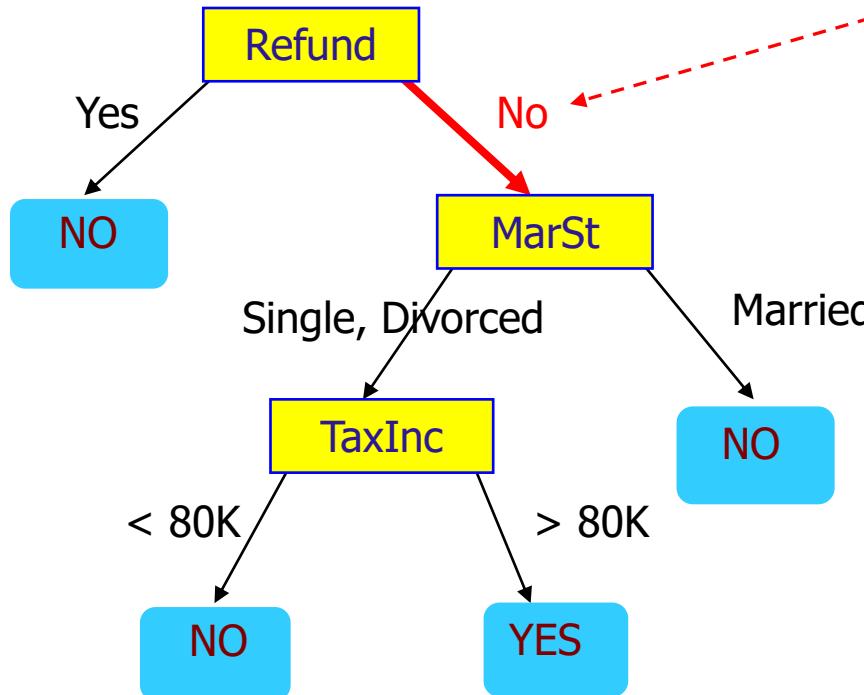




# Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

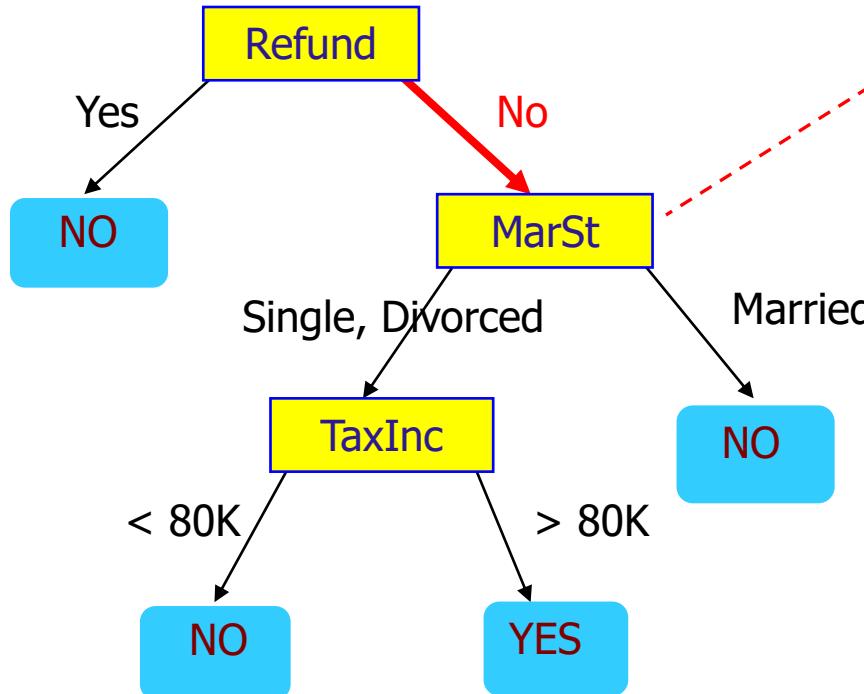




# Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

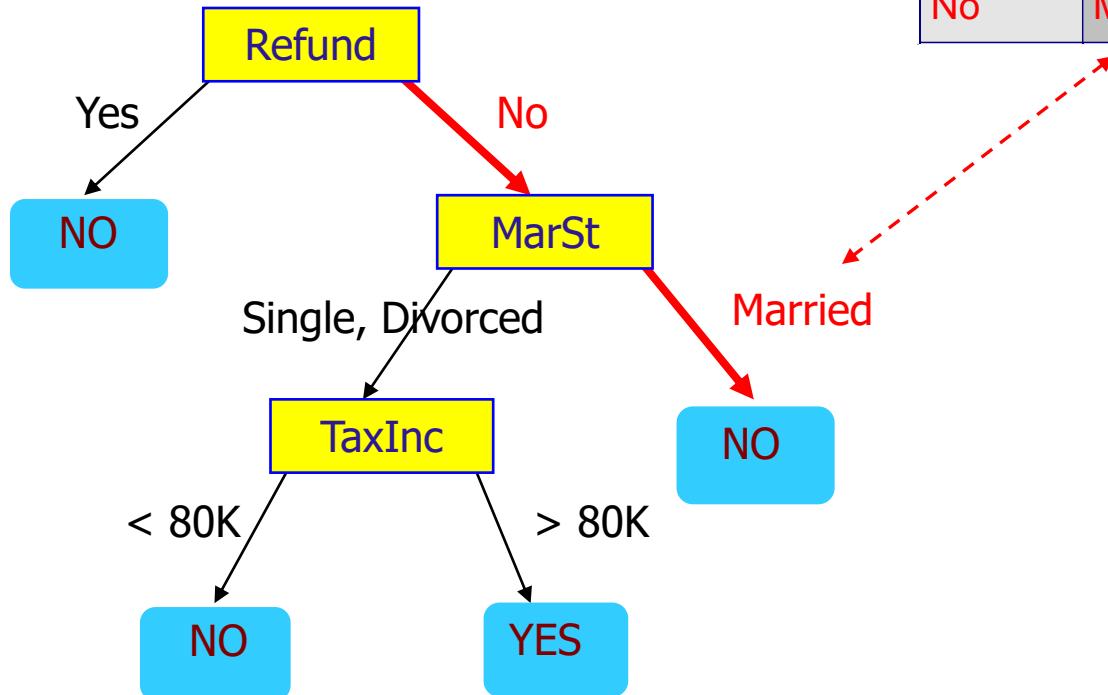




# Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

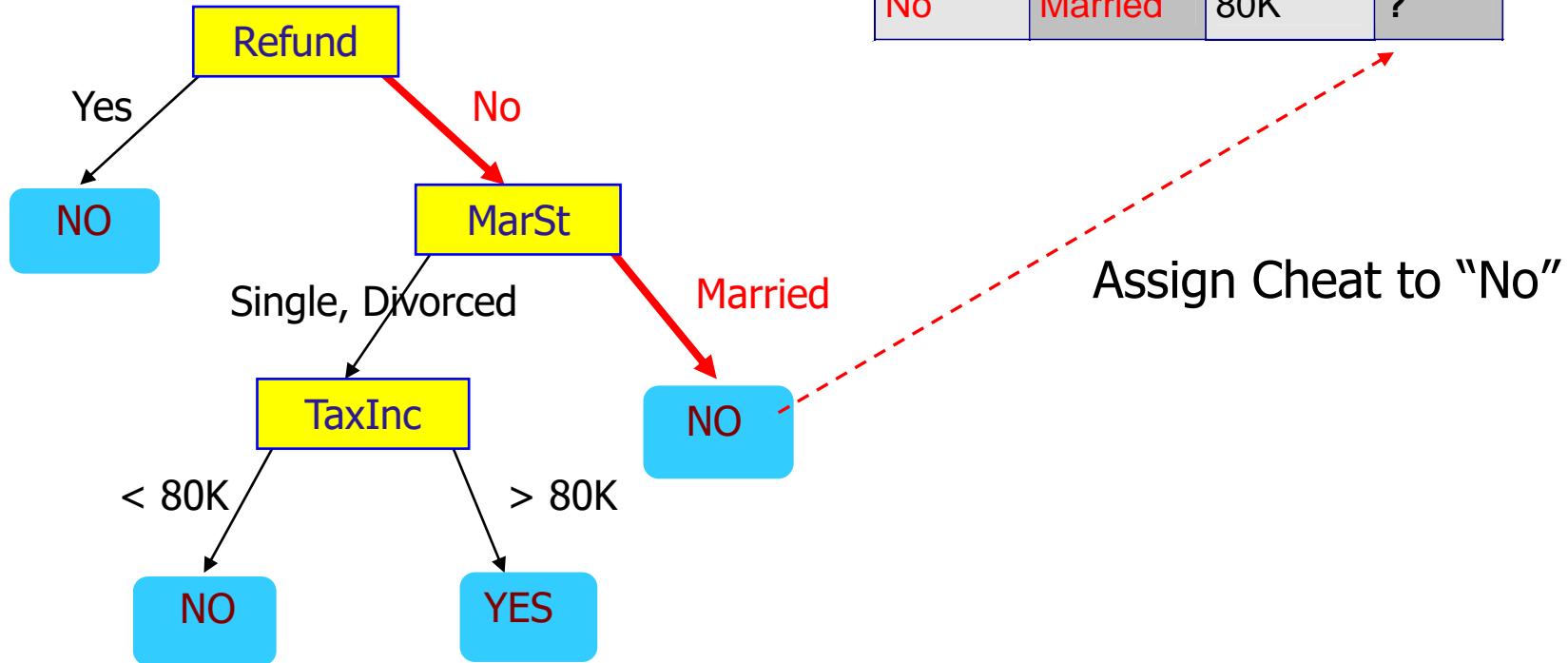




# Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?





# Decision tree induction

- Many algorithms to build a decision tree
  - Hunt's Algorithm (one of the earliest)
  - CART
  - ID3, C4.5, C5.0
  - SLIQ, SPRINT



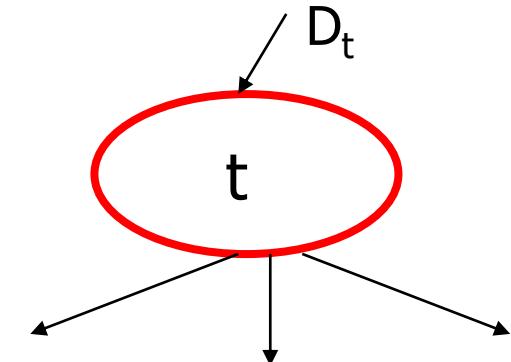
# General structure of Hunt's algorithm

## Basic steps

- If  $D_t$  contains records that belong to more than one class
  - select the “best” attribute  $A$  on which to split  $D_t$  and label node  $t$  as  $A$
  - split  $D_t$  into smaller subsets and recursively apply the procedure to each subset
- If  $D_t$  contains records that belong to the same class  $y_t$ 
  - then  $t$  is a leaf node labeled as  $y_t$
- If  $D_t$  is an empty set
  - then  $t$  is a leaf node labeled as the default (majority) class,  $y_d$

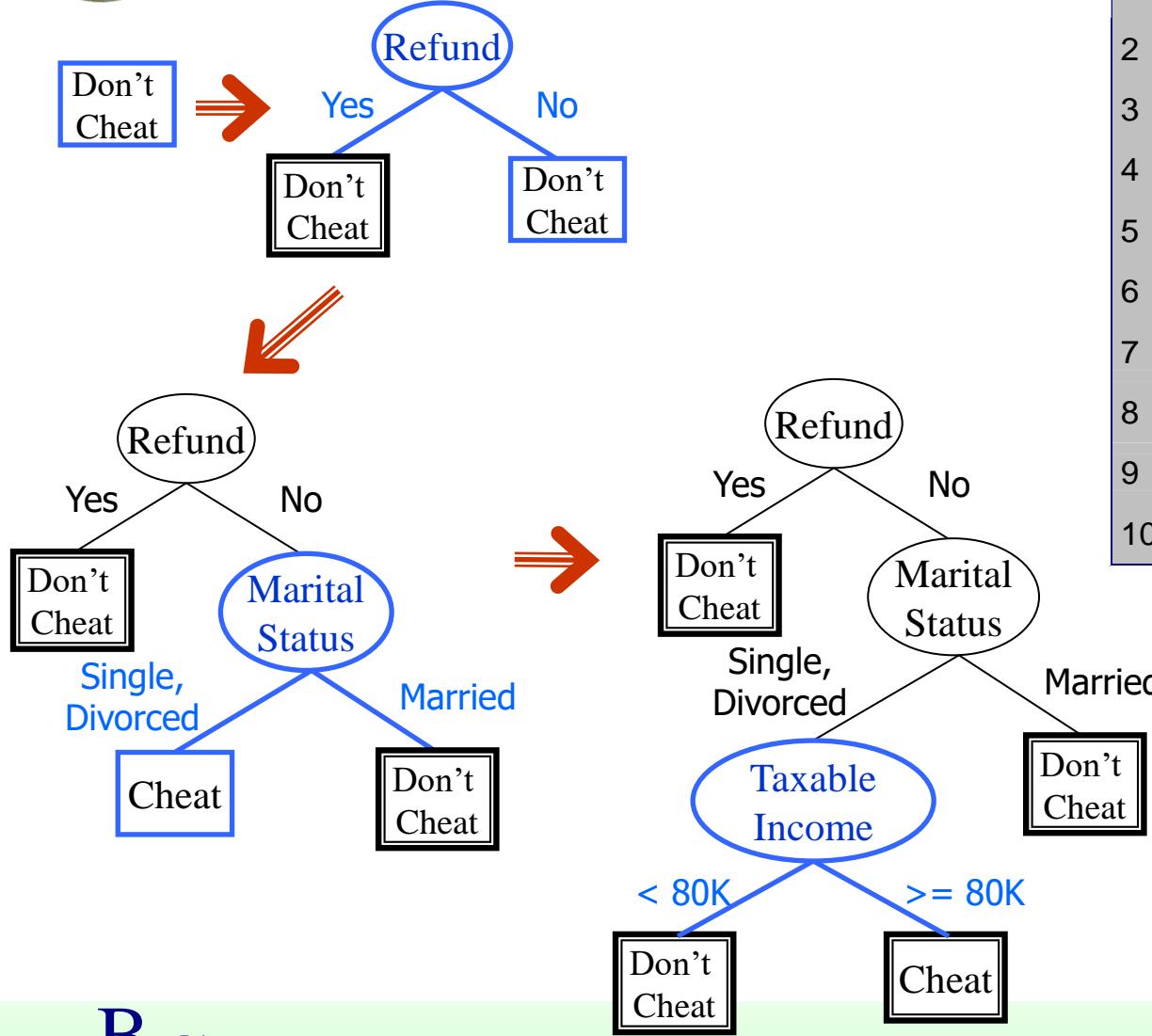
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

$D_t$ , set of training records that reach a node  $t$





# Hunt's algorithm



Tid	Refund	Marital Status	Taxable Income	Cheat
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



# Decision tree induction

- Adopts a greedy strategy
  - “Best” attribute for the split is selected locally at each step
    - not a global optimum
- Issues
  - Structure of test condition
    - Binary split versus multiway split
  - Selection of the best attribute for the split
  - Stopping condition for the algorithm



# Structure of test condition

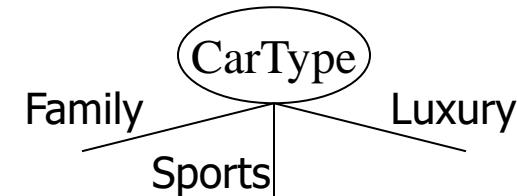
- Depends on attribute type
  - nominal
  - ordinal
  - continuous
- Depends on number of outgoing edges
  - 2-way split
  - multi-way split



# Splitting on nominal attributes

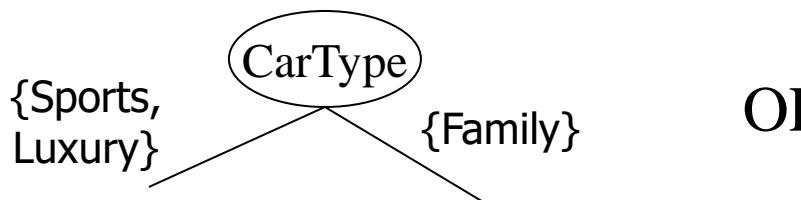
## ■ Multi-way split

- use as many partitions as distinct values

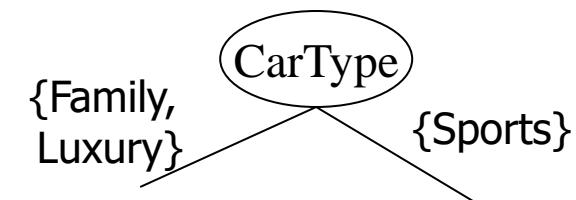


## ■ Binary split

- Divides values into two subsets
- Need to find optimal partitioning



OR





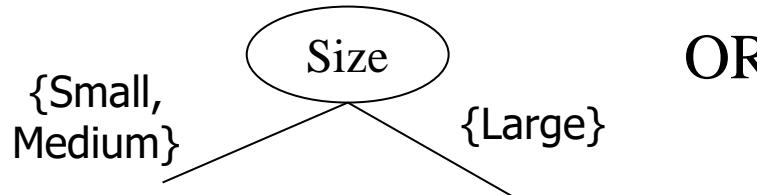
# Splitting on ordinal attributes

## ■ Multi-way split

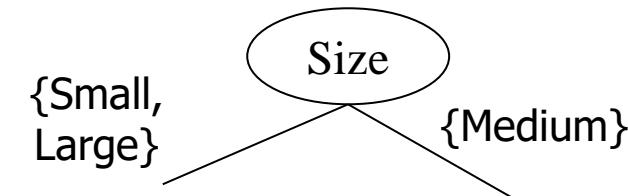
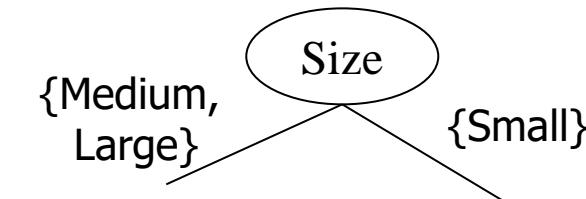
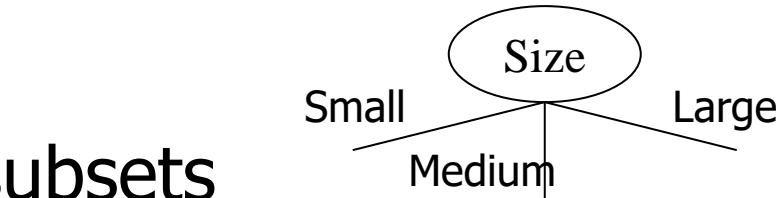
- use as many partitions as distinct values

## ■ Binary split

- Divides values into two subsets
- Need to find optimal partitioning



OR



What about this split?



# Splitting on continuous attributes

## ■ Different techniques

- **Discretization** to form an ordinal categorical attribute

- Static – discretize once at the beginning
  - Dynamic – discretize during tree induction

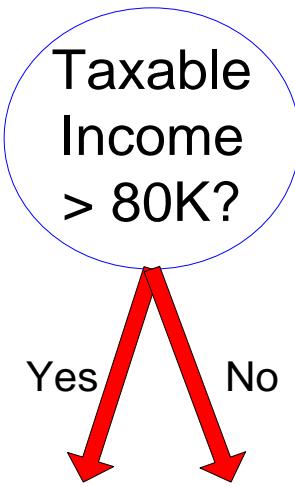
Ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering

- **Binary decision** ( $A < v$ ) or ( $A \geq v$ )

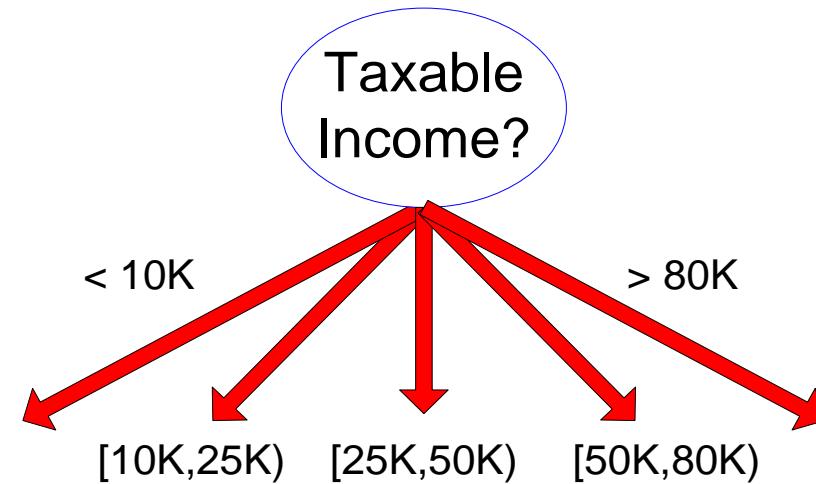
- consider all possible splits and find the best cut
  - more computationally intensive



# Splitting on continuous attributes



(i) Binary split



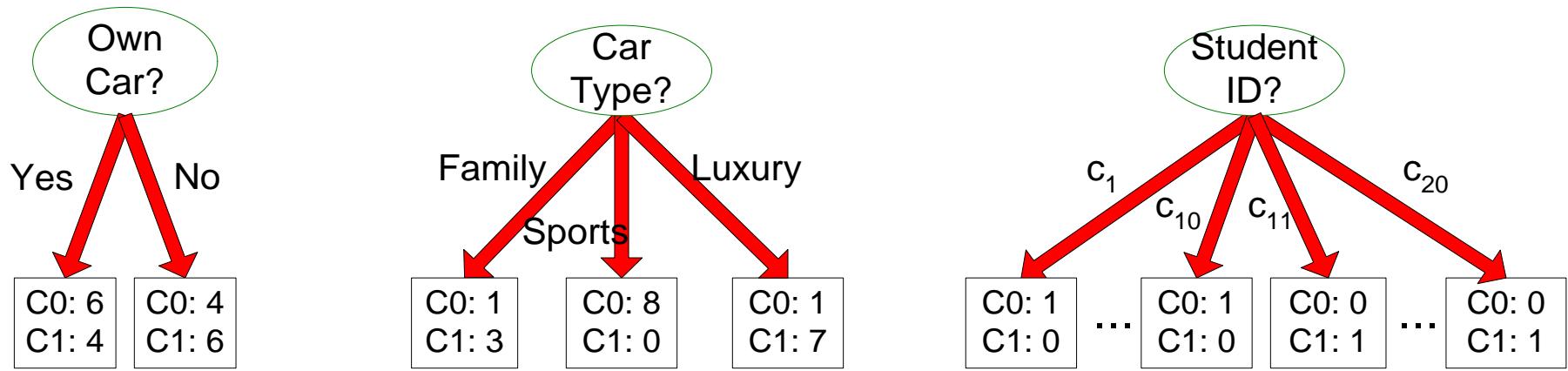
(ii) Multi-way split



# Selection of the best attribute

Before splitting:

10 records of class 0,  
10 records of class 1



Which attribute (test condition) is the best?



# Selection of the best attribute

- Attributes with *homogeneous* class distribution are preferred
- Need a measure of node impurity

C0: 5
C1: 5

Non-homogeneous,  
high degree of impurity

C0: 9
C1: 1

Homogeneous, low  
degree of impurity

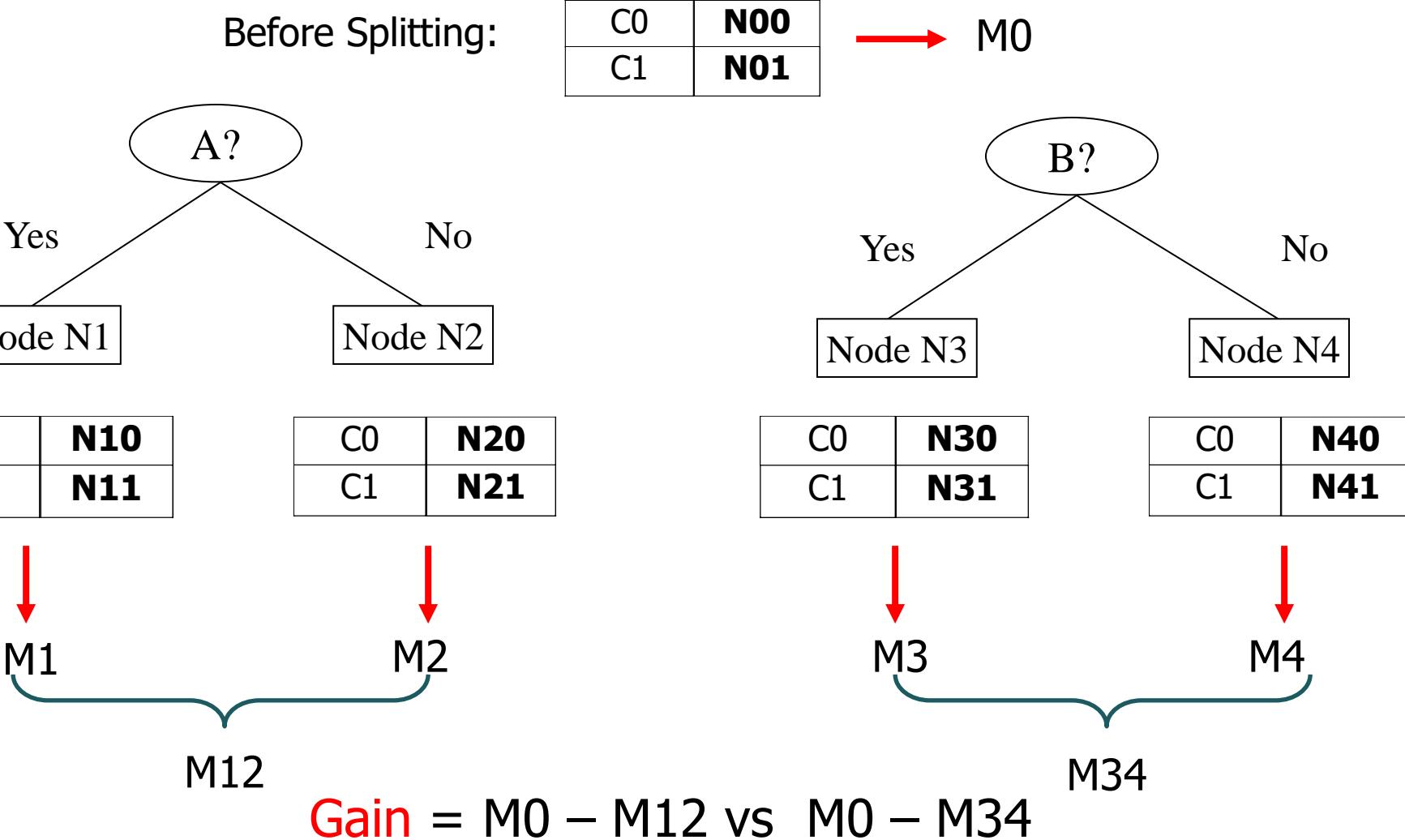


# Measures of node impurity

- Many different measures available
  - Gini index
  - Entropy
  - Misclassification error
- Different algorithms rely on different measures



# How to find the best attribute



From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



# GINI impurity measure

- Gini Index for a given node t

$$GINI(t) = 1 - \sum_j [p(j | t)]^2$$

$p(j / t)$  is the relative frequency of class j at node t

- Maximum ( $1 - 1/n_c$ ) when records are equally distributed among all classes, implying higher impurity degree
- Minimum (0.0) when all records belong to one class, implying lower impurity degree

C1	<b>0</b>
C2	<b>6</b>
<b>Gini=0.000</b>	

C1	<b>1</b>
C2	<b>5</b>
<b>Gini=0.278</b>	

C1	<b>2</b>
C2	<b>4</b>
<b>Gini=0.444</b>	

C1	<b>3</b>
C2	<b>3</b>
<b>Gini=0.500</b>	



# Examples for computing GINI

$$GINI(t) = 1 - \sum_j [p(j | t)]^2$$

C1	<b>0</b>
C2	<b>6</b>

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$

$$\text{Gini} = 1 - P(C1)^2 - P(C2)^2 = 1 - 0 - 1 = 0$$

C1	<b>1</b>
C2	<b>5</b>

$$P(C1) = 1/6 \quad P(C2) = 5/6$$

$$\text{Gini} = 1 - (1/6)^2 - (5/6)^2 = 0.278$$

C1	<b>2</b>
C2	<b>4</b>

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

$$\text{Gini} = 1 - (2/6)^2 - (4/6)^2 = 0.444$$



# Splitting based on GINI

- Used in CART, SLIQ, SPRINT
- When a node p is split into k partitions (children), the quality of the split is computed as

$$GINI_{split} = \sum_{i=1}^k \frac{n_i}{n} GINI(i)$$

where

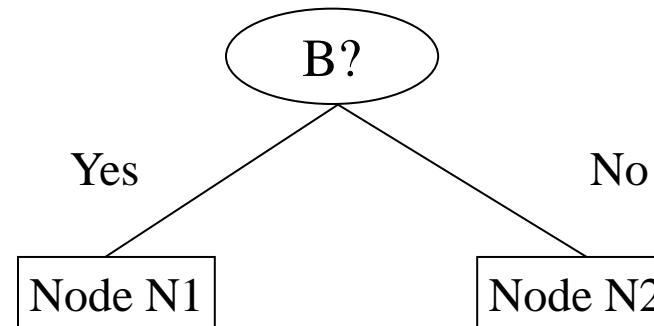
$n_i$  = number of records at child i

$n$  = number of records at node p



# Computing GINI index: Boolean attribute

- Splits into two partitions
  - larger and purer partitions are sought for



	<b>Parent</b>
C1	<b>6</b>
C2	<b>6</b>
<b>Gini</b>	<b>= 0.500</b>

$\text{Gini}(N1)$

$$\begin{aligned} &= 1 - (5/7)^2 - (2/7)^2 \\ &= 0.408 \end{aligned}$$

$\text{Gini}(N2)$

$$\begin{aligned} &= 1 - (1/5)^2 - (4/5)^2 \\ &= 0.32 \end{aligned}$$

	<b>N1</b>	<b>N2</b>
C1	<b>5</b>	<b>1</b>
C2	<b>2</b>	<b>4</b>
<b>Gini=?</b>		

$\text{Gini}(\text{split on } B)$

$$\begin{aligned} &= 7/12 * 0.408 + \\ &\quad 5/12 * 0.32 \\ &= 0.371 \end{aligned}$$



# Computing GINI index: Categorical attribute

- For each distinct value, gather counts for each class in the dataset
- Use the count matrix to make decisions

Multi-way split

	CarType		
	Family	Sports	Luxury
C1	1	2	1
C2	4	1	1
Gini	<b>0.393</b>		

Two-way split  
(find best partition of values)

-----

	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	<b>0.400</b>	

-----

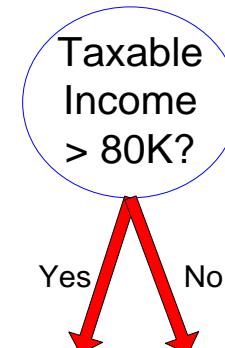
	CarType	
	{Sports}	{Family, Luxury}
C1	2	2
C2	1	5
Gini	<b>0.419</b>	



# Computing GINI index: Continuous attribute

- Binary decision on one splitting value
  - Number of possible splitting values  
= Number of distinct values
- Each splitting value  $v$  has a count matrix
  - class counts in the two partitions
    - $A < v$
    - $A \geq v$

Tid	Refund	Marital Status	Taxable Income	Cheat
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





# Computing GINI index: Continuous attribute

- For each attribute
  - Sort the attribute on values
  - Linearly scan these values, each time updating the count matrix and computing gini index
  - Choose the split position that has the least gini index

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No	No
Taxable Income											
	60	70	75	85	90	95	100	120	125	172	220
Sorted Values	55	65	72	80	87	92	97	110	122	172	230
Split Positions	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >
Yes	0 3	0 3	0 3	0 3	1 2	2 1	3 0	3 0	3 0	3 0	3 0
No	0 7	1 6	2 5	3 4	3 4	3 4	3 4	4 3	5 2	6 1	7 0
Gini	0.420	0.400	0.375	0.343	0.417	0.400	0.300	0.343	0.375	0.400	0.420



# Entropy impurity measure (INFO)

- Entropy at a given node t

$$Entropy(t) = -\sum_j p(j | t) \log_2 p(j | t)$$

$p(j | t)$  is the relative frequency of class j at node t

- Maximum ( $\log n_c$ ) when records are equally distributed among all classes, implying higher impurity degree
- Minimum (0.0) when all records belong to one class, implying lower impurity degree
- Entropy based computations are similar to GINI index computations



# Examples for computing entropy

$$Entropy(t) = -\sum_j p(j | t) \log_2 p(j | t)$$

C1	<b>0</b>
C2	<b>6</b>

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$

$$\text{Entropy} = -0 \log 0 - 1 \log 1 = -0 - 0 = 0$$

C1	<b>1</b>
C2	<b>5</b>

$$P(C1) = 1/6 \quad P(C2) = 5/6$$

$$\text{Entropy} = -(1/6) \log_2 (1/6) - (5/6) \log_2 (5/6) = 0.65$$

C1	<b>2</b>
C2	<b>4</b>

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

$$\text{Entropy} = -(2/6) \log_2 (2/6) - (4/6) \log_2 (4/6) = 0.92$$



# Splitting Based on INFO

- Information Gain

$$GAIN_{split} = Entropy(p) - \left( \sum_{i=1}^k \frac{n_i}{n} Entropy(i) \right)$$

Parent Node, p is split into k partitions;  
 $n_i$  is number of records in partition i

- Measures reduction in entropy achieved because of the split. Choose the split that achieves most reduction (maximizes GAIN)
- Used in ID3 and C4.5
- Disadvantage: Tends to prefer splits yielding a large number of partitions, each small but pure



# Splitting Based on INFO

- Gain Ratio

$$GainRATIO_{split} = \frac{GAIN_{split}}{SplitINFO}$$

$$SplitINFO = -\sum_{i=1}^k \frac{n_i}{n} \log \frac{n_i}{n}$$

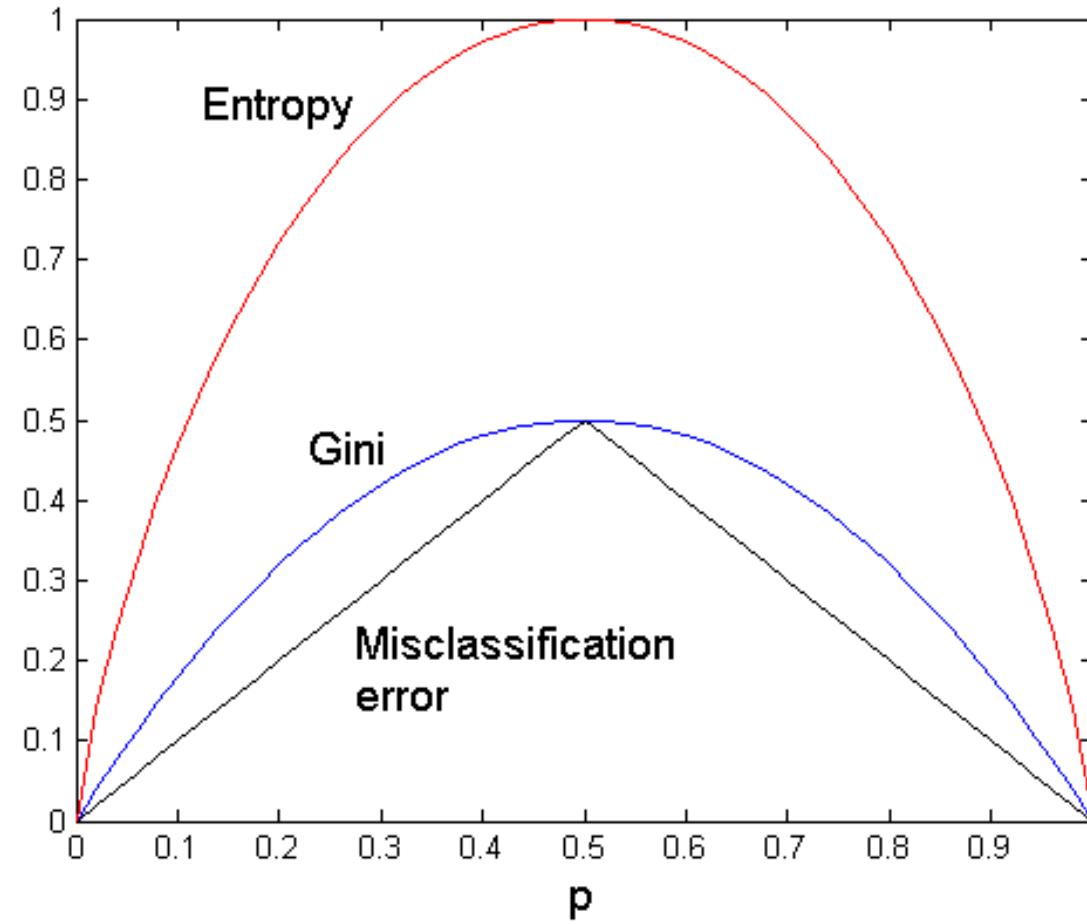
Parent Node, p is split into k partitions  
 $n_i$  is the number of records in partition i

- Adjusts Information Gain by the entropy of the partitioning (SplitINFO). Higher entropy partitioning (large number of small partitions) is penalized
- Used in C4.5
- Designed to overcome the disadvantage of Information Gain



# Comparison among splitting criteria

For a 2-class problem



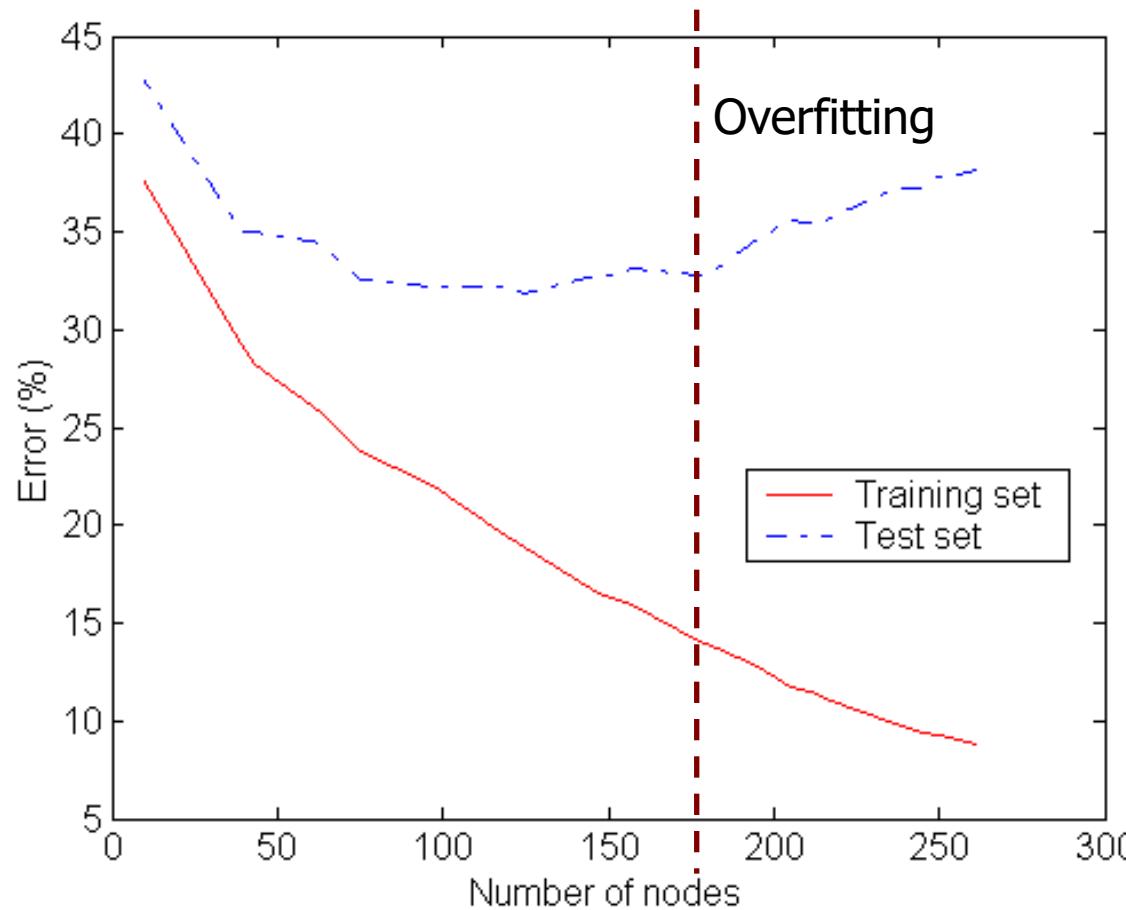


# Stopping Criteria for Tree Induction

- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when all the records have similar attribute values
- Early termination
  - Pre-pruning
  - Post-pruning



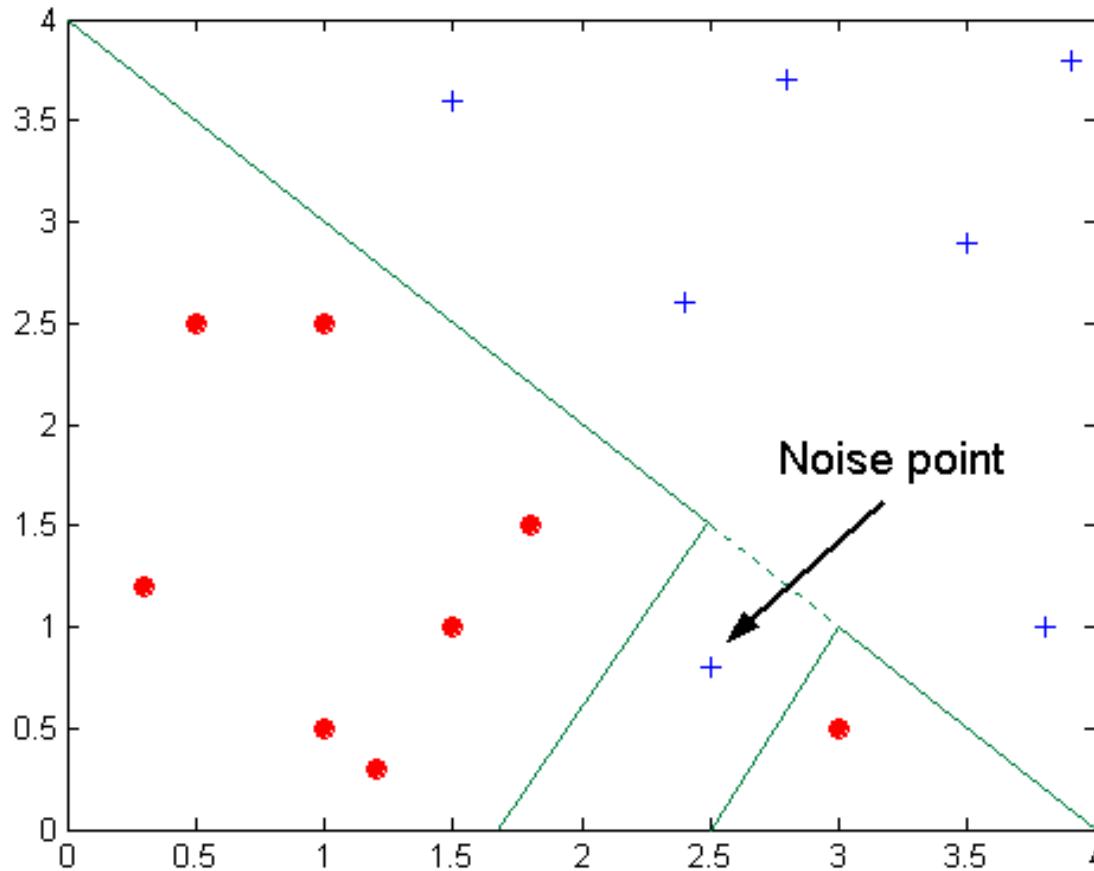
# Underfitting and Overfitting



Underfitting: when model is too simple, both training and test errors are large



# Overfitting due to Noise



Decision boundary is distorted by noise point



# How to address overfitting

- Pre-Pruning (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
  - Typical stopping conditions for a node
    - Stop if all instances belong to the same class
    - Stop if all the attribute values are the same
  - More restrictive conditions
    - Stop if number of instances is less than some user-specified threshold
    - Stop if class distribution of instances are independent of the available features (e.g., using  $\chi^2$  test)
    - Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain)



# How to address overfitting

## ■ Post-pruning

- Grow decision tree to its entirety
- Trim the nodes of the decision tree in a bottom-up fashion
- If generalization error improves after trimming, replace sub-tree by a leaf node.
- Class label of leaf node is determined from majority class of instances in the sub-tree



# Data fragmentation

- Number of instances gets smaller as you traverse down the tree
- Number of instances at the leaf nodes could be too small to make any statistically significant decision

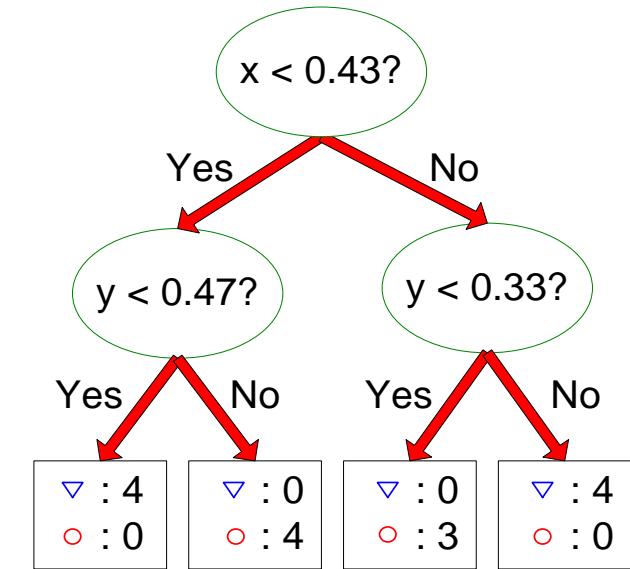
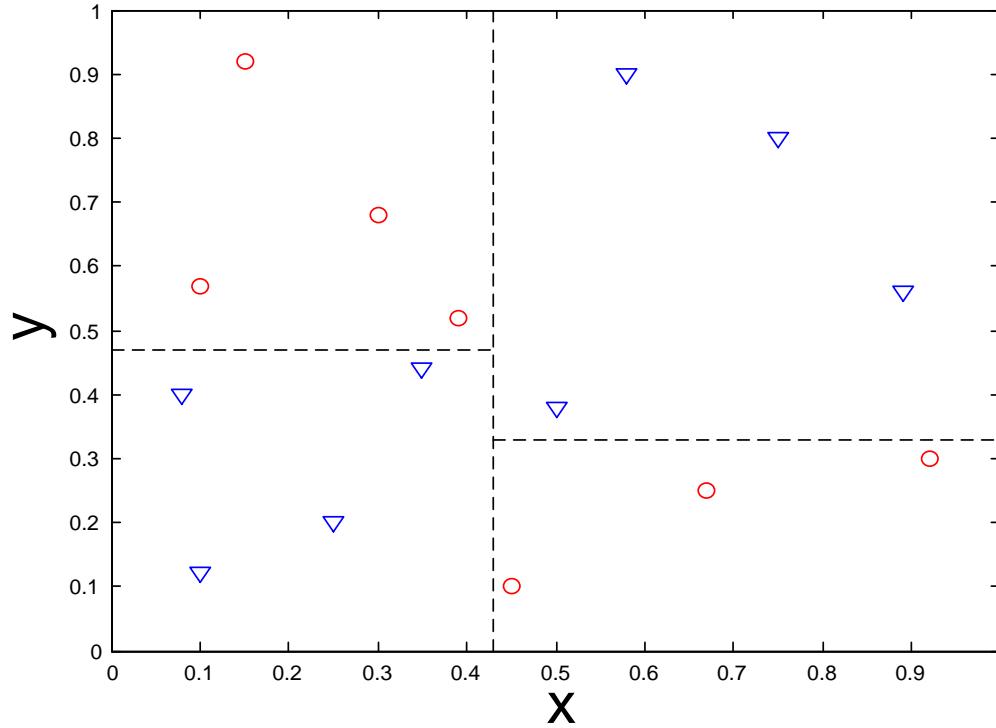


# Handling missing attribute values

- Missing values affect decision tree construction in three different ways
  - Affect how impurity measures are computed
  - Affect how to distribute instances with missing value to child nodes
  - Affect how a test instance with missing value is classified



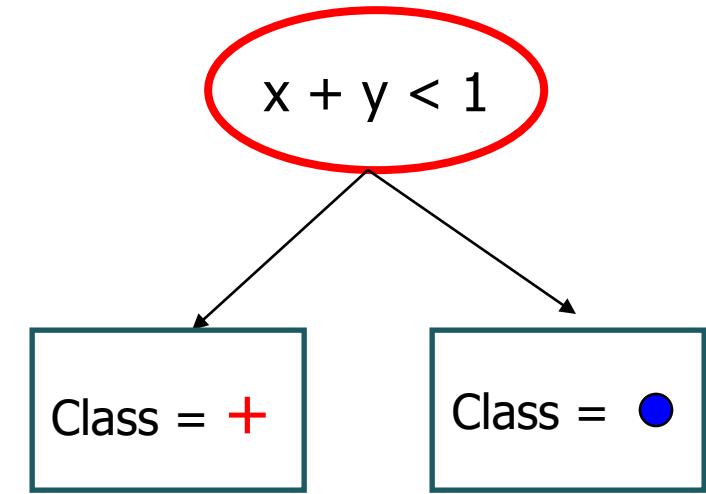
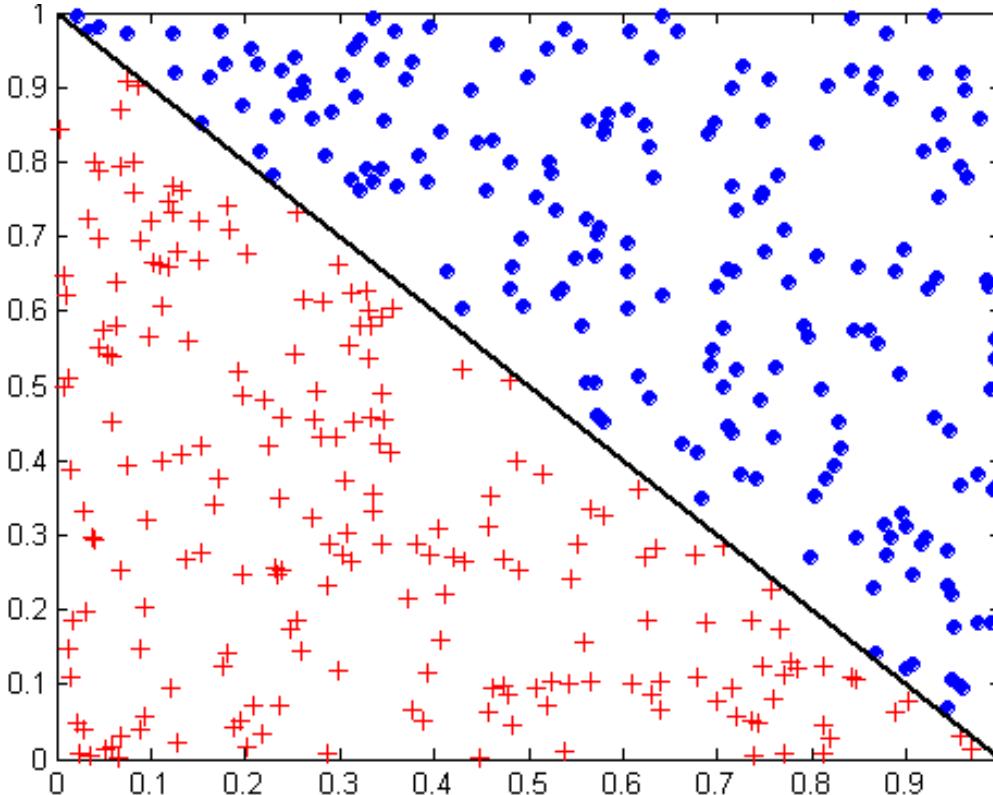
# Decision boundary



- Border line between two neighboring regions of different classes is known as decision boundary
- Decision boundary is parallel to axes because test condition involves a single attribute at-a-time



# Oblique decision trees



- Test condition may involve multiple attributes
- More expressive representation
- Finding optimal test condition is computationally expensive



# Evaluation of decision trees

- Accuracy
  - For simple datasets, comparable to other classification techniques
- Interpretability
  - Model is interpretable for small trees
  - Single predictions are interpretable
- Incrementality
  - Not incremental
- Efficiency
  - Fast model building
  - Very fast classification
- Scalability
  - Scalable both in training set size and attribute number
- Robustness
  - Difficult management of missing data



# Random Forest

Elena Baralis  
*Politecnico di Torino*



# Random Forest

- Ensemble learning technique
  - multiple base models are combined
    - to improve accuracy and stability
    - to avoid overfitting
- Random forest = set of decision trees
  - a number of decision trees are built at training time
  - the class is assigned by majority voting



# Random Forest

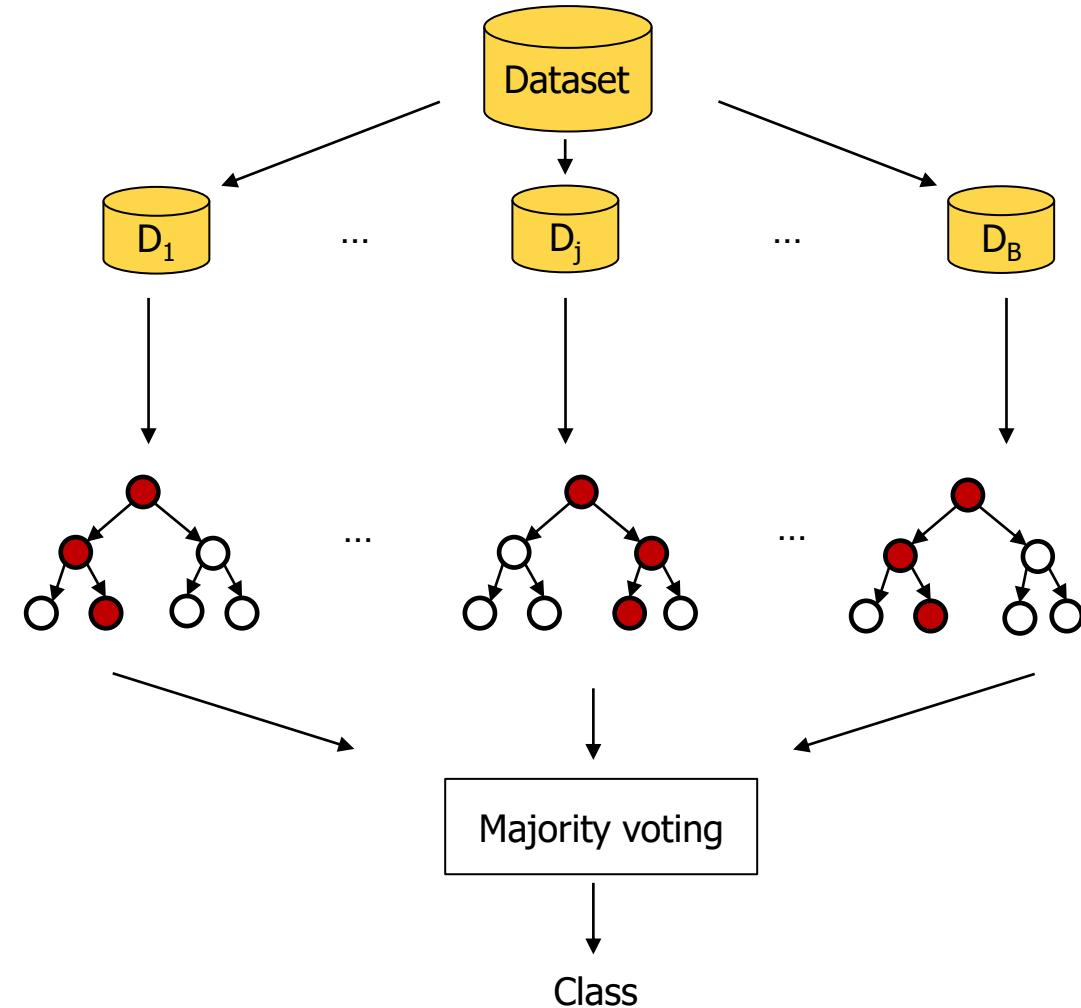
Original Training data

*Random* subsets

Multiple decision trees

For each subset, a tree is learned on a *random* set of features

Aggregating classifiers





# Bootstrap aggregation

- Given a training set  $D$  of  $n$  instances, it selects  $B$  times a *random* sample with replacement from  $D$  and trains trees on these dataset samples
  - For  $b = 1, \dots, B$ 
    - Sample with replacement  $n'$  training examples,  $n' \leq n$ 
      - A dataset subset  $D_b$  is generated
    - Train a classification tree on  $D_b$



# Feature bagging

- Selects, *for each candidate split* in the learning process, a *random* subset of the features
  - Being  $p$  the number of features,  $\sqrt{p}$  features are typically selected
- Trees are decorrelated
  - Feature subsets are sampled randomly, hence different features can be selected as best attributes for the split



# Random Forest – Algorithm Recap

- Given a training set  $D$  of  $n$  instances with  $p$  features
- For  $b = 1, \dots, B$ 
  - Sample randomly with replacement  $n'$  training examples. A subset  $D_b$  is generated
  - Train a classification tree on  $D_b$ 
    - During the tree construction, for each candidate split
      - $m \ll p$  random features are selected (typically  $m \approx \sqrt{p}$ )
      - the best split is computed among these  $m$  features
- Class is assigned by majority voting among the  $B$  predictions



# Evaluation of random forests

- Accuracy
  - Higher than decision trees
- Interpretability
  - Model and prediction are not interpretable
    - A prediction may be given by hundreds of trees
  - Provide global feature importance
    - an estimate of which features are important in the classification
- Incrementality
  - Not incremental
- Efficiency
  - Fast model building
  - Very fast classification
- Scalability
  - Scalable both in training set size and attribute number
- Robustness
  - Robust to noise and outliers

# Rule-based classification



Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis  
*Politecnico di Torino*



# Rule-based classifier

- Classify records by using a collection of “if...then...” rules
- Rule:  $(\textit{Condition}) \rightarrow y$ 
  - where
    - *Condition* is a conjunction of simple predicates
    - $y$  is the class label
  - *LHS*: rule antecedent or condition
  - *RHS*: rule consequent
- Examples of classification rules
  - $(\text{Blood Type}=\text{Warm}) \wedge (\text{Lay Eggs}=\text{Yes}) \rightarrow \text{Birds}$
  - $(\text{Taxable Income} < 50K) \wedge (\text{Refund}=\text{Yes}) \rightarrow \text{Cheat}=\text{No}$



# Rule-based Classifier (Example)

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
human	warm	yes	no	no	mammals
python	cold	no	no	no	reptiles
salmon	cold	no	no	yes	fishes
whale	warm	yes	no	yes	mammals
frog	cold	no	no	sometimes	amphibians
komodo	cold	no	no	no	reptiles
bat	warm	yes	yes	no	mammals
pigeon	warm	no	yes	no	birds
cat	warm	yes	no	no	mammals
leopard shark	cold	yes	no	yes	fishes
turtle	cold	no	no	sometimes	reptiles
penguin	warm	no	no	sometimes	birds
porcupine	warm	yes	no	no	mammals
eel	cold	no	no	yes	fishes
salamander	cold	no	no	sometimes	amphibians
gila monster	cold	no	no	no	reptiles
platypus	warm	no	no	no	mammals
owl	warm	no	yes	no	birds
dolphin	warm	yes	no	yes	mammals
eagle	warm	no	yes	no	birds

R1: (Give Birth = no)  $\wedge$  (Can Fly = yes)  $\rightarrow$  Birds

R2: (Give Birth = no)  $\wedge$  (Live in Water = yes)  $\rightarrow$  Fishes

R3: (Give Birth = yes)  $\wedge$  (Blood Type = warm)  $\rightarrow$  Mammals

R4: (Give Birth = no)  $\wedge$  (Can Fly = no)  $\rightarrow$  Reptiles

R5: (Live in Water = sometimes)  $\rightarrow$  Amphibians



# Rule-based classification

- A rule  $r$  **covers** an instance  $\mathbf{x}$  if the attributes of the instance satisfy the condition of the rule

R1: (Give Birth = no)  $\wedge$  (Can Fly = yes)  $\rightarrow$  Birds

R2: (Give Birth = no)  $\wedge$  (Live in Water = yes)  $\rightarrow$  Fishes

R3: (Give Birth = yes)  $\wedge$  (Blood Type = warm)  $\rightarrow$  Mammals

R4: (Give Birth = no)  $\wedge$  (Can Fly = no)  $\rightarrow$  Reptiles

R5: (Live in Water = sometimes)  $\rightarrow$  Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
hawk	warm	no	yes	no	?
grizzly bear	warm	yes	no	no	?

Rule R1 covers a hawk => Bird

Rule R3 covers the grizzly bear => Mammal



# Rule-based classification

R1: (Give Birth = no)  $\wedge$  (Can Fly = yes)  $\rightarrow$  Birds

R2: (Give Birth = no)  $\wedge$  (Live in Water = yes)  $\rightarrow$  Fishes

R3: (Give Birth = yes)  $\wedge$  (Blood Type = warm)  $\rightarrow$  Mammals

R4: (Give Birth = no)  $\wedge$  (Can Fly = no)  $\rightarrow$  Reptiles

R5: (Live in Water = sometimes)  $\rightarrow$  Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
lemur	warm	yes	no	no	?
turtle	cold	no	no	sometimes	?
dogfish shark	cold	yes	no	yes	?

A lemur triggers (only) rule R3, so it is classified as a mammal

A turtle triggers both R4 and R5

A dogfish shark triggers none of the rules

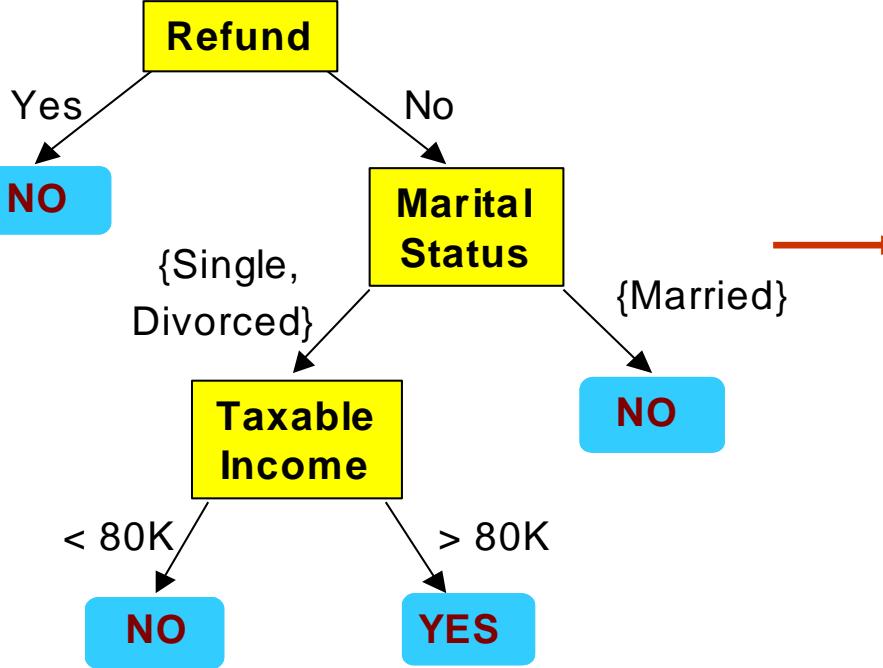


# Characteristics of rules

- *Mutually exclusive* rules
  - Two rule conditions can't be true at the same time
  - Every record is covered by at most one rule
- *Exhaustive* rules
  - Classifier rules account for every possible combination of attribute values
  - Each record is covered by at least one rule



# From decision trees to rules



## Classification Rules

(Refund=Yes) ==> No

(Refund>No, Marital Status={Single,Divorced},  
Taxable Income<80K) ==> No

(Refund>No, Marital Status={Single,Divorced},  
Taxable Income>80K) ==> Yes

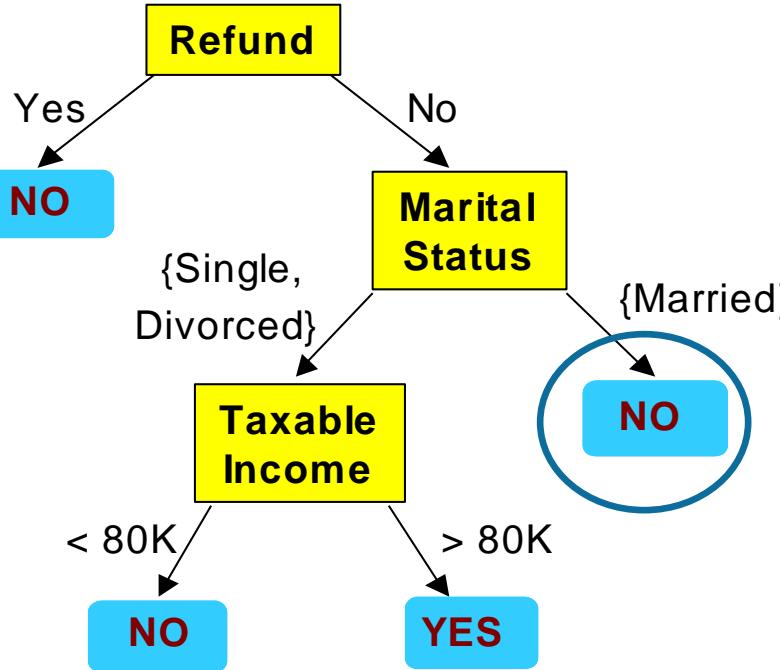
(Refund>No, Marital Status={Married}) ==> No

Rules are mutually exclusive and exhaustive

Rule set contains as much information as the tree



# Rules can be simplified



Tid	Refund	Marital Status	Taxable Income	Cheat
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

Initial Rule:  $(\text{Refund}=\text{No}) \wedge (\text{Status}=\text{Married}) \rightarrow \text{No}$

Simplified Rule:  $(\text{Status}=\text{Married}) \rightarrow \text{No}$



# Effect of rule simplification

- Rules are no longer mutually exclusive
  - A record may trigger more than one rule
  - Solution?
    - Ordered rule set
    - Unordered rule set – use voting schemes
- Rules are no longer exhaustive
  - A record may not trigger any rules
  - Solution?
    - Use a default class



# Ordered rule set

- Rules are rank ordered according to their priority
  - An ordered rule set is known as a decision list
- When a test record is presented to the classifier
  - It is assigned to the class label of the highest ranked rule it has triggered
  - If none of the rules fired, it is assigned to the default class

R1: (Give Birth = no)  $\wedge$  (Can Fly = yes)  $\rightarrow$  Birds

R2: (Give Birth = no)  $\wedge$  (Live in Water = yes)  $\rightarrow$  Fishes

R3: (Give Birth = yes)  $\wedge$  (Blood Type = warm)  $\rightarrow$  Mammals

R4: (Give Birth = no)  $\wedge$  (Can Fly = no)  $\rightarrow$  Reptiles

R5: (Live in Water = sometimes)  $\rightarrow$  Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
turtle	cold	no	no	sometimes	?



# Building classification rules

## ■ Direct Method

- Extract rules directly from data
- e.g.: RIPPER, CN2, Holte's 1R

## ■ Indirect Method

- Extract rules from other classification models (e.g. decision trees, neural networks, etc).
- e.g: C4.5rules



# Evaluation of rule based classifiers

- Accuracy
  - Higher than decision trees
- Interpretability
  - Model and prediction are interpretable
- Incrementality
  - Not incremental
- Efficiency
  - Fast model building
  - Very fast classification
- Scalability
  - Scalable both in training set size and attribute number
- Robustness
  - Robust to outliers

# Associative classification



Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis  
*Politecnico di Torino*



# Associative classification

- The classification model is defined by means of association rules

$$(Condition) \rightarrow y$$

- rule body is an itemset
- Model generation
  - Rule selection & sorting
    - based on support, confidence and correlation thresholds
  - Rule pruning
    - Database coverage: the training set is covered by selecting topmost rules according to previous sort



# Evaluation of associative classifiers

- Accuracy
  - Higher than decision trees and rule-based classifiers
    - *correlation* among attributes is considered
- Interpretability
  - Model and prediction are interpretable
- Incrementality
  - Not incremental
- Efficiency
  - Rule generation may be slow
    - It depends on support threshold
  - Very fast classification
- Scalability
  - Scalable in training set size
  - Reduced scalability in attribute number
    - Rule generation may become unfeasible
- Robustness
  - Unaffected by missing data
  - Robust to outliers

# K-Nearest Neighbor

D<sup>B</sup><sub>M</sub>G



Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis  
*Politecnico di Torino*



# Instance-Based Classifiers

## Set of Stored Cases

Atr1	.....	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

- Store the training records
- Use training records to predict the class label of unseen cases

## Unseen Case

Atr1	.....	AtrN



# Instance Based Classifiers

## ■ Examples

### ■ Rote-learner

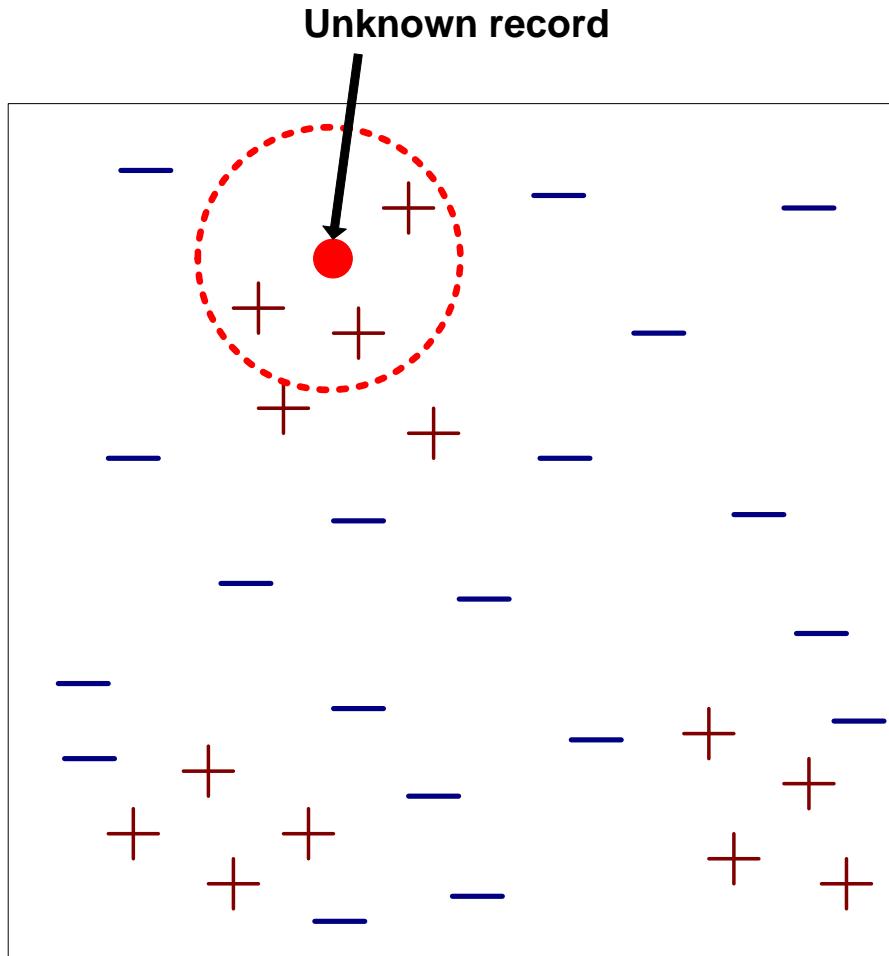
- Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly

### ■ Nearest neighbor

- Uses  $k$  “closest” points (nearest neighbors) for performing classification



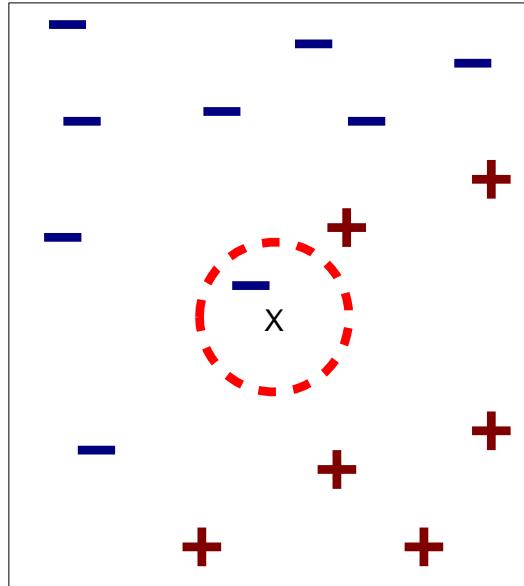
# Nearest-Neighbor Classifiers



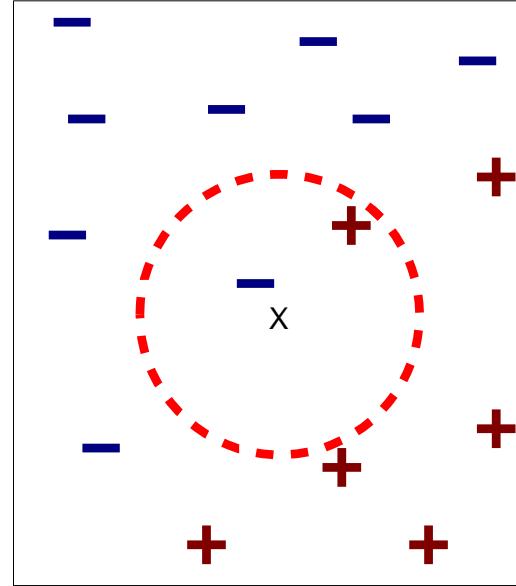
- Requires
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of  $k$ , the number of nearest neighbors to retrieve
  
- To classify an unknown record
  - Compute distance to other training records
  - Identify  $k$  nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)



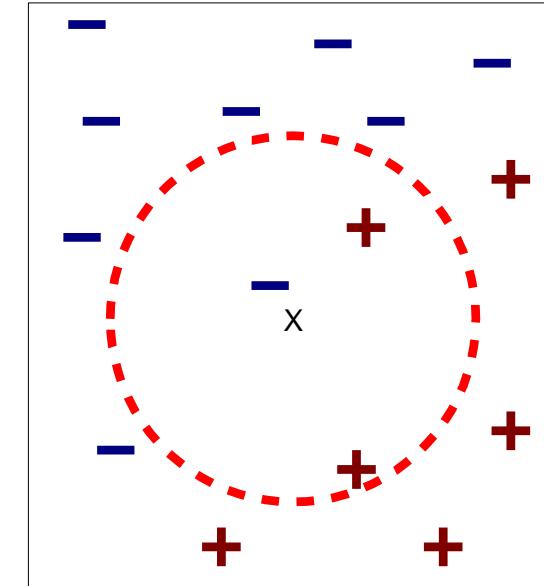
# Definition of Nearest Neighbor



(a) 1-nearest neighbor



(b) 2-nearest neighbor



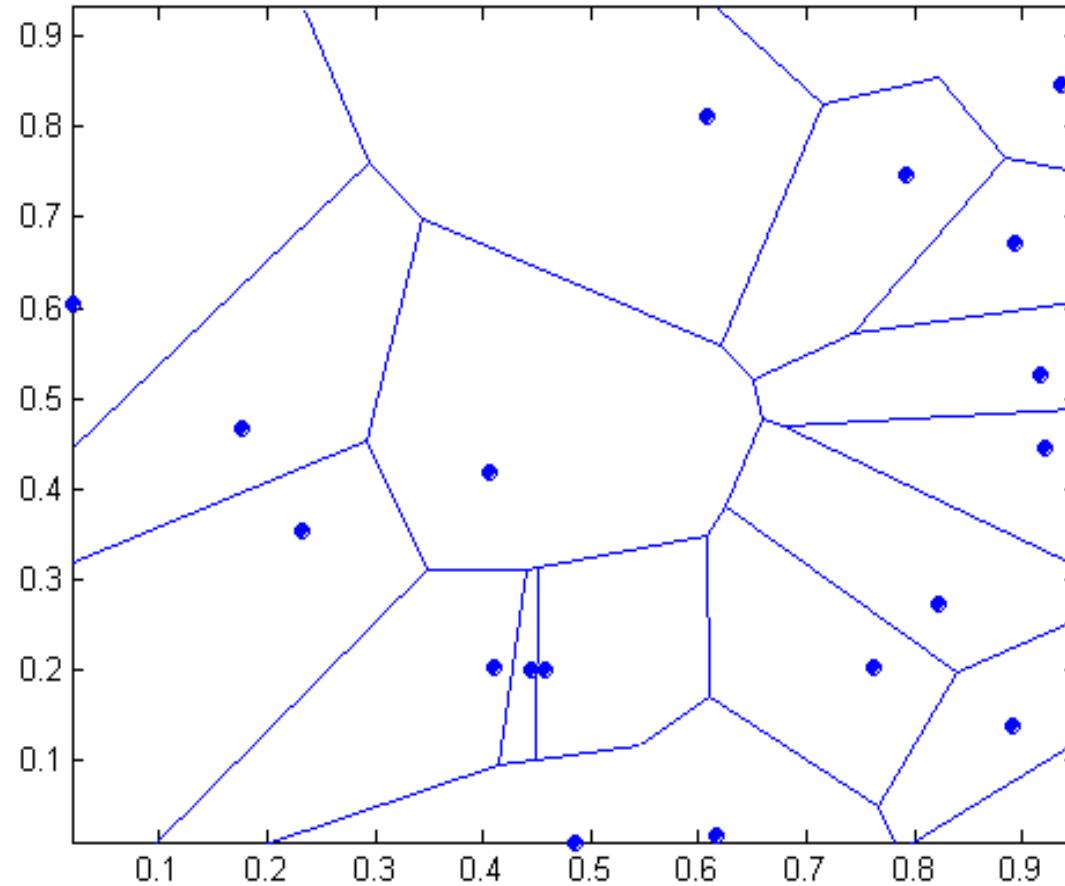
(c) 3-nearest neighbor

K-nearest neighbors of a record  $x$  are data points that have the  $k$  smallest distance to  $x$



# 1 nearest-neighbor

## Voronoi Diagram





# Nearest Neighbor Classification

- Compute distance between two points
  - Euclidean distance

$$d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$$

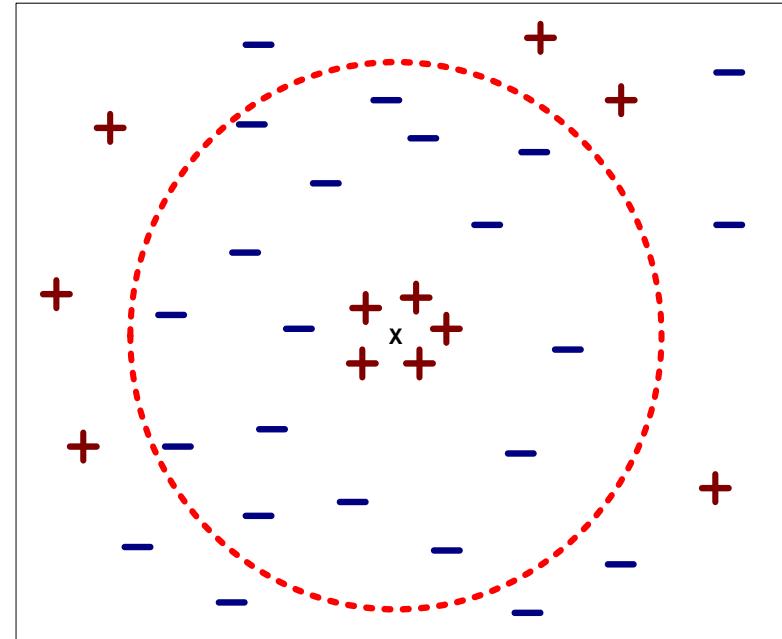
- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor,  $w = 1/d^2$



# Nearest Neighbor Classification

## ■ Choosing the value of k:

- If  $k$  is too small, sensitive to noise points
- If  $k$  is too large, neighborhood may include points from other classes





# Nearest Neighbor Classification

## ■ Scaling issues

- Attribute domain should be normalized to prevent distance measures from being dominated by one of the attributes
- Example: height [1.5m to 2.0m] vs. income [\$10K to \$1M]

## ■ Problem with distance measures

- High dimensional data
  - curse of dimensionality



# Evaluation of KNN

- Accuracy
  - Comparable to other classification techniques for simple datasets
- Interpretability
  - Model is not interpretable
  - Single predictions can be "described" by neighbors
- Incrementality
  - Incremental
  - Training set *must* be available
- Efficiency
  - (Almost) no model building
  - Slower classification, requires computing distances
- Scalability
  - Weakly scalable in training set size
  - Curse of dimensionality for increasing attribute number
- Robustness
  - Depends on distance computation

# Bayesian Classification



Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis  
*Politecnico di Torino*



# Bayes theorem

- Let  $C$  and  $X$  be random variables

$$P(C,X) = P(C|X) P(X)$$

$$P(C,X) = P(X|C) P(C)$$

- Hence

$$P(C|X) P(X) = P(X|C) P(C)$$

- and also

$$P(C|X) = P(X|C) P(C) / P(X)$$



# Bayesian classification

- Let the class attribute and all data attributes be random variables
  - $C = \text{any class label}$
  - $X = \langle x_1, \dots, x_k \rangle$  record to be classified
- Bayesian classification
  - compute  $P(C|X)$  for all classes
    - probability that record  $X$  belongs to  $C$
  - assign  $X$  to the class with *maximal*  $P(C|X)$
- Applying Bayes theorem

$$P(C|X) = P(X|C) \cdot P(C) / P(X)$$

- $P(X)$  constant for all  $C$ , disregarded for maximum computation
- $P(C)$  a priori probability of  $C$

$$P(C) = N_c/N$$



# Bayesian classification

- How to estimate  $P(X|C)$ , i.e.  $P(x_1, \dots, x_k|C)$ ?
- Naïve hypothesis

$$P(x_1, \dots, x_k|C) = P(x_1|C) P(x_2|C) \dots P(x_k|C)$$

- *statistical independence* of attributes  $x_1, \dots, x_k$
- not always true
  - model quality may be affected
- Computing  $P(x_k|C)$ 
  - for discrete attributes
$$P(x_k|C) = |x_{kC}| / N_c$$
    - where  $|x_{kC}|$  is number of instances having value  $x_k$  for attribute k and belonging to class C
  - for continuous attributes, use probability distribution
- Bayesian networks
  - allow specifying a subset of dependencies among attributes



# Bayesian classification: Example

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N



# Bayesian classification: Example

outlook		
$P(\text{sunny} p) = 2/9$	$P(\text{sunny} n) = 3/5$	$P(p) = 9/14$
$P(\text{overcast} p) = 4/9$	$P(\text{overcast} n) = 0$	$P(n) = 5/14$
$P(\text{rain} p) = 3/9$	$P(\text{rain} n) = 2/5$	
temperature		
$P(\text{hot} p) = 2/9$	$P(\text{hot} n) = 2/5$	
$P(\text{mild} p) = 4/9$	$P(\text{mild} n) = 2/5$	
$P(\text{cool} p) = 3/9$	$P(\text{cool} n) = 1/5$	
humidity		
$P(\text{high} p) = 3/9$	$P(\text{high} n) = 4/5$	
$P(\text{normal} p) = 6/9$	$P(\text{normal} n) = 2/5$	
windy		
$P(\text{true} p) = 3/9$	$P(\text{true} n) = 3/5$	
$P(\text{false} p) = 6/9$	$P(\text{false} n) = 2/5$	



# Bayesian classification: Example

- Data to be labeled

$$X = \langle \text{rain}, \text{hot}, \text{high}, \text{false} \rangle$$

- For class p

$$P(X|p) \cdot P(p) =$$

$$= P(\text{rain}|p) \cdot P(\text{hot}|p) \cdot P(\text{high}|p) \cdot P(\text{false}|p) \cdot P(p)$$

$$= 3/9 \cdot 2/9 \cdot 3/9 \cdot 6/9 \cdot 9/14 = 0.010582$$

- For class n

$$P(X|n) \cdot P(n) =$$

$$= P(\text{rain}|n) \cdot P(\text{hot}|n) \cdot P(\text{high}|n) \cdot P(\text{false}|n) \cdot P(n)$$

$$= 2/5 \cdot 2/5 \cdot 4/5 \cdot 2/5 \cdot 5/14 = \textcolor{red}{0.018286}$$



# Evaluation of Naïve Bayes Classifiers

- Accuracy
  - Similar or lower than decision trees
    - Naïve hypothesis simplifies model
- Interpretability
  - Model and prediction are not interpretable
    - The weights of contributions in a single prediction may be used to explain
- Incrementality
  - Fully incremental
  - Does *not* require availability of training data
- Efficiency
  - Fast model building
  - Very fast classification
- Scalability
  - Scalable both in training set size and attribute number
- Robustness
  - Affected by attribute correlation

# Support Vector Machines

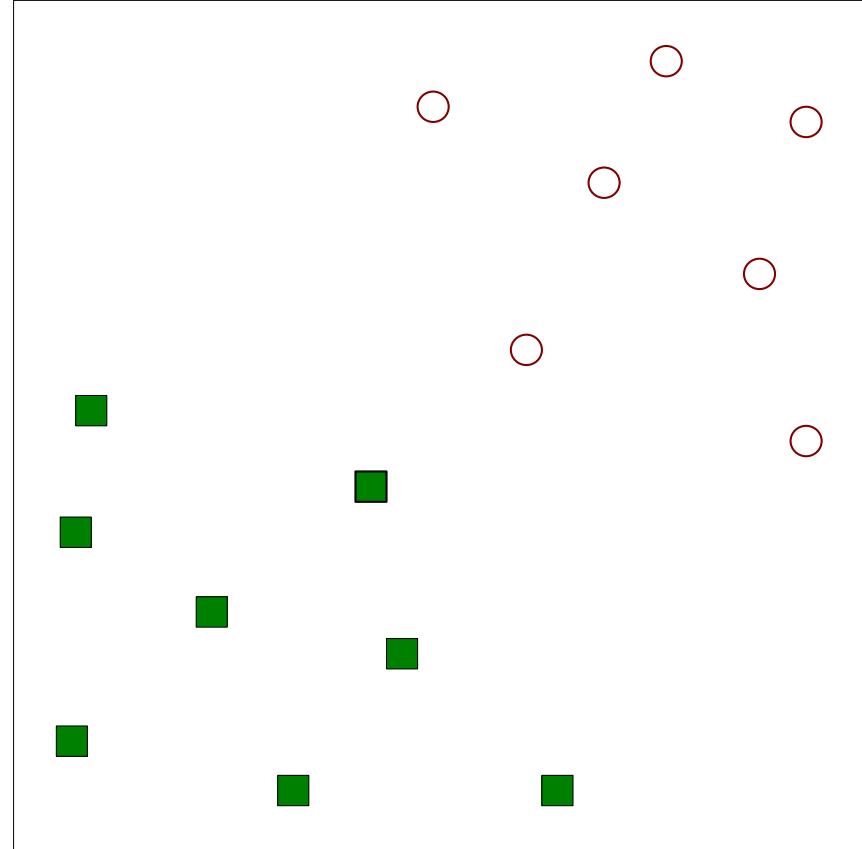


Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis  
*Politecnico di Torino*



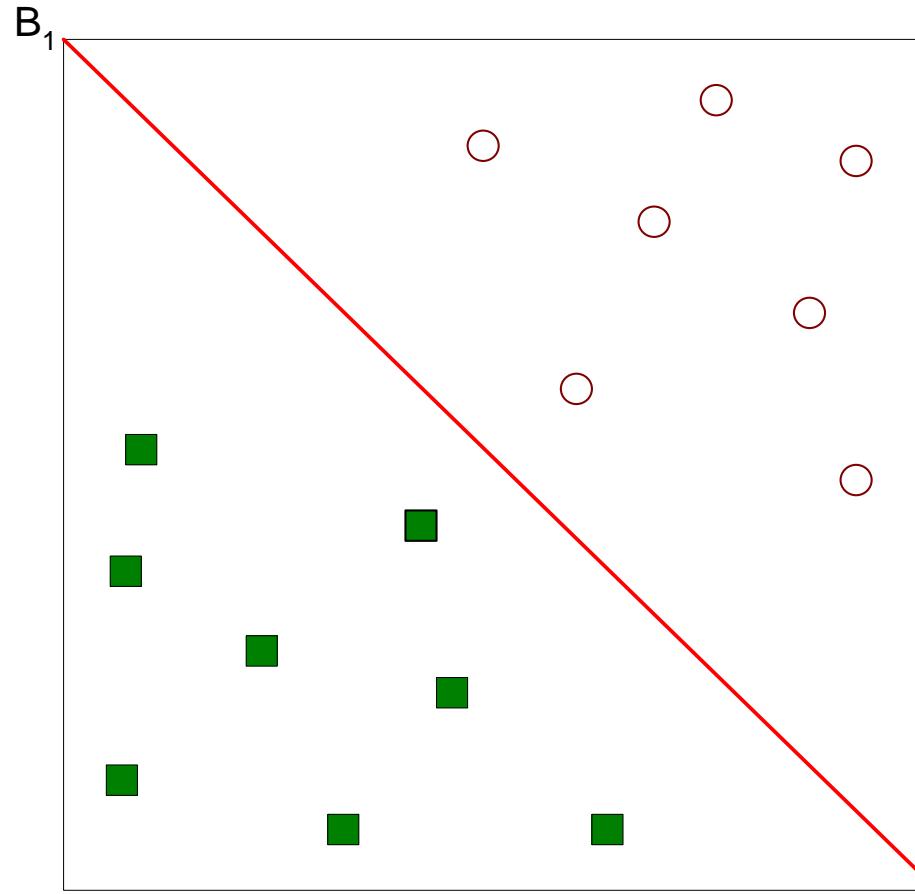
# Support Vector Machines



- Find a linear hyperplane (decision boundary) that will separate the data



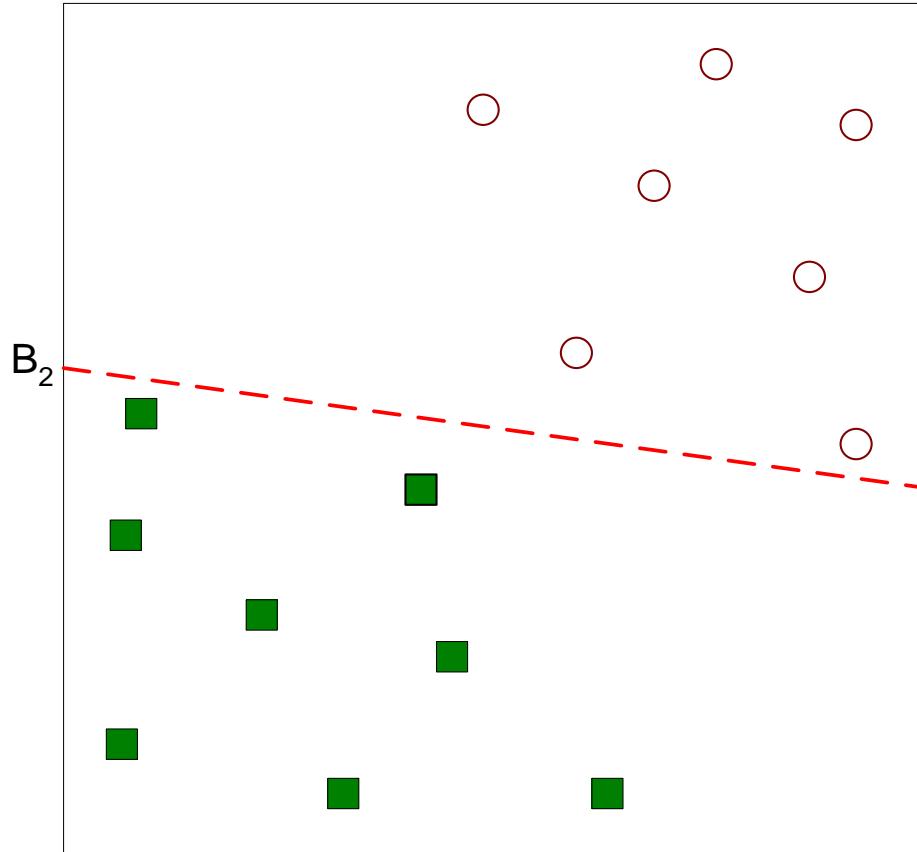
# Support Vector Machines



- One Possible Solution



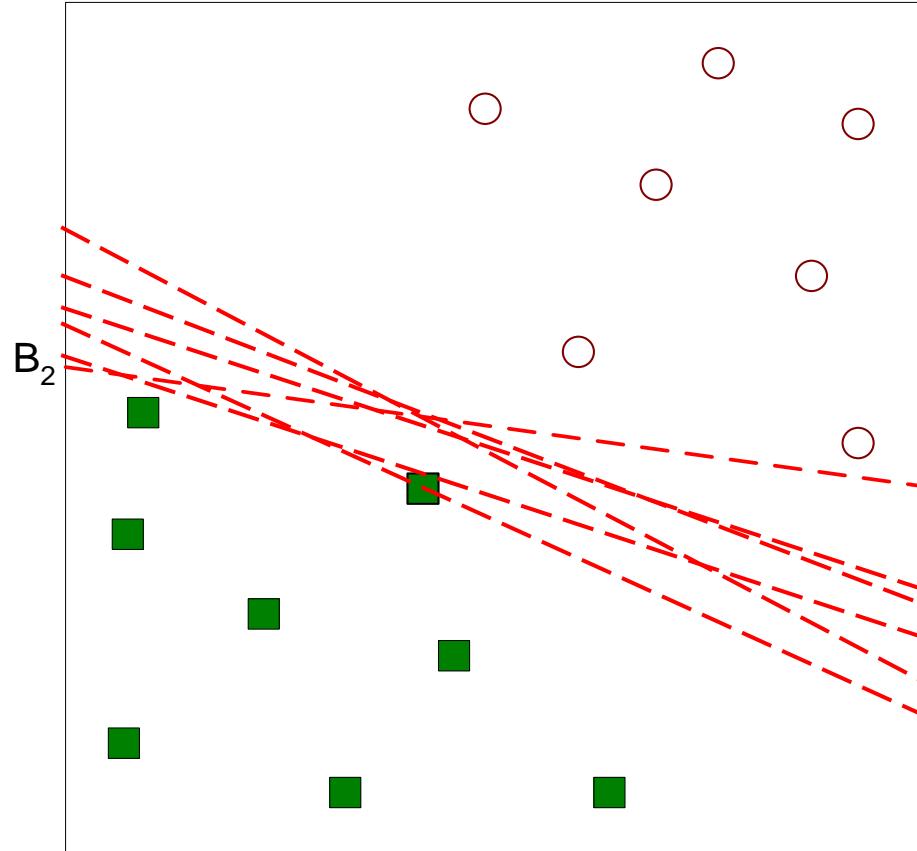
# Support Vector Machines



- Another possible solution



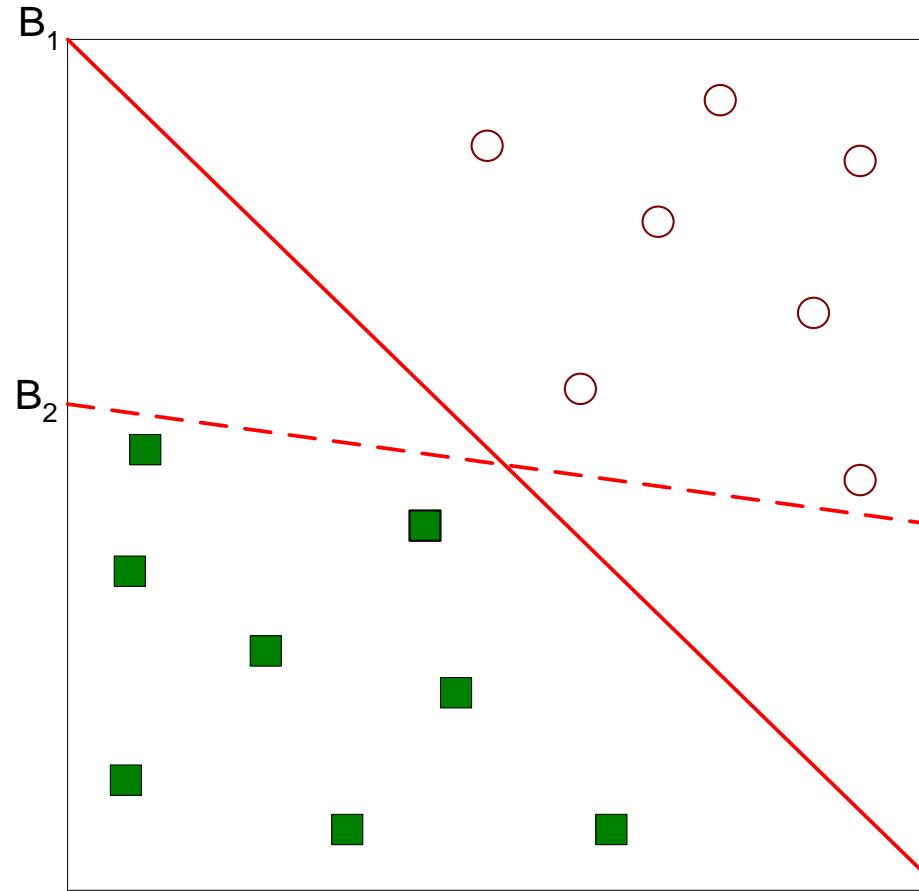
# Support Vector Machines



- Other possible solutions



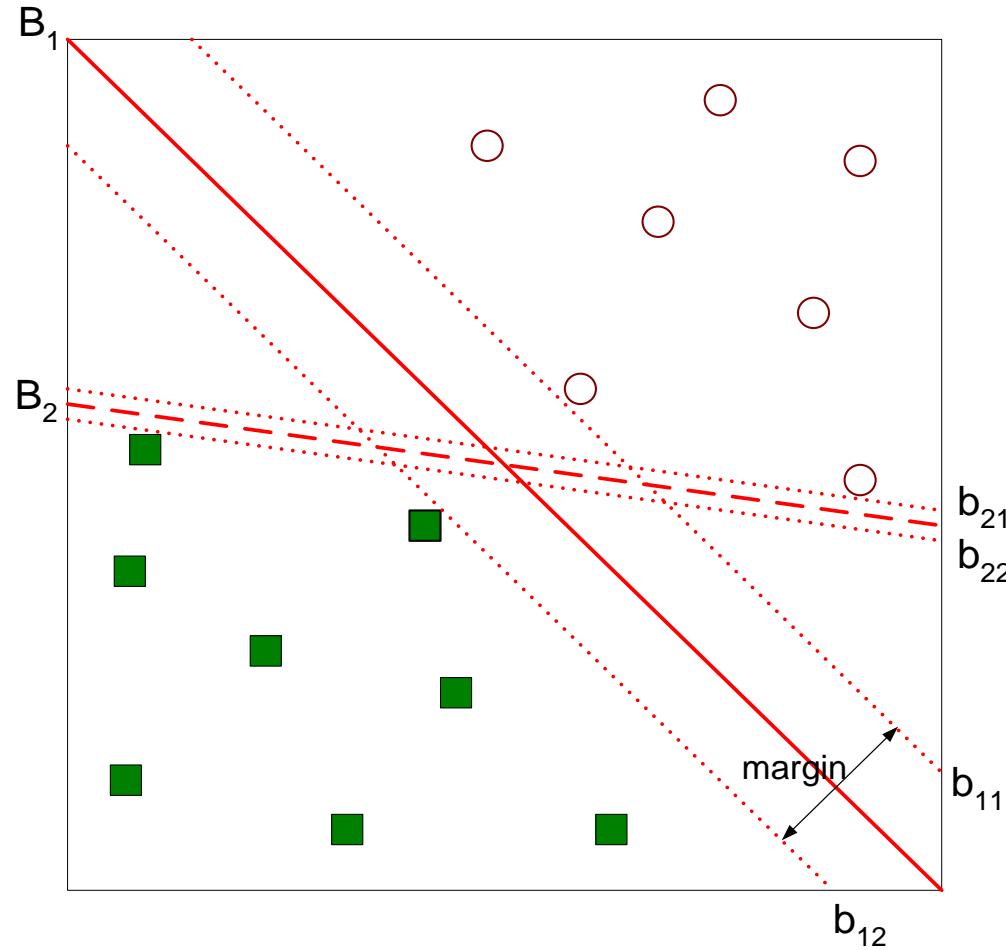
# Support Vector Machines



- Which one is better? B1 or B2?
- How do you define better?



# Support Vector Machines

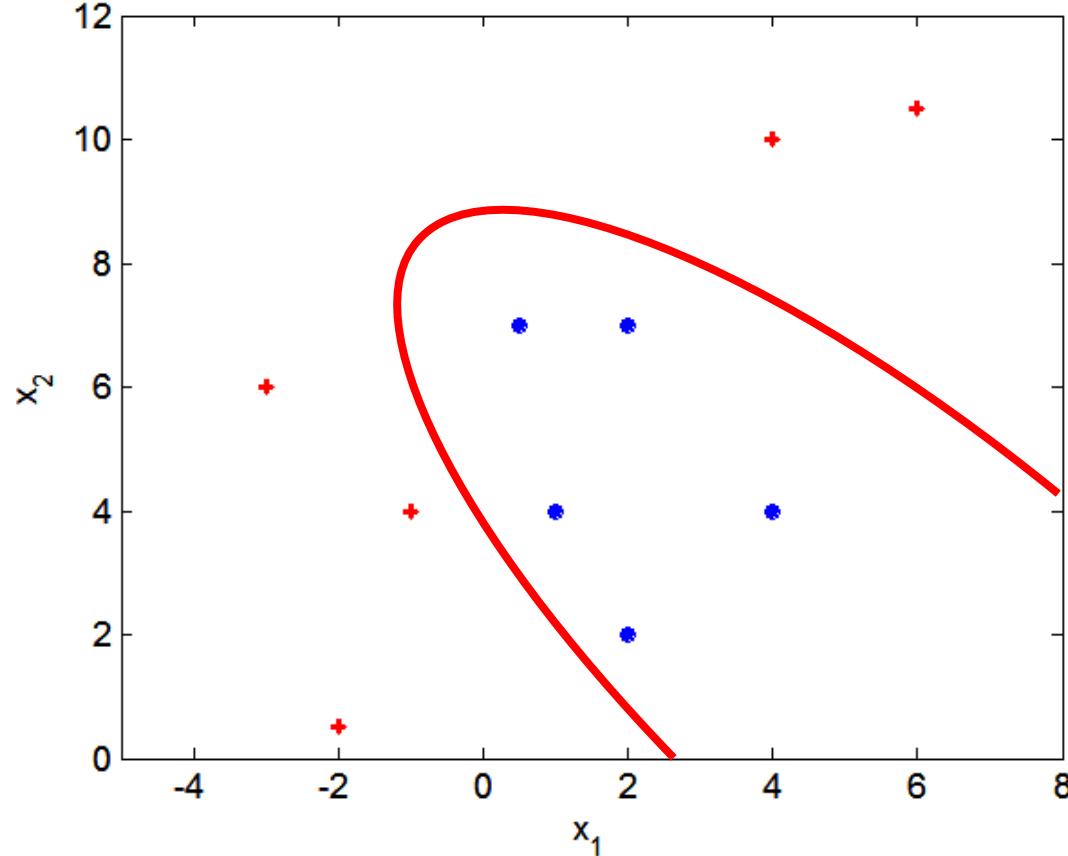


- Find hyperplane **maximizes** the margin => B1 is better than B2



# Nonlinear Support Vector Machines

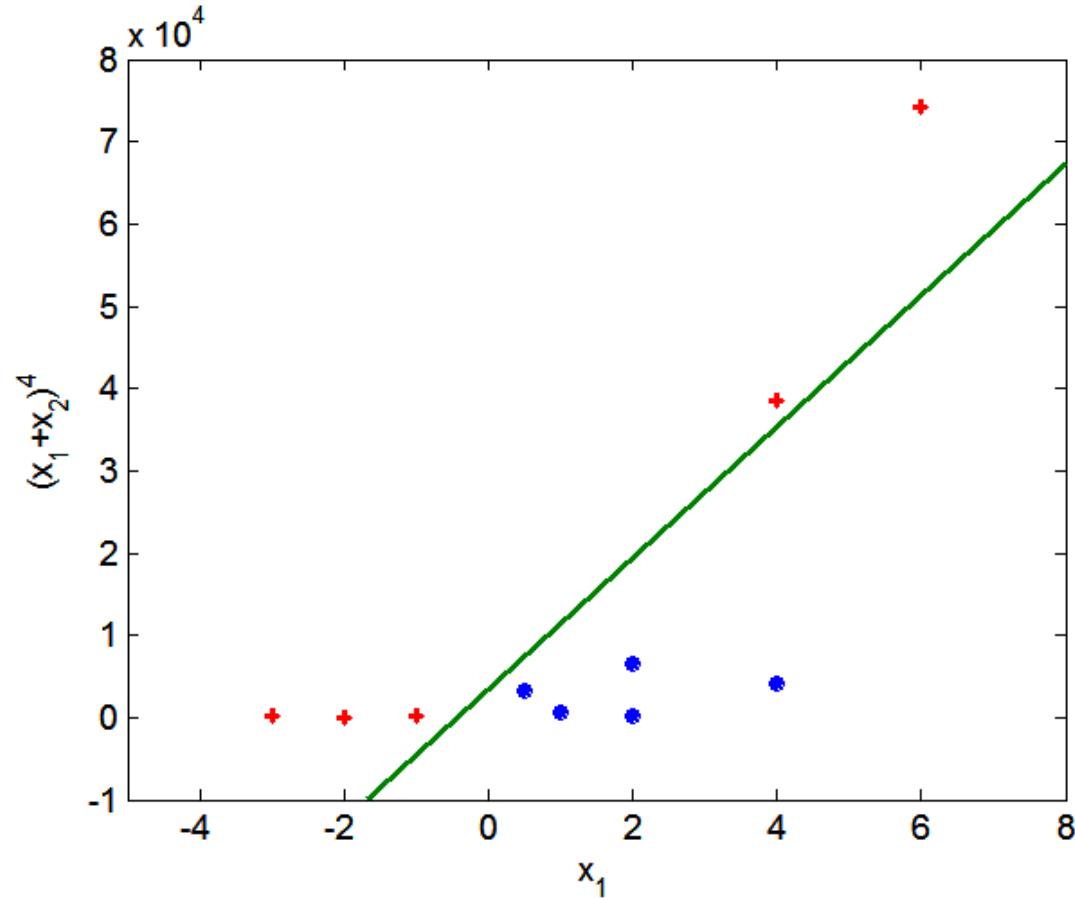
- What if decision boundary is not linear?





# Nonlinear Support Vector Machines

- Transform data into higher dimensional space





# Evaluation of Support Vector Machines

- Accuracy
  - Among best performers
- Interpretability
  - Model and prediction are not interpretable
    - Black box model
- Incrementality
  - Not incremental
- Efficiency
  - Model building requires significant parameter tuning
  - Very fast classification
- Scalability
  - Medium scalable both in training set size and attribute number
- Robustness
  - Robust to noise and outliers

# Artificial Neural Networks



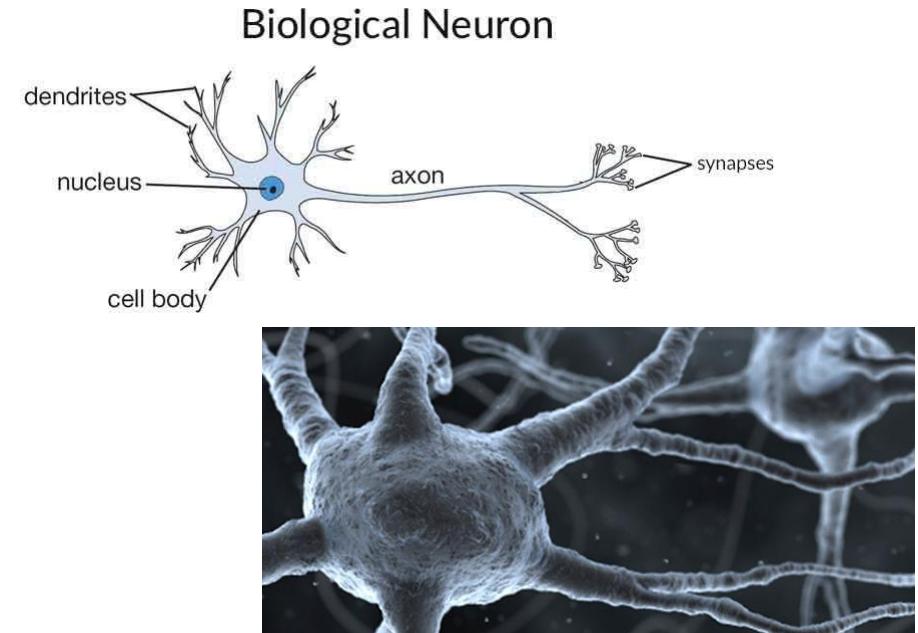
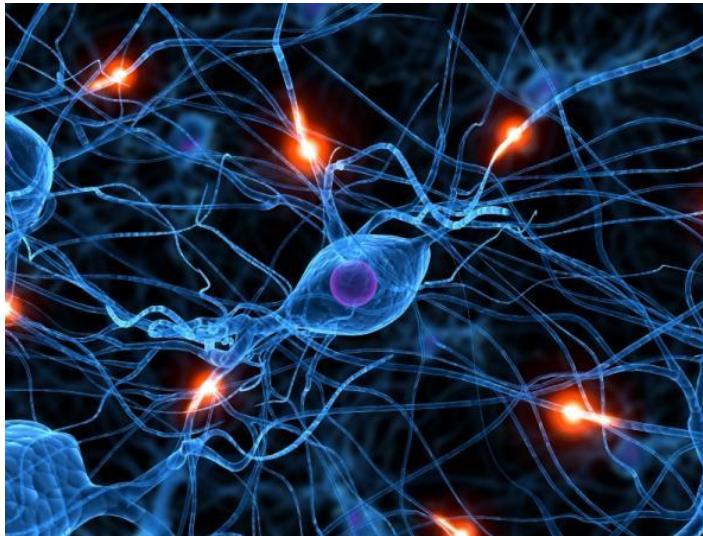
Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis  
*Politecnico di Torino*



# Artificial Neural Networks

- Inspired to the structure of the human brain
  - Neurons as elaboration units
  - Synapses as connection network

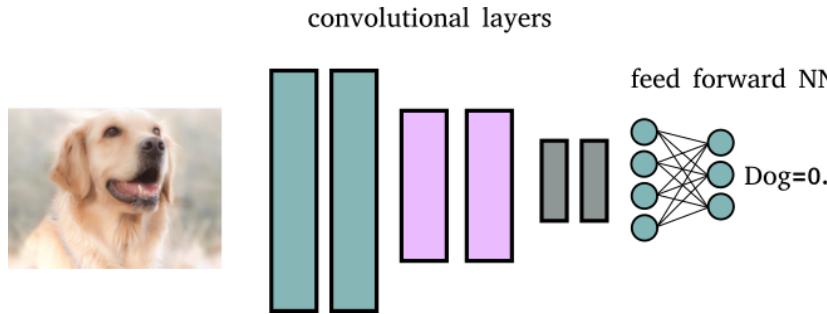




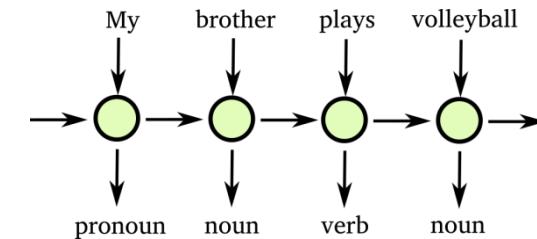
# Artificial Neural Networks

## ■ Different tasks, different architectures

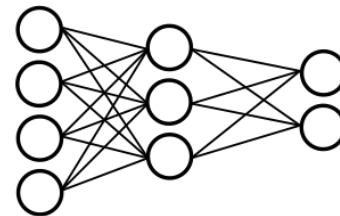
image understanding: convolutional NN (CNN)



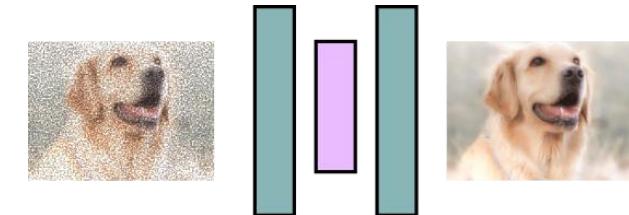
time series analysis: recurrent NN (RNN)



numerical vectors classification: feed forward NN (FFNN)

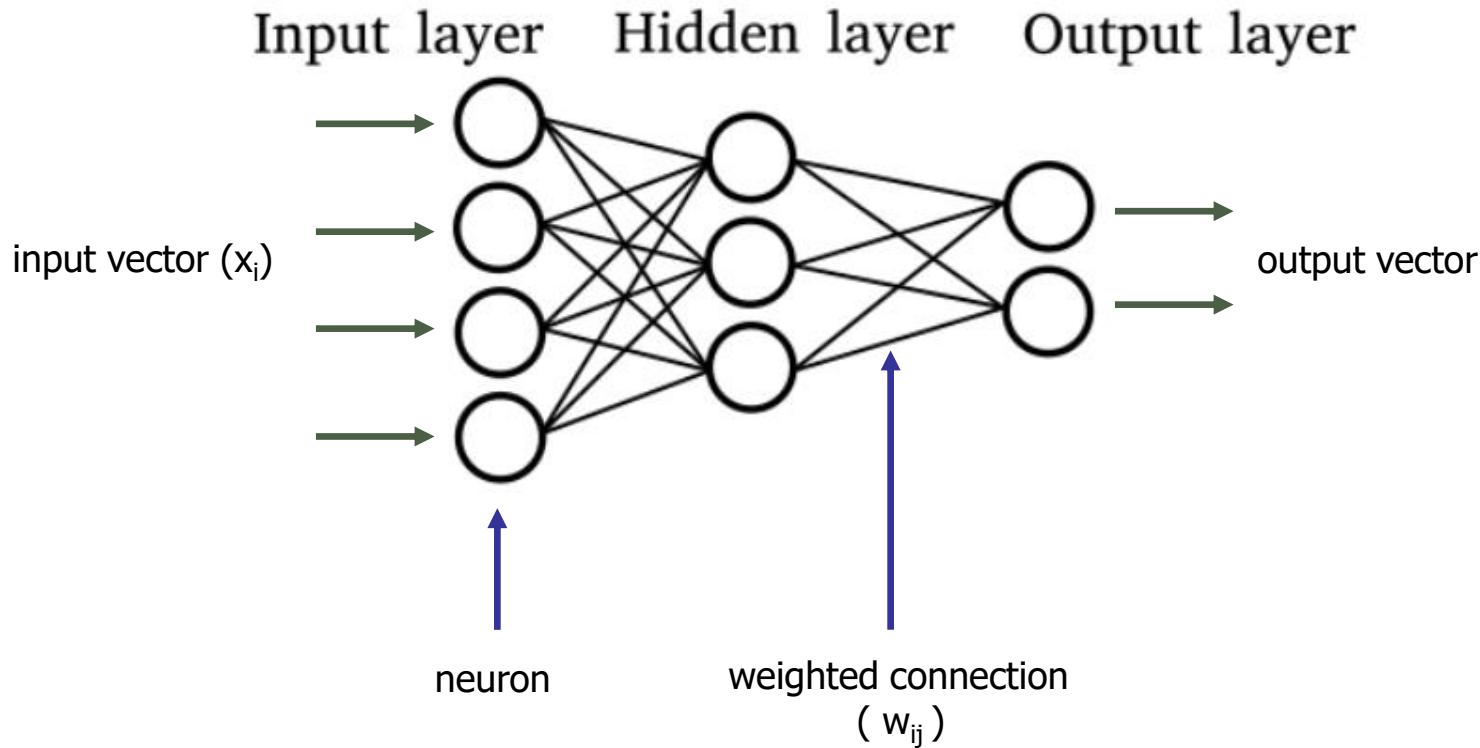


denoising: auto-encoders



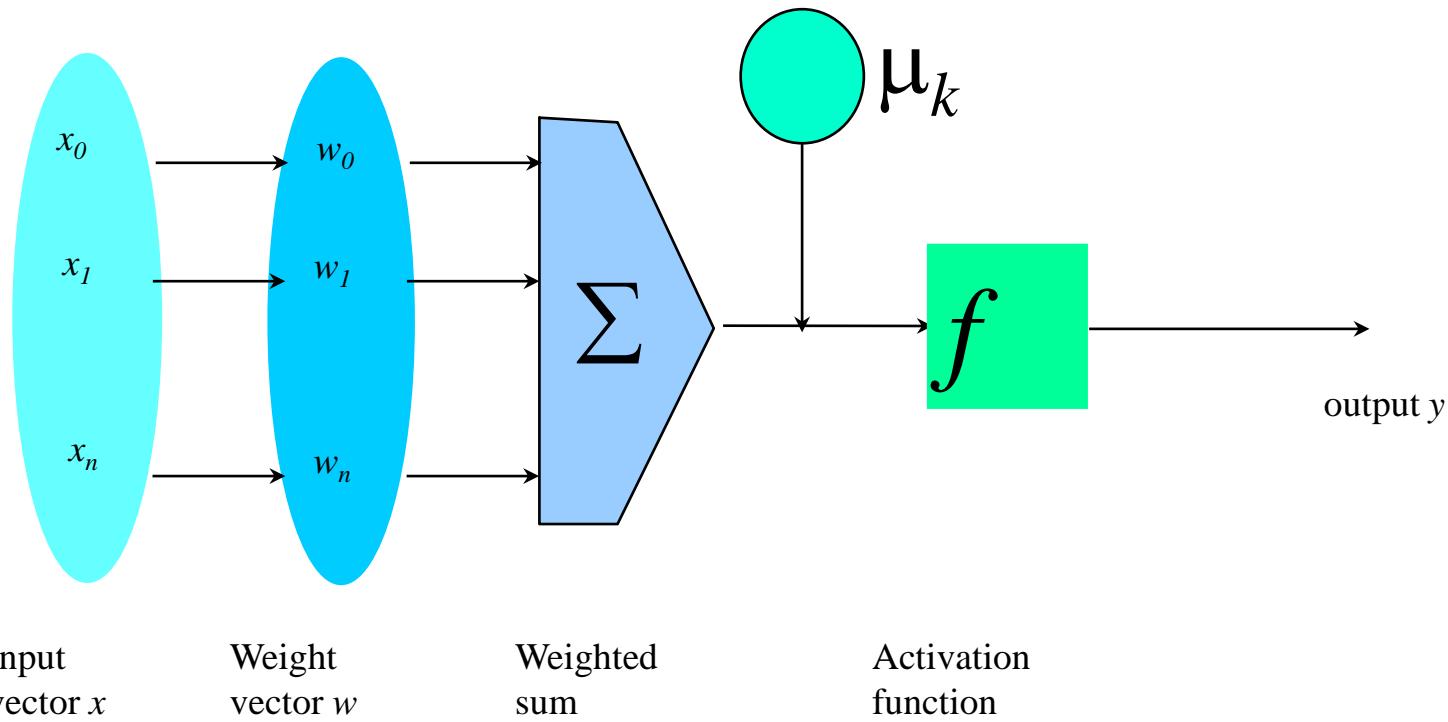


# Feed Forward Neural Network





# Structure of a neuron

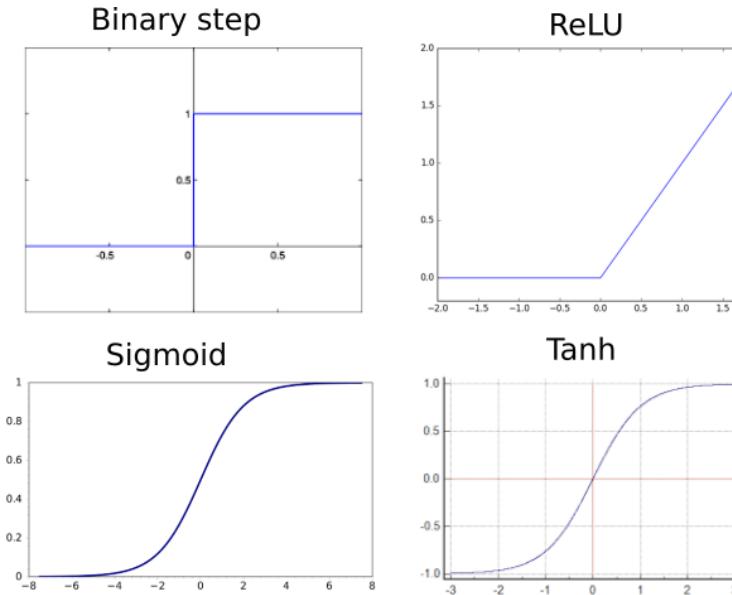




# Activation Functions

## ■ Activation

- simulates biological activation to input stimulus
- provides non-linearity to the computation
- may help to saturate neuron outputs in fixed ranges

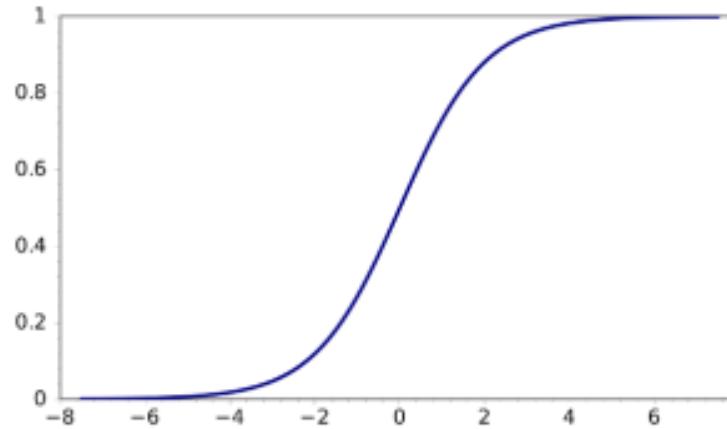




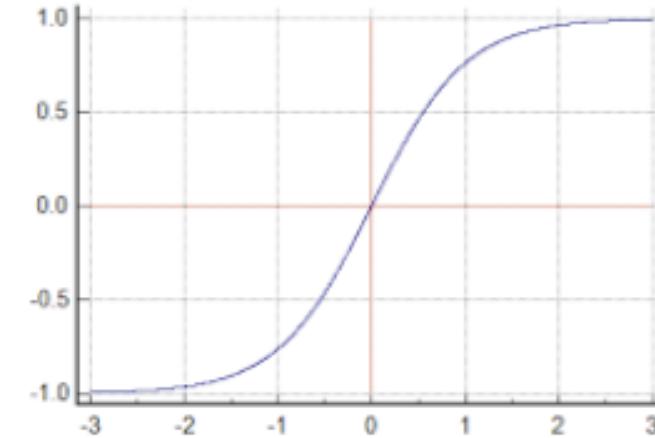
# Activation Functions

- Sigmoid, tanh
  - saturate input value in a fixed range
  - non linear for all the input scale
  - typically used by FFNNs for both hidden and output layers
    - E.g. *sigmoid* in output layers allows generating values between 0 and 1 (useful when output must be interpreted as likelihood)

Sigmoid



Tanh



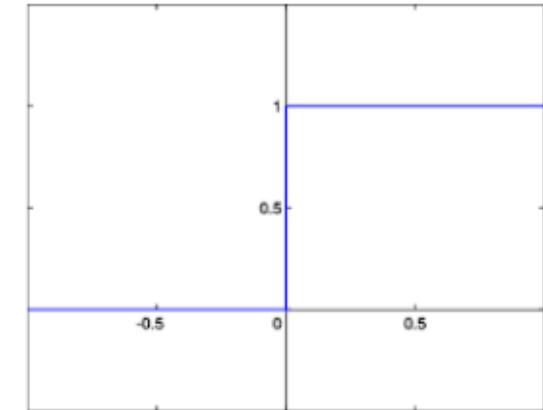


# Activation Functions

## ■ Binary Step

- outputs 1 when input is non-zero
- useful for binary outputs
- **issues:** not appropriate for gradient descent
  - derivative not defined in  $x=0$
  - derivative equal to 0 in every other position

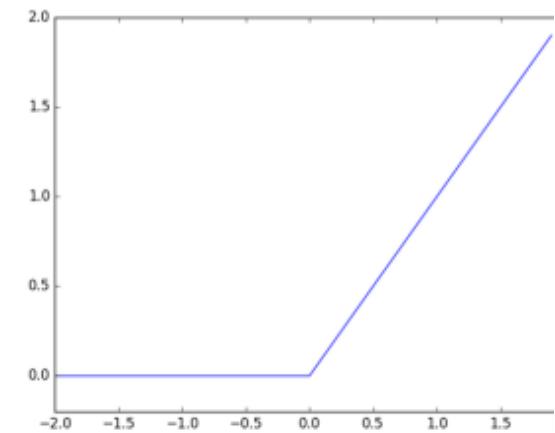
Binary step



## ■ ReLU (Rectified Linear Unit)

- used in deep networks (e.g. CNNs)
  - avoids vanishing gradient
  - does not saturate
- neurons activate linearly only for positive input

ReLU



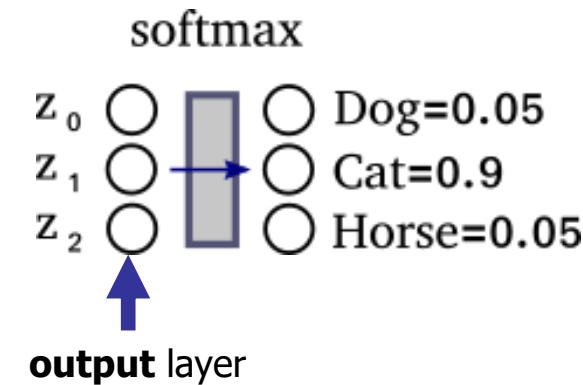


# Activation Functions

## ■ Softmax

- differently to other activation functions
  - it is applied only to the **output** layer
  - works by considering **all the neurons** in the layer
- after softmax, the output vector can be interpreted as a discrete **distribution of probabilities**
  - e.g. the probabilities for the input pattern of belonging to each class

$$\text{softmax}(z_j) = \frac{e^{z_j}}{\sum_{i=0}^{N-1} e^{z_i}}$$





# Building a FFNN

- For each node, definition of
  - set of weights
  - offset valueproviding the highest accuracy on the training data
- Iterative approach on training data instances



# Building a FFNN

## ■ Base algorithm

- Initially assign random values to weights and offsets
- Process instances in the training set one at a time
  - For each neuron, compute the result when applying weights, offset and activation function for the instance
  - Forward propagation until the output is computed
  - Compare the computed output with the expected output, and evaluate error
  - Backpropagation of the error, by updating weights and offset for each neuron
- The process ends when
  - % of accuracy above a given threshold
  - % of parameter variation (error) below a given threshold
  - The maximum number of epochs is reached



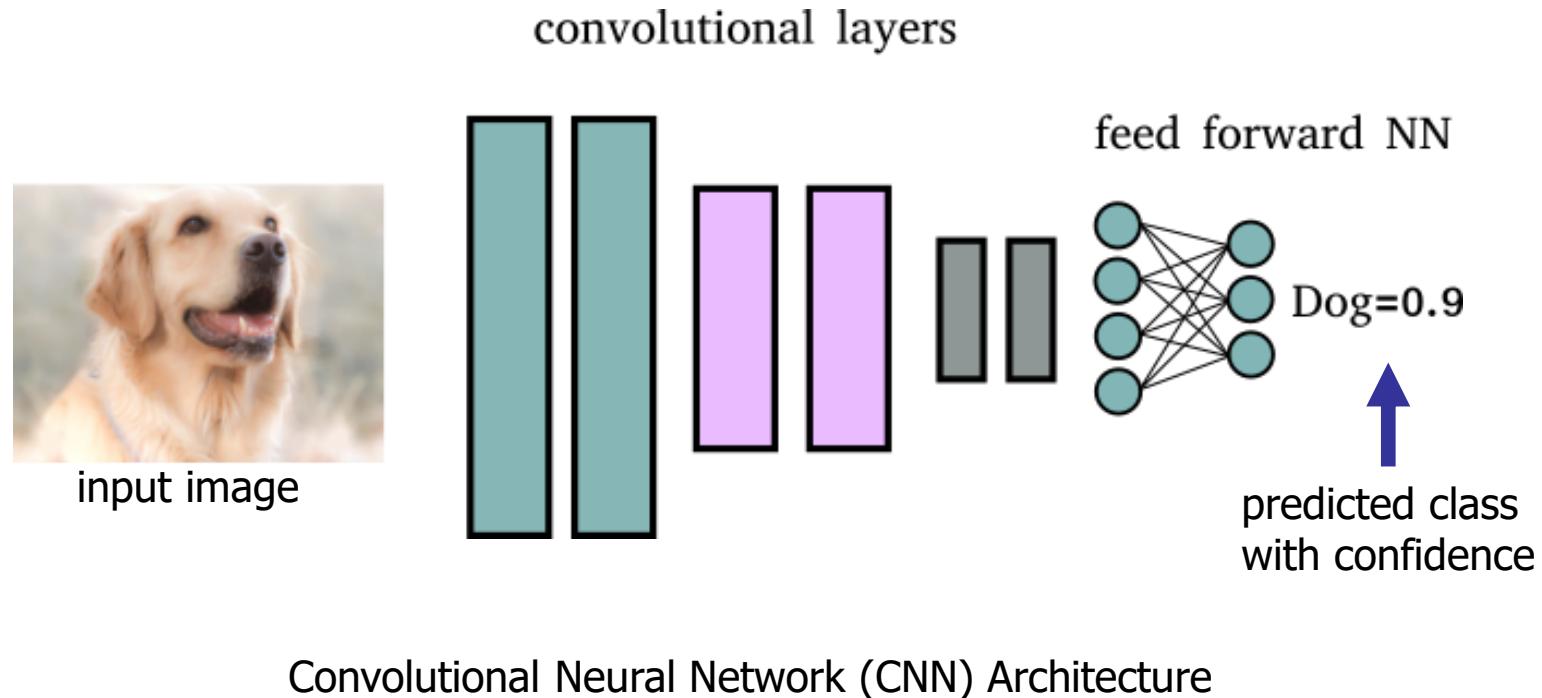
# Evaluation of Feed Forward NN

- Accuracy
  - Among best performers
- Interpretability
  - Model and prediction are not interpretable
    - Black box model
- Incrementality
  - Not incremental
- Efficiency
  - Model building requires *very complex* parameter tuning
    - It requires significant time
  - Very fast classification
- Scalability
  - Medium scalable both in training set size and attribute number
- Robustness
  - Robust to noise and outliers
  - Requires large training set
    - Otherwise unstable when tuning parameters



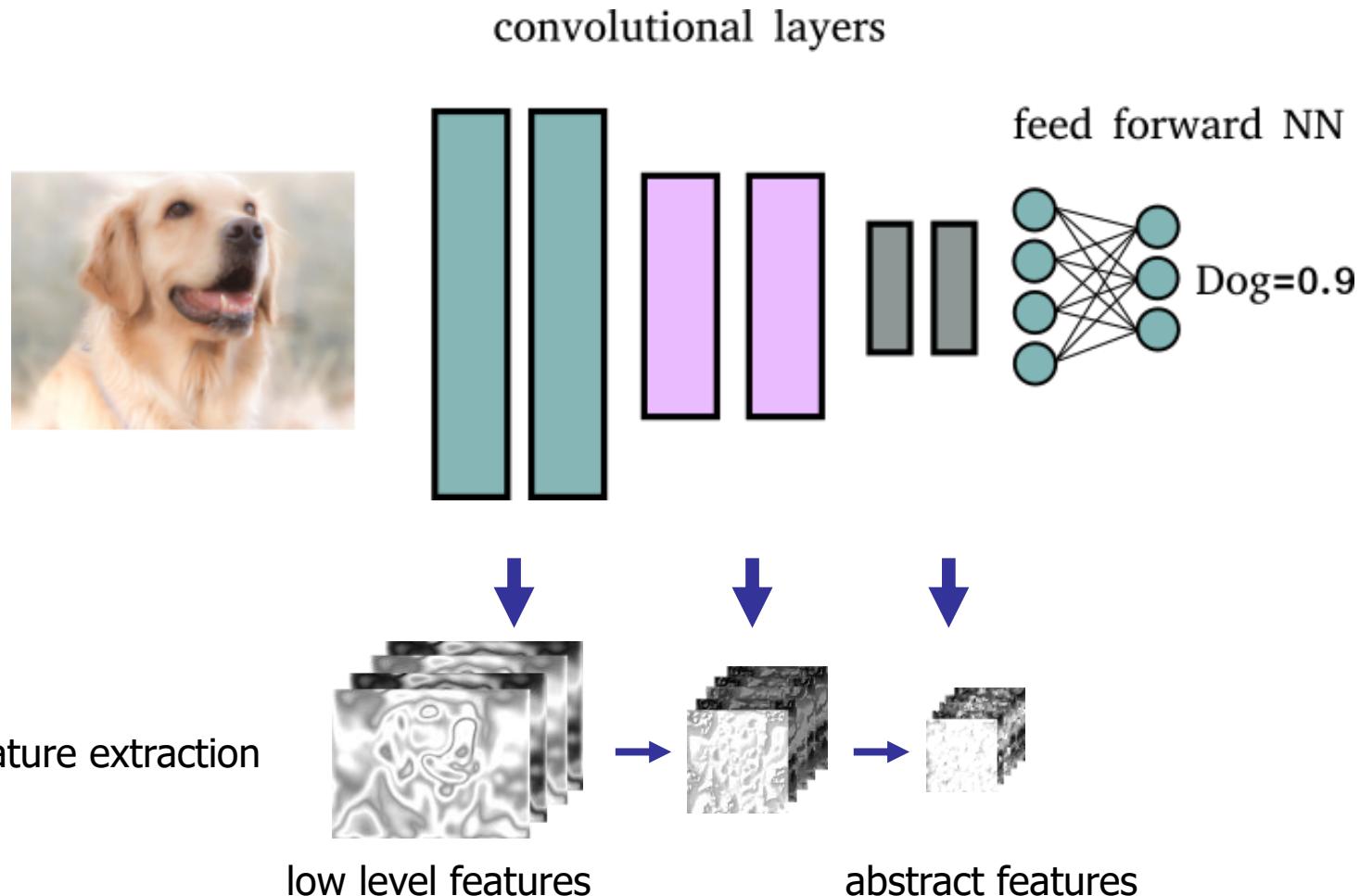
# Convolutional Neural Networks

- Allow automatically extracting **features** from images and performing **classification**



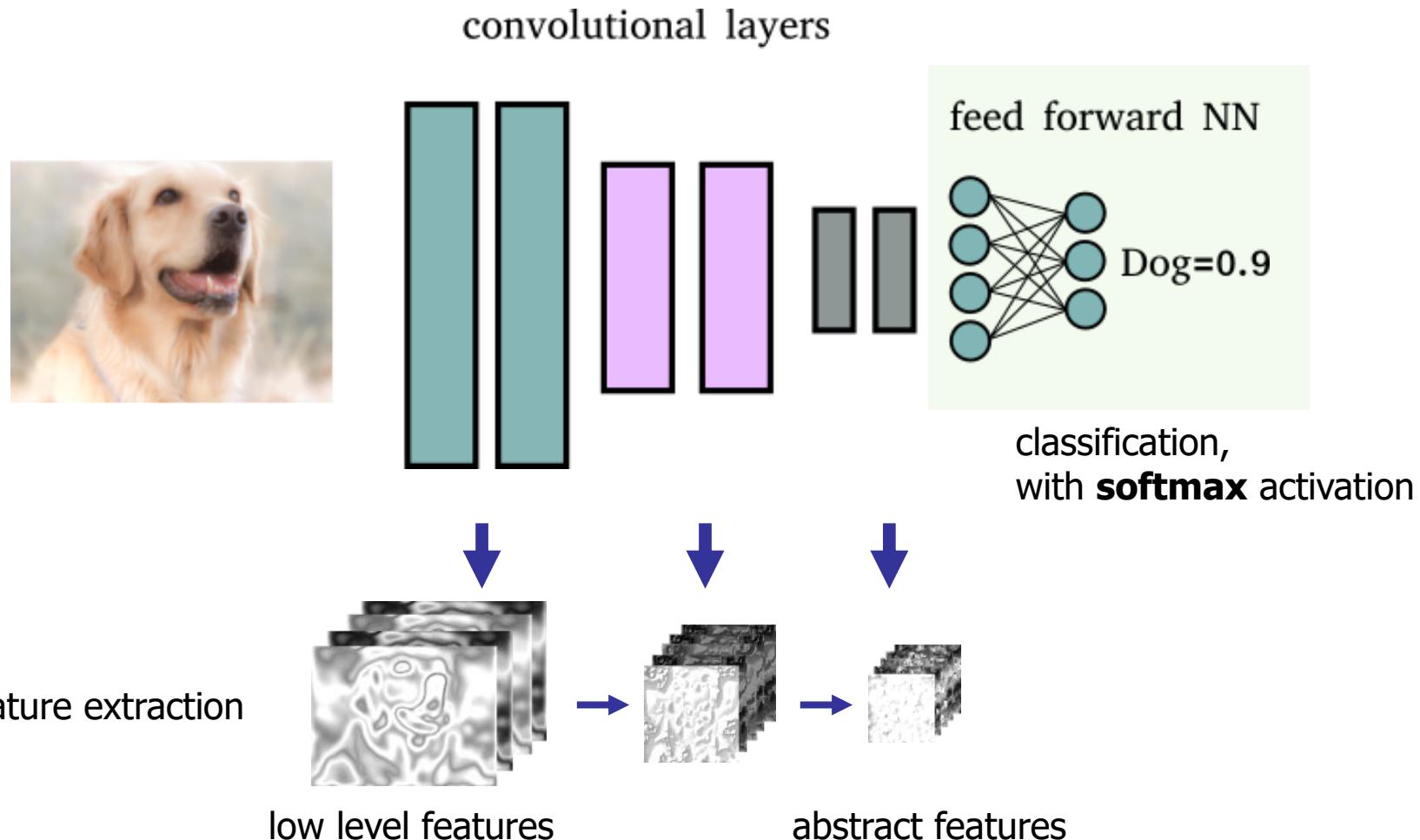


# Convolutional Neural Networks





# Convolutional Neural Networks

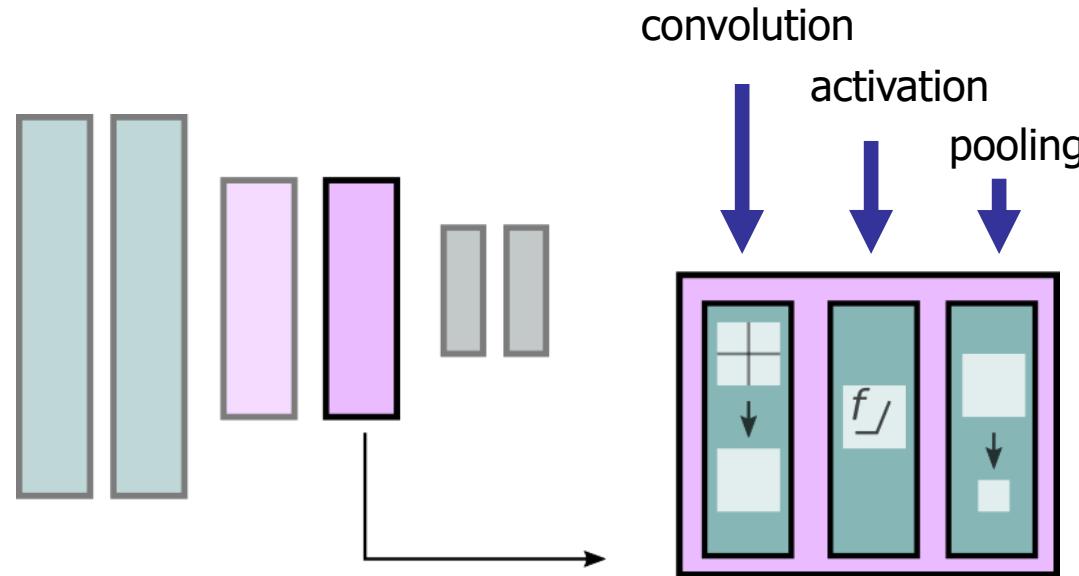




# Convolutional Neural Networks

## ■ Typical convolutional layer

- *convolution* stage: feature extraction by means of (hundreds to thousands) sliding filters
- sliding filters *activation*: apply activation functions to input tensor
- *pooling*: tensor downsampling



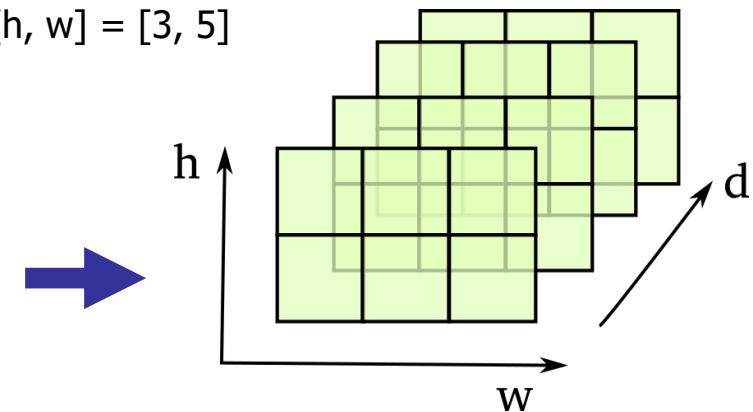


# Convolutional Neural Networks

## ■ Tensors

- data flowing through CNN layers is represented in the form of *tensors*
- *Tensor* = N-dimensional vector
- *Rank* = number of dimensions
  - scalar: rank 0
  - 1-D vector: rank 1
  - 2-D matrix: rank 2
- *Shape* = number of elements for each dimension
  - e.g. a vector of length 5 has shape [5]
  - e.g. a matrix  $w \times h$ ,  $w=5$ ,  $h=3$  has shape  $[h, w] = [3, 5]$

rank-3 tensor with shape  
 $[d, h, w] = [4, 2, 3]$

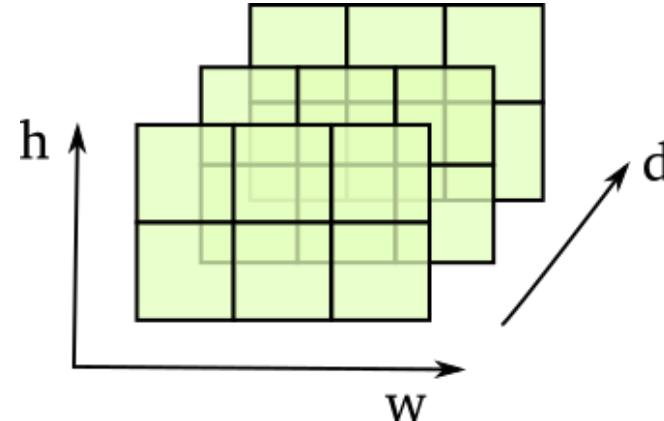




# Convolutional Neural Networks

## ■ Images

- rank-3 tensors with shape  $[d, h, w]$
- where  $h = \text{height}$ ,  $w = \text{width}$ ,  $d = \text{image depth}$  (1 for grayscale, 3 for RGB colors)

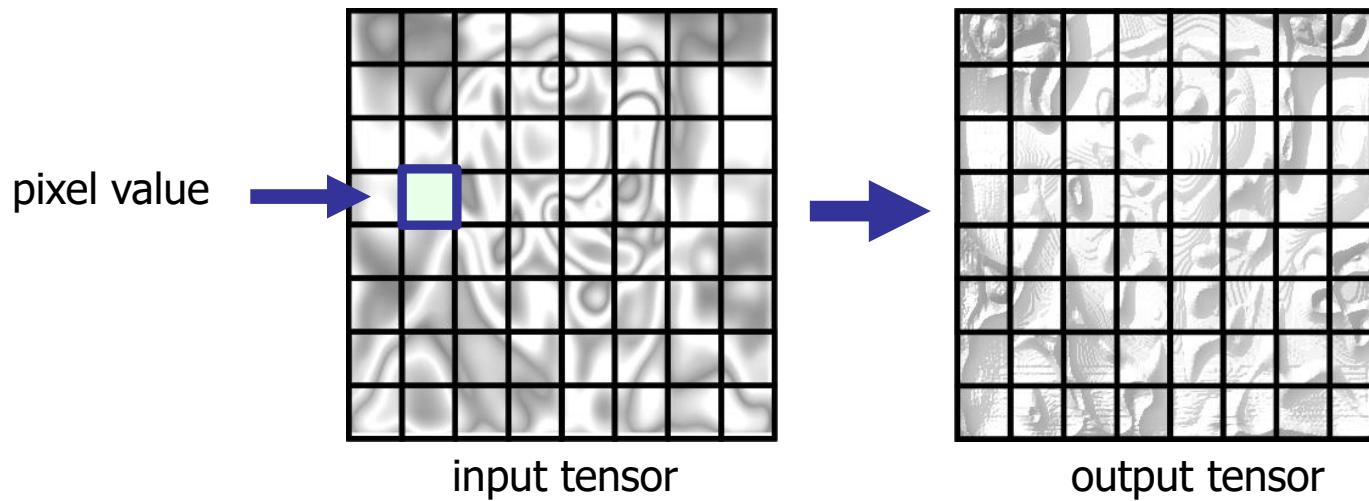




# Convolutional Neural Networks

## ■ Convolution

- processes data in form of *tensors* (multi-dimensional matrices)
- **input:** input image or intermediate features (tensor)
- **output:** a tensor with the extracted features

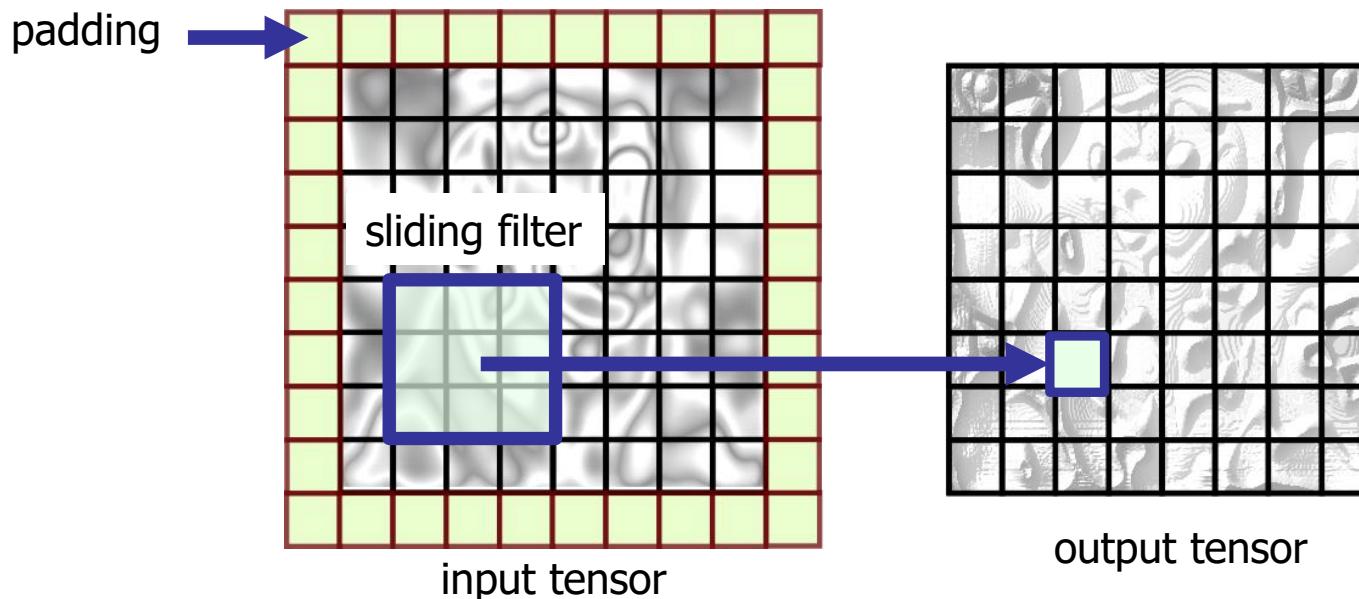




# Convolutional Neural Networks

## ■ Convolution

- a *sliding filter* produces the values of the output tensor
- sliding filters contain the trainable *weights* of the neural network
- each convolutional layer contains many (hundreds) filters

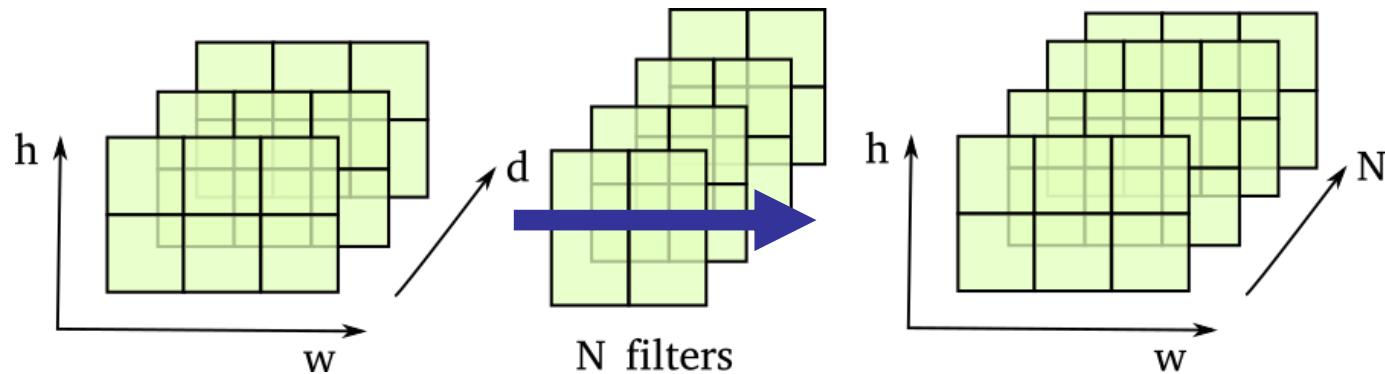




# Convolutional Neural Networks

## ■ Convolution

- images are transformed into features by convolutional filters
- after convolving a tensor  $[d, h, w]$  with *N filters* we obtain
  - a rank-3 tensor with shape  $[N, h, w]$
  - hence, each filter generates a layer in the depth of the output tensor

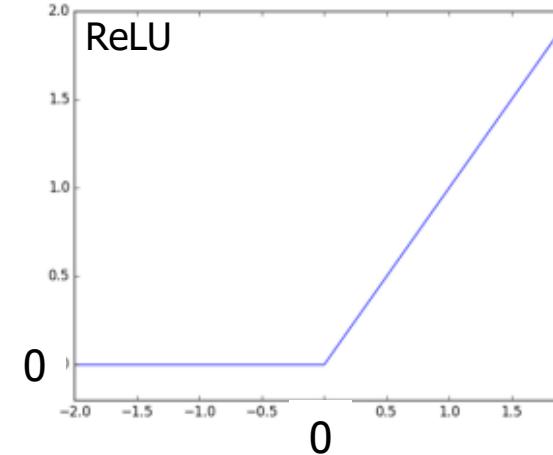
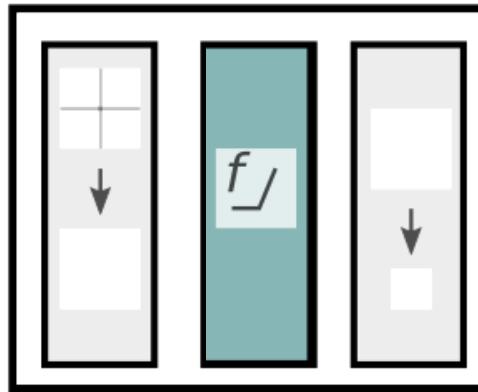




# Convolutional Neural Networks

## ■ Activation

- simulates biological activation to input stimulus
- provides non-linearity to the computation
- ReLU is typically used for CNNs
  - faster training (no vanishing gradients)
  - does not saturate
  - faster computation of derivatives for backpropagation

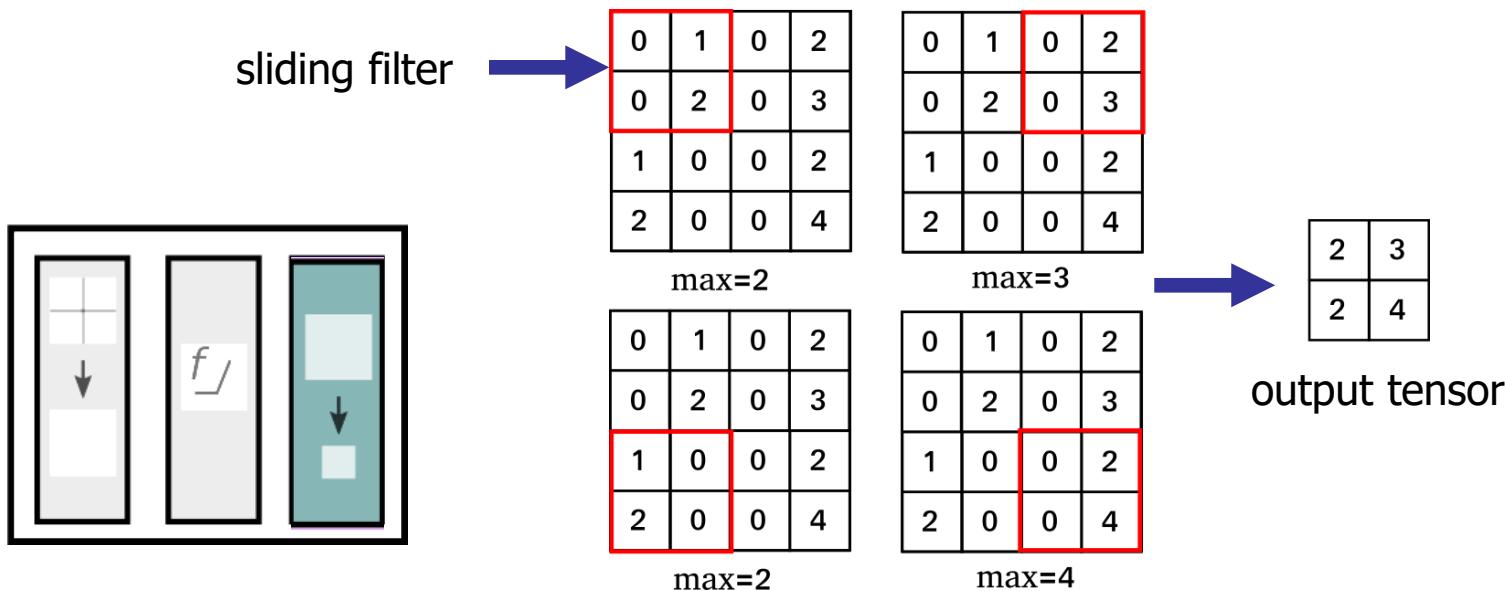




# Convolutional Neural Networks

## ■ Pooling

- performs tensor *downsampling*
- *sliding filter* which replaces tensor values with a *summary* statistic of the nearby outputs
- *maxpool* is the most common: computes the maximum value as statistic

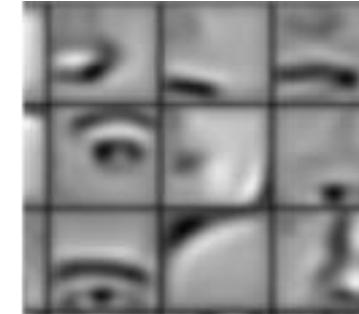
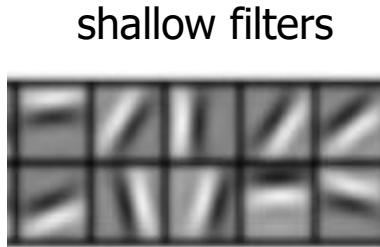




# Convolutional Neural Networks

## ■ Convolutional layers training

- during training each sliding filter learns to recognize a particular *pattern* in the input tensor
- filters in *shallow layers* recognize textures and edges
- filters in *deeper layers* can recognize objects and parts (e.g. eye, ear or even faces)



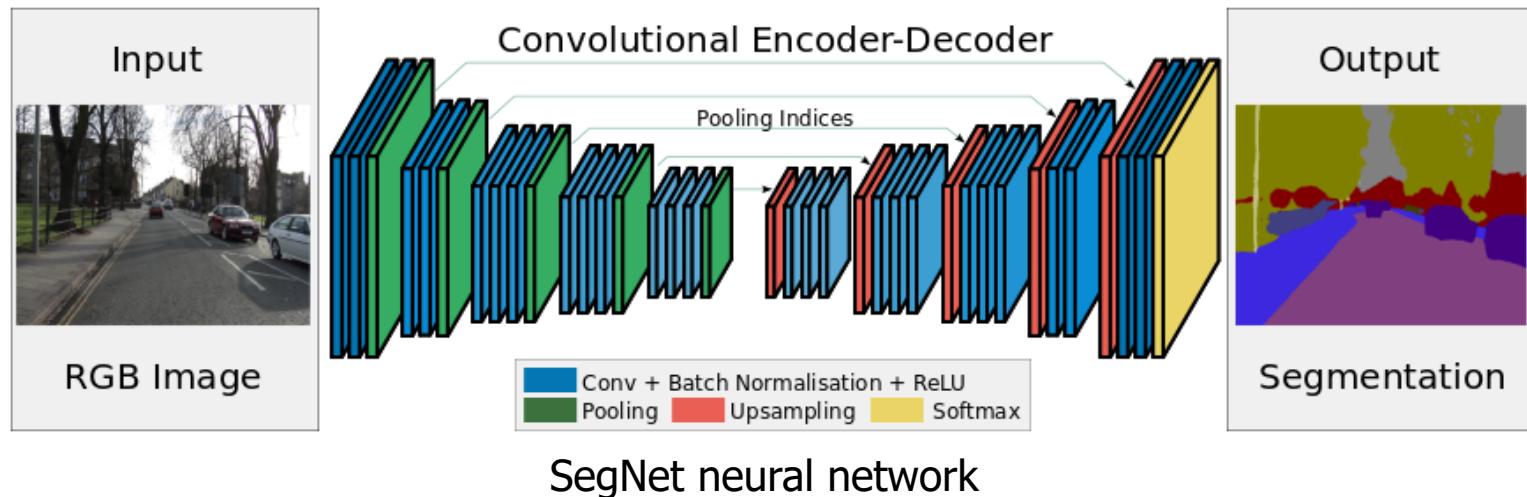
deeper filters



# Convolutional Neural Networks

## ■ Semantic segmentation CNNs

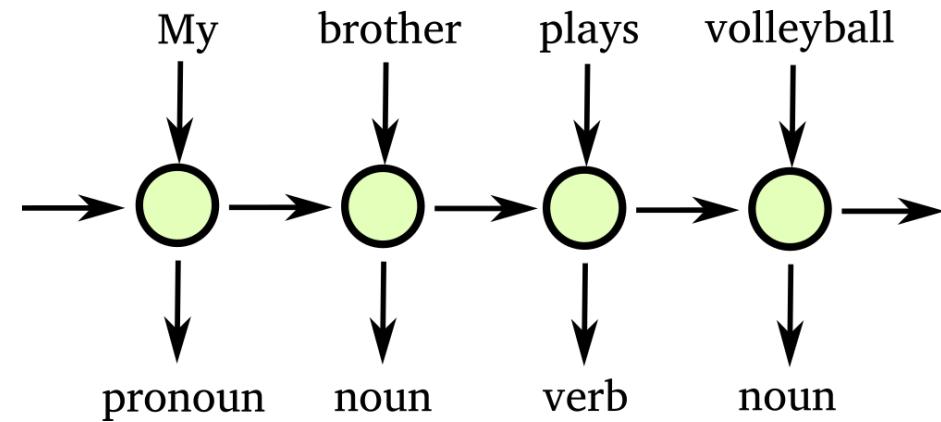
- allow assigning a class to each pixel of the input image
- composed of 2 parts
  - **encoder network**: convolutional layers to extract abstract features
  - **decoder network**: deconvolutional layers to obtain the output image from the extracted features





# Recurrent Neural Networks

- Allow processing *sequential* data  $x(t)$
- Differently from normal FFNN they are able to keep a *state* which evolves during time
- Applications
  - machine translation
  - time series prediction
  - speech recognition
  - part of speech (POS) tagging

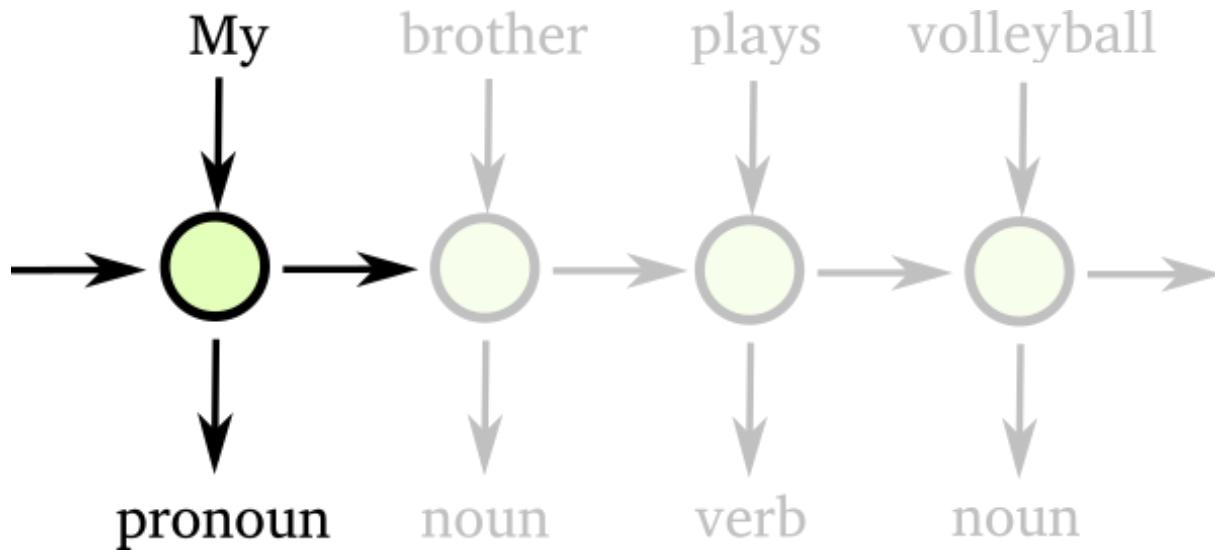




# Recurrent Neural Networks

- RNN execution during time

instance of the RNN at time t1

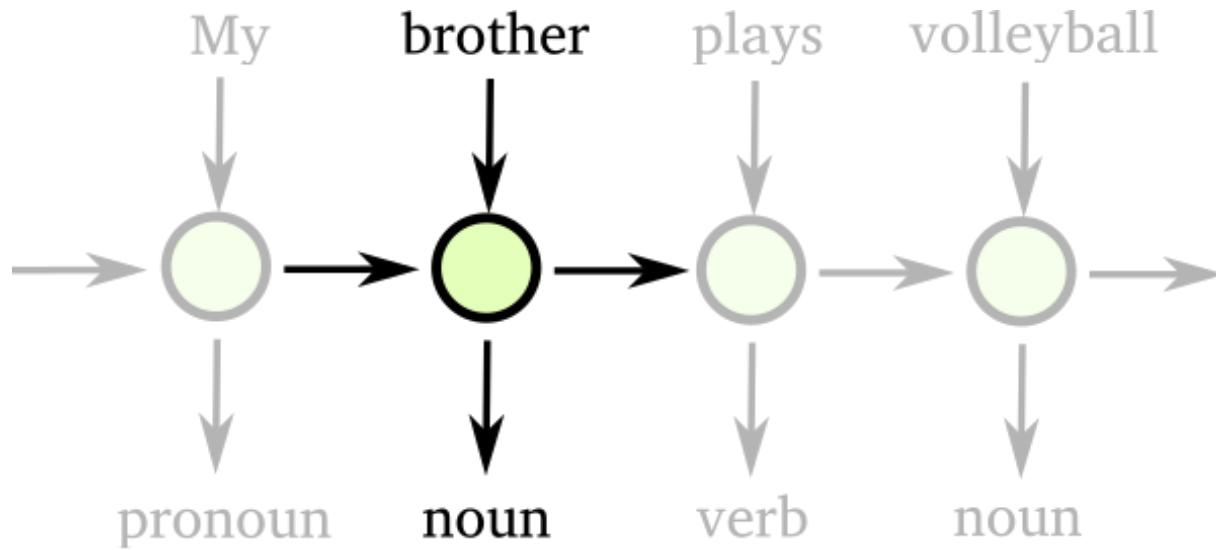




# Recurrent Neural Networks

- RNN execution during time

instance of the RNN at time t2

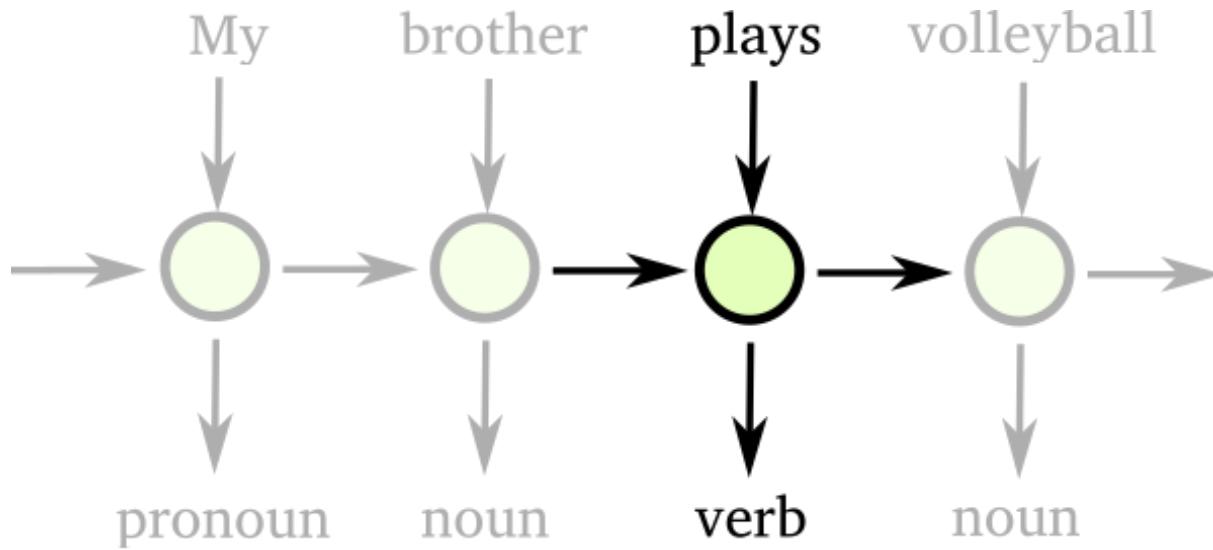




# Recurrent Neural Networks

- RNN execution during time

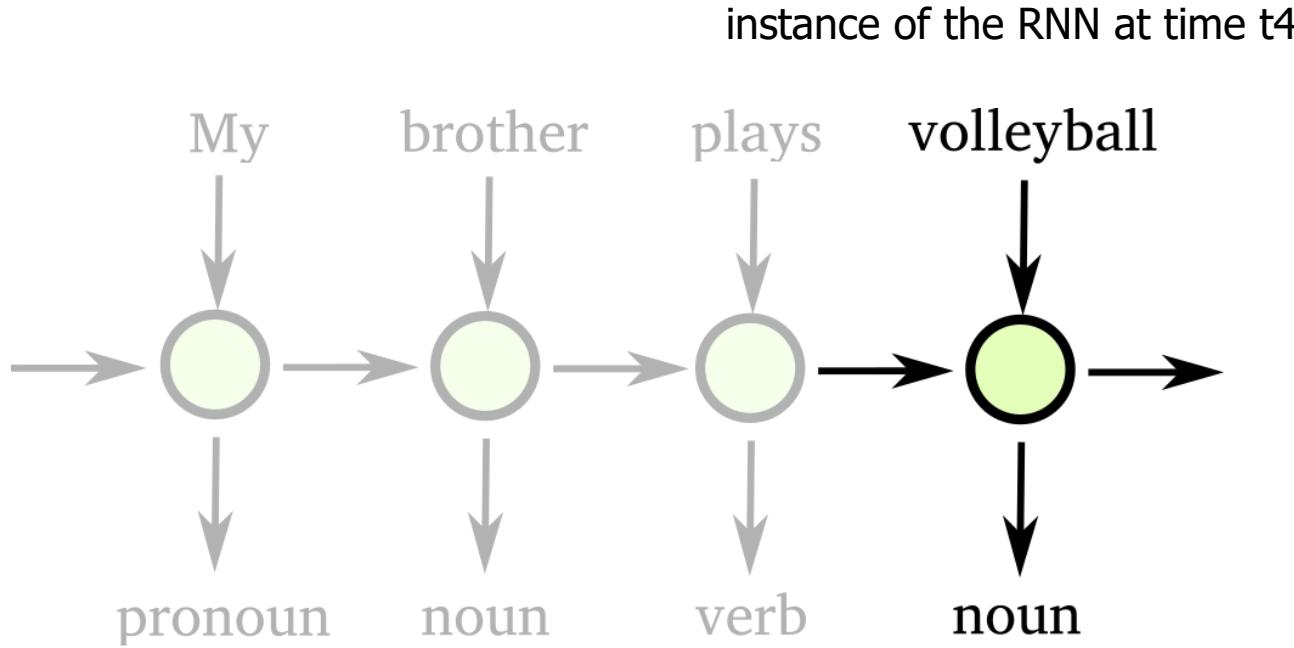
instance of the RNN at time t3





# Recurrent Neural Networks

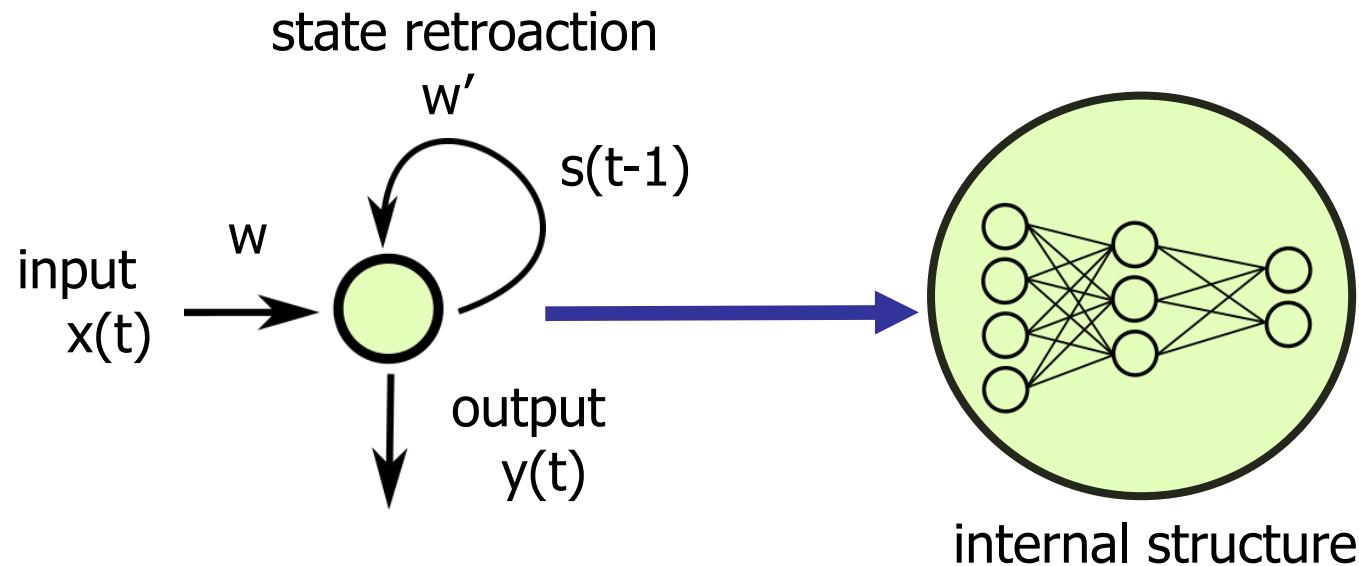
- RNN execution during time





# Recurrent Neural Networks

- A RNN receives as input a vector  $x(t)$  and the state at previous time step  $s(t-1)$
- A RNN typically contains many *neurons organized in different layers*

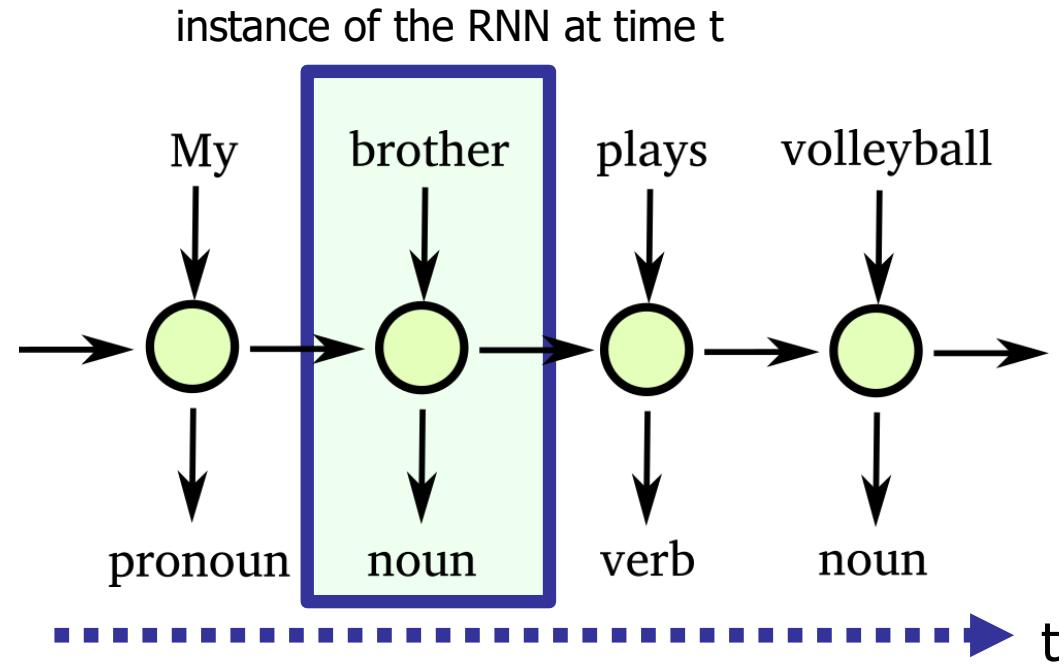




# Recurrent Neural Networks

- Training is performed with *Backpropagation Through Time*
- Given a pair training sequence  $x(t)$  and expected output  $y(t)$ 
  - error is propagated through time
  - weights are updated to minimize the error across all the time steps

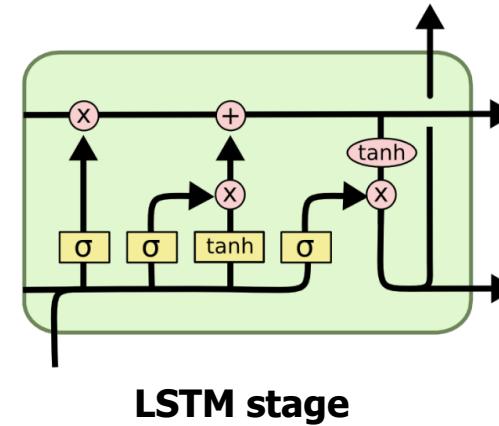
unrolled RNN diagram: shows the **same** neural network at different time steps





# Recurrent Neural Networks

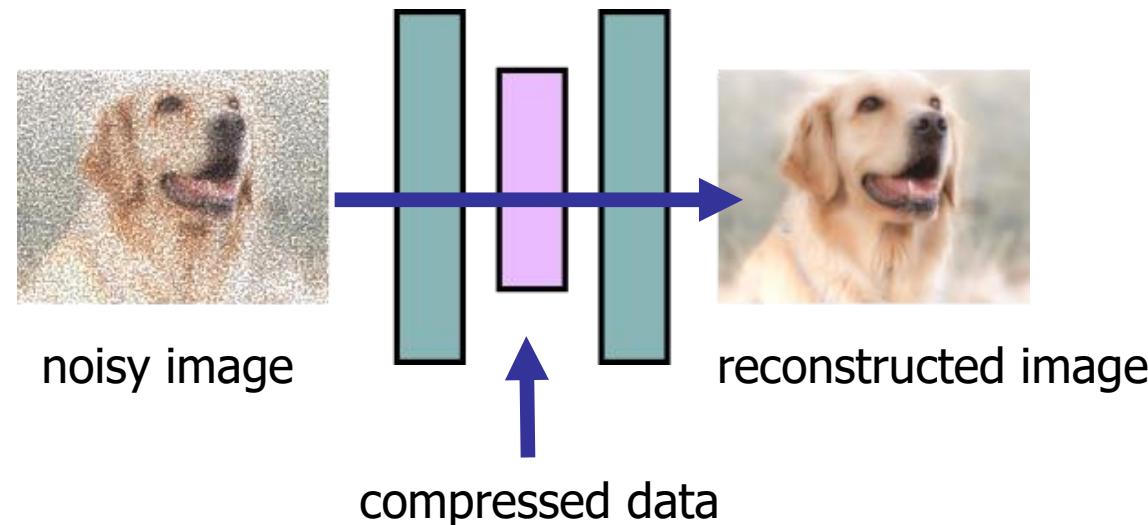
- Issues
  - *vanishing gradient*: error gradient decreases rapidly over time, weights are not properly updated
  - this makes harder having RNN with *long-term* memories
- Solution: *LSTM* (Long Short Term Memories)
  - RNN with “gates” which encourage the state information to flow through long time intervals





# Autoencoders

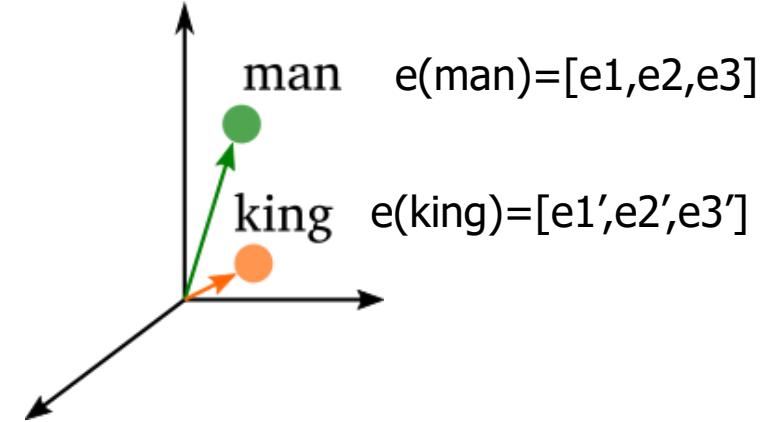
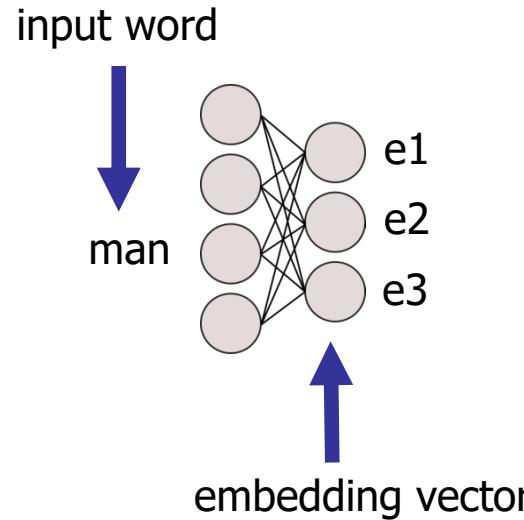
- Autoencoders allow *compressing* input data by means of compact representations and from them *reconstruct* the initial input
  - for feature extraction: the compressed representation can be used as significant set of features representing input data
  - for image (or signal) *denoising*: the image reconstructed from the abstract representation is denoised with respect to the original one





# Word Embeddings (Word2Vec)

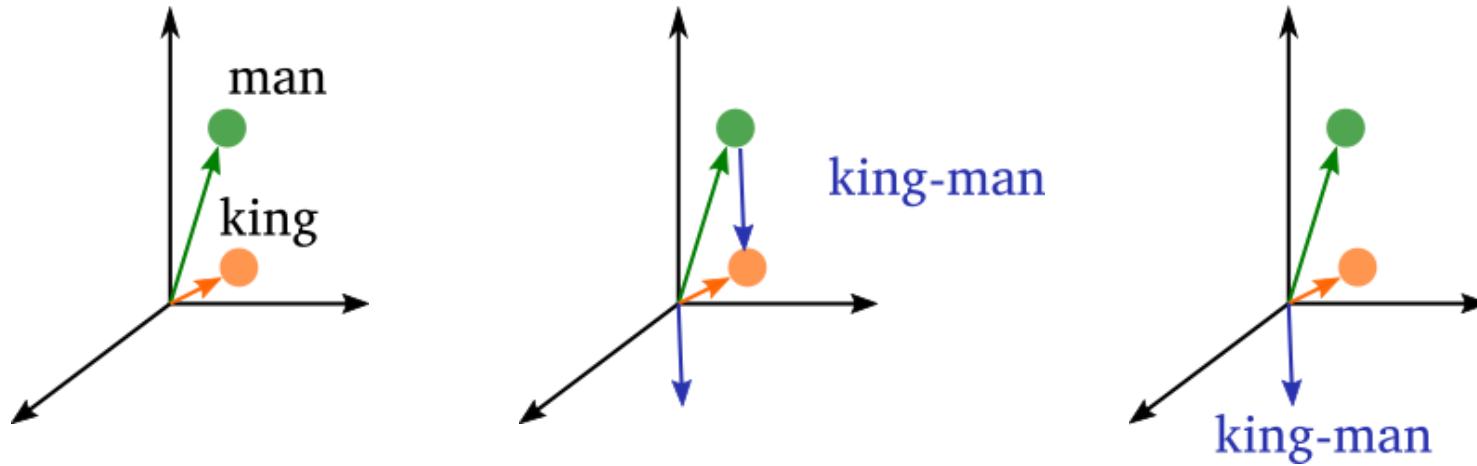
- Word *embeddings* associate words to n-dimensional vectors
  - trained on big text collections to model the word distributions in different sentences and contexts
  - able to capture the *semantic* information of each word
  - words with similar *meaning* share vectors with similar characteristics





# Word Embeddings (Word2Vec)

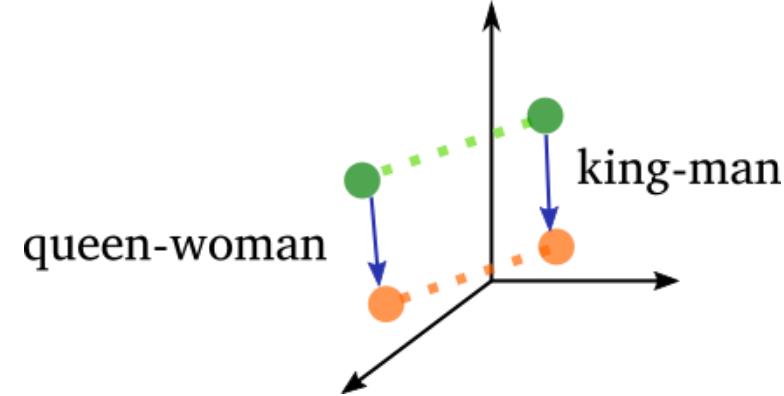
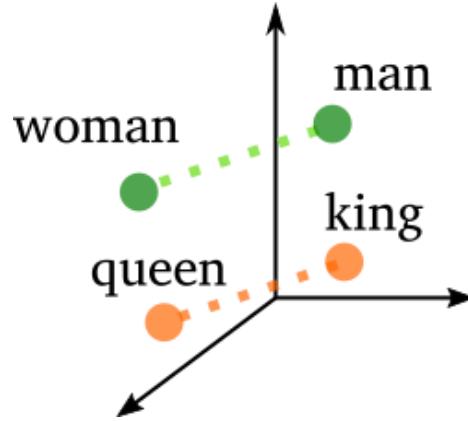
- Since each word is represented with a vector, operations among words (e.g. difference, addition) are allowed





# Word Embeddings (Word2Vec)

- Semantic relationships among words are captured by vector positions



$$\begin{aligned}\text{king} - \text{man} &= \text{queen} - \text{woman} \\ \text{king} - \text{man} + \text{woman} &= \text{queen}\end{aligned}$$

# Model evaluation



Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis  
*Politecnico di Torino*



# Model evaluation

- Methods for performance evaluation
  - Partitioning techniques for training and test sets
- Metrics for performance evaluation
  - Accuracy, other measures
- Techniques for model comparison
  - ROC curve

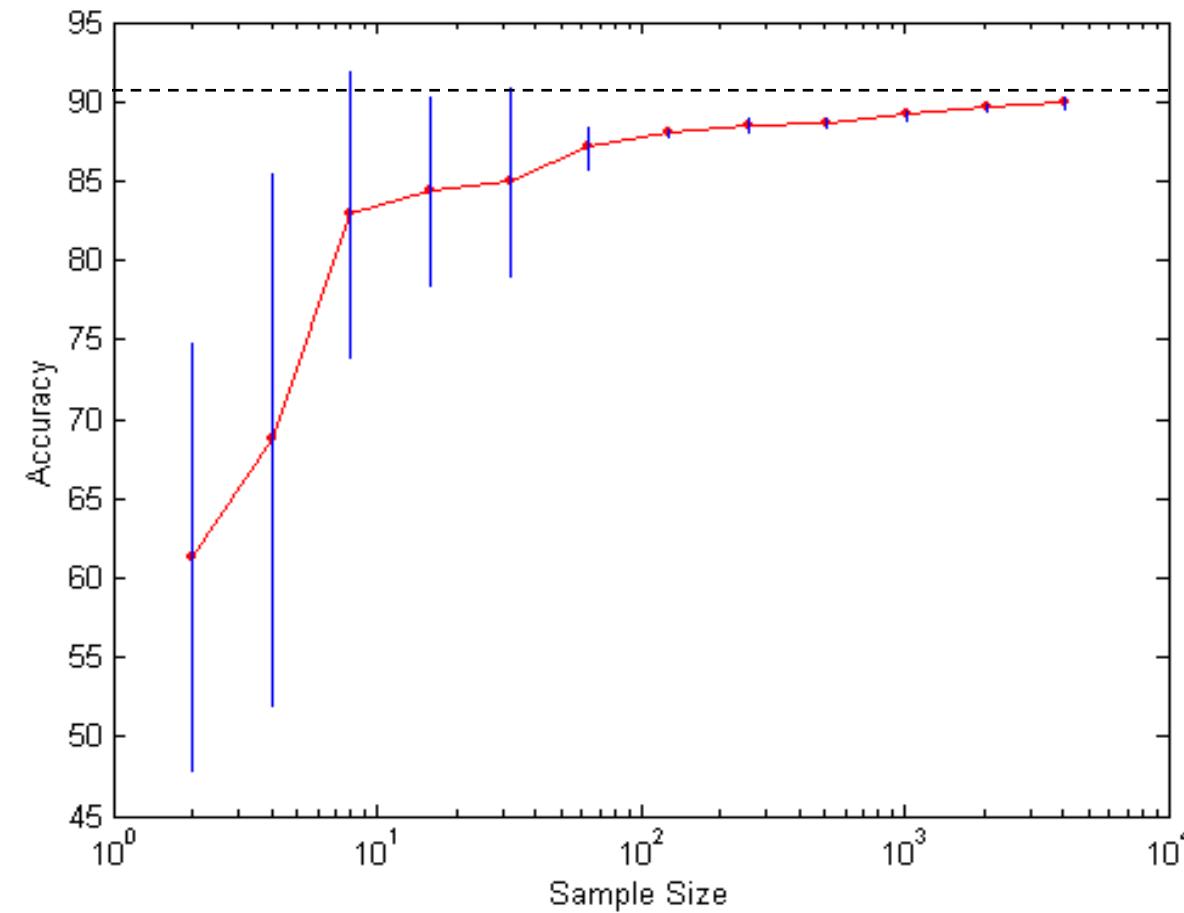


# Methods for performance evaluation

- Objective
  - reliable estimate of performance
- Performance of a model may depend on other factors besides the learning algorithm
  - Class distribution
  - Cost of misclassification
  - Size of training and test sets



# Learning curve



- Learning curve shows how accuracy changes with varying training sample size
- Requires a sampling schedule for creating learning curve:
  - Arithmetic sampling (Langley, et al)
  - Geometric sampling (Provost et al)
- Effect of small sample size:
  - Bias in the estimate
  - Variance of estimate



# Partitioning data

- Several partitioning techniques
  - holdout
  - cross validation
- Stratified sampling to generate partitions
  - without replacement
- Bootstrap
  - Sampling with replacement



# Methods of estimation

- Partitioning labeled data for training, validation and test
- Several partitioning techniques
  - holdout
  - cross validation
- Stratified sampling to generate partitions
  - without replacement
- Bootstrap
  - Sampling with replacement



# Holdout

- Fixed partitioning
  - Typically, may reserve 80% for training, 20% for test
  - Other proportions may be appropriate, depending on the dataset size
- Appropriate for large datasets
  - may be repeated several times
    - repeated holdout



# Cross validation

## ■ Cross validation

- partition data into  $k$  disjoint subsets (i.e., folds)
- $k$ -fold: train on  $k-1$  partitions, test on the remaining one
  - repeat for all folds
- reliable accuracy estimation, not appropriate for very large datasets

## ■ Leave-one-out

- cross validation for  $k=n$
- only appropriate for very small datasets



# Model performance estimation

- Model training step
  - Building a new model
- Model validation step
  - Hyperparameter tuning
  - Algorithm selection
- Model test step
  - Estimation of model performance



# Model performance estimation

- Typical dataset size
  - Training set 60% of labeled data
  - Validation set 20% of labeled data
  - Test set 20% of labeled data
- Splitting labeled data
  - Use hold-out to split in
    - training+validation
    - test
  - Use cross validation to split in
    - training
    - validation



# Metrics for model evaluation

- Evaluate the predictive accuracy of a model
- Confusion matrix
  - binary classifier

		PREDICTED CLASS	
		Class=Yes	Class>No
ACTUAL CLASS	Class=Yes	a	b
	Class>No	c	d

- a: TP (true positive)
- b: FN (false negative)
- c: FP (false positive)
- d: TN (true negative)



# Accuracy

- Most widely-used metric for model evaluation

$$\text{Accuracy} = \frac{\text{Number of correctly classified objects}}{\text{Number of classified objects}}$$

- Not always a reliable metric



# Accuracy

- For a binary classifier

		PREDICTED CLASS	
		Class=Yes	Class>No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class>No	c (FP)	d (TN)

$$\text{Accuracy} = \frac{a + d}{a + b + c + d} = \frac{TP + TN}{TP + TN + FP + FN}$$



# Limitations of accuracy

- Consider a binary problem

- Cardinality of Class 0 = 9900
  - Cardinality of Class 1 = 100

- Model

$() \rightarrow \text{class } 0$

- Model predicts everything to be class 0
    - accuracy is  $9900/10000 = 99.0\%$
- Accuracy is misleading because the model does not detect any class 1 object



# Limitations of accuracy

- Classes may have different importance
  - Misclassification of objects of a given class is more important
  - e.g., ill patients erroneously assigned to the healthy patients class
- Accuracy is not appropriate for
  - unbalanced class label distribution
  - different class relevance



# Class specific measures

- Evaluate separately for each class C

$$\text{Recall (r)} = \frac{\text{Number of objects correctly assigned to C}}{\text{Number of objects belonging to C}}$$

$$\text{Precision (p)} = \frac{\text{Number of objects correctly assigned to C}}{\text{Number of objects assigned to C}}$$

- Maximize

$$\text{F - measure (F)} = \frac{2rp}{r + p}$$



# Class specific measures

- For a binary classification problem
  - on the confusion matrix, for the positive class

$$\text{Precision (p)} = \frac{a}{a + c}$$

$$\text{Recall (r)} = \frac{a}{a + b}$$

$$\text{F - measure (F)} = \frac{2rp}{r + p} = \frac{2a}{2a + b + c}$$



# ROC (Receiver Operating Characteristic)

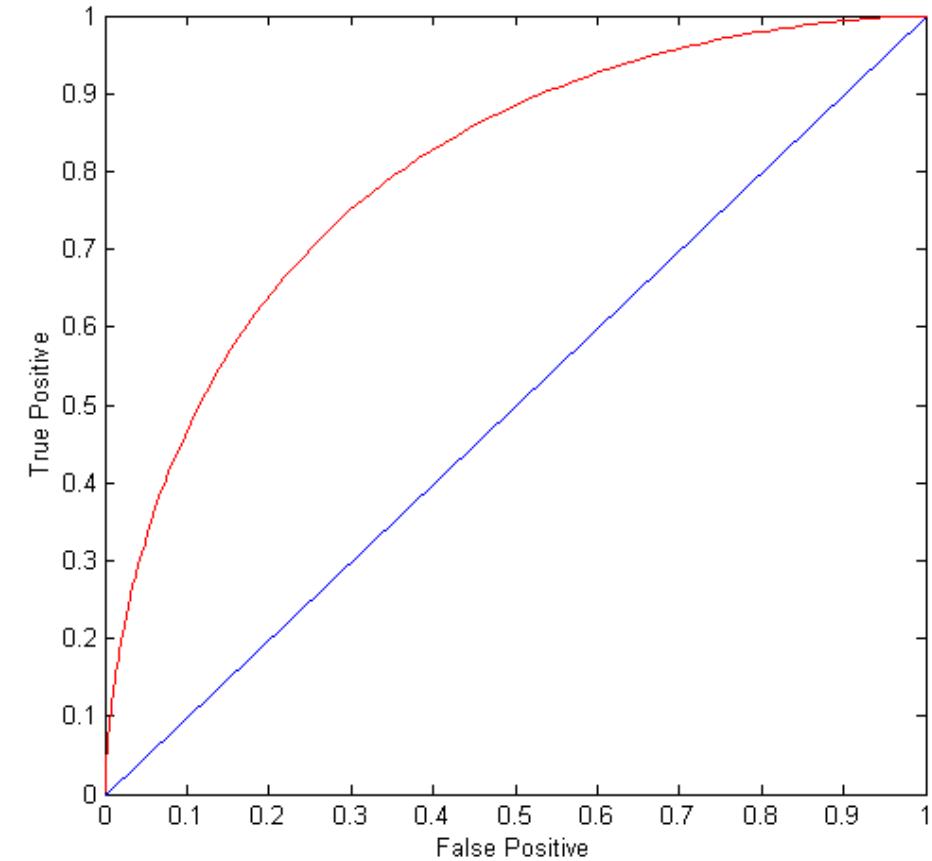
- Developed in 1950s for signal detection theory to analyze noisy signals
  - characterizes the trade-off between positive hits and false alarms
- ROC curve plots
  - TPR, True Positive Rate (on the y-axis)  
$$TPR = TP / (TP + FN)$$
against
  - FPR, False Positive Rate (on the x-axis)  
$$FPR = FP / (FP + TN)$$



# ROC curve

(FPR, TPR)

- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (0,1): ideal
  
- Diagonal line
  - Random guessing
  - Below diagonal line
    - prediction is opposite of the true class





# How to build a ROC curve

Instance	P(+ A)	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use classifier that produces posterior probability for each test instance  $P(+|A)$
- Sort the instances according to  $P(+|A)$  in decreasing order
- Apply threshold at each unique value of  $P(+|A)$
- Count the number of TP, FP, TN, FN at each threshold
  - TP rate  
$$TPR = TP / (TP + FN)$$
  - FP rate  
$$FPR = FP / (FP + TN)$$



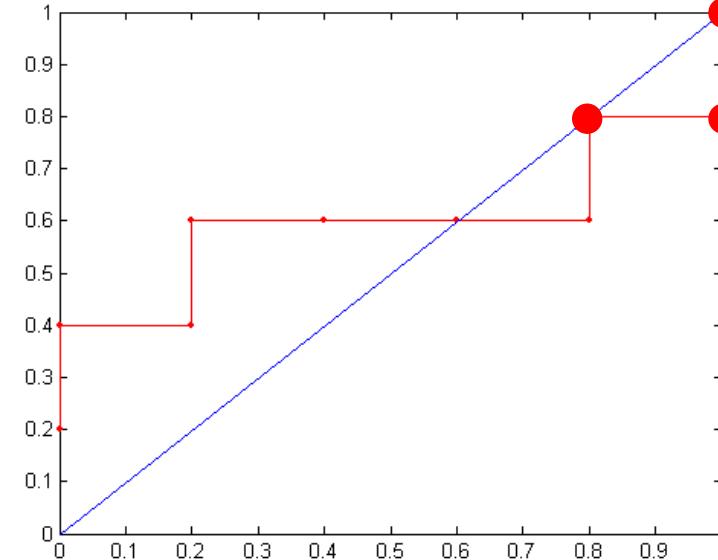
# How to build a ROC curve

Class	+	-	+	-	-	-	+	-	+	+	
$P(+ A)$	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
<b>TP</b>	5	4	4	3	3	3	3	2	2	1	0
<b>FP</b>	5	5	4	4	3	2	1	1	0	0	0
<b>TN</b>	0	0	1	1	2	3	4	4	5	5	5
<b>FN</b>	0	1	1	2	2	2	2	3	3	4	5
→ <b>TPR</b>	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
→ <b>FPR</b>	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

→

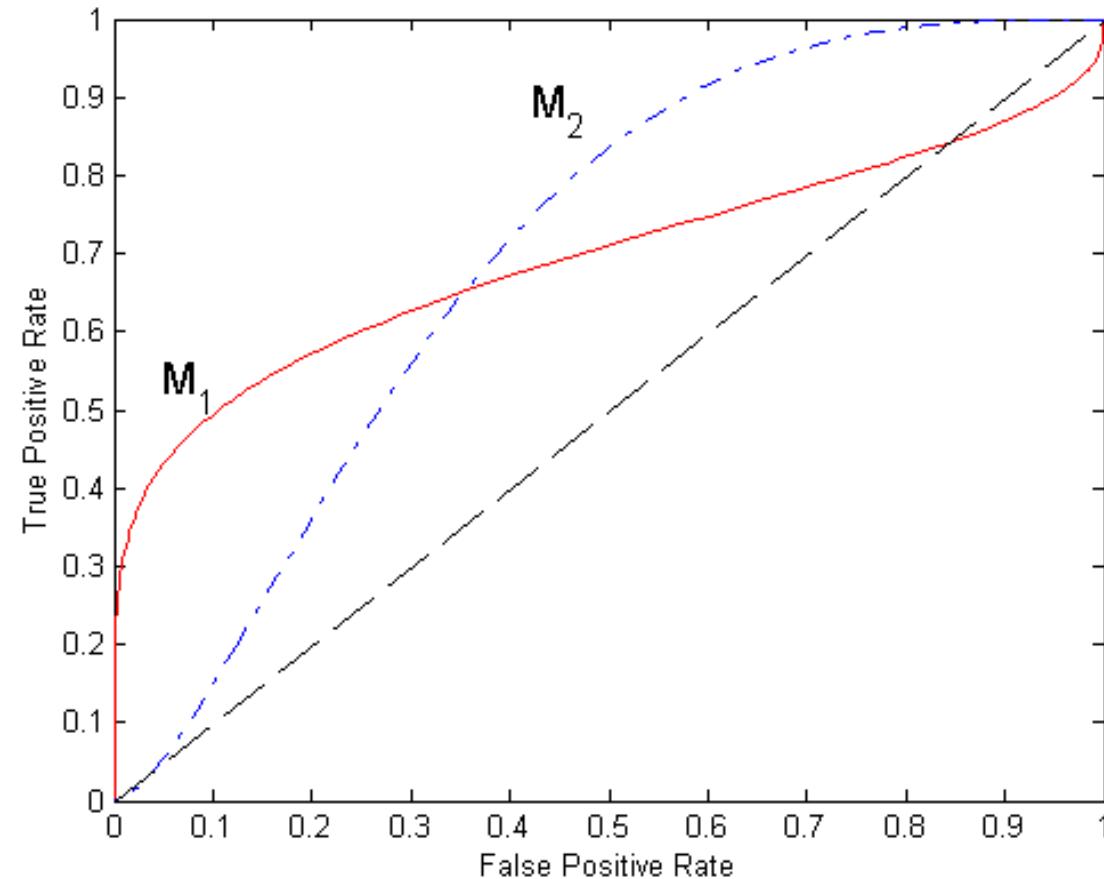
→

ROC Curve





# Using ROC for Model Comparison



- No model consistently outperforms the other
  - $M_1$  is better for small FPR
  - $M_2$  is better for large FPR
- Area under ROC curve
  - Ideal Area = 1.0
  - Random guess Area = 0.5

# Clustering fundamentals



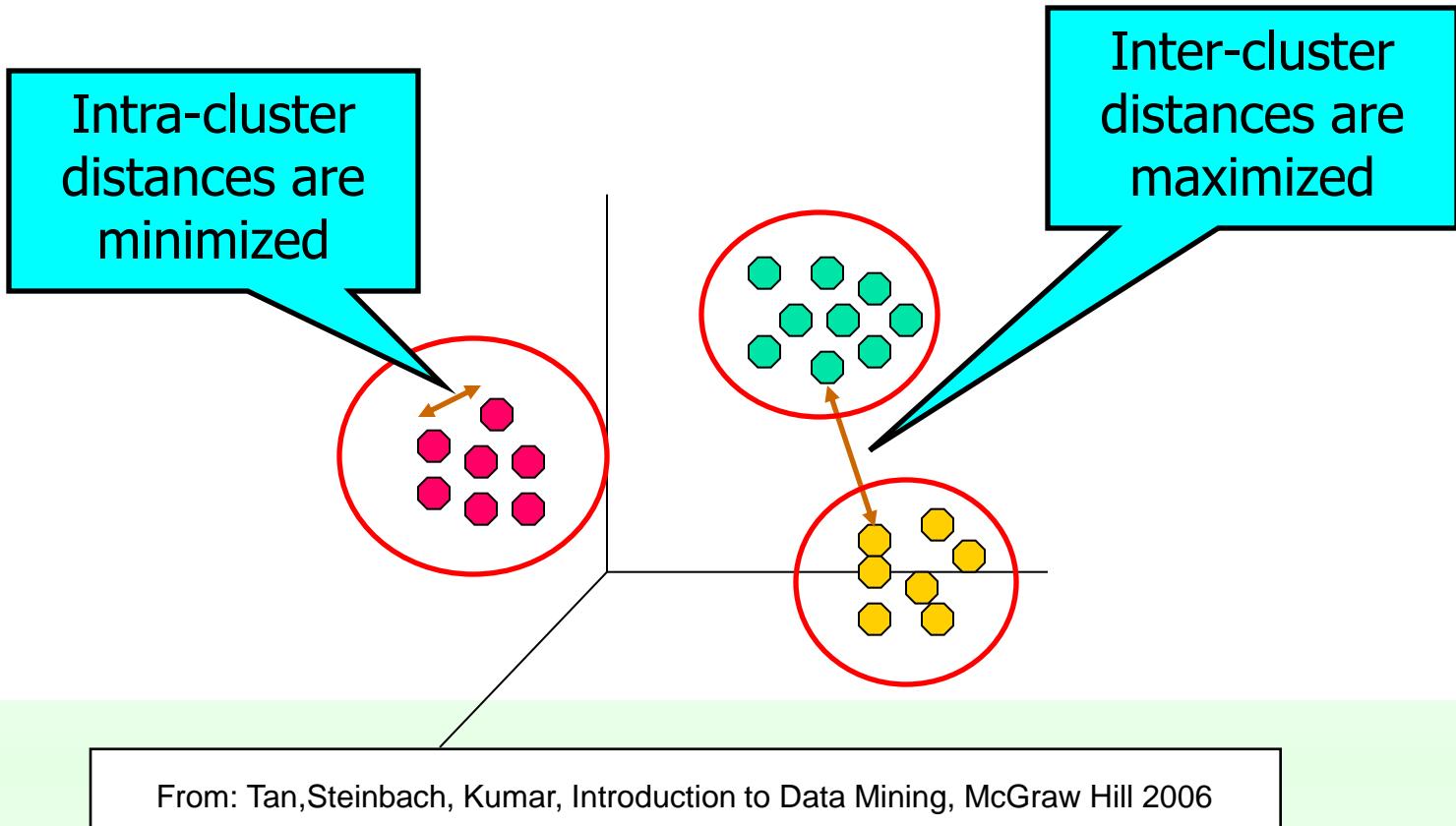
Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis, Tania Cerquitelli  
*Politecnico di Torino*



# What is 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





# Applications of Cluster Analysis

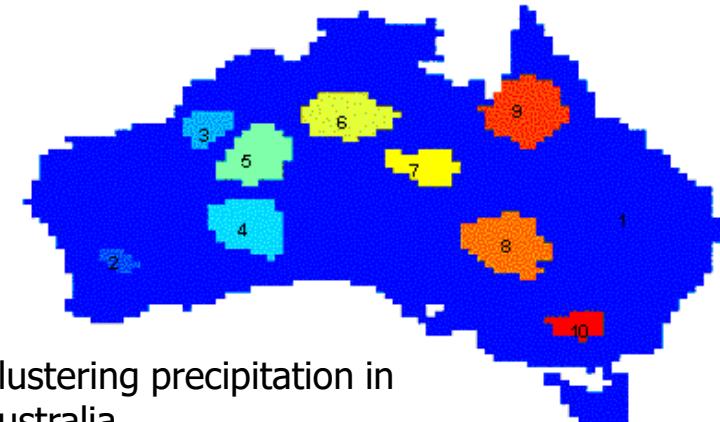
## ■ Understanding

- Group related documents for browsing, group genes and proteins that have similar functionality, or group stocks with similar price fluctuations

	<i>Discovered Clusters</i>	<i>Industry Group</i>
1	Applied-Matl-DOWN,Bay-Network-Down,3-COM-DOWN, Cabletron-Sys-DOWN,CISCO-DOWN,HP-DOWN, DSC-Comm-DOWN,INTEL-DOWN,LSI-Logic-DOWN, Micron-Tech-DOWN,Texas-Inst-Down,Tellabs-Inc-Down, Natl-Semiconduct-DOWN,Oracl-DOWN,SGI-DOWN, Sun-DOWN	Technology1-DOWN
2	Apple-Comp-DOWN,Autodesk-DOWN,DEC-DOWN, ADV-Micro-Device-DOWN,Andrew-Corp-DOWN, Computer-Assoc-DOWN,Circuit-City-DOWN, Compaq-DOWN, EMC-Corp-DOWN, Gen-Inst-DOWN, Motorola-DOWN,Microsoft-DOWN,Scientific-Atl-DOWN	Technology2-DOWN
3	Fannie-Mae-DOWN,Fed-Home-Loan-DOWN, MBNA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
4	Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP, Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP, Schlumberger-UP	Oil-UP

## ■ Summarization

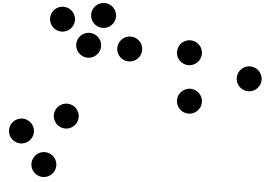
- Reduce the size of large data sets



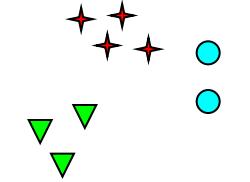
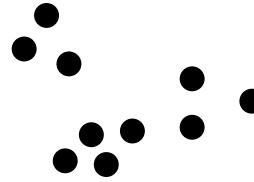
Clustering precipitation in Australia



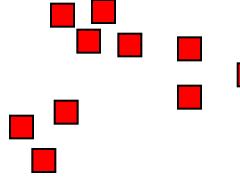
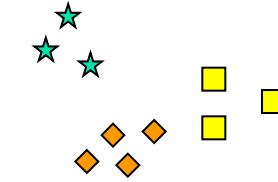
# Notion of a Cluster can be Ambiguous



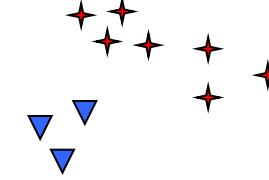
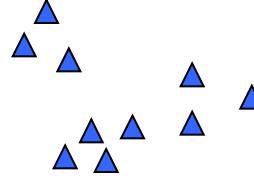
How many clusters?



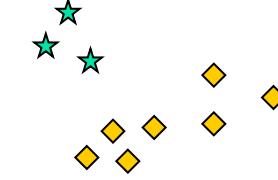
Six Clusters



Two Clusters



Four Clusters



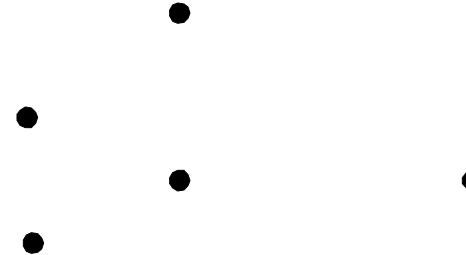
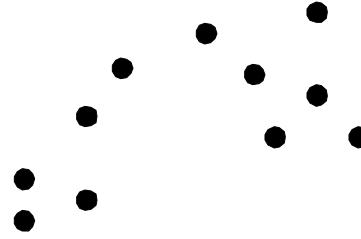


# Types of Clusterings

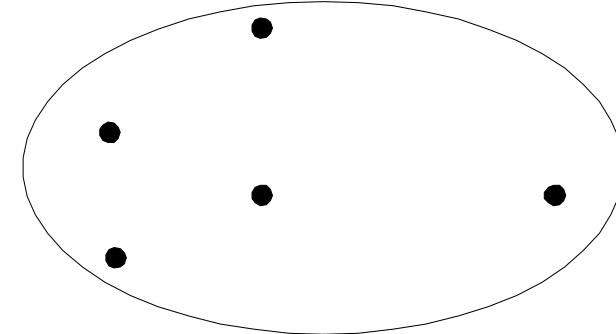
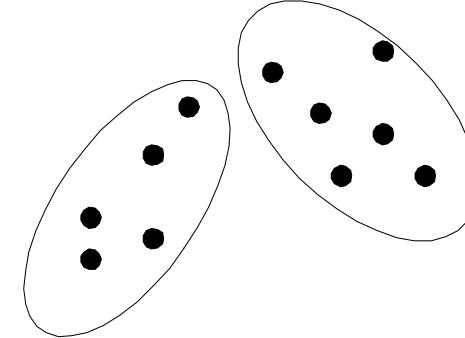
- A **clustering** is a set of clusters
- Important distinction between **hierarchical** and **partitional** sets of clusters
- **Partitional Clustering**
  - Divides data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
- **Hierarchical clustering**
  - A set of nested clusters organized as a hierarchical tree



# Partitional Clustering



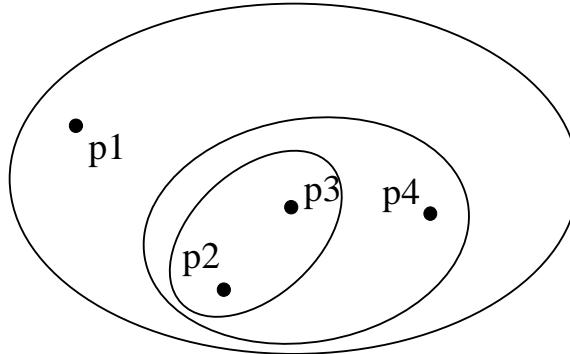
Original Points



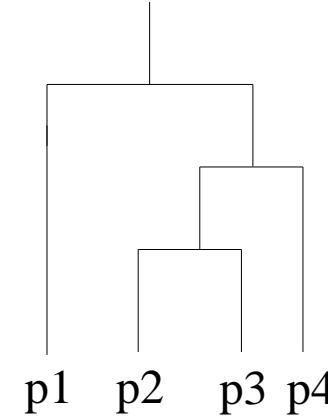
A Partitional Clustering



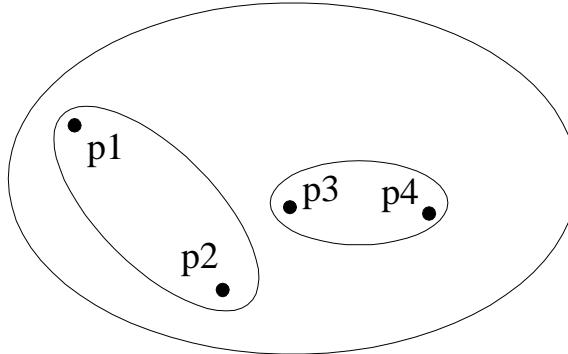
# Hierarchical Clustering



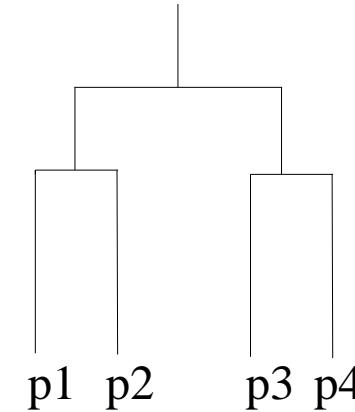
Traditional Hierarchical Clustering



Traditional Dendrogram



Non-traditional Hierarchical Clustering



Non-traditional Dendrogram



# Other Distinctions Between Sets of Clusters

- Exclusive versus non-exclusive
  - In non-exclusive clustering, points may belong to multiple clusters.
- Fuzzy versus non-fuzzy
  - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
  - Weights must sum to 1
  - Probabilistic clustering has similar characteristics
- Partial versus complete
  - In some cases, we only want to cluster some of the data
- Heterogeneous versus homogeneous
  - Cluster of widely different sizes, shapes, and densities



# Types of Clusters

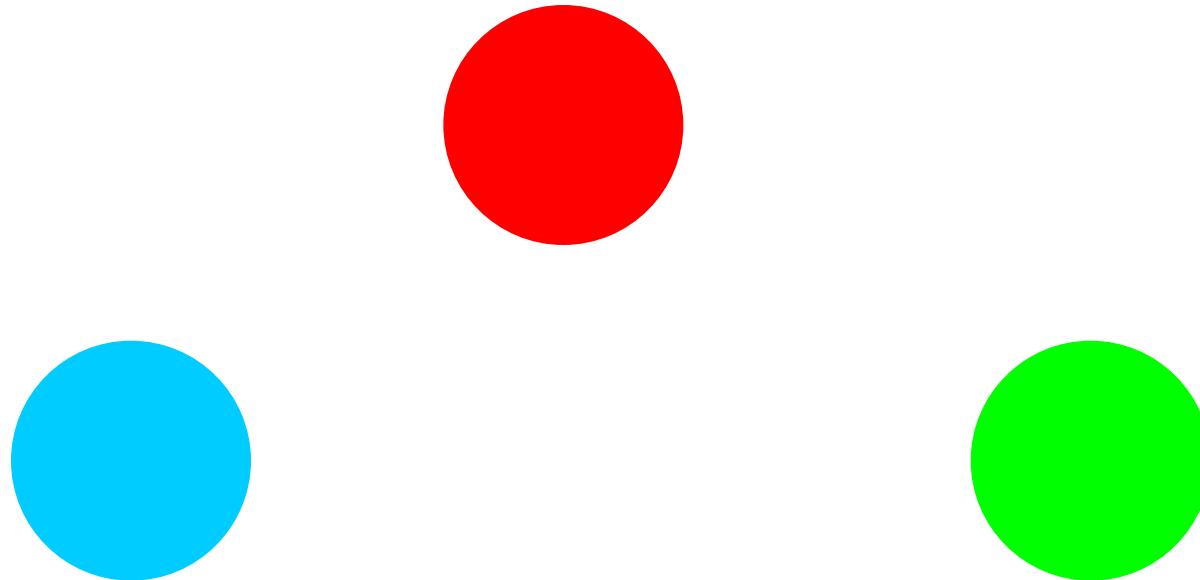
- Well-separated clusters
- Center-based clusters
- Contiguous clusters
- Density-based clusters
- Property or Conceptual
- Described by an Objective Function



# Types of Clusters: Well Separated

## ■ Well-Separated Clusters:

- A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.



3 well-separated clusters



# Types of Clusters: Center-Based

## ■ Center-based

- A cluster is a set of objects such that an object in a cluster is closer (more similar) to the “center” of a cluster, than to the center of any other cluster
- The center of a cluster is often a **centroid**, the average of all the points in the cluster, or a **medoid**, the most “representative” point of a cluster

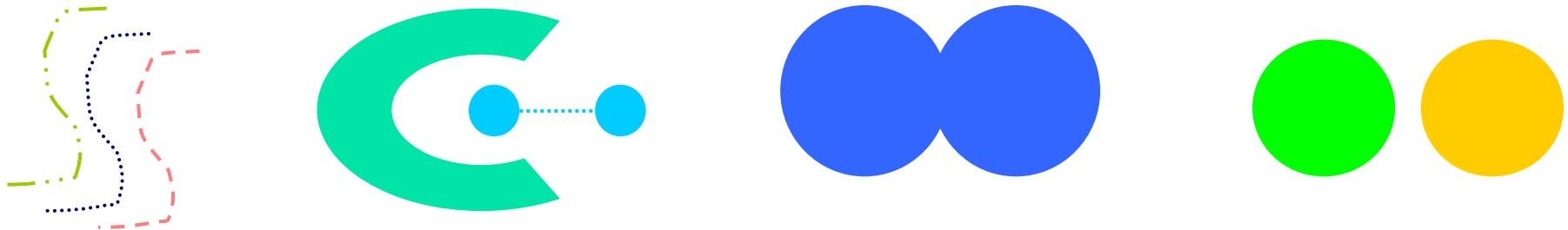


4 center-based clusters



# Types of Clusters: Contiguity-Based

- Contiguous Cluster (Nearest neighbor or Transitive)
  - A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.



8 contiguous clusters

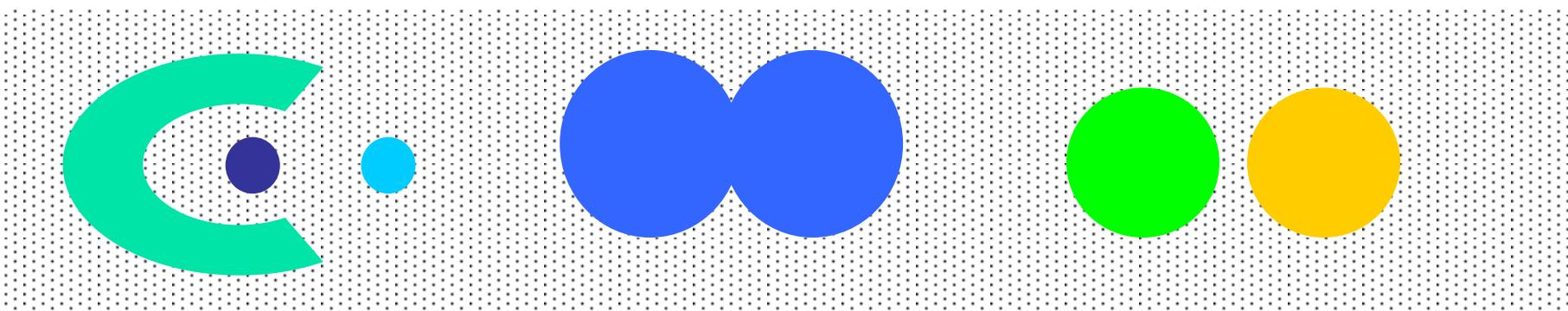
From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



# Types of Clusters: Density-Based

## ■ Density-based

- A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.
- Used when the clusters are irregular or intertwined, and when noise and outliers are present.

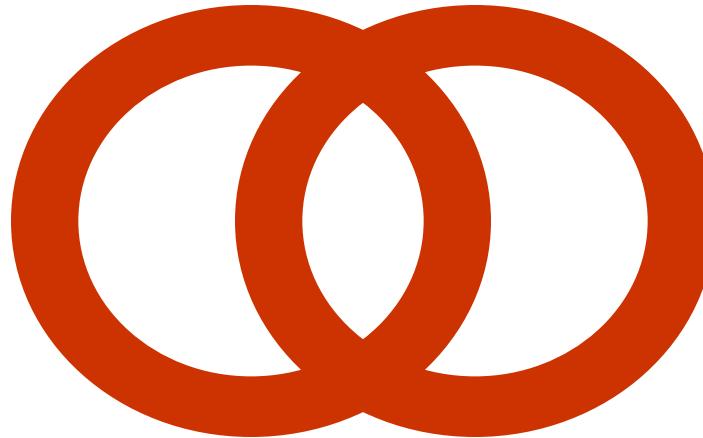


6 density-based clusters



# Types of Clusters: Conceptual Clusters

- Shared Property or Conceptual Clusters
  - Finds clusters that share some common property or represent a particular concept.
  -



2 Overlapping Circles

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



# Clustering Algorithms

- K-means and its variants
- Hierarchical clustering
- Density-based clustering



# K-means Clustering

- Partitional clustering approach
- Each cluster is associated with a **centroid** (center 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

- 
- 1: Select  $K$  points as the initial centroids.
  - 2: **repeat**
  - 3:   Form  $K$  clusters by assigning all points to the closest centroid.
  - 4:   Recompute the centroid of each cluster.
  - 5: **until** The centroids don't change
-

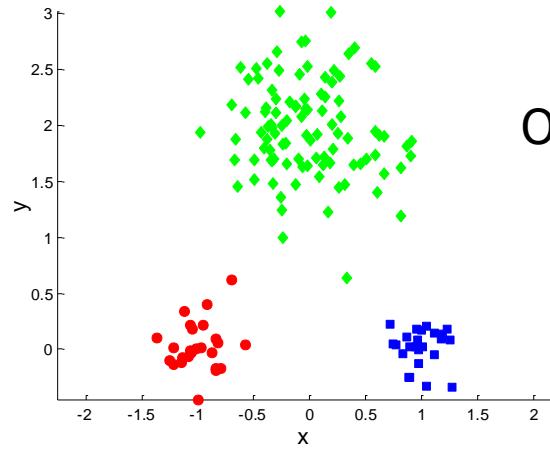


# K-means Clustering – Details

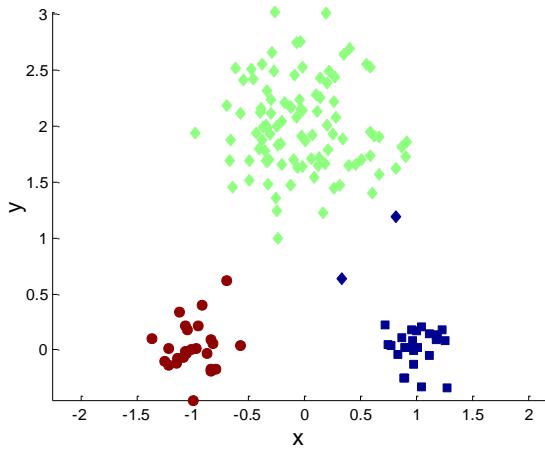
- Initial centroids are often chosen randomly.
  - Clusters produced vary from one run to another.
- The centroid is (typically) the mean of the points in the cluster.
- ‘Closeness’ is measured by Euclidean distance, cosine similarity, correlation, etc.
- K-means will converge for common similarity measures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



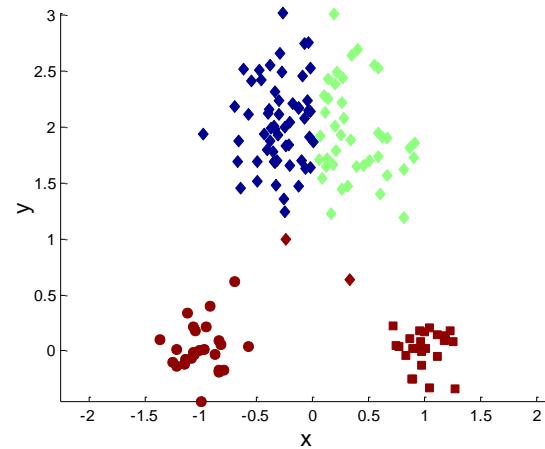
# Two different K-means Clusterings



Original Points



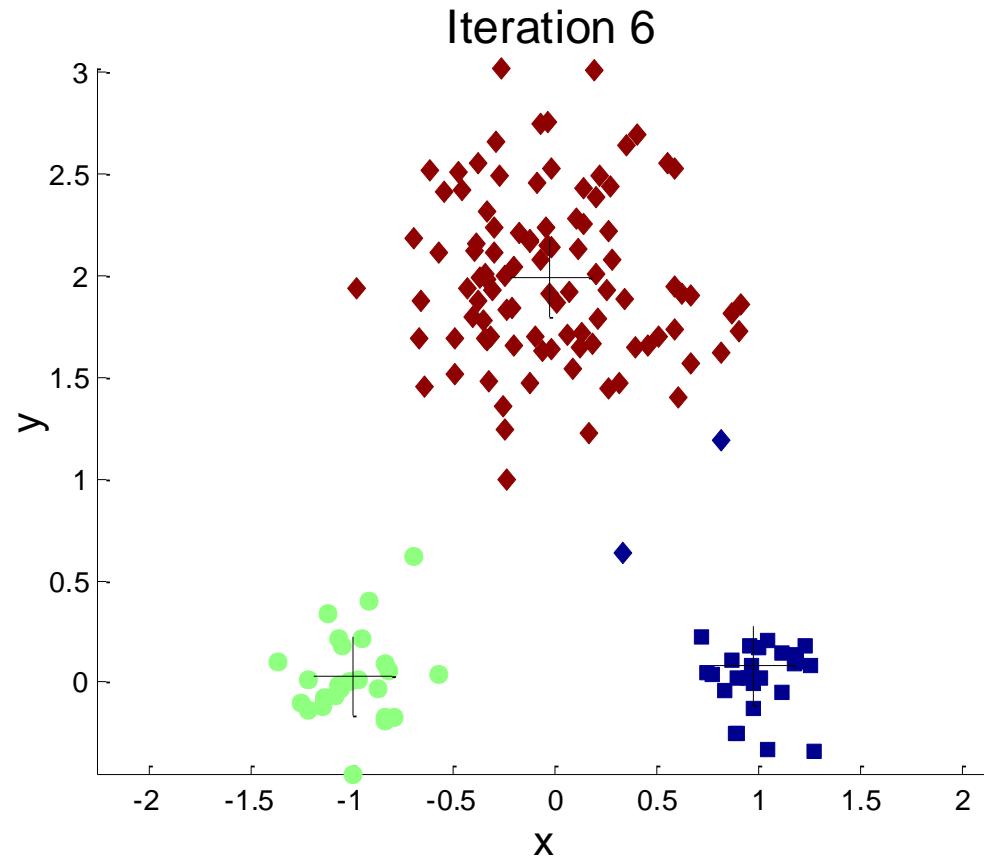
Optimal Clustering



Sub-optimal Clustering

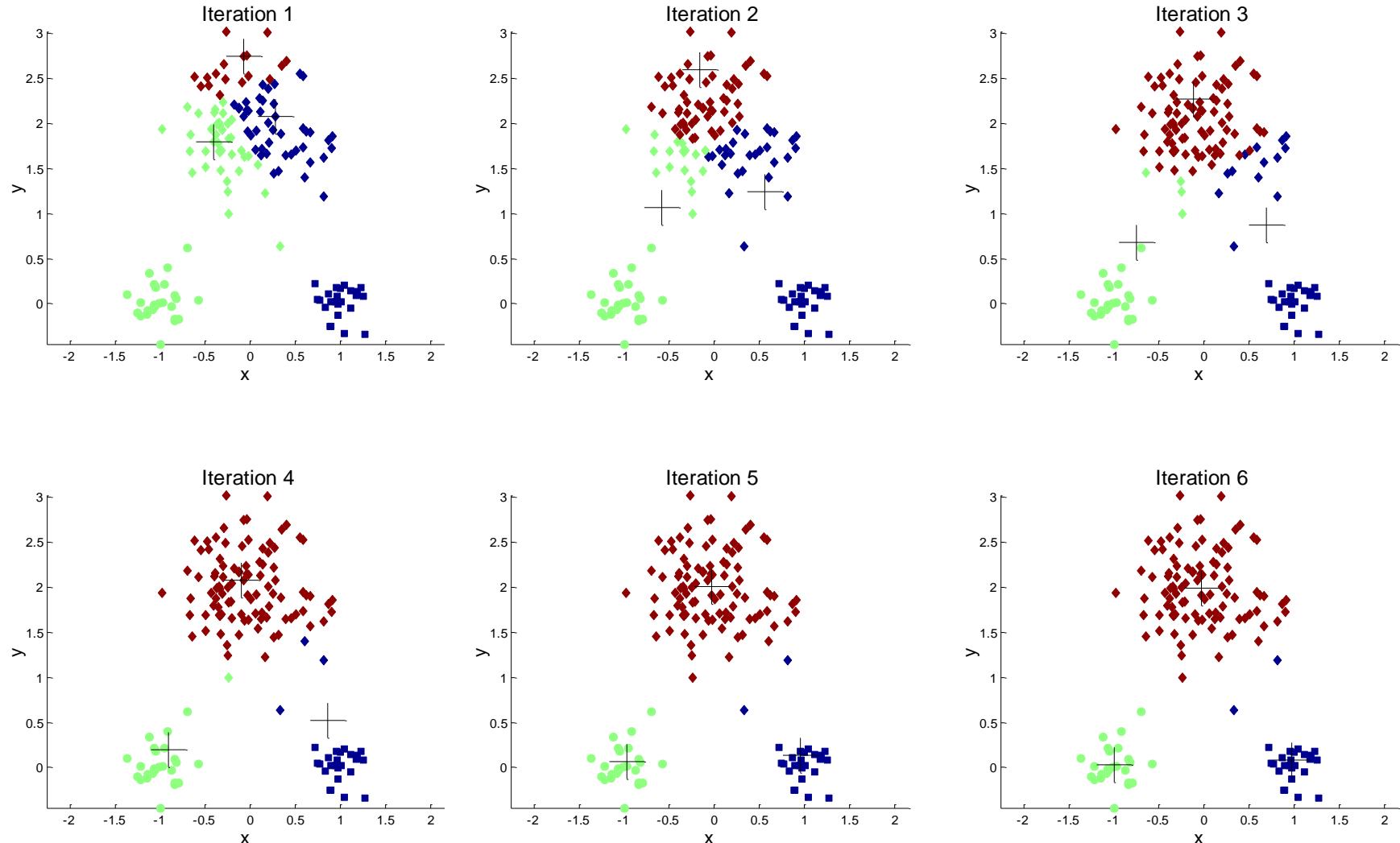


# Importance of Choosing Initial Centroids



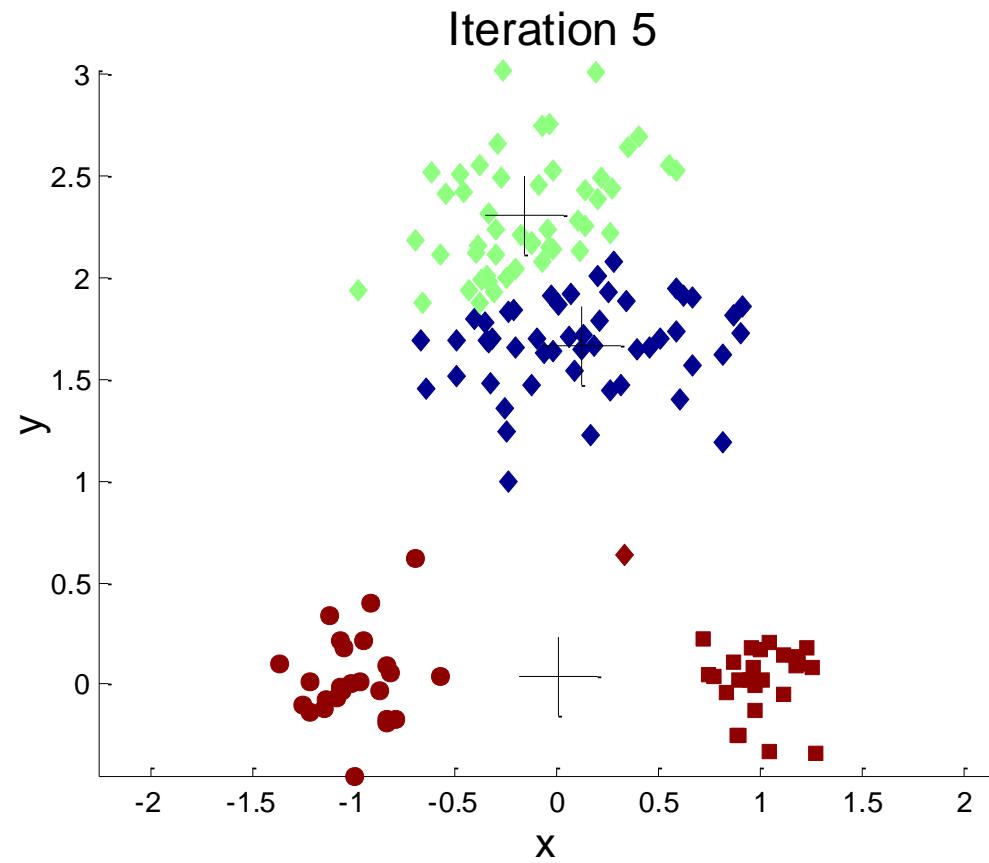


# Importance of Choosing Initial Centroids



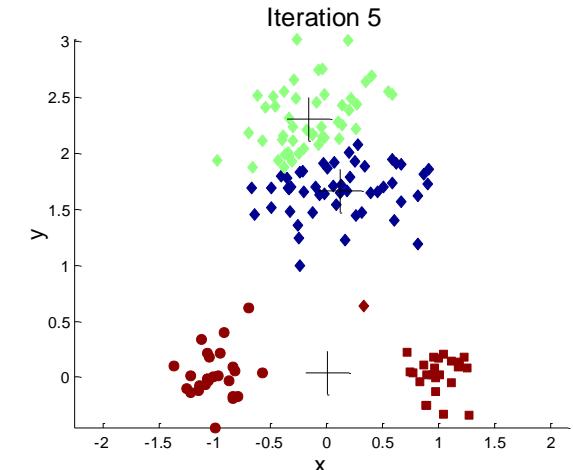
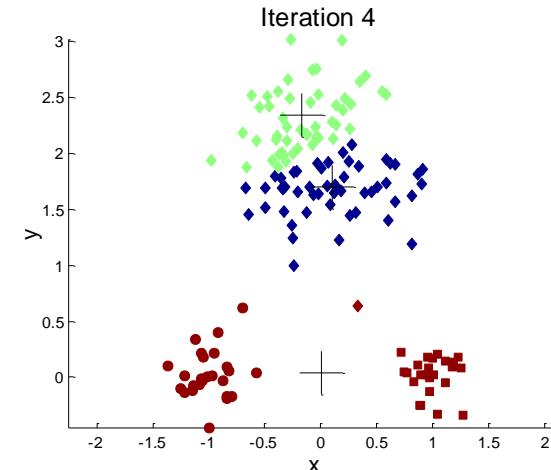
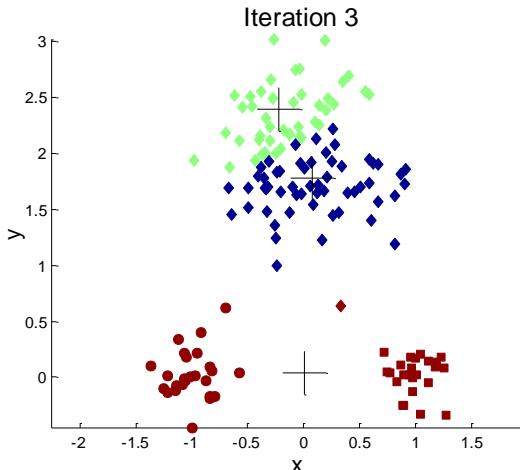
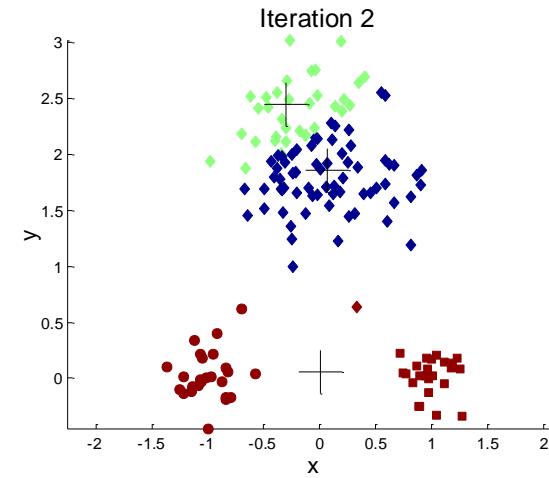
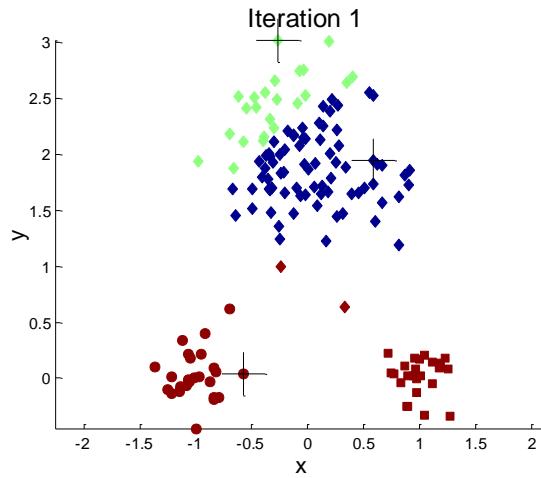


# Importance of Choosing Initial Centroids





# Importance of Choosing Initial Centroids





# Evaluating K-means Clusters

- Most common measure is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster
  - To get SSE, we square these errors and sum them.

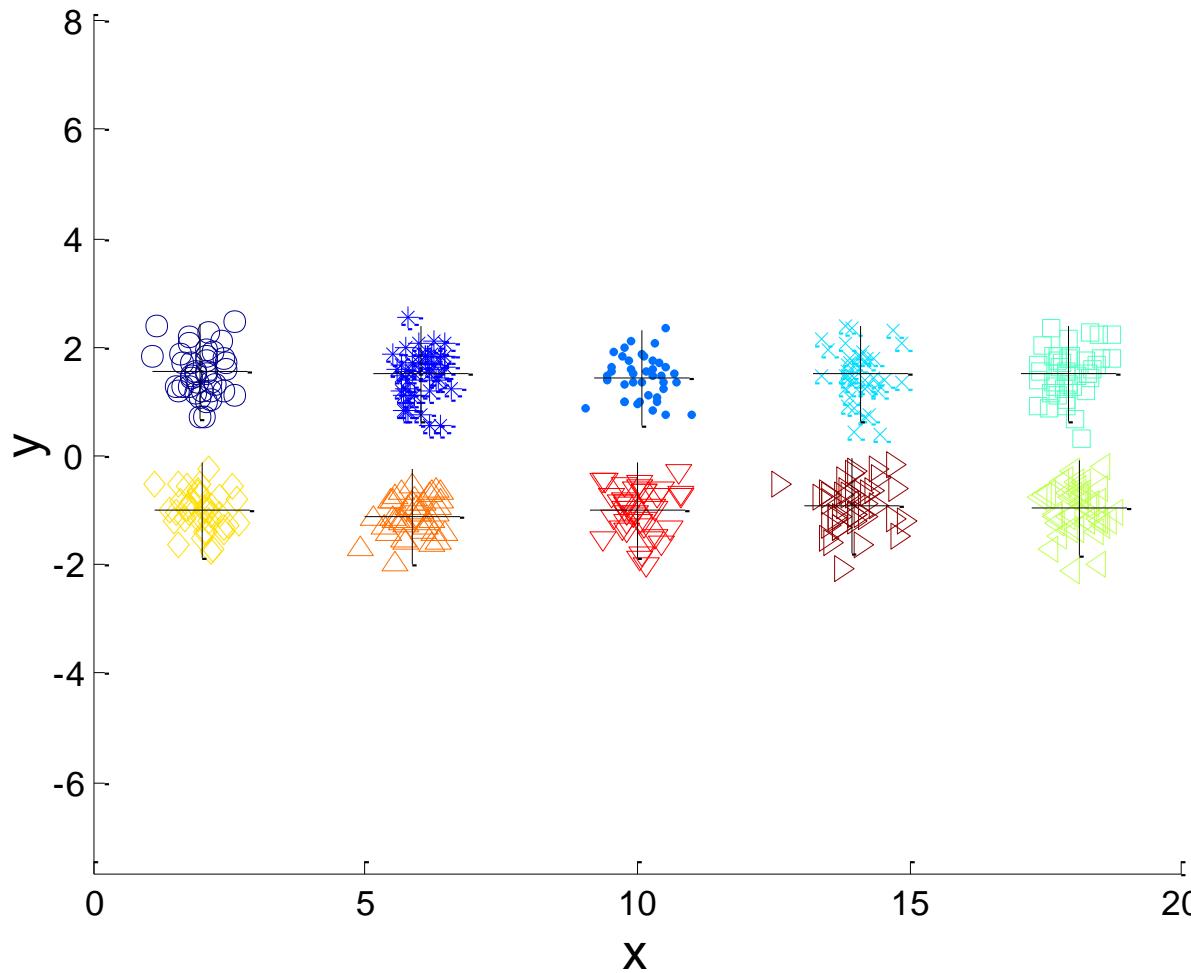
$$SSE = \sum_{i=1}^K \sum_{x \in C_i} dist^2(m_i, x)$$

- $x$  is a data point in cluster  $C_i$  and  $m_i$  is the representative point for cluster  $C_i$ 
  - can show that  $m_i$  corresponds to the center (mean) of the cluster
- Given two clusters, we can choose the one with the smallest error
- One easy way to reduce SSE is to increase K, the number of clusters
  - A good clustering with smaller K can have a lower SSE than a poor clustering with higher K



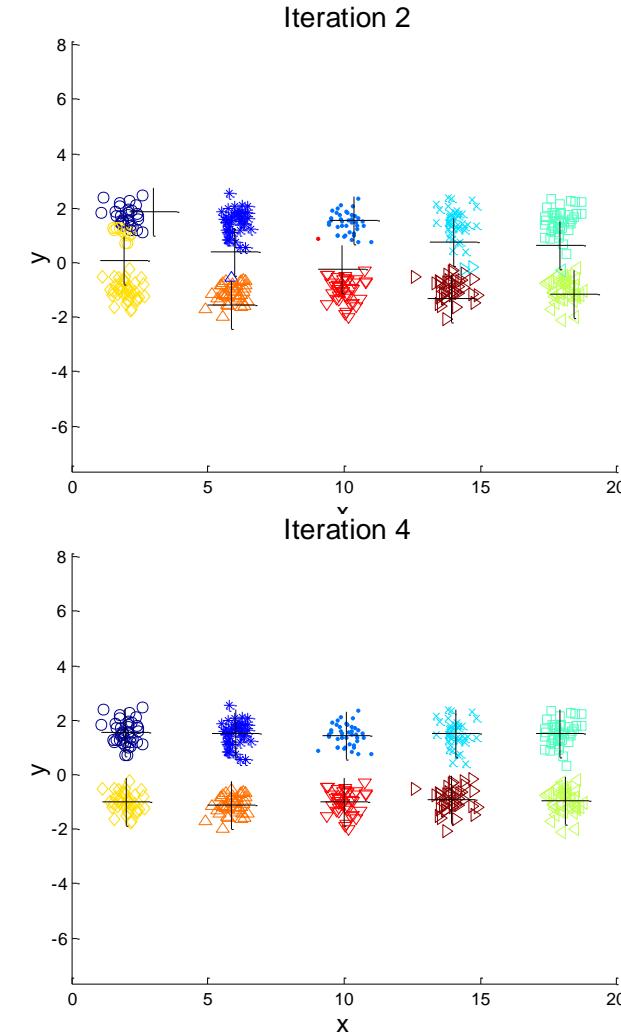
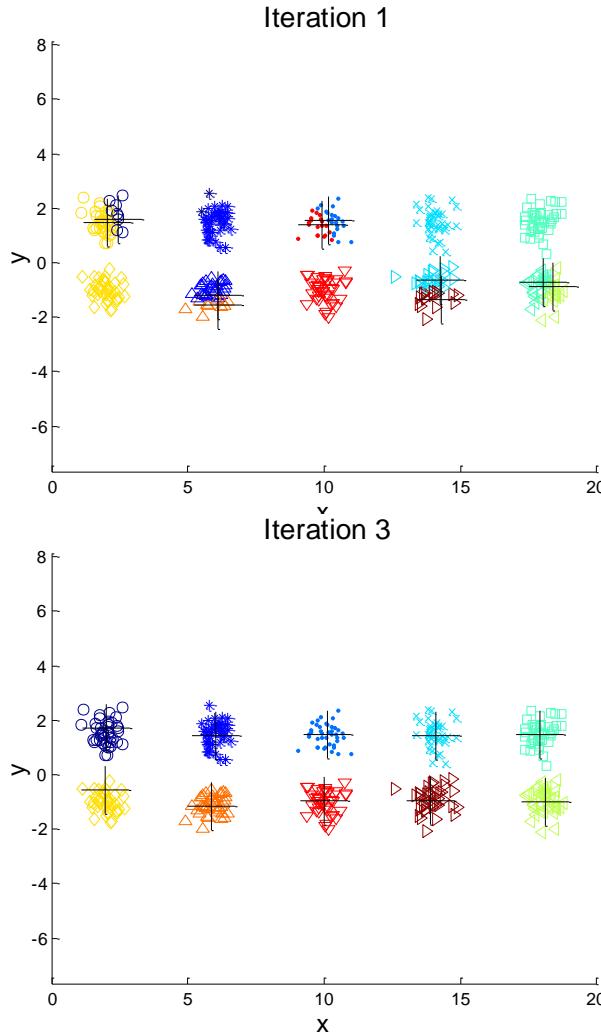
# 10 Clusters Example

Iteration 4





# 10 Clusters Example



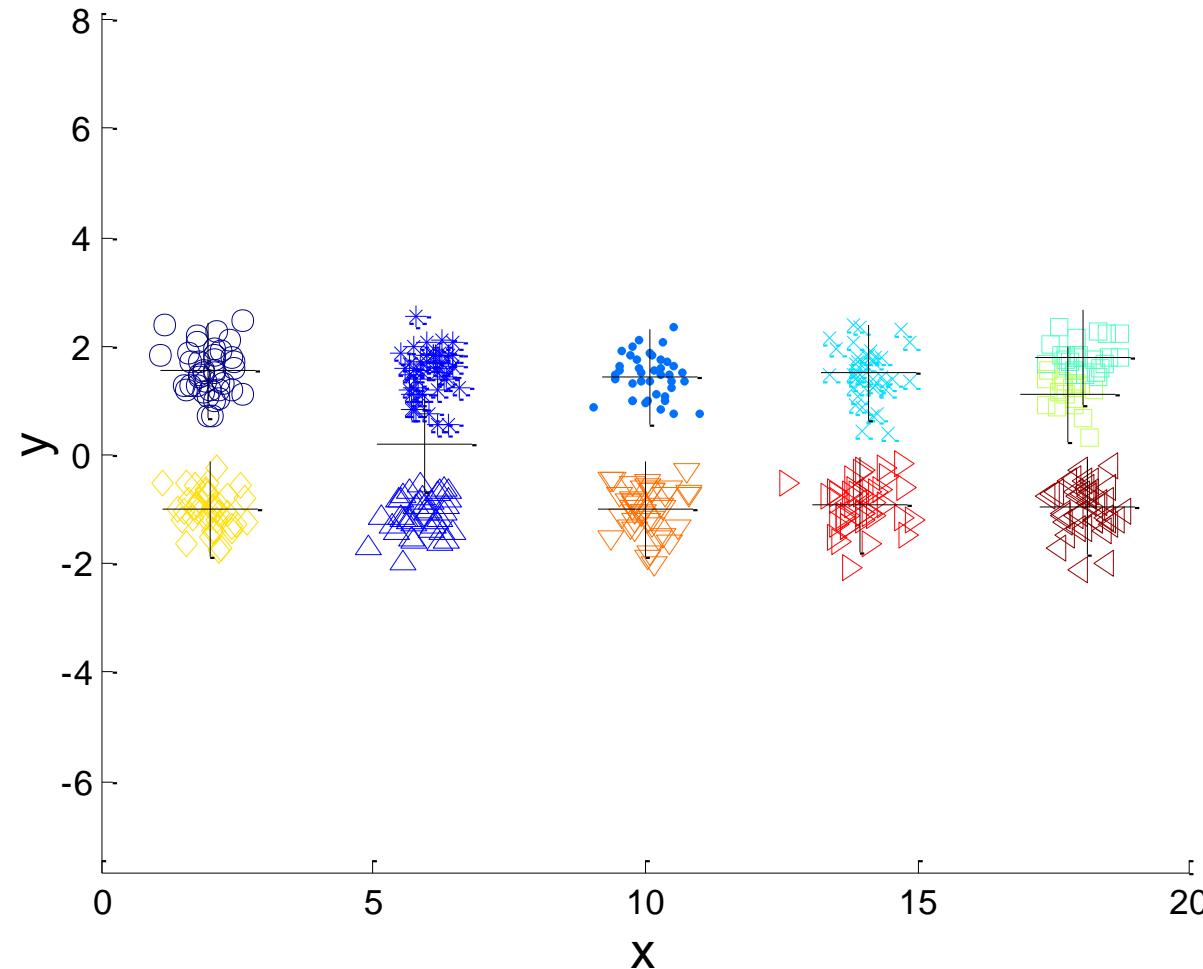
Starting with two initial centroids in one cluster of each pair of clusters

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



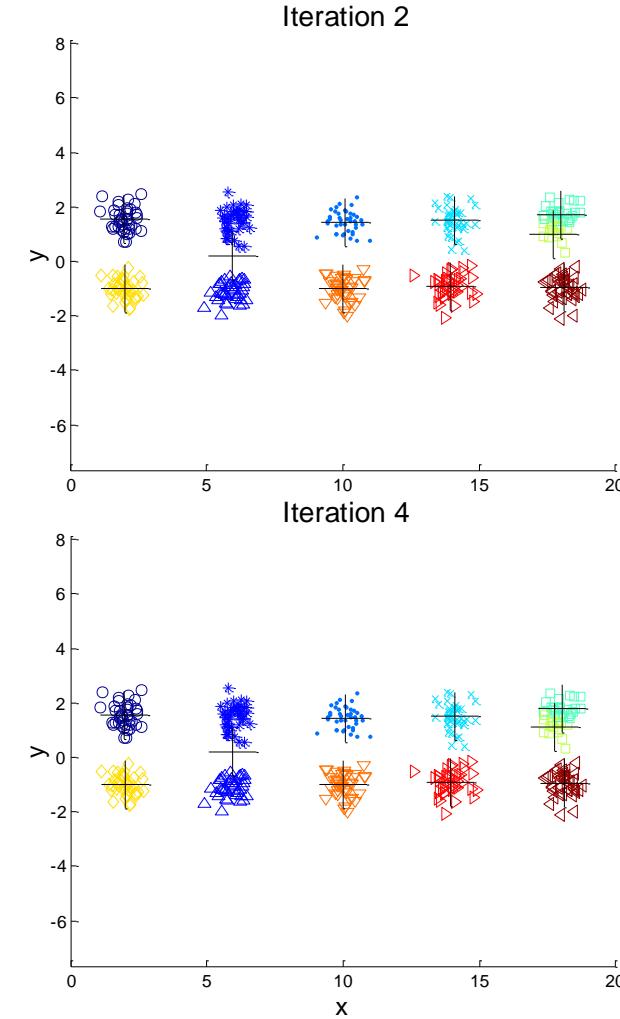
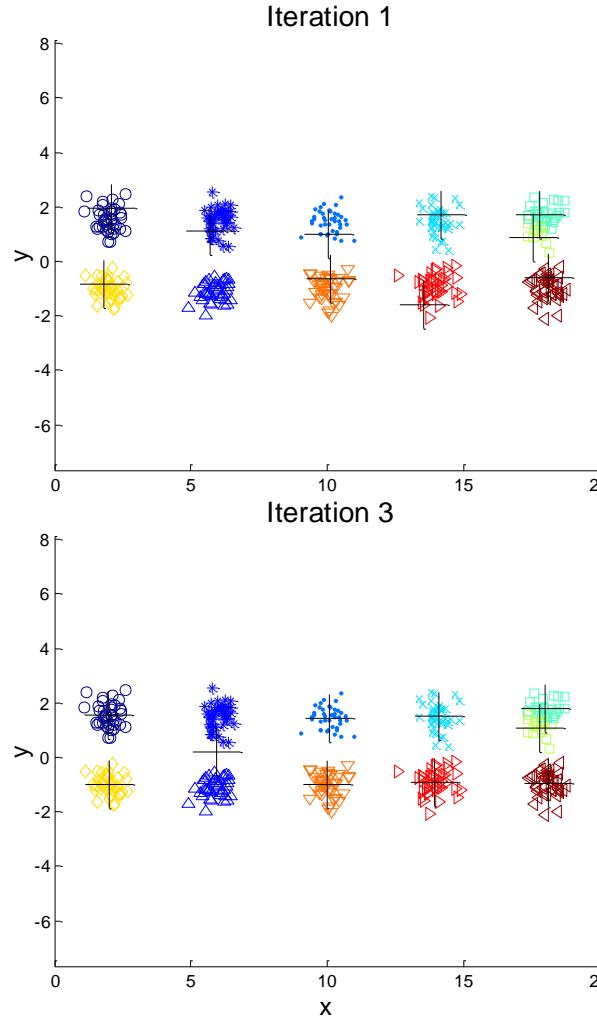
# 10 Clusters Example

Iteration 4





# 10 Clusters Example





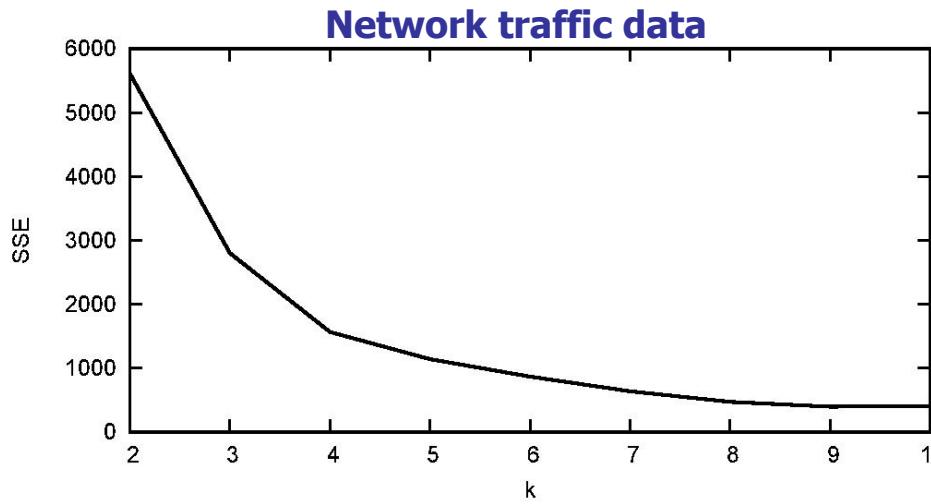
# Solutions to Initial Centroids Problem

- Multiple runs
  - Helps, but probability is not on your side
- Sample and use hierarchical clustering to determine initial centroids
- Select more than  $k$  initial centroids and then select among these initial centroids
  - Select most widely separated
- Postprocessing
- Bisecting K-means
  - Not as susceptible to initialization issues



# K-means parameter setting

- Elbow graph (Knee approach)
  - Plotting the quality measure trend (e.g., SSE) against K
  - Choosing the value of K
    - the gain from adding a centroid is negligible
    - The reduction of the quality measure is not interesting anymore





# Handling Empty Clusters

- Basic K-means algorithm can yield empty clusters
- Several strategies
  - Choose the point that contributes most to SSE
  - Choose a point from the cluster with the highest SSE
  - If there are several empty clusters, the above can be repeated several times.



# Pre-processing and Post-processing

- Pre-processing
  - Normalize the data
  - Eliminate outliers
- Post-processing
  - Eliminate small clusters that may represent outliers
  - Split 'loose' clusters, i.e., clusters with relatively high SSE
  - Merge clusters that are 'close' and that have relatively low SSE
  - Can use these steps during the clustering process



# Bisecting K-means

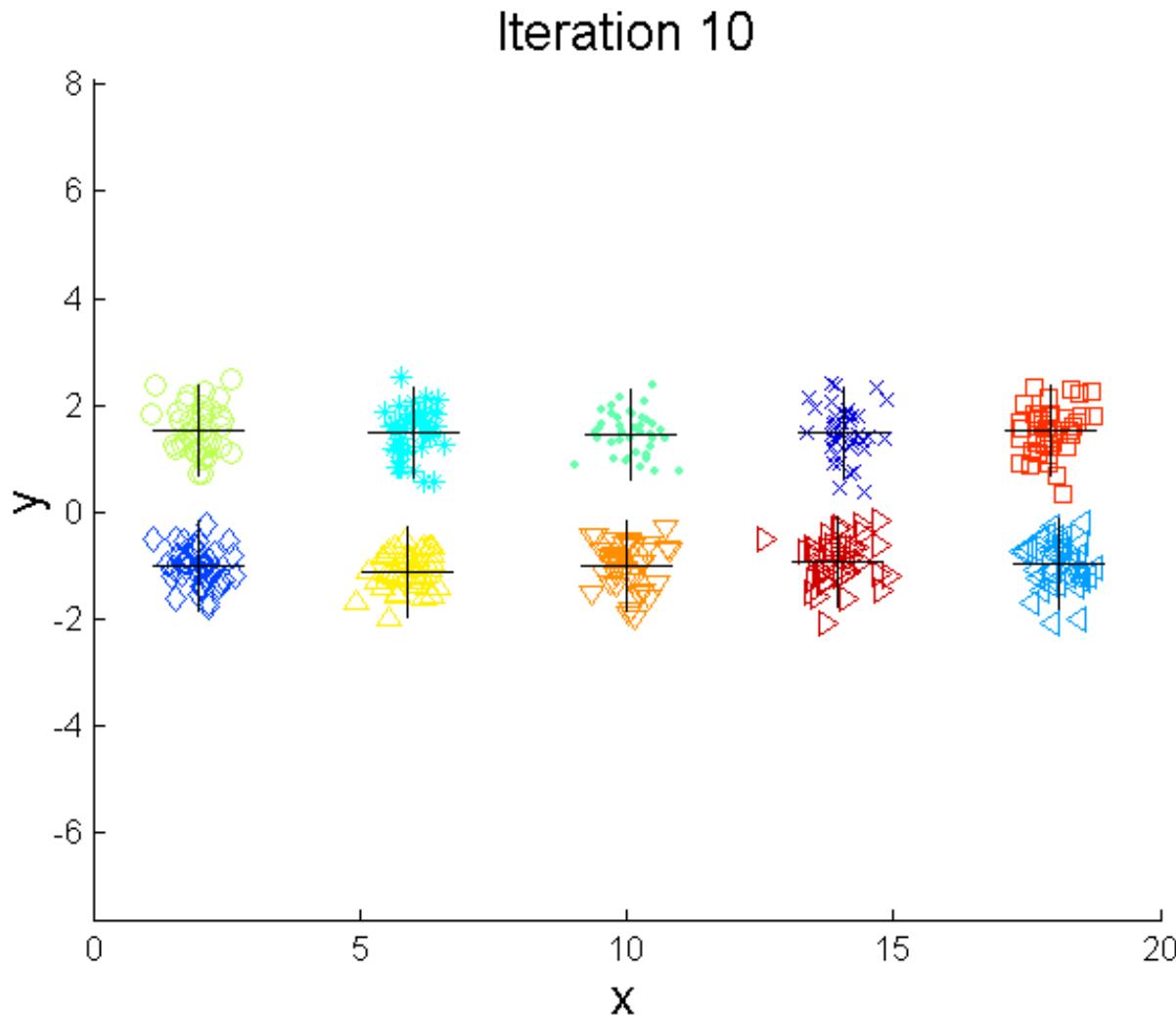
## ■ Bisecting K-means algorithm

- Variant of K-means that can produce a partitional or a hierarchical clustering

- 
- 1: Initialize the list of clusters to contain the cluster containing all points.
  - 2: **repeat**
  - 3:   Select a cluster from the list of clusters
  - 4:   **for**  $i = 1$  to *number\_of\_iterations* **do**
  - 5:     Bisect the selected cluster using basic K-means
  - 6:   **end for**
  - 7:   Add the two clusters from the bisection with the lowest SSE to the list of clusters.
  - 8: **until** Until the list of clusters contains  $K$  clusters
-



# Bisecting K-means Example



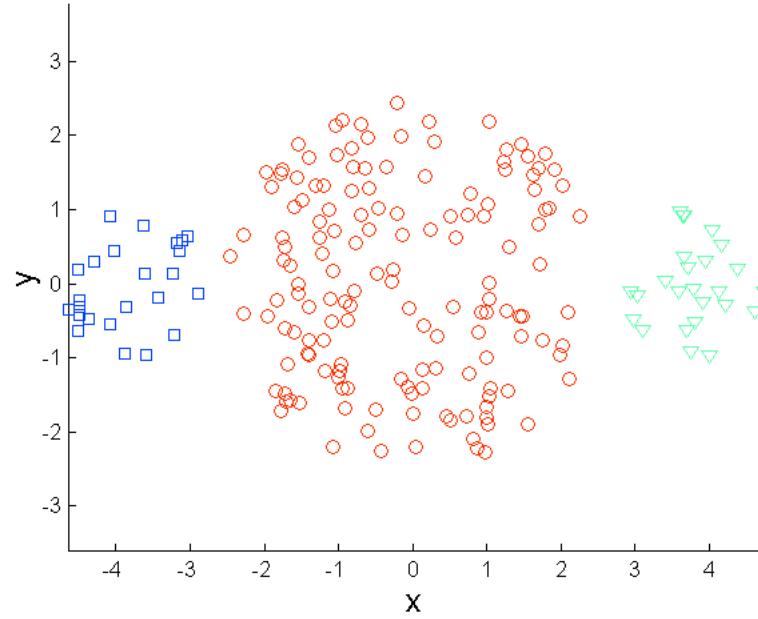


# Limitations of K-means

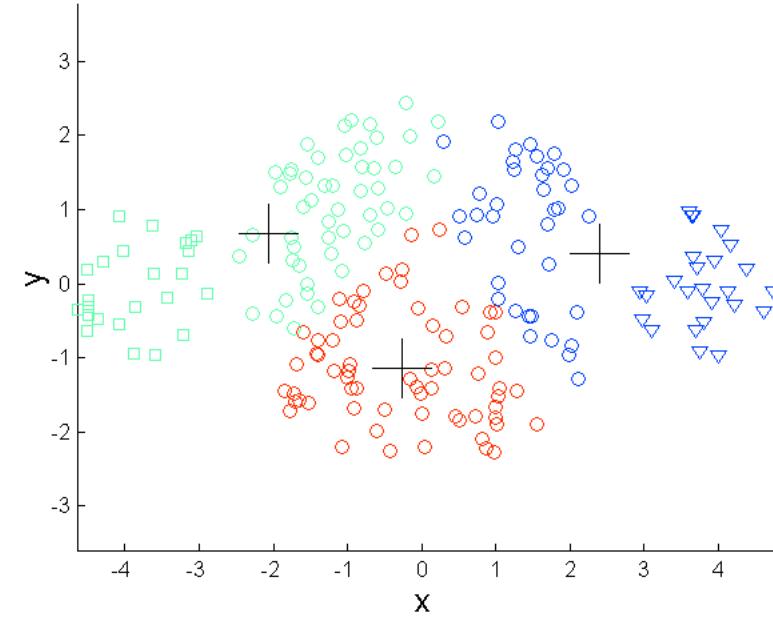
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes
- K-means has problems when the data contains outliers.



# Limitations of K-means: Differing Sizes



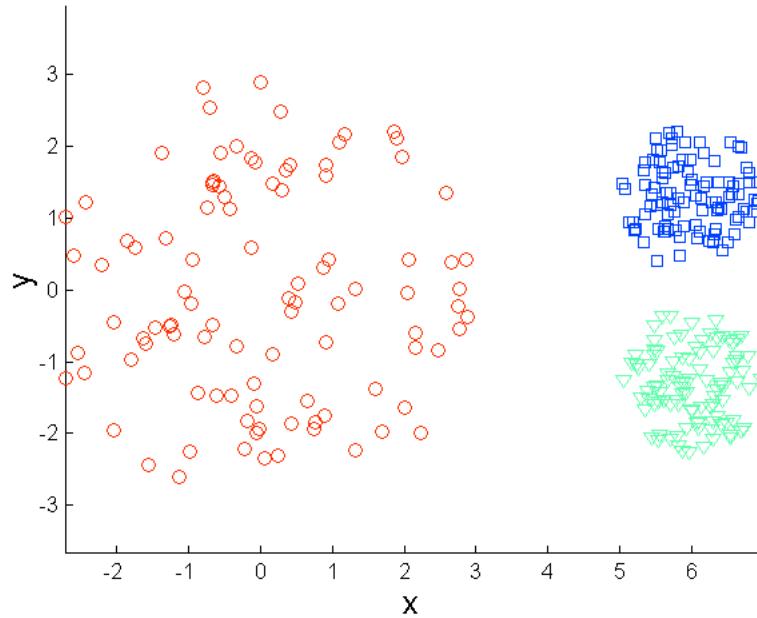
Original Points



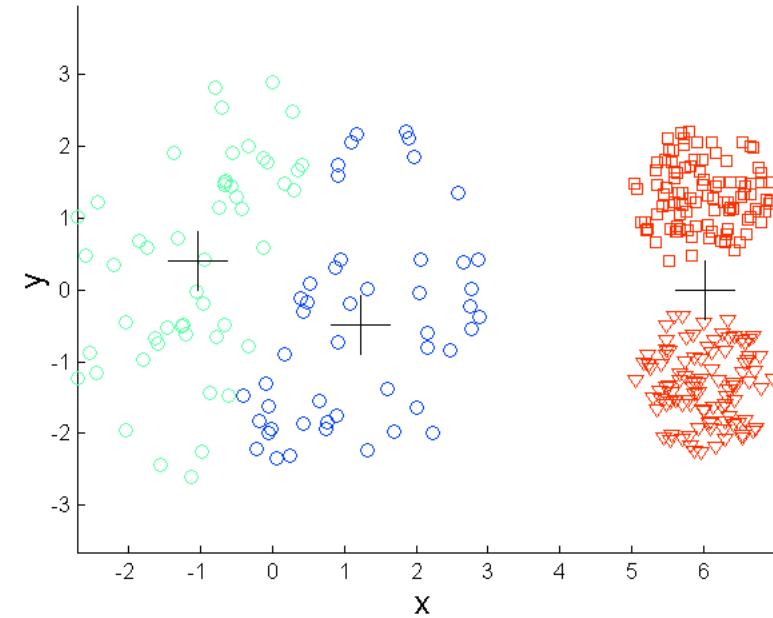
K-means (3 Clusters)



# Limitations of K-means: Differing Density



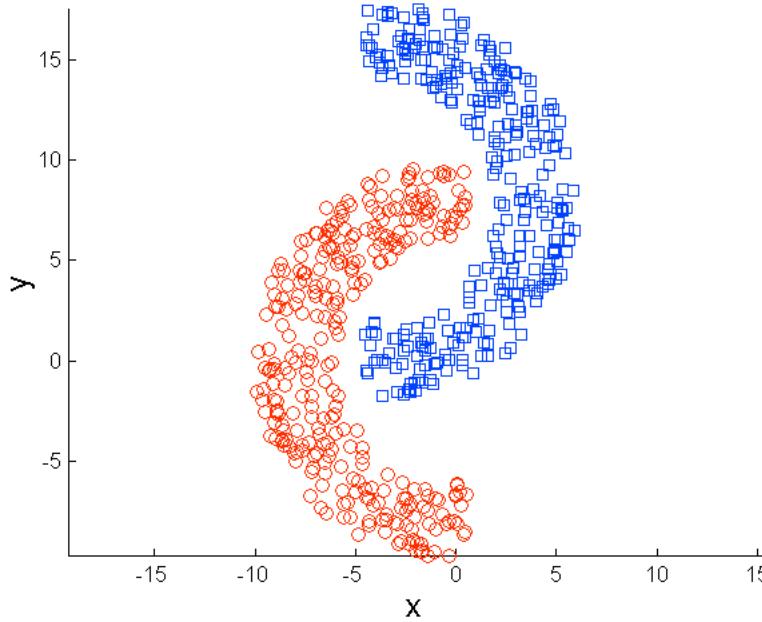
Original Points



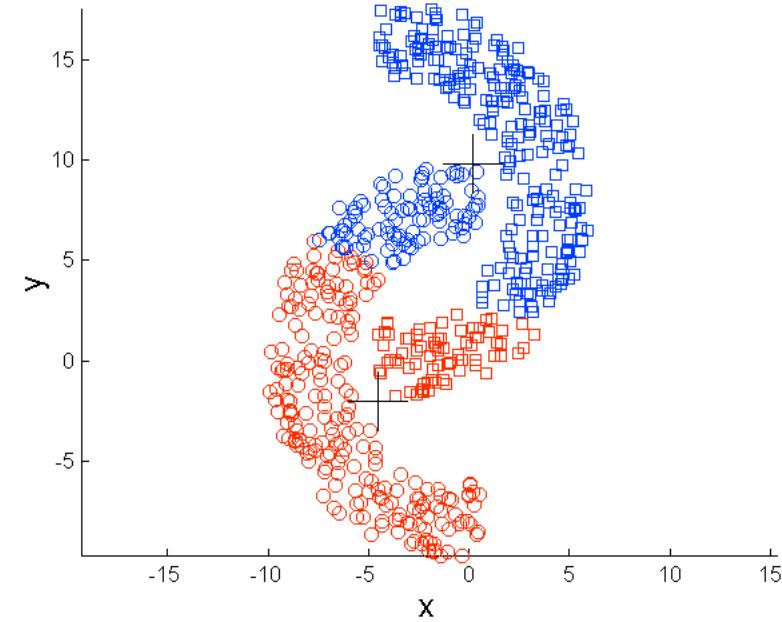
K-means (3 Clusters)



# Limitations of K-means: Non-globular Shapes



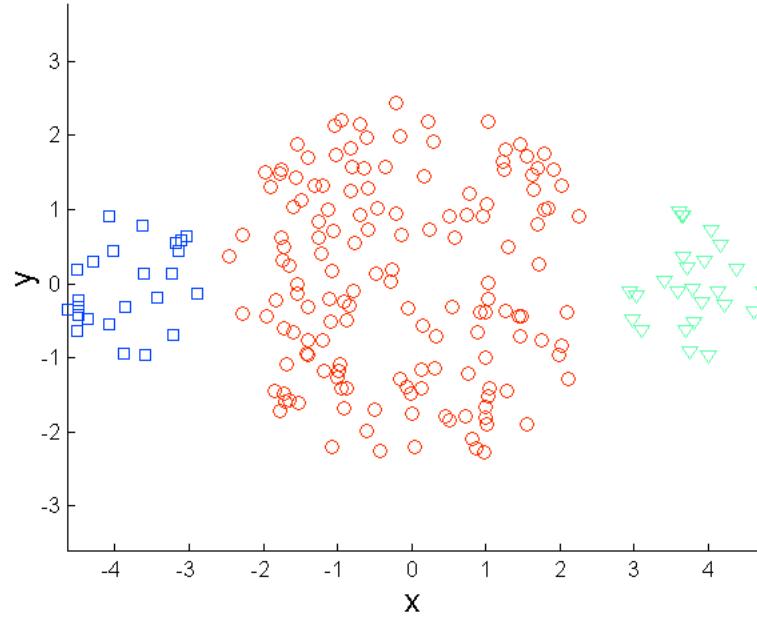
Original Points



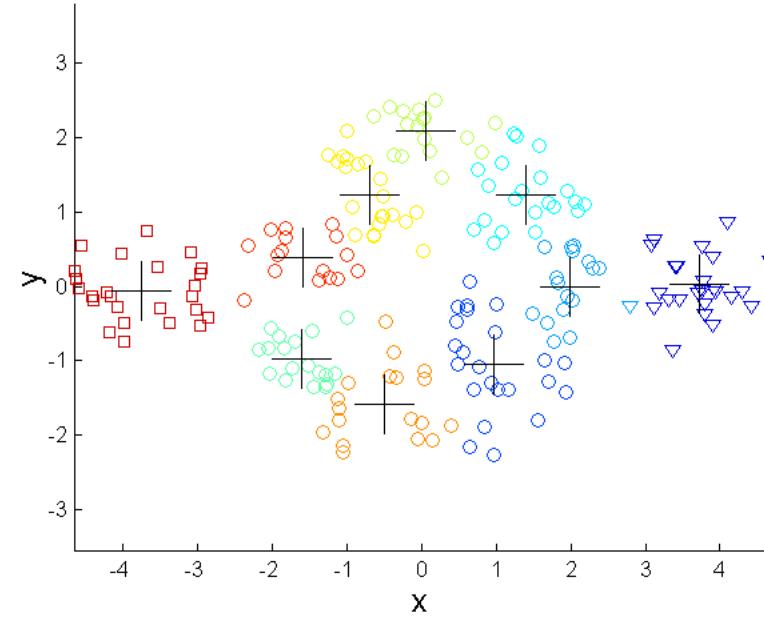
K-means (2 Clusters)



# Overcoming K-means Limitations



Original Points



K-means Clusters

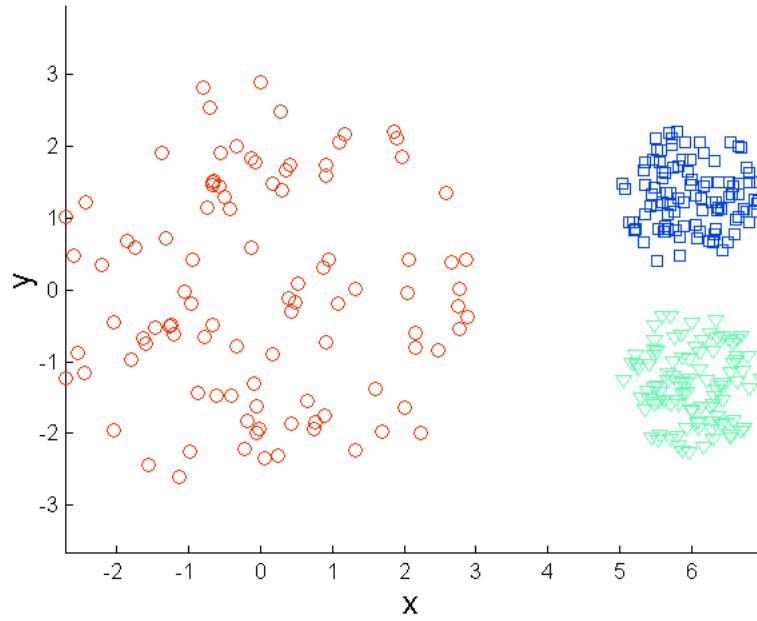
One solution is to use many clusters.

Find parts of clusters, but need to put together.

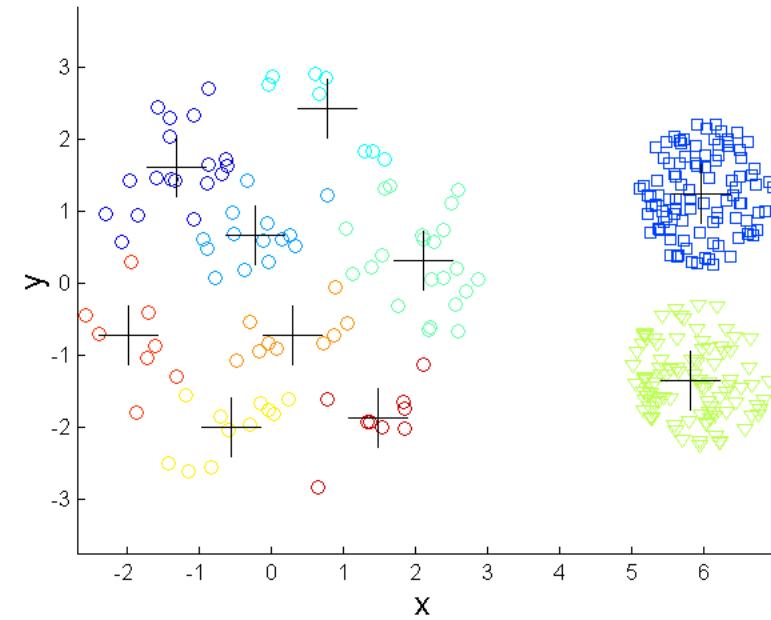
From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



# Overcoming K-means Limitations



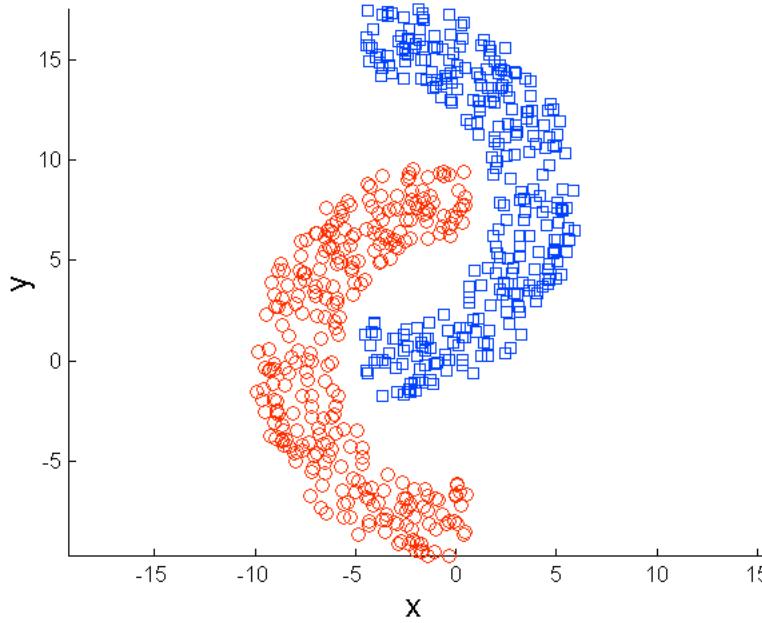
Original Points



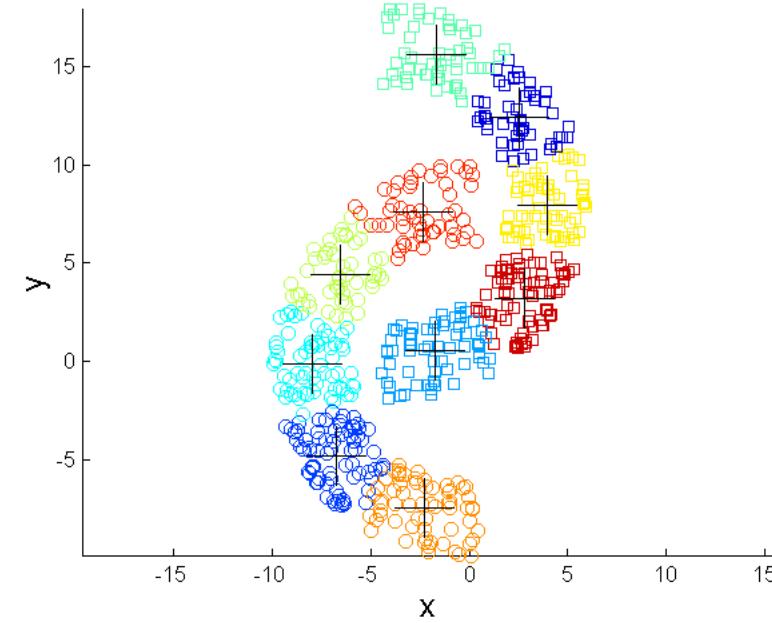
K-means Clusters



# Overcoming K-means Limitations



Original Points

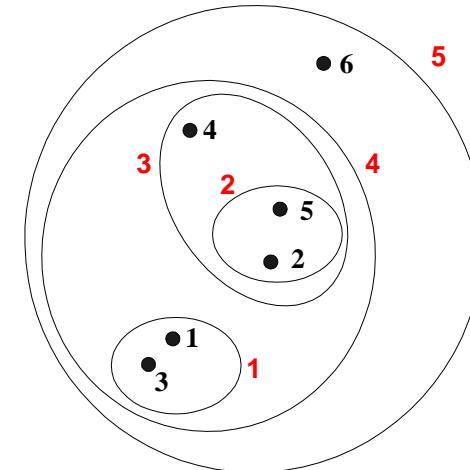
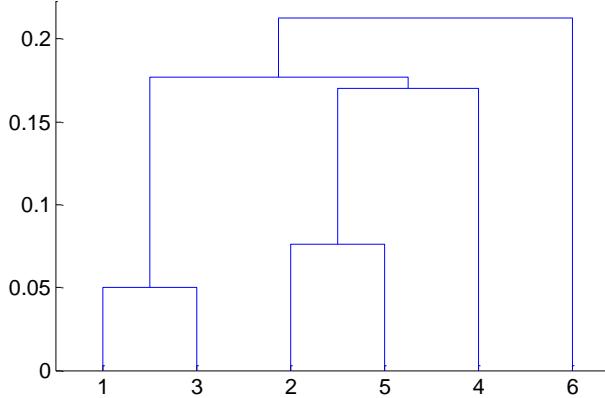


K-means Clusters



# Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree like diagram that records the sequences of merges or splits





# 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 dendrogram at the proper level
- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, ...)



# Hierarchical Clustering

- Two main types of hierarchical clustering
  - Agglomerative:
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or  $k$  clusters) left
  - Divisive:
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains a point (or there are  $k$  clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time



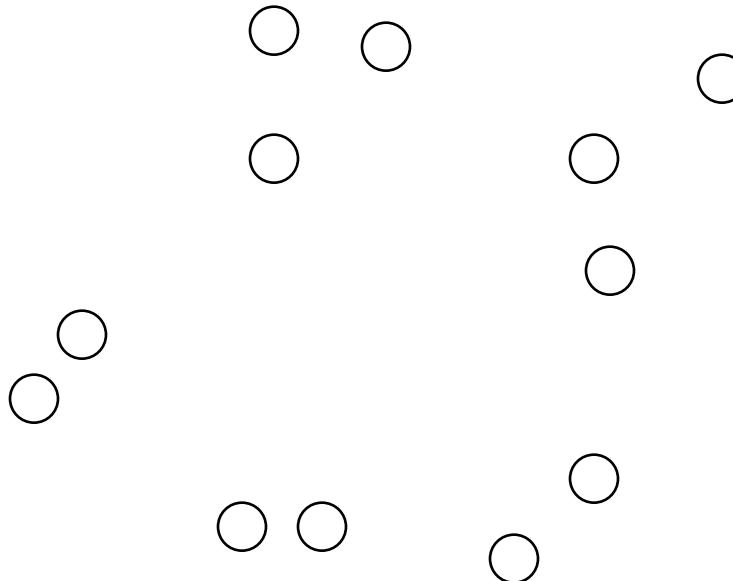
# Agglomerative Clustering Algorithm

- More popular hierarchical clustering technique
- Basic algorithm is straightforward
  1. Compute the proximity matrix
  2. Let each data point be a cluster
  3. **Repeat**
    4. Merge the two closest clusters
    5. Update the proximity matrix
  6. **Until** only a single cluster remains
- Key operation is the computation of the proximity of two clusters
  - Different approaches to defining the distance between clusters distinguish the different algorithms



# Starting Situation

- Start with clusters of individual points and a proximity matrix



	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

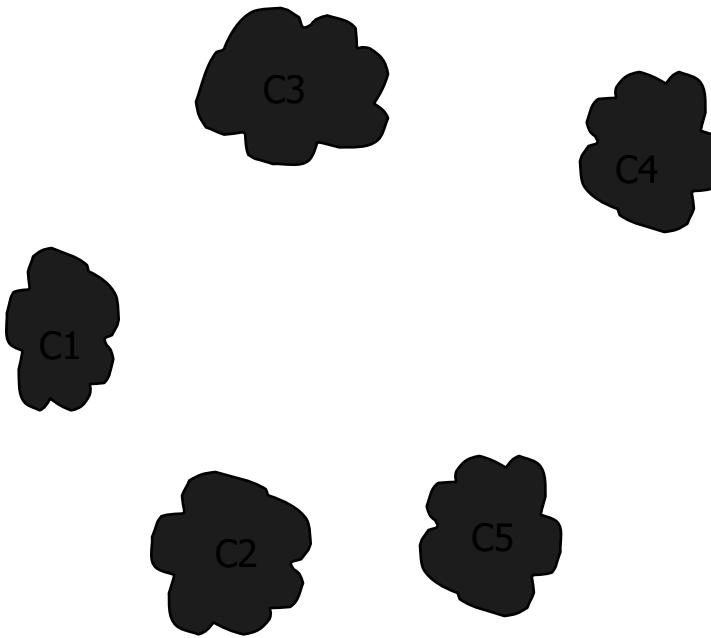
p1 p2 p3 p4 ... p9 p10 p11 p12

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



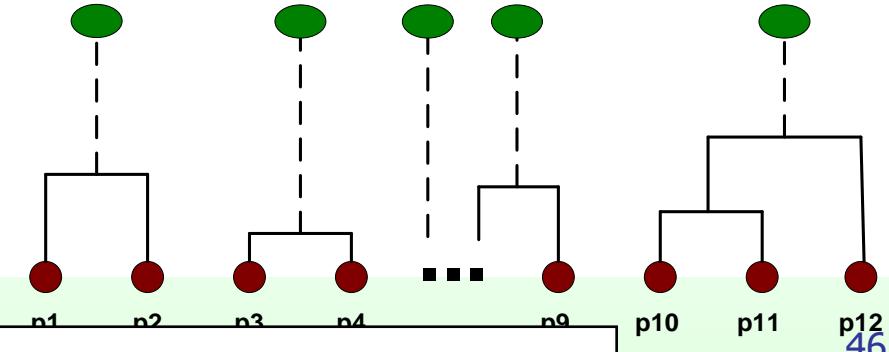
# Intermediate Situation

- After some merging steps, we have some clusters



	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix

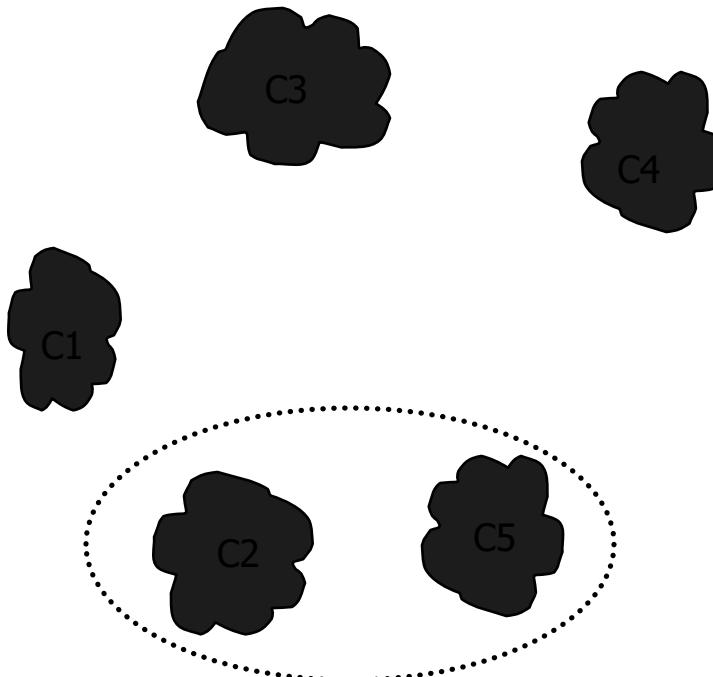


From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



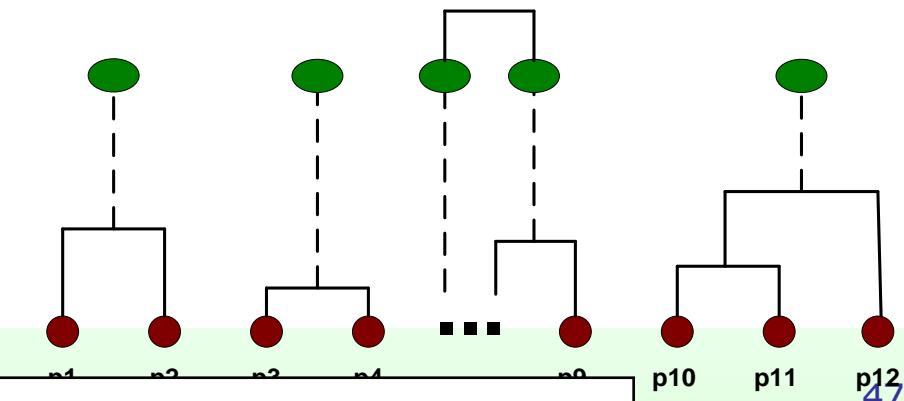
# Intermediate Situation

- We want to merge the two closest clusters ( $C_2$  and  $C_5$ ) and update the proximity matrix.



	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

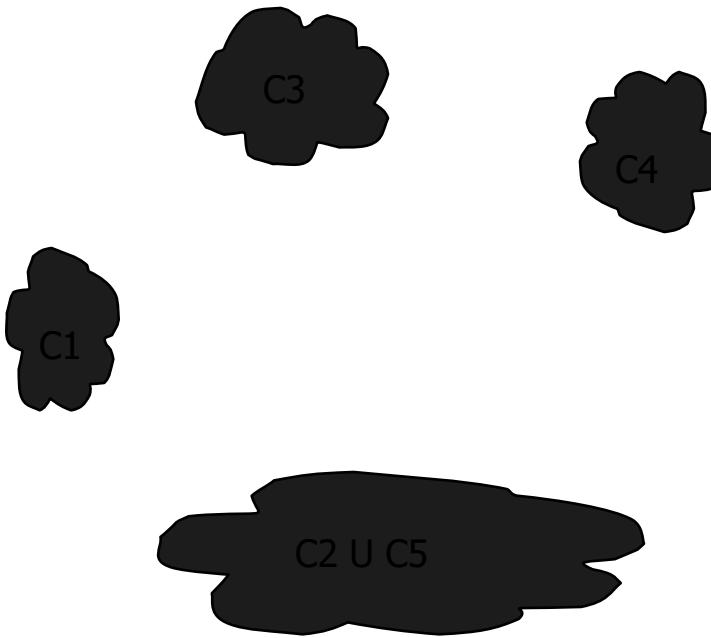
Proximity Matrix





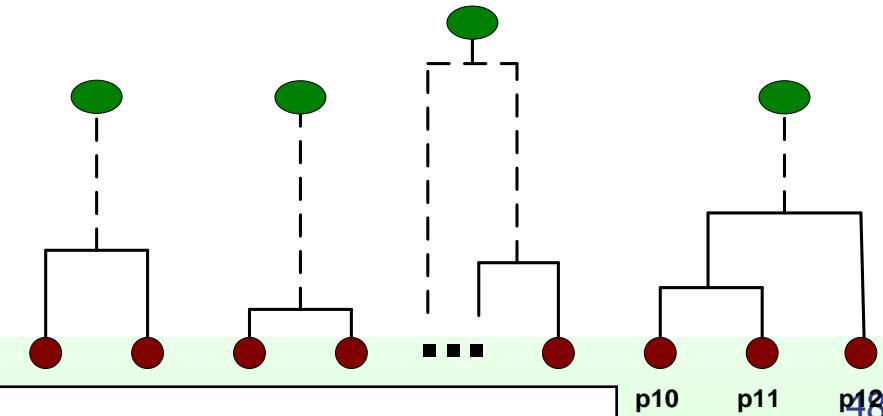
# After Merging

- The question is “How do we update the proximity matrix?”



		C2 U C5	C3	C4
C1	C1	?		
	?	?	?	?
C3		?		
C4	?			

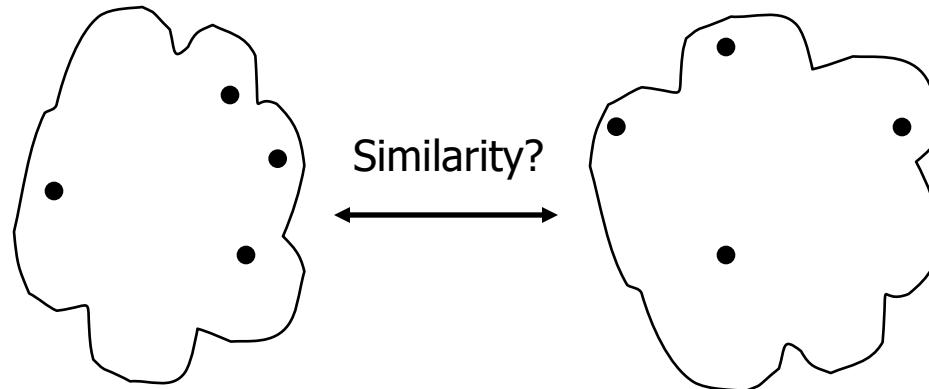
Proximity Matrix



From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



# How to Define Inter-Cluster Similarity



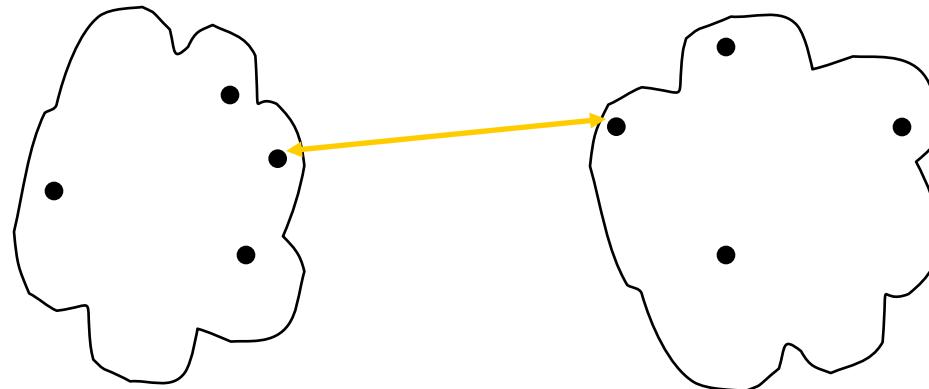
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

Proximity Matrix



# How to Define Inter-Cluster Similarity



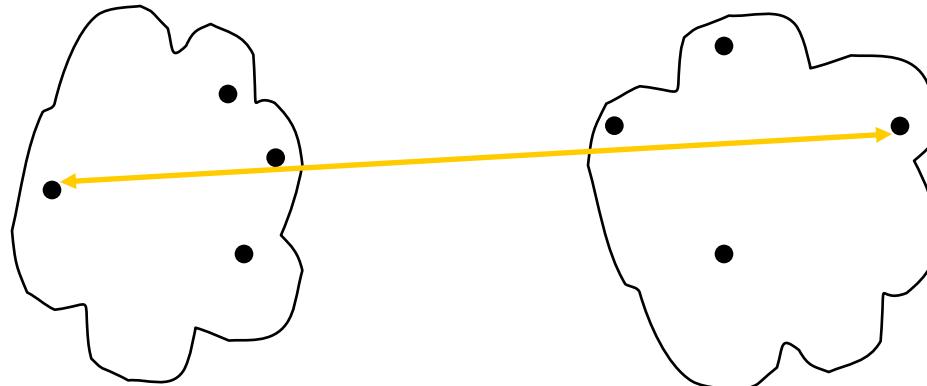
- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

Proximity Matrix



# How to Define Inter-Cluster Similarity



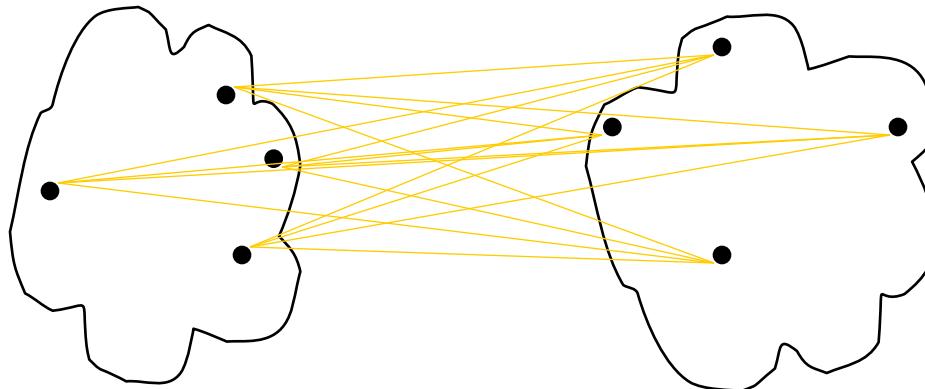
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

Proximity Matrix



# How to Define Inter-Cluster Similarity



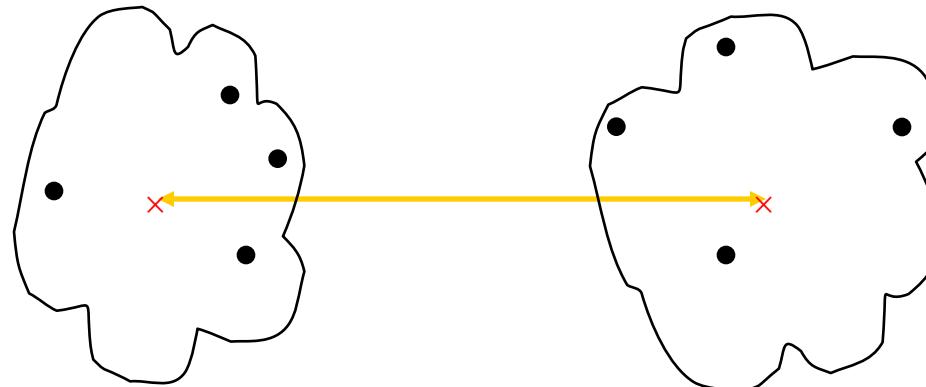
- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

Proximity Matrix



# How to Define Inter-Cluster Similarity



- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

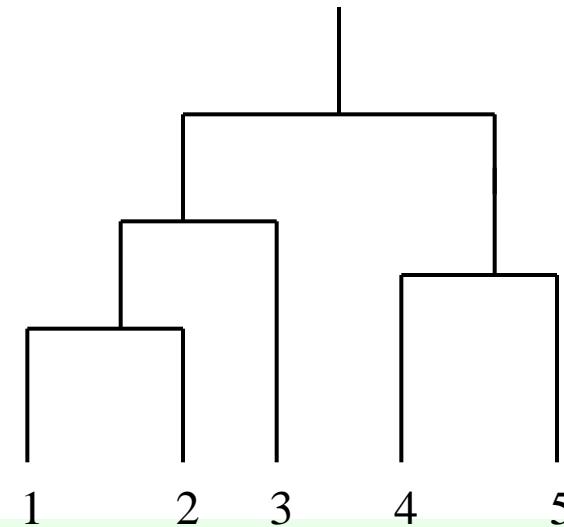
Proximity Matrix



# Cluster Similarity: MIN or Single Link

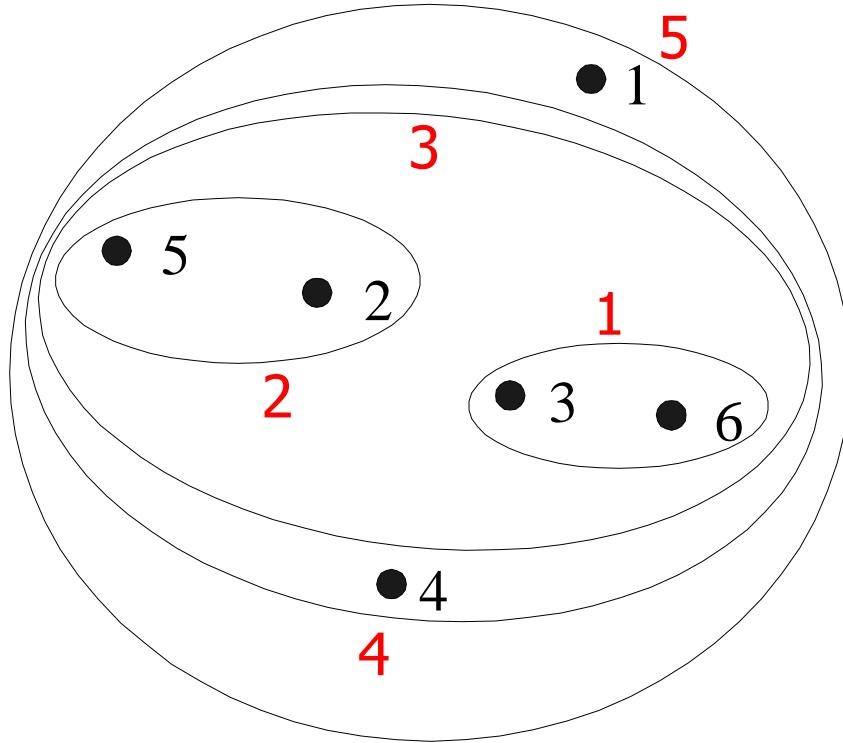
- Similarity of two clusters is based on the two most similar (closest) points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph.

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

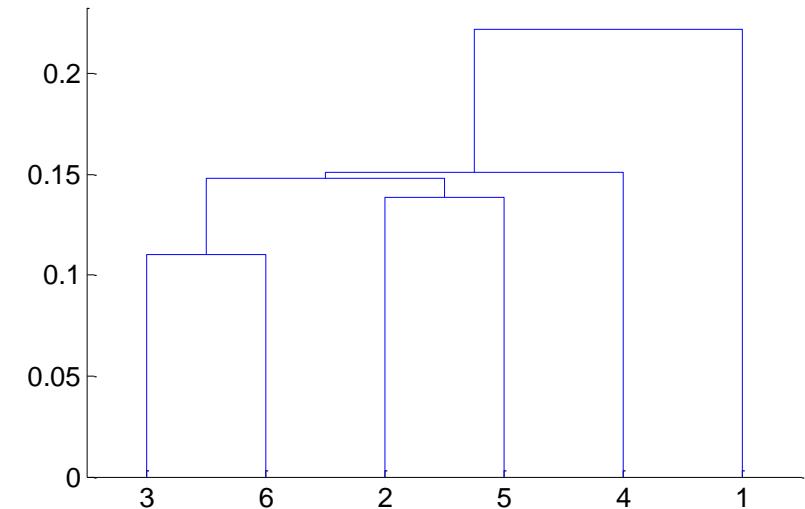




# Hierarchical Clustering: MIN



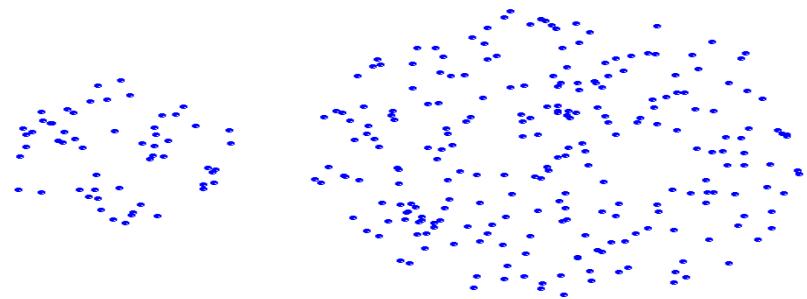
Nested Clusters



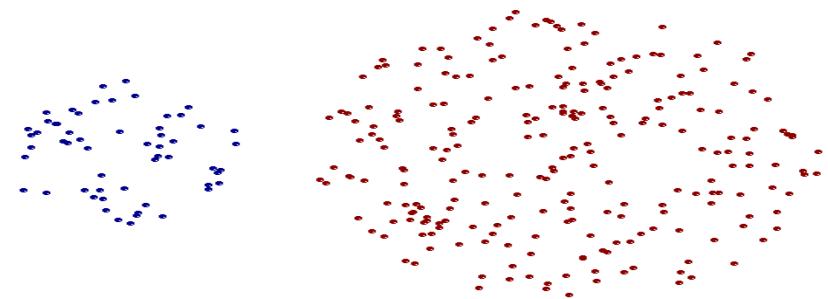
Dendrogram



# Strength of MIN



Original Points

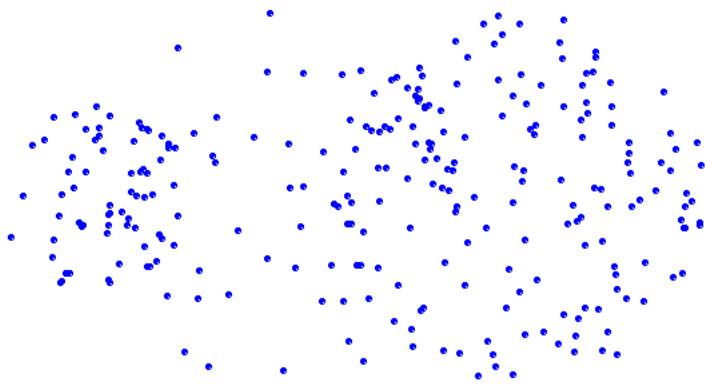


Two Clusters

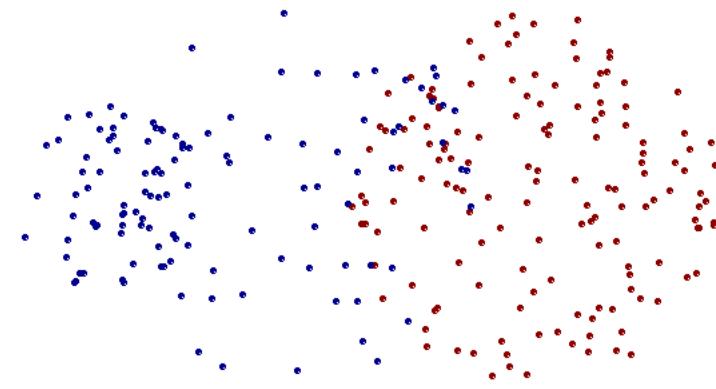
- Can handle non-elliptical shapes



# Limitations of MIN



Original Points



Two Clusters

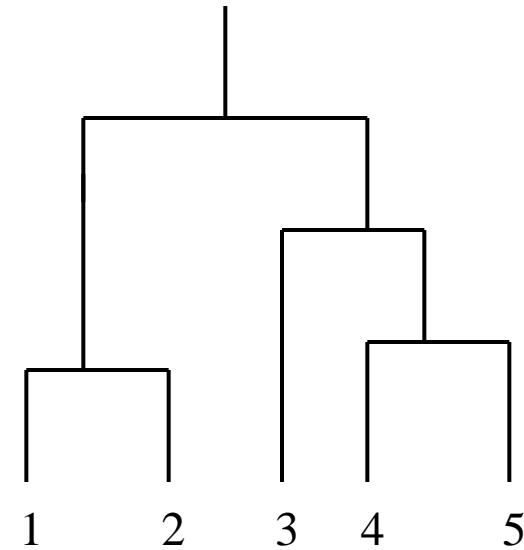
- Sensitive to noise and outliers



## Cluster Similarity: MAX or Complete Linkage

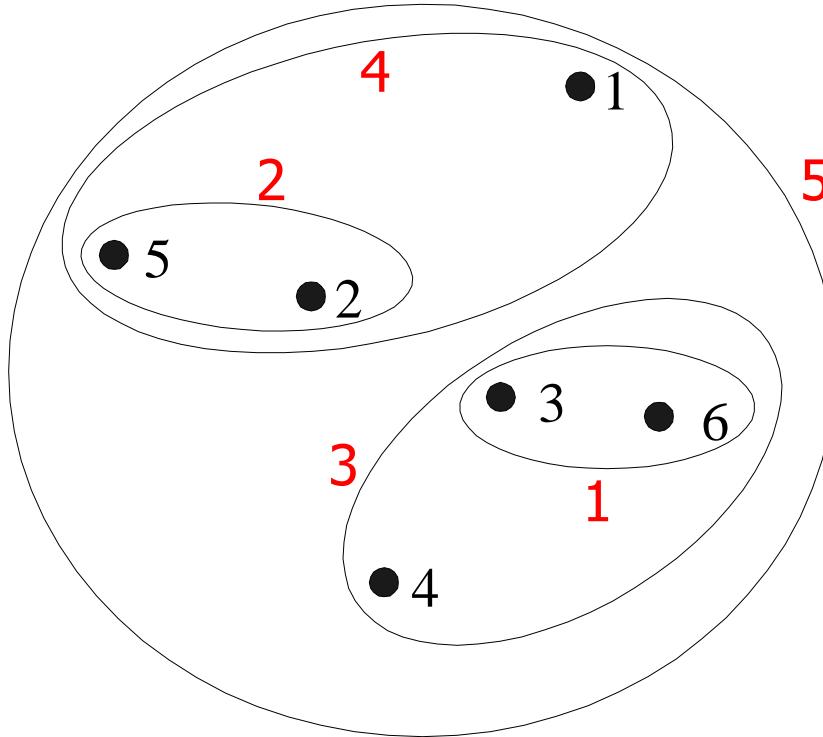
- Similarity of two clusters is based on the two least similar (most distant) points in the different clusters
  - Determined by all pairs of points in the two clusters

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

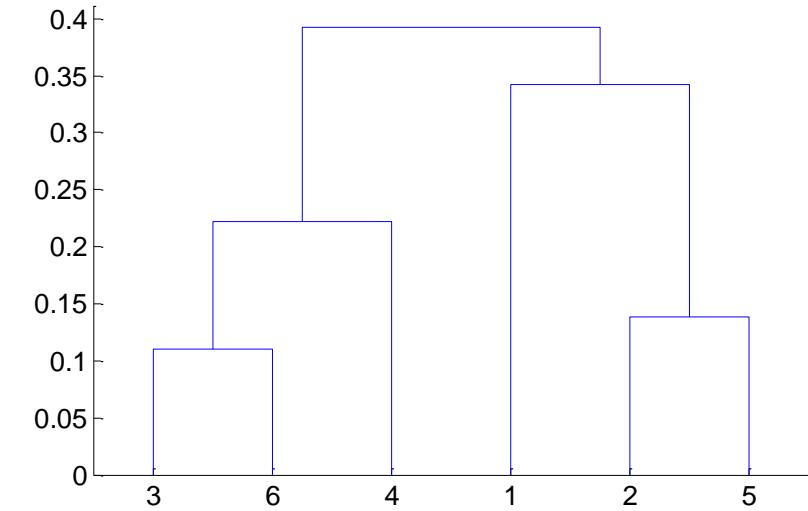




# Hierarchical Clustering: MAX



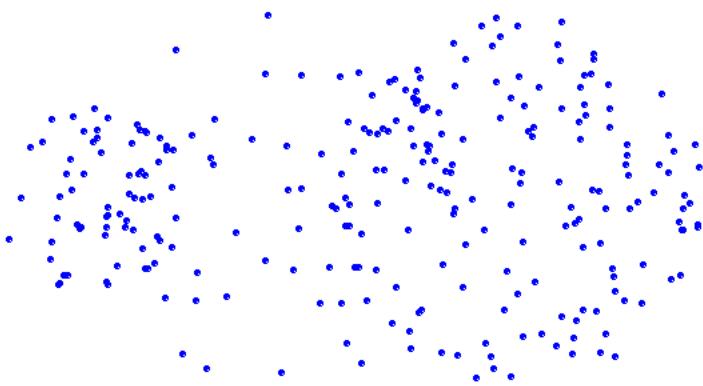
Nested Clusters



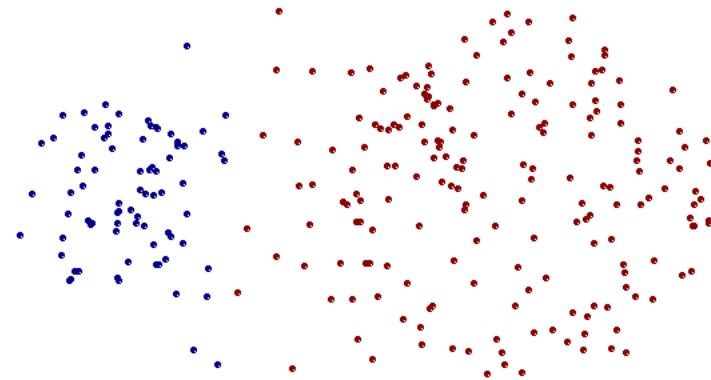
Dendrogram



# Strength of MAX



Original Points

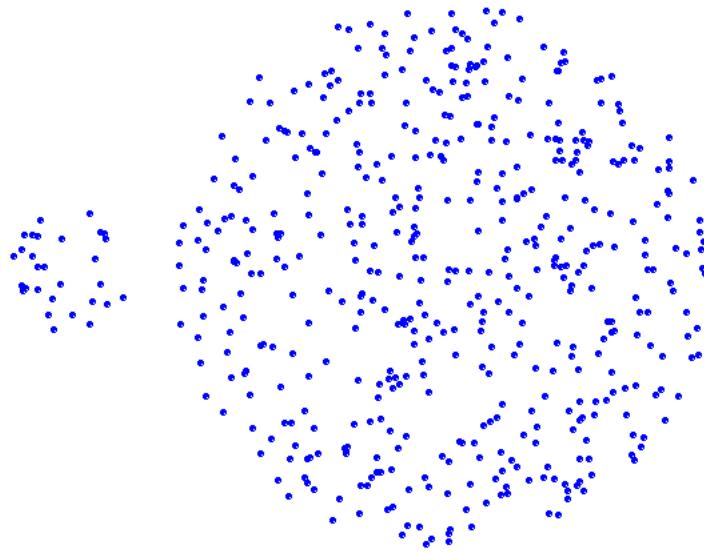


Two Clusters

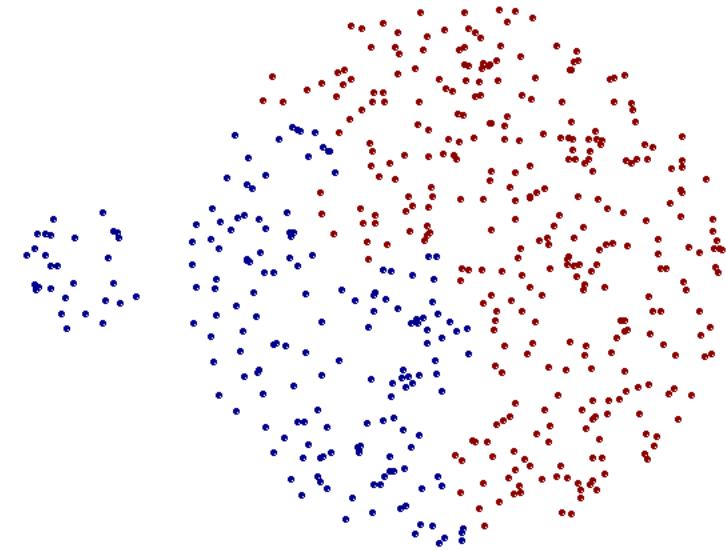
- Less susceptible to noise and outliers



# Limitations of MAX



Original Points



Two Clusters

- Tends to break large clusters
- Biased towards globular clusters



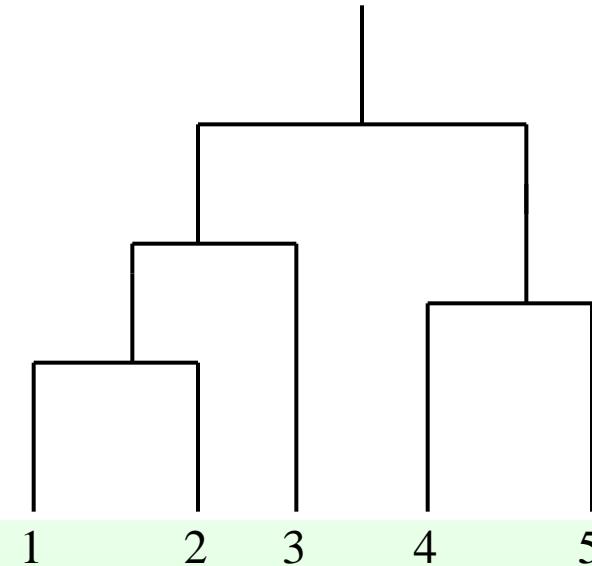
# Cluster Similarity: Group Average

- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

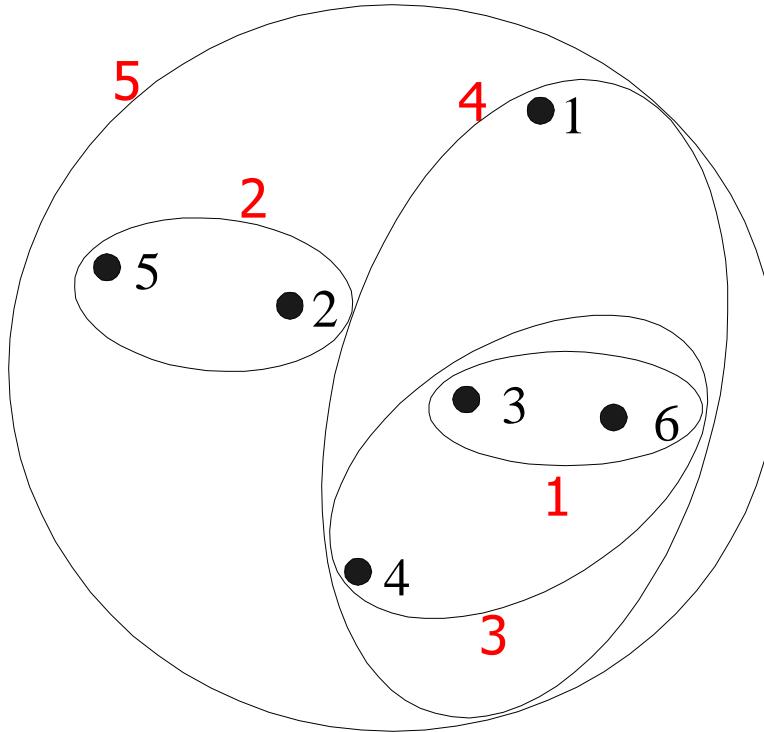
- Need to use average connectivity for scalability since total proximity favors large clusters

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

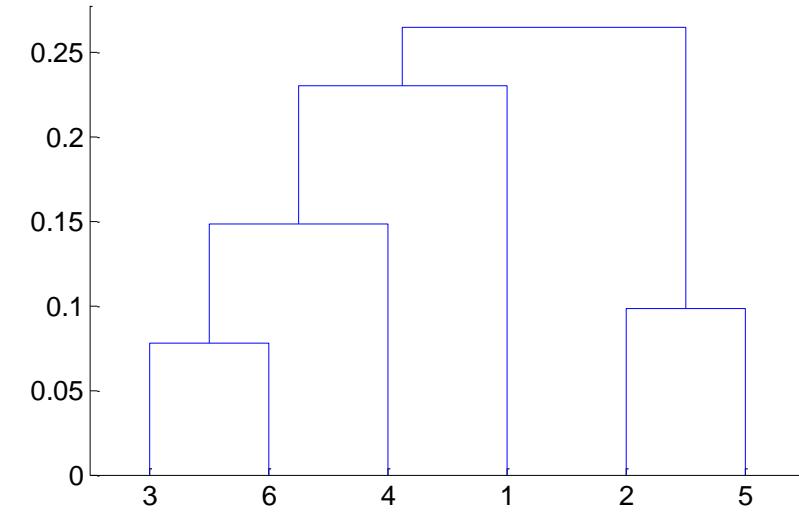




# Hierarchical Clustering: Group Average



Nested Clusters



Dendrogram



# Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise and outliers
- Limitations
  - Biased towards globular clusters

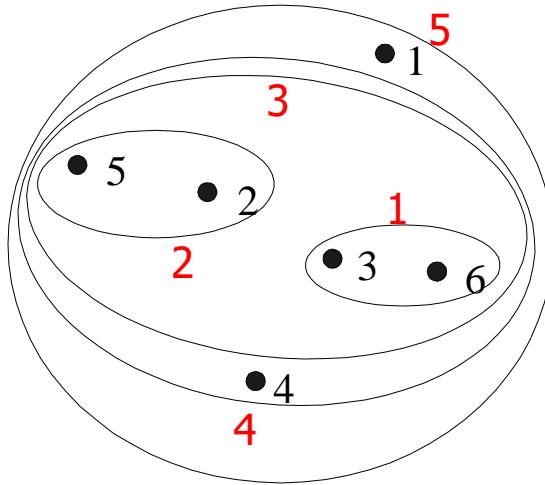


# Cluster Similarity: Ward's Method

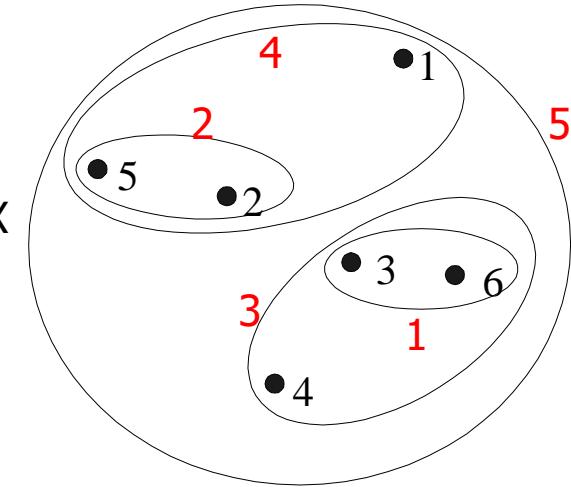
- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used to initialize K-means



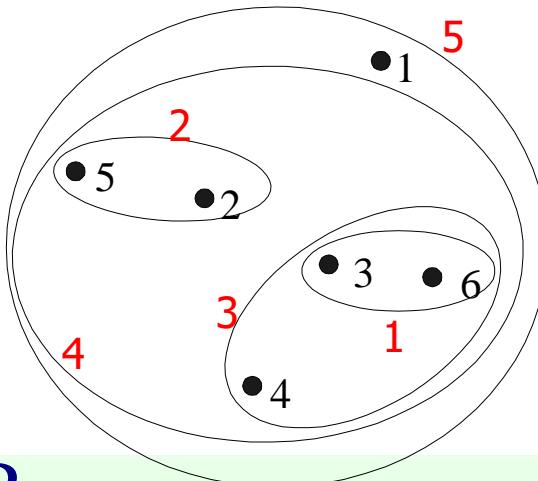
# Hierarchical Clustering: Comparison



MIN

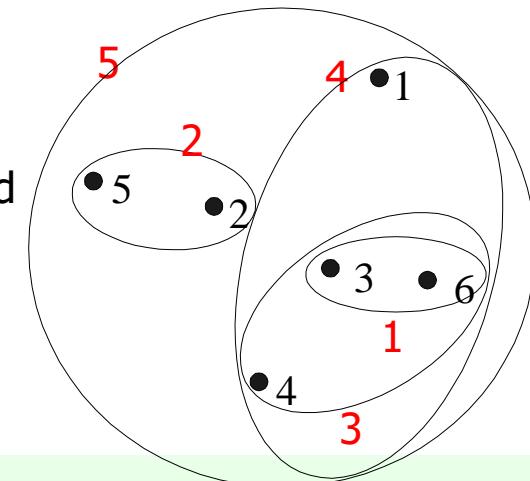


MAX



Group Average

Ward's Method





## Hierarchical Clustering: Time and Space requirements

- $O(N^2)$  space since it uses the proximity matrix.
  - N is the number of points.
- $O(N^3)$  time in many cases
  - There are N steps and at each step the size,  $N^2$ , proximity matrix must be updated and searched
  - Complexity can be reduced to  $O(N^2 \log(N))$  time for some approaches

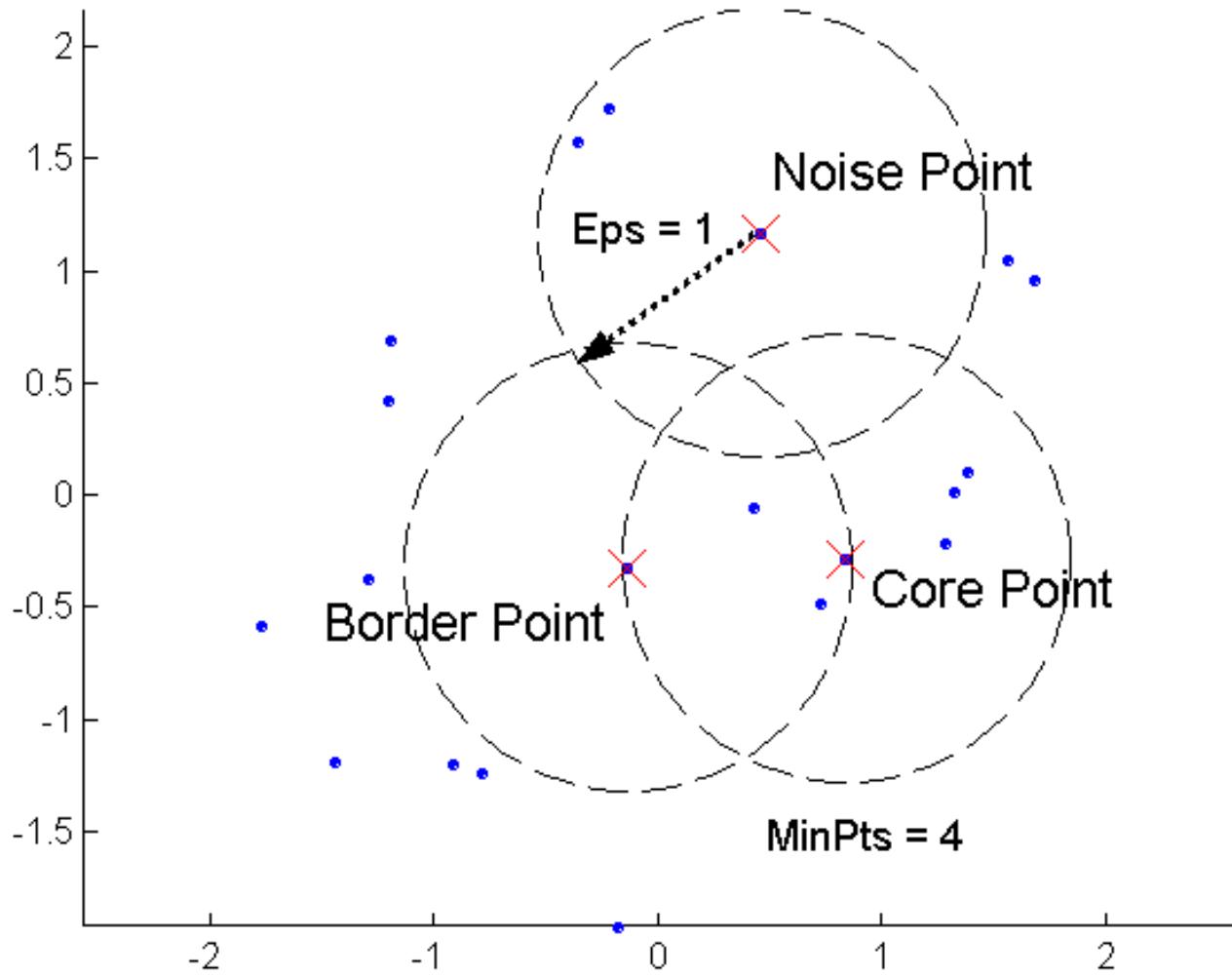


# DBSCAN

- DBSCAN is a density-based algorithm
  - Density = number of points within a specified radius (Eps)
  - A point is a **core point** if it has more than a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
  - A **border point** has fewer than MinPts within Eps, but is in the neighborhood of a core point
  - A **noise point** is any point that is not a core point or a border point.



# DBSCAN: Core, Border, and Noise Points





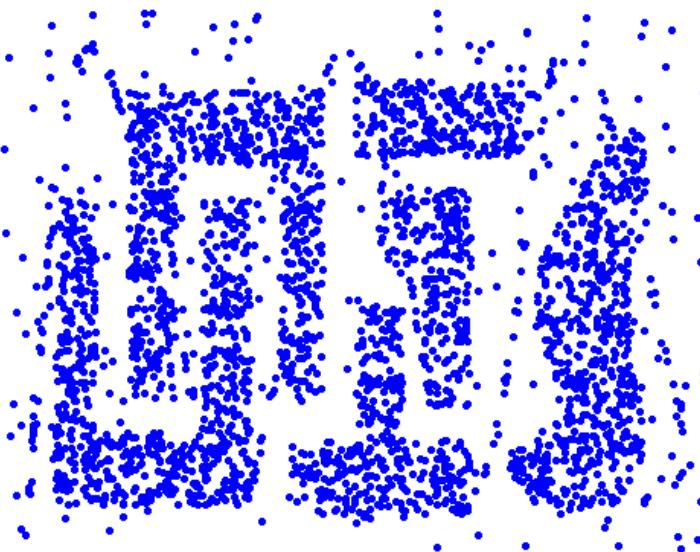
# DBSCAN Algorithm

- Eliminate noise points
- Perform clustering on the remaining points

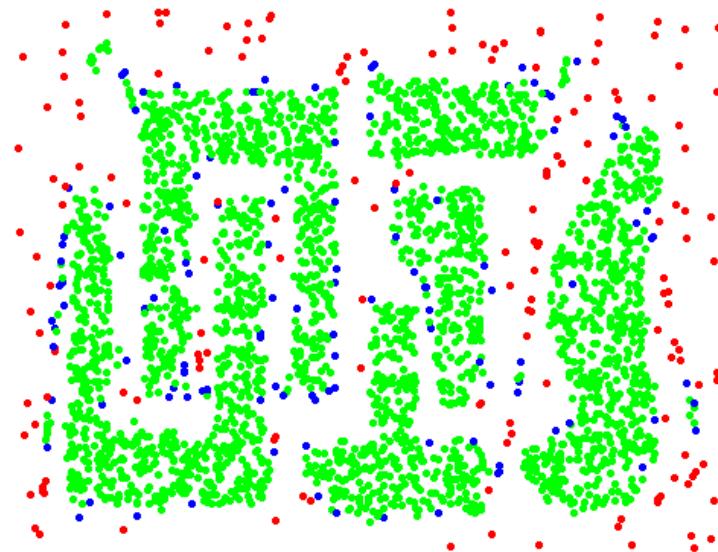
```
current_cluster_label ← 1
for all core points do
    if the core point has no cluster label then
        current_cluster_label ← current_cluster_label + 1
        Label the current core point with cluster label current_cluster_label
    end if
    for all points in the  $Eps$ -neighborhood, except  $i^{th}$  the point itself do
        if the point does not have a cluster label then
            Label the point with cluster label current_cluster_label
        end if
    end for
end for
```



# DBSCAN: Core, Border, and Noise Points



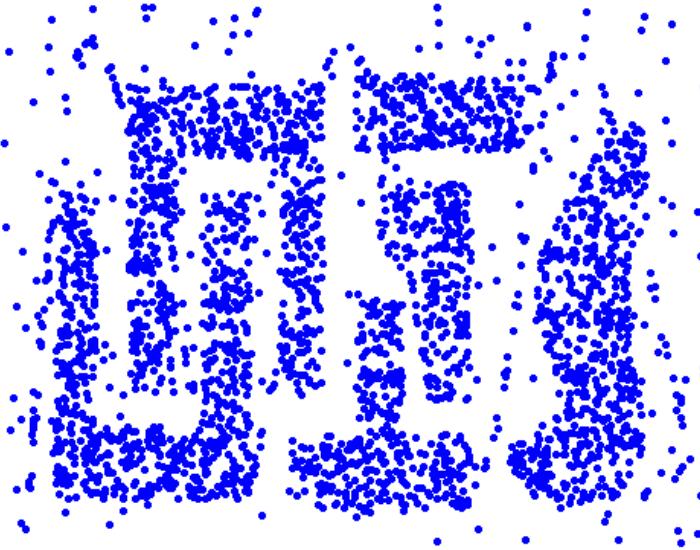
Original Points



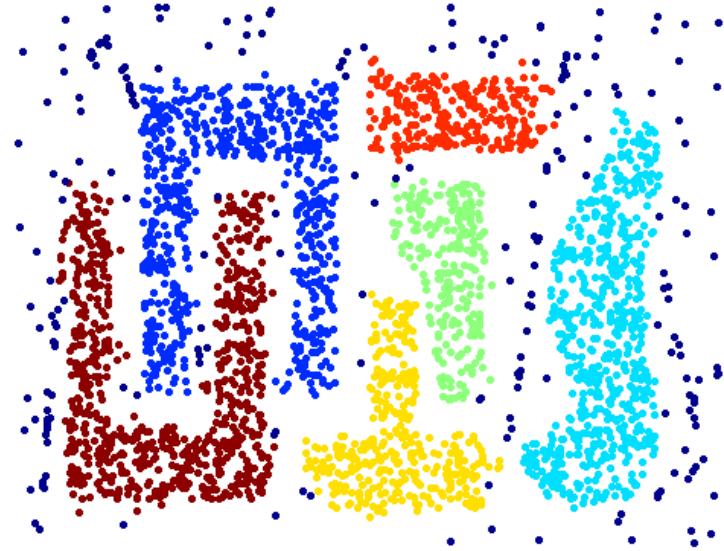
Point types: core,  
border and noise



# When DBSCAN Works Well



Original Points

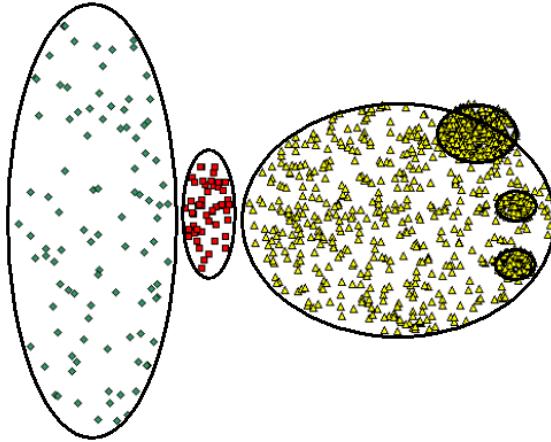


Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

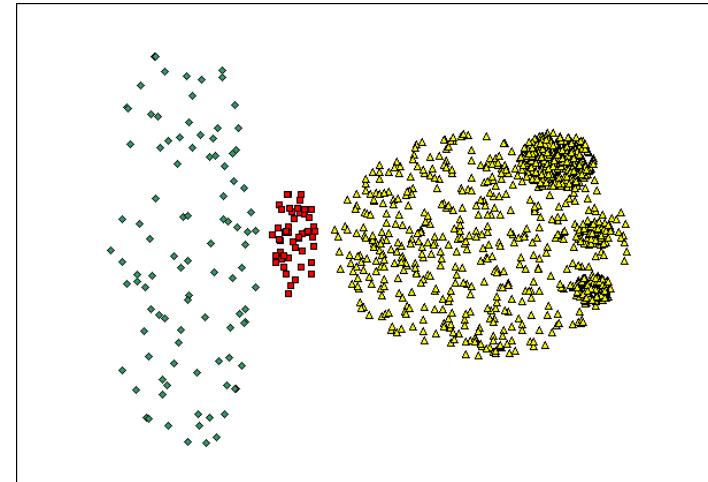


# When DBSCAN Does NOT Work Well

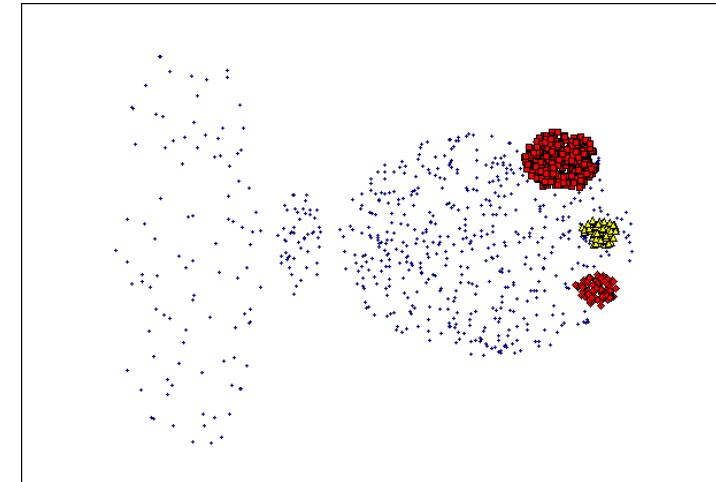


Original Points

- Varying densities
- High-dimensional data



(MinPts=4, Eps=9.75).



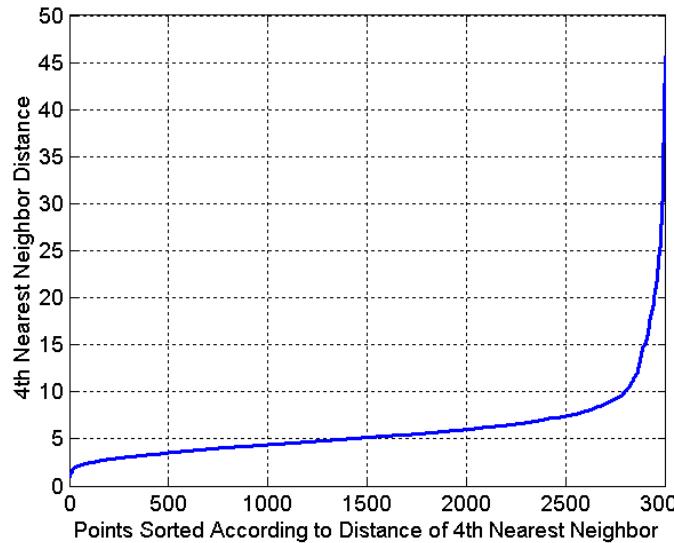
(MinPts=4, Eps=9.62)

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006



# DBSCAN: Determining EPS and MinPts

- Idea is that for points in a cluster, their  $k^{\text{th}}$  nearest neighbors are at roughly the same distance
- Noise points have the  $k^{\text{th}}$  nearest neighbor at farther distance
- So, plot sorted distance of every point to its  $k^{\text{th}}$  nearest neighbor



# Cluster Validity



Data Base and Data Mining Group of Politecnico di Torino

Elena Baralis, Tania Cerquitelli  
*Politecnico di Torino*



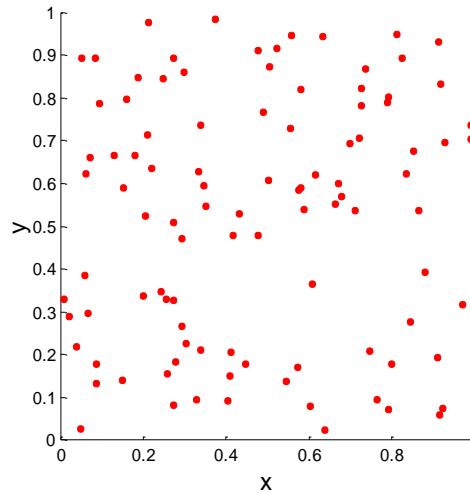
# Cluster Validity

- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall
- For cluster analysis, the analogous question is how to evaluate the “goodness” of the resulting clusters?
- But “clusters are in the eye of the beholder”!
- Then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

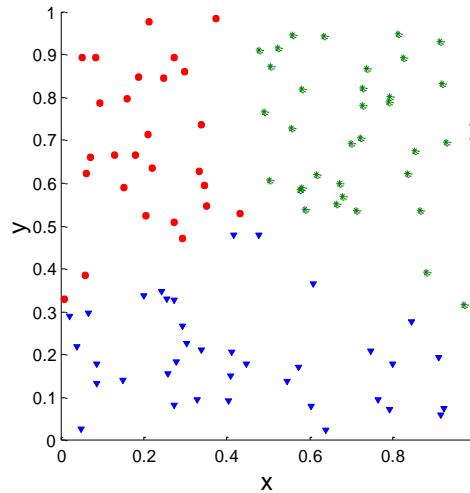


# Clusters found in Random Data

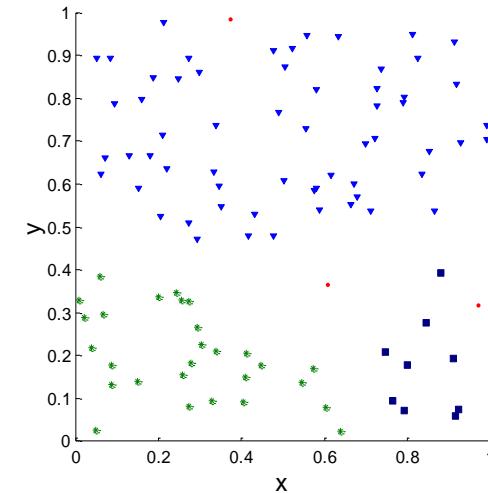
**Random Points**



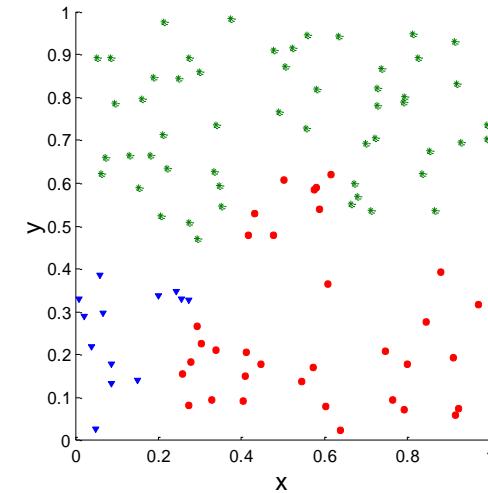
**K-means**



**DBSCAN**



**Complete Link**





# Different Aspects of Cluster Validation

1. Determining the **clustering tendency** of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.
2. Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.
3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
  - Use only the data
4. Comparing the results of two different sets of cluster analyses to determine which is better.
5. Determining the 'correct' number of clusters.

For 2, 3, and 4, we can further distinguish whether we want to evaluate the entire clustering or just individual clusters.



# Measures of Cluster Validity

- Numerical measures are applied to judge various aspects of cluster validity
- Numerical measures can be classified into three classes
  - **External Index:** Used to measure the extent to which cluster labels match externally supplied class labels.
    - e.g., entropy, purity
  - **Internal Index:** Used to measure the goodness of a clustering structure *without* respect to external information.
    - e.g., Sum of Squared Error (SSE), cluster cohesion, cluster separation, Rand-Index, adjusted rand-index, Silhouette index
  - **Relative Index:** Used to compare two different clusterings or clusters.
    - Often an external or internal index is used for this function, e.g., SSE or entropy



# Internal Measures: Cohesion and Separation

- **Cluster Cohesion:** Measures how closely related are objects in a cluster

- Cohesion is measured by the within cluster sum of squares (SSE)

$$WSS = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

- **Cluster Separation:** Measure how distinct or well-separated a cluster is from other clusters

- Separation is measured by the between cluster sum of squares

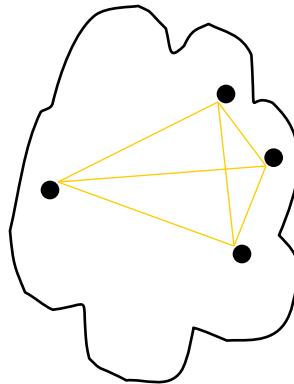
$$BSS = \sum_i |C_i| (m - m_i)^2$$

- Where  $|C_i|$  is the size of cluster i

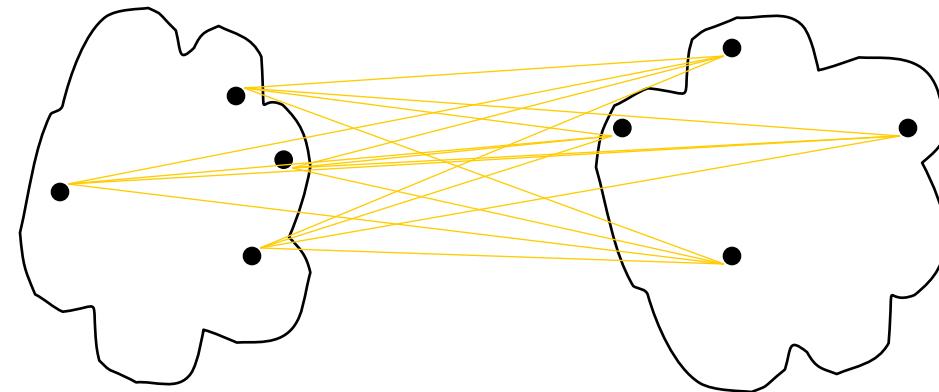


# Internal Measures: Cohesion and Separation

- A proximity graph based approach can also be used for cohesion and separation.
  - Cluster cohesion is the sum of the weight of all links within a cluster.
  - Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.



cohesion



separation



# Internal measures: Silhouette

- A succinct measure to evaluate how well each object lies within its cluster
- It is defined for single points
- It considers both cohesion and separation
- Can be computed for
  - Individual points
  - Individual clusters
  - Clustering result



# Internal measures: Silhouette

- For each object  $i$ 
  - $a(i)$ : the average dissimilarity of  $i$  with all other objects within the same cluster (the smaller the value, the better the assignment)
  - $b(i)$ :  $\min(\text{average dissimilarity of } i \text{ to any other cluster, of which } i \text{ is not a member})$
- Ranges between -1 and +1
  - Typically between 0 and 1
  - The closer to 1, the better
- Silhouette for clusters and clusterings
  - The average  $s(i)$  over all data of a *cluster* measures how tightly grouped all the data in the cluster are
  - The average  $s(i)$  over all data of the *dataset* measures how appropriately the data has been clustered



# External Measures of Cluster Validity: Entropy and Purity

**Table 5.9.** K-means Clustering Results for LA Document Data Set

Cluster	Entertainment	Financial	Foreign	Metro	National	Sports	Entropy	Purity
1	3	5	40	506	96	27	1.2270	0.7474
2	4	7	280	29	39	2	1.1472	0.7756
3	1	1	1	7	4	671	0.1813	0.9796
4	10	162	3	119	73	2	1.7487	0.4390
5	331	22	5	70	13	23	1.3976	0.7134
6	5	358	12	212	48	13	1.5523	0.5525
Total	354	555	341	943	273	738	1.1450	0.7203

**entropy** For each cluster, the class distribution of the data is calculated first, i.e., for cluster  $j$  we compute  $p_{ij}$ , the ‘probability’ that a member of cluster  $j$  belongs to class  $i$  as follows:  $p_{ij} = m_{ij}/m_j$ , where  $m_j$  is the number of values in cluster  $j$  and  $m_{ij}$  is the number of values of class  $i$  in cluster  $j$ . Then using this class distribution, the entropy of each cluster  $j$  is calculated using the standard formula  $e_j = \sum_{i=1}^L p_{ij} \log_2 p_{ij}$ , where the  $L$  is the number of classes. The total entropy for a set of clusters is calculated as the sum of the entropies of each cluster weighted by the size of each cluster, i.e.,  $e = \sum_{i=1}^K \frac{m_i}{m} e_j$ , where  $m_j$  is the size of cluster  $j$ ,  $K$  is the number of clusters, and  $m$  is the total number of data points.

**purity** Using the terminology derived for entropy, the purity of cluster  $j$ , is given by  $purity_j = \max p_{ij}$  and the overall purity of a clustering by  $purity = \sum_{i=1}^K \frac{m_i}{m} purity_j$ .



# Rand Index

- Idea
  - Any two objects that are in the same cluster should be in the same class and vice versa
- Given
  - $f_{00}$  = number of pairs of objects having a different class and a different cluster
  - $f_{01}$  = number of pairs of objects having a different class and the same cluster
  - $f_{10}$  = number of pairs of objects having the same class and a different cluster
  - $f_{11}$  = number of pairs of objects having the same class and the same cluster
- Rand Index

$$Rand\ Index = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}$$



# Final Comment on Cluster Validity

“The validation of clustering structures is the most difficult and frustrating part of cluster analysis.

Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage.”

*Algorithms for Clustering Data*, Jain and Dubes



POLITECNICO  
DI TORINO

# Data Science Lab

## Regression Analysis: Fundamentals

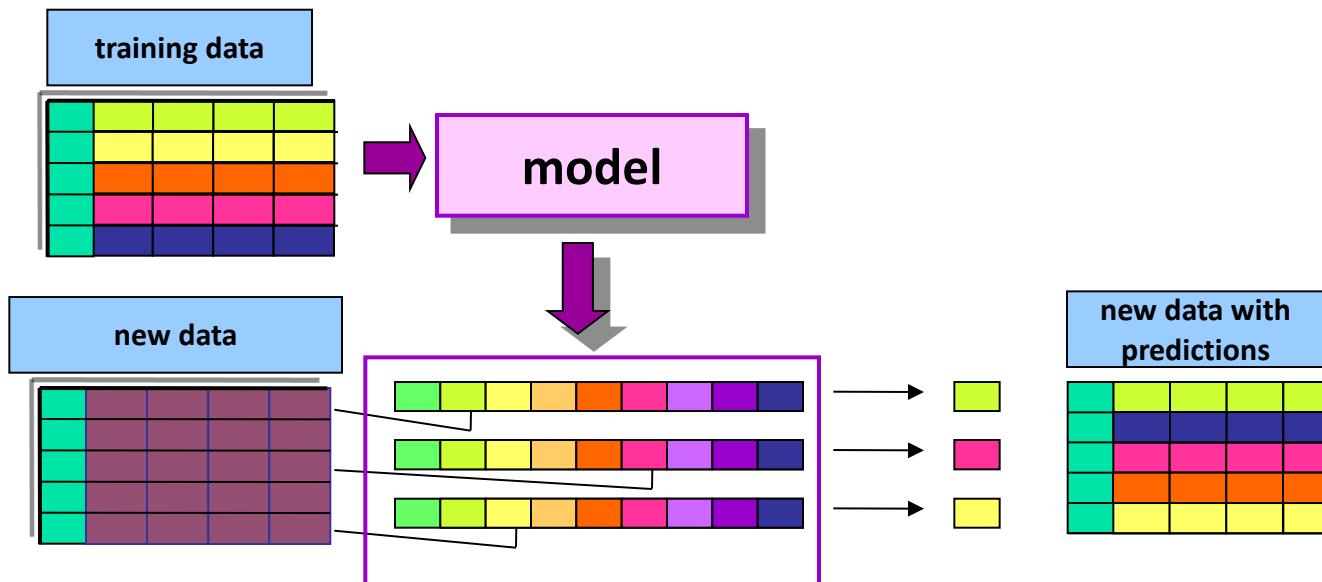
DataBase and Data Mining Group

Tania Cerquitelli and Elena Baralis

# Introduction to the regression analysis

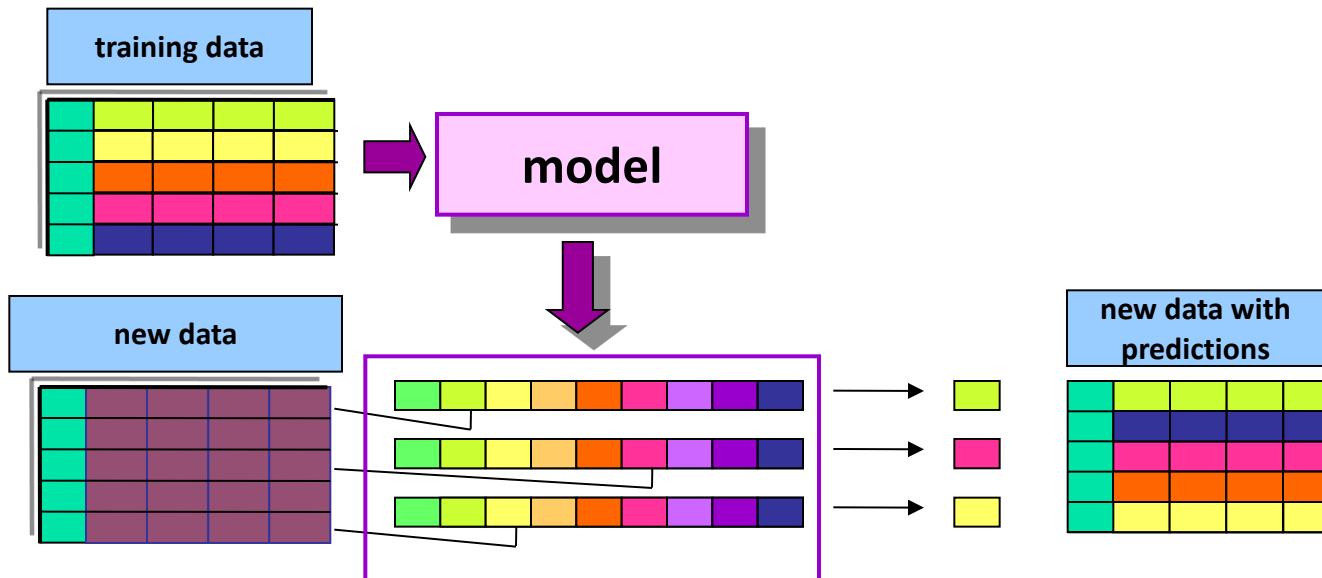
## Objectives

- Prediction of a **numerical target variable**
- Definition of a **model** of a given phenomenon



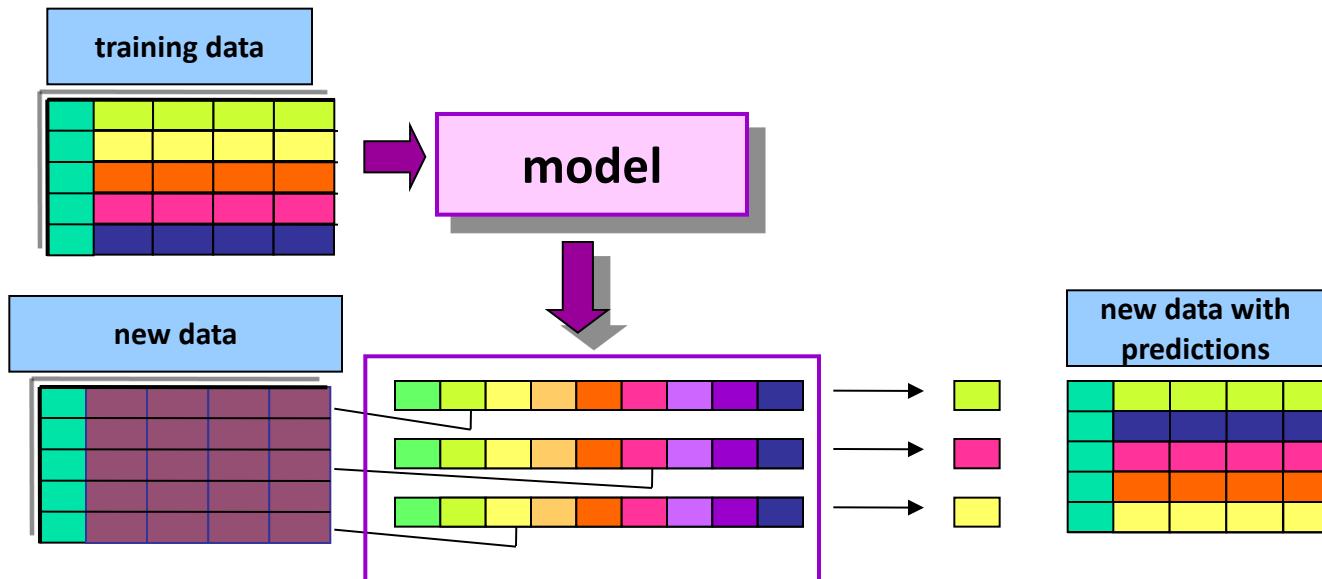
# Introduction to the regression analysis

- Approach discussed in this set of slides
  - **Linear regression**
  - **SVMs (SVR)**
- Other approaches
  - k-Nearest Neighbours
  - Decision trees
  - ..



# Introduction to the regression analysis

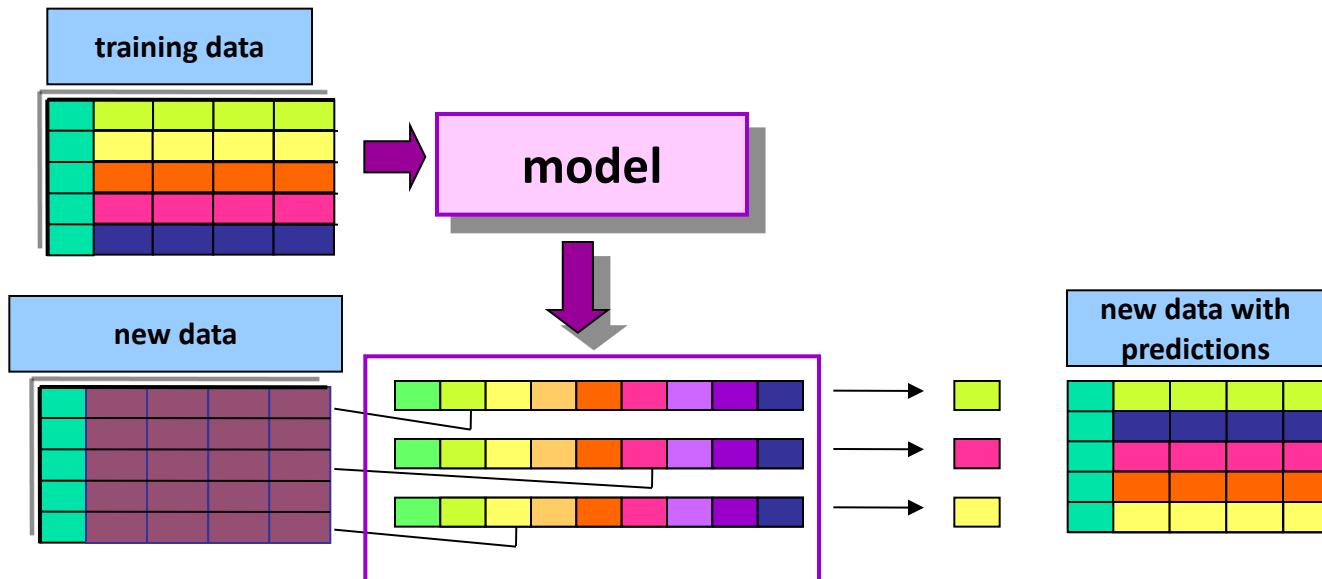
- Requirements
  - **accuracy**
  - interpretability
  - scalability
  - **noise and outlier management**



# Introduction to the regression analysis

## Applications

- Estimating the cost of a house
- Estimating the remaining useful life (RUL) of an industrial equipment
- Industrial Vehicle Usage Predictions
- Predicting the Number of Free Floating Car Sharing Vehicles within Urban Areas
- ...



- The term "regression" was coined by **Francis Galton** in 1877 to describe a biological phenomenon
  - the heights of descendants of tall ancestors tend to regress down towards a normal average (i.e., regression toward the mean)
- Father of regression **Carl F. Gauss** (1777–1855)

- Given
  - A numerical target attribute
  - A collection of data objects also characterized by the target attribute
- The regression task finds a model that allows predicting the target variable value of new objects through
  - $y=f(x_1, x_2, \dots x_n)$

- Regression analysis can be classified based on
  - **Number of explanatory variables**
    - Simple regression: single explanatory variable
    - Multiple regression: includes any number of explanatory variables
  - **Types of relationship**
    - Linear regression: straight-line relationship
    - Non-linear: implies curved relationships (e.g., logarithmic relationships)
  - **Temporal dimension**
    - Cross Sectional: data gathered from the same time period
    - Time Series: involves data observed over equally spaced points in time

$$y = \beta_0 + \beta_1 x$$

- The regression line provides an **interpretable model** of the phenomenon under analysis
  - $y$ : **estimated** (or predicted) **value**
  - $\beta_0$ : estimation of the **regression intercept**
    - The intercept represents the estimated value of  $y$  when  $x$  assumes 0
  - $\beta_1$ : estimation of the **regression slope**
  - $x$ : **independent variable**

$$y = \beta_0 + \beta_1 x$$

- *Least squares method*
  - $\beta_0$  and  $\beta_1$  can be obtained by **minimizing the Residual sum of squares (RSS)** that is the sum of the squared residuals
    - differences between actual values ( $y$ ) and estimated ones ( $\hat{y}$ )

$$\min RSS = \min \sum_i (y_i - \hat{y}_i)^2 =$$

$$\min \sum_i (y_i - (\beta_0 + \beta_1 x_i))^2$$

# Estimation of the parameters by least squares

$$y = \beta_0 + \beta_1 x$$

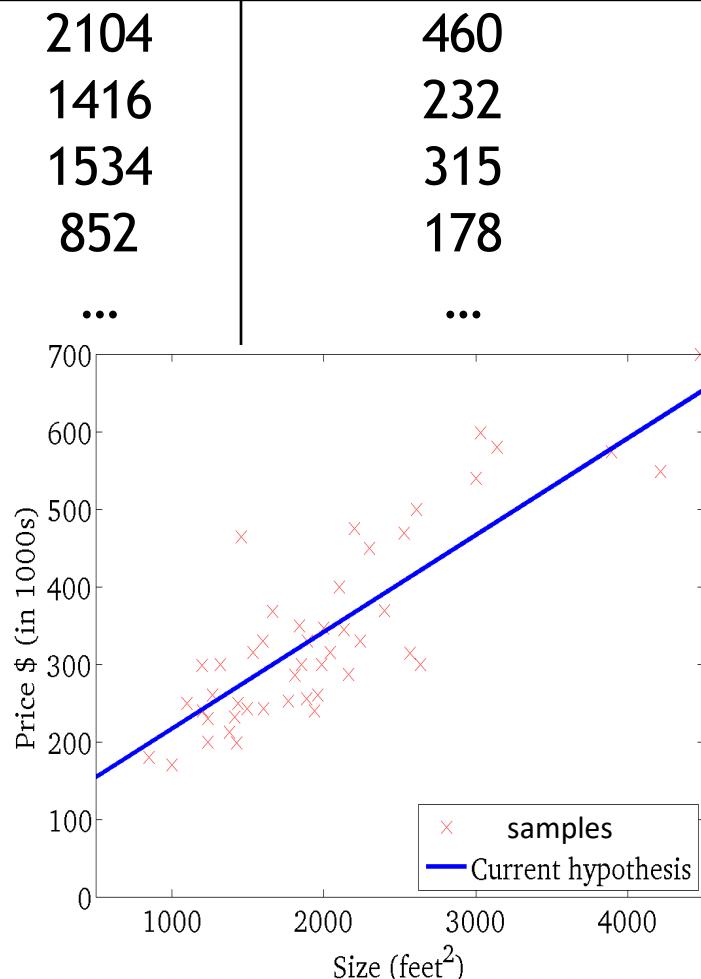
$$\beta_1 = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

- where  $\bar{y} = \frac{1}{n} \sum_i y_i$  and  $\bar{x} = \frac{1}{n} \sum_i x_i$  are the sample means

# Simple linear regression: example

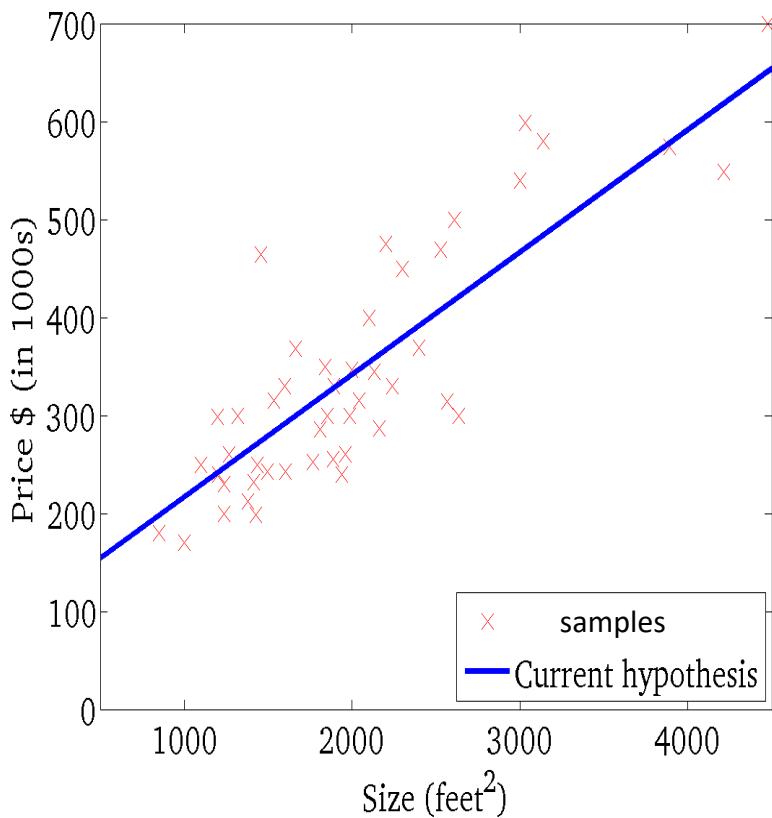
Size in feet <sup>2</sup>	Price (\$) in 1000's
---------------------------	----------------------



- Goal of a **real estate agency**
  - Estimate the selling price of a home based on the value of size in square feet
- Simple linear regression finds a **linear model** of the problem
  - $x$  = Size in feet<sup>2</sup>
  - $y$  = Price (\$) in 1000's

$$y = \beta_0 + \beta_1 x$$

# Simple linear regression: example



- $\beta_0$ : The **intercept** represents the estimated value of  $y$  when  $x$  assumes 0
  - No house had 0 square feet, but  $\beta_0$  is the portion of house price not explained by square feet
- $\beta_1$ : the **slope** measures the estimated change in the  $y$  value as for every one-unit change in  $x$ 
  - The average value of a square foot of size

$$y = f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots + \beta_n x_n + \xi$$

- **Dependant variable ( $y$ ):** the single variable being explained/predicted by the regression model
- **Independent or explanatory variables ( $x_i$ ):** The variables used to predict/explain the dependant variable
- **Coefficients ( $\beta_i$ ):** values, computed by the regression task, reflecting explanatory to dependent variable relationships
- **Residuals ( $\xi$ ):** the portion of the dependent variable that is not explained by the model
  - The model performs under or over predictions

# Interpreting regression coefficients

- Uncorrelated predictors
  - Each coefficient can be estimated and tested separately
  - Interpretation: a unit change in  $x_i$  is associated with a  $\beta_i$  change in  $y$ , while all the other variables stay fixed
    - $\beta_i$  represents the average effect on  $y$  of a one unit increase in  $x_i$ , holding all other predictors fixed
- Correlation among predictors cause problems
  - The variance of all coefficients tends to increase, sometimes dramatically
  - Interpretations become complex: when  $x_j$  changes, everything else changes
- The claim of causality should be avoided for the observational data

- In case of a high dimensional data set, in terms of number of dependent variables, **some of the variables** might provide **redundant information**.
- Feature selection and removal (correlation-based approach)
  - simplifying the model computation
  - improving the model performance
  - Enhancing the model interpretation (i.e., better explainability of the dependent variables)
- Variable/feature selection
  - Driven by the business understanding and domain knowledge
  - Feature selection based on correlation test
    - Features highly-correlated with other attributes could be discarded from the analysis
    - having dependence or association in any statistical relationship, whether causal or not

# Polynomial regression

- The polynomial models can be used in those situations where the **relationship** between dependent and explanatory variables is **curvilinear**.
- Polynomial regression consists of:
  - Computing new **features** that are power functions of the input features
  - Applying **linear** regression on these new features

$$y = \beta + \beta_1 x + \beta_2 x^2 + \varepsilon$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \beta_{12} x_1 x_2 + \varepsilon$$

- The above models are also linear (i.e., A model is linear when it is linear in parameters)
- They are the second order polynomials in one and two variables respectively.
- Sometimes a nonlinear relationship in a small range of explanatory variables can also be modeled by polynomials.

# Polynomial model in one variable

- The  $k^{\text{th}}$  order polynomial model in one variable is given by

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_k x^k + \varepsilon$$

- It is included in the linear regression model below

$$y = X\beta + \varepsilon$$

- Techniques for fitting linear model can be used for fitting the polynomial regression model

- For example,  $y = \beta_0 + \beta_1 x + \beta_2 x^2$

- Is a polynomial regression model in one variable and is called as **second order model** or **quadratic model**, where the coefficients

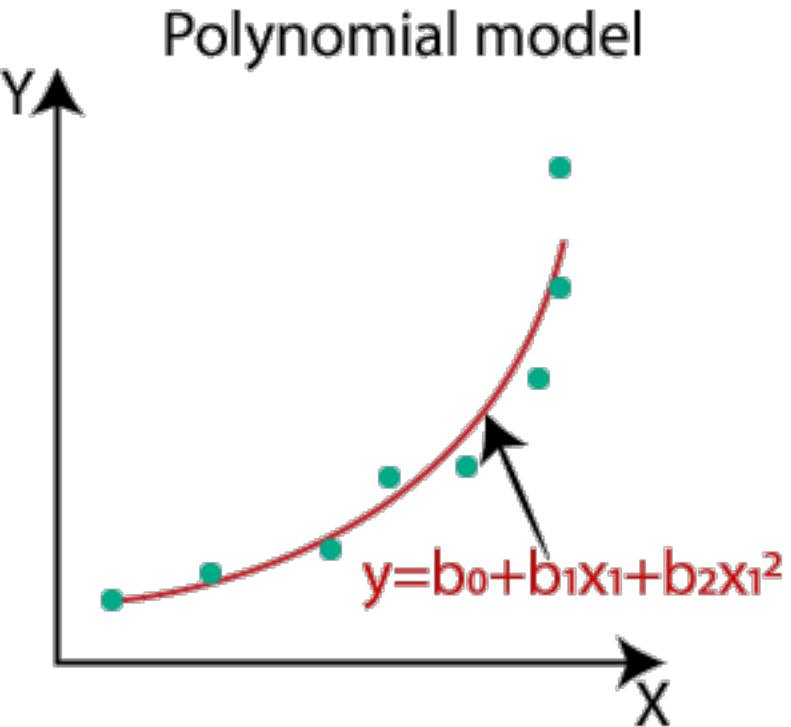
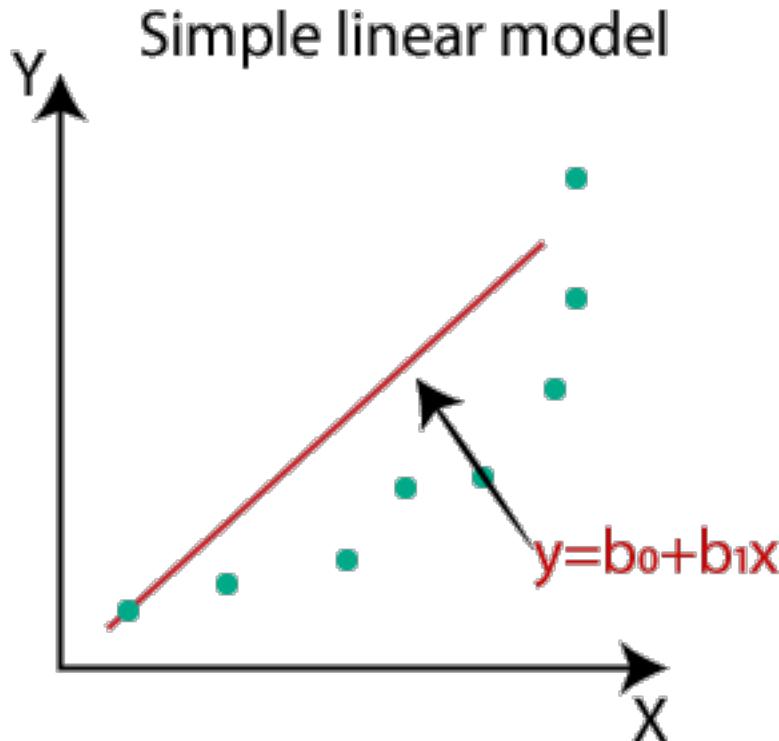
- $\beta_1$  is the linear effect parameter
    - $\beta_2$  is the quadratic effect parameter

- The polynomial models can be used to approximate a complex nonlinear relationship

# Polynomial model in one variable

Example.

Second order model or quadratic model



# Polynomial regression: considerations in case of one variable

- Order of the model
  - Keep the order of the polynomial model as low as possible
    - Up to the **second order** polynomial
    - If necessary, you should apply some **data transformations**
  - Arbitrary fitting of higher order polynomials can be a serious abuse of regression analysis.
    - Data overfitting issue
- Different model building strategies do not necessarily lead to the same model
  - **Forward selection procedure:** to successively fit the models in increasing order and test the significance of regression coefficients at each step of model fitting.
    - Keep the order increasing until t-test for the highest order term is nonsignificant
    - The significance of highest order term is tested through the null hypothesis
  - **Backward elimination:** to fit the appropriate highest order model and then delete terms one at a time starting with highest order. This is continued until the highest order remaining term has a significant t-test
- The first and second order polynomials are mostly used in practice.

# Polynomial models in two or more variables

- The techniques of fitting of polynomial model in one variable can be extended to fitting of polynomial models in two or more variables.
- A second order polynomial is more used in practice and its model is specified by

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \beta_{12} x_1 x_2 + \varepsilon$$

- This is also called **response surface**.

# Strong and weak points of Polynomial Regression

- Advantages of using Polynomial Regression:
  - **Broad range of function** can be fit under it.
  - Polynomial basically fits **wide range of curvature**.
  - Polynomial usually provides the **best approximation** of the relationship between dependent and independent variable.
- Disadvantages of using Polynomial Regression
  - They are **too sensitive to the outliers**.
    - The presence of a few outliers in the data can seriously affect the results of a nonlinear analysis.
  - Higher polynomial degree means **higher flexibility** of your model, but also **data overfitting**
    - Overfitting occurs in those cases when you have a few samples and a model that has high flexibility
    - It is always possible for a polynomial of order  $(n-1)$  to pass through  $n$  points so that a polynomial of sufficiently high degree can always be found that provides a “good” fit to the data.
    - Those models **never enhance the understanding** of the unknown function and they are **never good predictors**.

# To avoid data overfitting

- Use more training data (if possible)
- Use lower model complexity
- Use regularization techniques
  - e.g., Ridge and Lasso

- Regression analysis methods that perform both **variable selection** and **regularization** in order to enhance the prediction **accuracy** and **interpretability** of the statistical model it produces.
- Useful **to reduce model complexity** and **prevent overfitting** when
  - The number of variables describing each observation exceeds the number of observations
  - The number of variables does not exceed the number of observations, but the learned model suffers from poor generalization.
- Techniques of training a linear regression (or a linear regression with polynomial features)
  - They try to assign values **closer to zero (RIDGE)** or **zero (LASSO)** to the coefficients assigned to features that are not useful for the regression
  - The effect is the **decreasing of the complexity of the model**

# Regularization: RIDGE and LASSO

## Cost function

### Linear regression

$$RSS = \sum_i (y_i - \hat{y}_i)^2 = \sum_i (y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij})^2$$

### Ridge regression

$$RSS + \lambda \sum_{j=1}^p \beta_j^2$$

### Lasso regression

$$RSS + \lambda \sum_{j=1}^p |\beta_j|$$

Penalty term  $\lambda \rightarrow$  amount of shrinkage (or constraint)

## Ridge regression

- It adds L2 as the penalty
- L2 is the sum of the square of the magnitude of beta coefficients

$$RSS + \lambda \sum_{j=1}^p \beta_j^2$$

This is equivalent to minimizing the RSS under the condition

$$\text{For } c > 0, \sum_{j=1}^p \beta_j^2 < c$$

- Penalty term  $\lambda \rightarrow$  amount of shrinkage (or constraint)
  - Regularizes the coefficients, penalizing coefficients taking large values

- LASSO means **Least Absolute Shrinkage and Selection Operator**
- Term coined by Robert Tibshirani in 1996, but it was originally introduced in geophysics literature 10 years before
- Lasso **regularization** was originally defined for **least squares**, but it is easily extended to a wide variety of statistical models in a straightforward fashion
  - E.g., generalized linear models
- The Lasso's **variable selection** relies on the form of the **constraint**
  - It forces the sum of the absolute value of the regression coefficients to be less than a fixed constraint, which forces some coefficients to be set to zero
  - The selected model is simpler since it does not include coefficients set to zero.
- It is similar to RIDGE regression but usually identifier a simpler model
  - **RIDGE** simplifies the model by shrinking the size of some coefficients, while **LASSO** sets some coefficients to zero.

## Lasso regression

- It adds L1 the penalty
- L1 is the sum of the absolute value of the beta coefficients

$$RSS + \lambda \sum_{j=1}^p |\beta_j|$$

This is equivalent to minimizing the RSS under the condition

$$\text{For } c > 0, \sum_{j=1}^p |\beta_j| < c$$

The regularization (L1) can lead to zero coefficients

- i.e. some of the features are completely neglected for the evaluation  
It not only helps in reducing overfitting but also in feature selection

# Simple linear regression vs Support Vector Regression

Recall that for linear regression, the parameters and the model can be derived by **minimizing the Residual sum of squares (RSS)**

$$\min RSS = \min \sum_i (y_i - \hat{y}_i)^2 =$$

We can instead be interested in reducing error to a certain degree

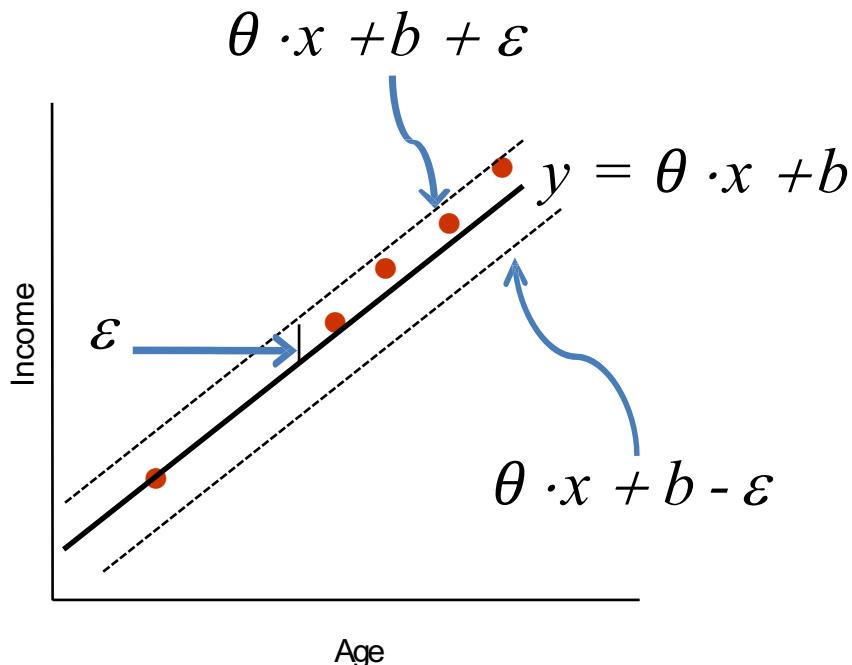
- errors within an acceptable range

## Support Vector Regression

- define how much error is acceptable in our model
- find an appropriate hyperplane to fit the data

# Support Vector Machine – Regression

- Find a function,  $f(x)$ , that performs a prediction of the target attribute  $y$  with a maximum error equal to  $\varepsilon$



We do not care about errors as long as they are less than  $\varepsilon$

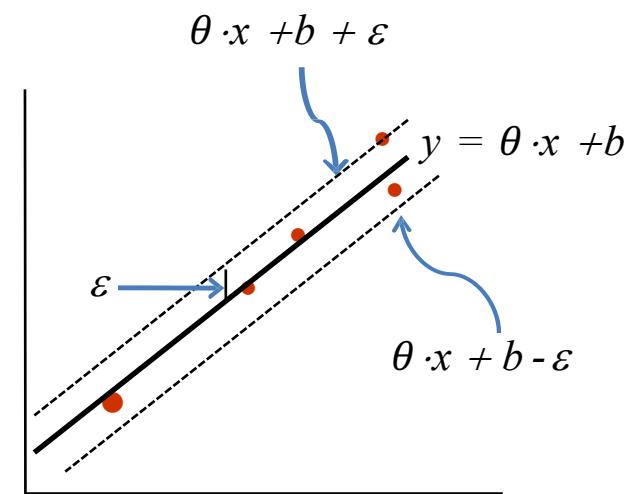
# Support Vector Regression: linear model

- The (training) problem can be formulated as a convex optimization problem

$$\min \frac{1}{2} \|\theta\|^2$$

$$s.t. \quad y^i - \theta \cdot x^i - b \leq \varepsilon; \\ \theta \cdot x^i + b - y^i \leq \varepsilon$$

Constraints



$y^i$  = value of the target attribute of the  $i^{\text{th}}$  training object

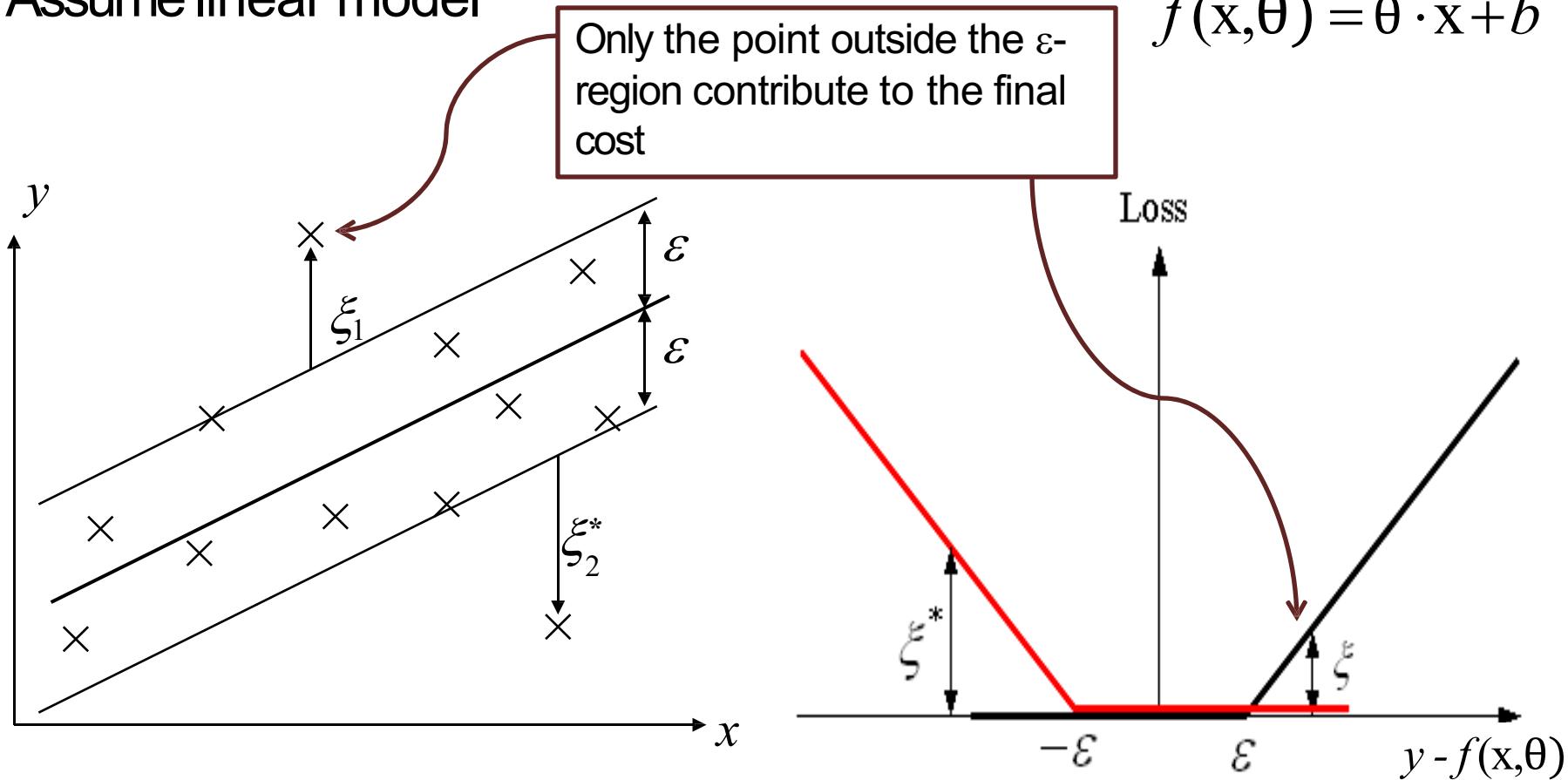
$x^i$  = value of the predictive attributes of the  $i^{\text{th}}$  training object

$\theta$  and  $b$  = parameter of the regression model

- Given a specific value of  $\varepsilon$ , the problem is not always feasible
- **Soft margin**
  - Reformulate the problem by considering the errors related to the predictions that do not satisfy the  **$\varepsilon$  maximum distance**

# Support Vector Regression: Soft margin

Assume linear model



For any value that falls outside of  $\varepsilon$ , we can denote its deviation from the margin as  $\xi$

# Support Vector Regression: Soft margin

- The (training) problem can be formulated as a convex optimization problem

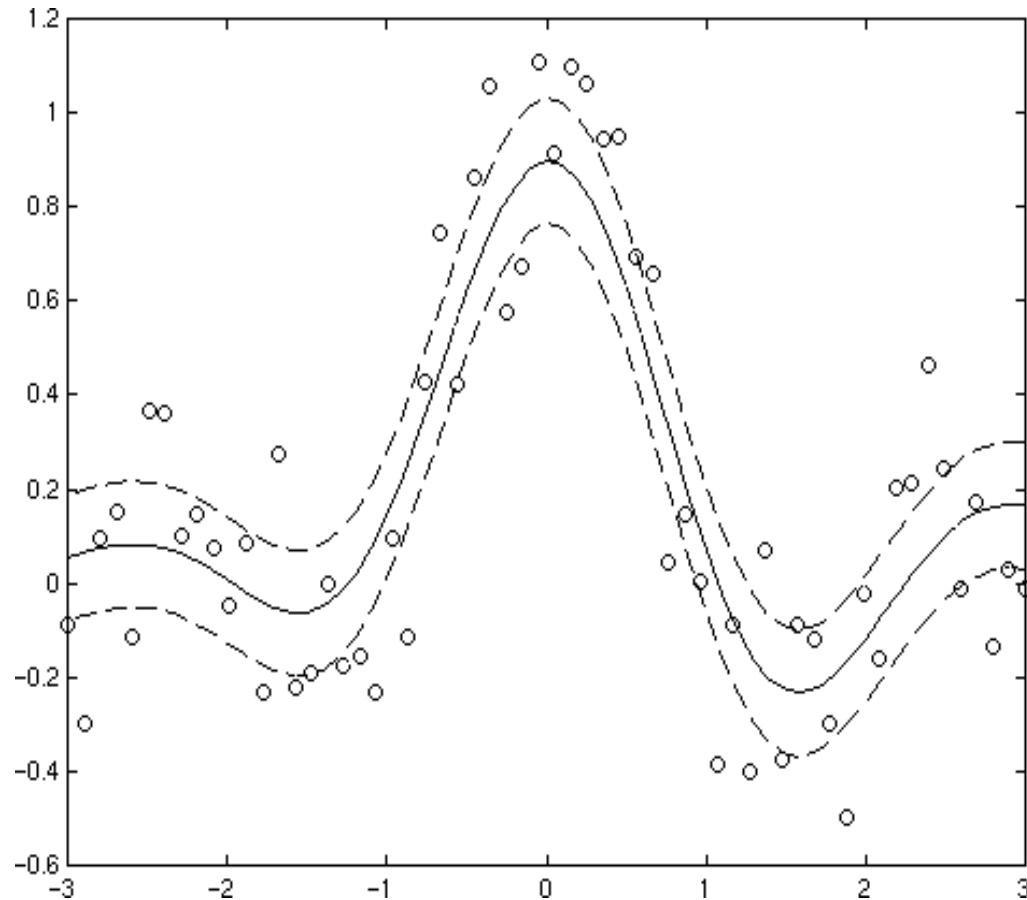
$$\min \frac{1}{2} \|\theta\|^2 + C \sum_{i=1}^m (\xi_i + \xi_i^*)$$

$$\begin{aligned}s.t. \quad & y^i - \theta \cdot x^i - b \leq \varepsilon + \xi_i; \\& \theta \cdot x^i + b - y^i \leq \varepsilon + \xi_i^* \\& \xi_i, \xi_i^* \geq 0, i = 1, \dots, m\end{aligned}$$

We minimize the deviation  $\xi$  from the margin  
C: additional hyperparameter.

- As C increases, also the tolerance for points outside of  $\varepsilon$  increases

# How about a non-linear case?



- Map the original features into a higher order dimensional space
- Apply a kernel transformation
  - Polynomial
  - Gaussian radial
  - ...
- Transform the input data by means of the kernel function  $\varphi$  and then solve the previous problem

# Linear versus Non-linear SVR

- $\phi$  maps the input data into a new dimensional space

$$\min \frac{1}{2} \|\theta\|^2 + C \sum_{i=1}^m (\xi_i + \xi_i^*)$$

$$s.t. \quad y^i - \theta \cdot \phi(x^i) - b \leq \varepsilon + \xi_i;$$

$$\theta \cdot \phi(x^i) + b - y^i \leq \varepsilon + \xi_i$$

$$\xi_i, \xi_i^* \geq 0, i = 1, \dots, m$$

- Evaluation metrics for regression:
  - MAE (Mean Absolute Error)
  - MSE (Mean Squared Error)
  - RSE: Residual Standard Error
  - $R^2$
  - Adjusted  $R^2$
- The evaluation is performed by comparing
  - $y$ : the actual value (**ground truth**)
  - $\hat{y}$ : the predicted value through the regression model

# Evaluating regression

- MAE (Mean Absolute Error)
  - the average vertical distance between each real value and the predicted one

$$MAE = \frac{1}{n} \sum_i |y_i - \hat{y}_i|$$

- MSE (Mean Squared Error)
  - the average of the squares of the errors
  - the average squared difference between the estimated values and the actual value.
  - MSE tends to penalize less errors close to 0

$$MSE = \frac{1}{n} \sum_i (y_i - \hat{y}_i)^2$$

- MAE and MSE always  $> 0$ 
  - The lower the values of MAE and MSE the better the model
  - It is mainly affected by the domains of data sample

- Overall accuracy of the model

- RSE: Residual Standard Error

$$RSE = \sqrt{\frac{1}{n-2} RSS} = \sqrt{\frac{1}{n-2} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

- n is the number of samples
  - RSS is the residual sum of squares
- RSE is always greater than 0
  - The lower the RSE value the better the regression model

# Evaluating regression

- $R^2$ : R-squared measures the goodness of fit of a model
  - how well the regression predictions approximate the real data points.
  - It estimates a normalized error

$$R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}$$

- RSS is the residual sum of squares  $RSS = \sum_i (y_i - \hat{y}_i)^2$
- TSS is the total sum of squares  $TSS = \sum_i (y_i - \bar{y}_i)^2$   
with  $\bar{y} = \frac{1}{n} \sum_i y_i$

# Evaluating regression: R<sup>2</sup>

$$R^2 = 1 - \frac{RSS}{TSS} = 1 - FVU$$

$$= 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y}_i)^2} = 1 - \frac{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}{\frac{1}{n} \sum_{i=1}^n (y_i - \bar{y}_i)^2} = 1 - \frac{MSE}{\sigma^2}$$

- R<sup>2</sup> represents the proportion of variance of y explained by variation in x
  - FVU means the fraction of variance unexplained
    - Ratio between the unexplained variance (variance of the model's errors) and the total variance

# Evaluating regression: $R^2$

- $R^2$  value

- $R^2 = 1$ 
  - A perfect linear relationship between x and y
  - 100% of the Y variation is explained by variation in x
- $R^2$  close to 1
  - A very good linear relationship between x and y
  - Good predictions
- $0 < R^2 << 1$ 
  - Weaker linear relationship between x and y
  - A portion of the variation in y is not explained by variation in x
- $R^2 = 0$ 
  - No linear relationship between x and y
  - The value of y does not depend on the value of x
- $R^2 < 0$ 
  - the model is predicting worse than the mean of the target values

# Evaluating regression: R<sup>2</sup> adjusted

- Drawback of R<sup>2</sup>
  - In the context of multiple linear regression, if new predictors ( $X_i$ ) are added to the model, R<sup>2</sup> only increases or remains constant but it never decreases.
  - However, it is not always true that by increasing the complexity of regression model, the latter will be more accurate
- The Adjusted R-Squared is the modified form of R-Squared that has been adjusted to incorporate model's degree of freedom.
- It should be used to evaluate the quality of a multiple linear regression model

$$\bar{R}^2 = 1 - (1 - R^2) \frac{n-1}{n-p-1}$$

- p = number of explanatory variables
- n = number of samples
- The adjusted R-Squared only increases if the new term improves the model accuracy.



POLITECNICO  
DI TORINO

# Data Science Lab

Time series analysis: fundamentals

DataBase and Data Mining Group

Tania Cerquitelli and Elena Baralis

- A time series is a **sequential** set of data points, measured typically over successive times
- A time series containing values of a single variable is termed as **univariate**. But if values of more than one variable are considered, it is termed as **multivariate**
- A time series can be **continuous** or **discrete**.
  - **continuous** time series: observations are measured at every instance of time
    - e.g., temperature readings, flow of a river
  - **discrete** time series: observations are measured at discrete points of time
    - e.g., city population, production of a company, exchange rates

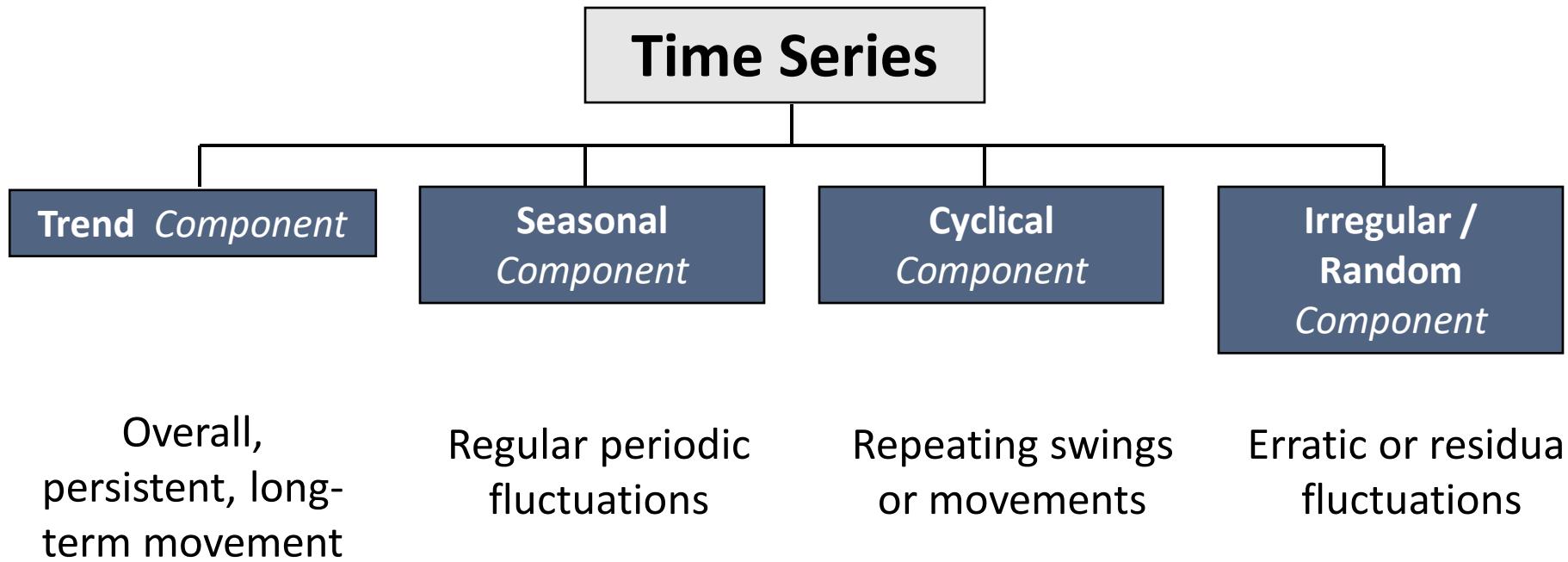
# A time series plot

- A time-series plot (time plot) is a two-dimensional plot of time series data
  - the vertical axis measures the **variable of interest**
  - the horizontal axis corresponds to the **time** periods

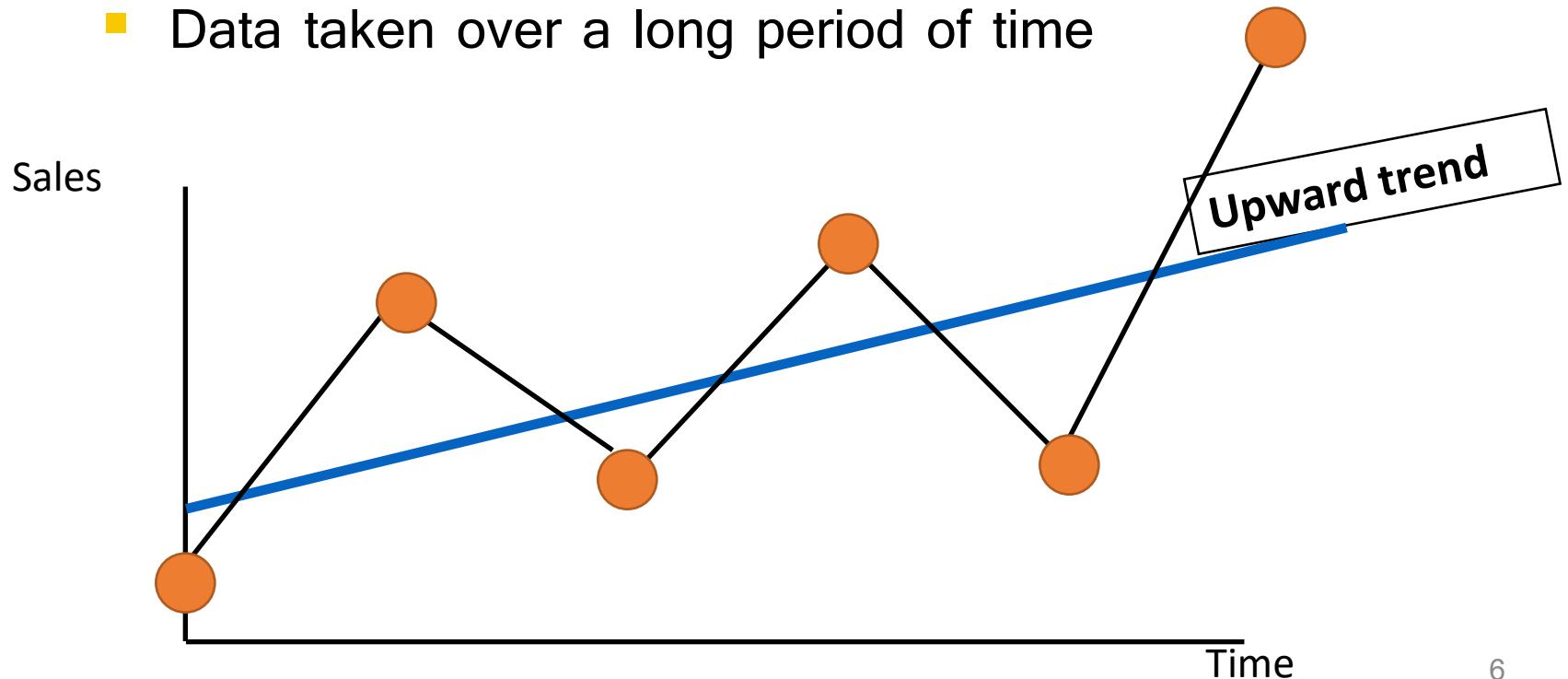


- Two main kinds of analysis can be performed on time series
  - Characterizing the nature of the phenomenon represented by the sequence of observations,
    - Time series components
  - Classification task vs. forecasting future values
    - Classification
      - speech recognition
      - classification of machine failures
      - ...
    - Forecasting
      - energy demand prediction
      - weather forecasting
      - traffic prediction
      - ...

# Time Series Components

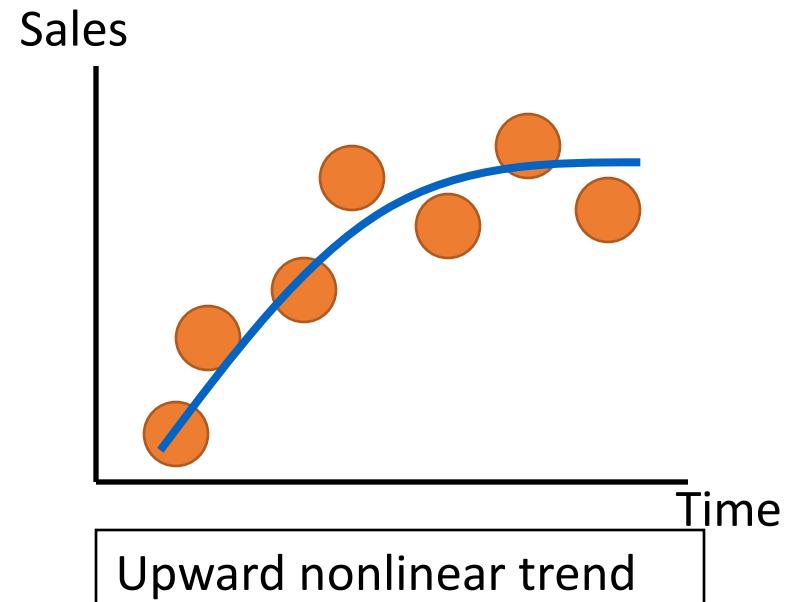
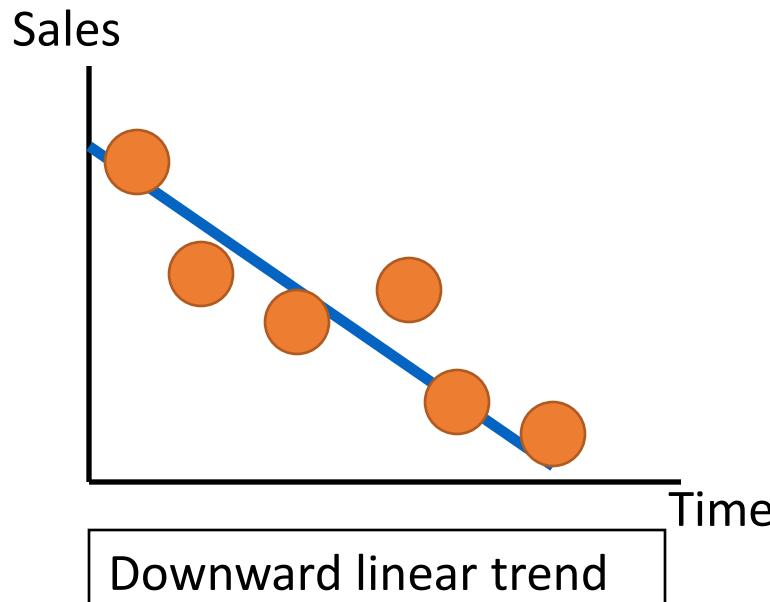


- Overall, persistent, long-term movement
  - increasing or decreasing over time
  - overall upward or downward movement
  - Data taken over a long period of time



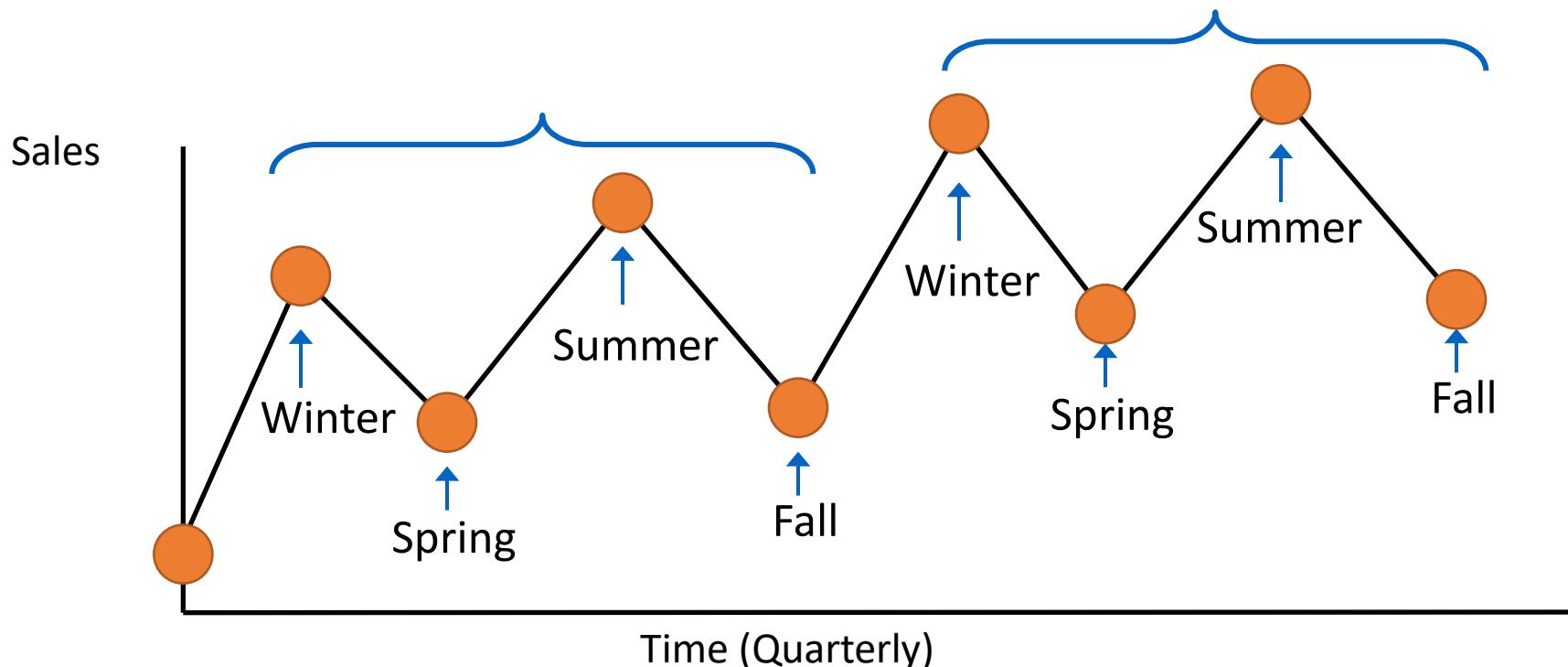
# Trend Component

- Different trends
  - Trend can be upward or downward
  - Trend can be linear or non-linear



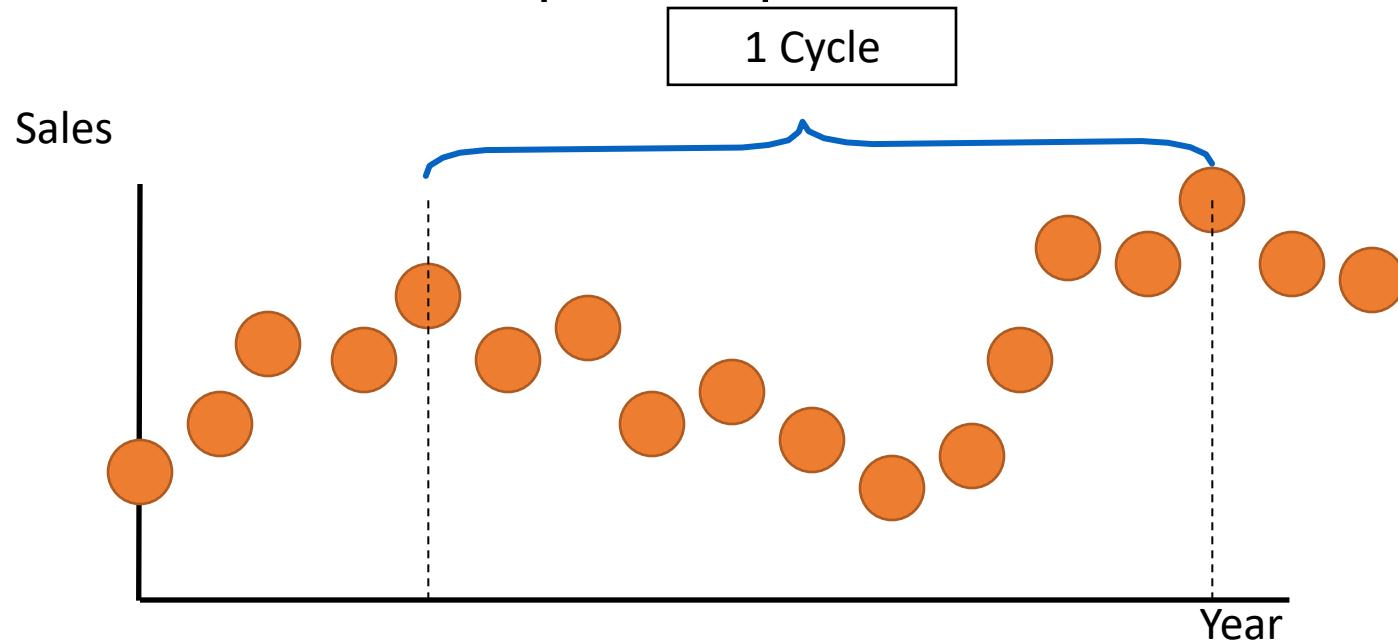
# Seasonal Component

- Regular periodic fluctuations
  - Short-term regular wave-like patterns



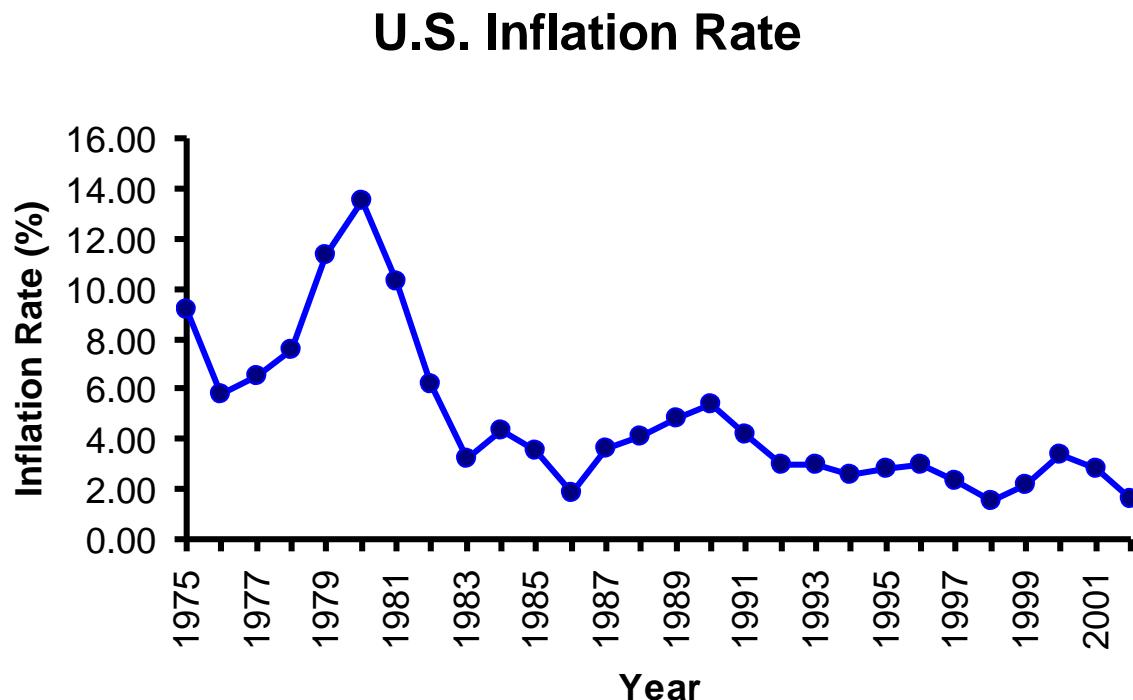
# Cyclical Component

- Repeating swings or movements
  - Long-term wave-like patterns
  - Regularly occur but may vary in length
  - Often measured peak to peak



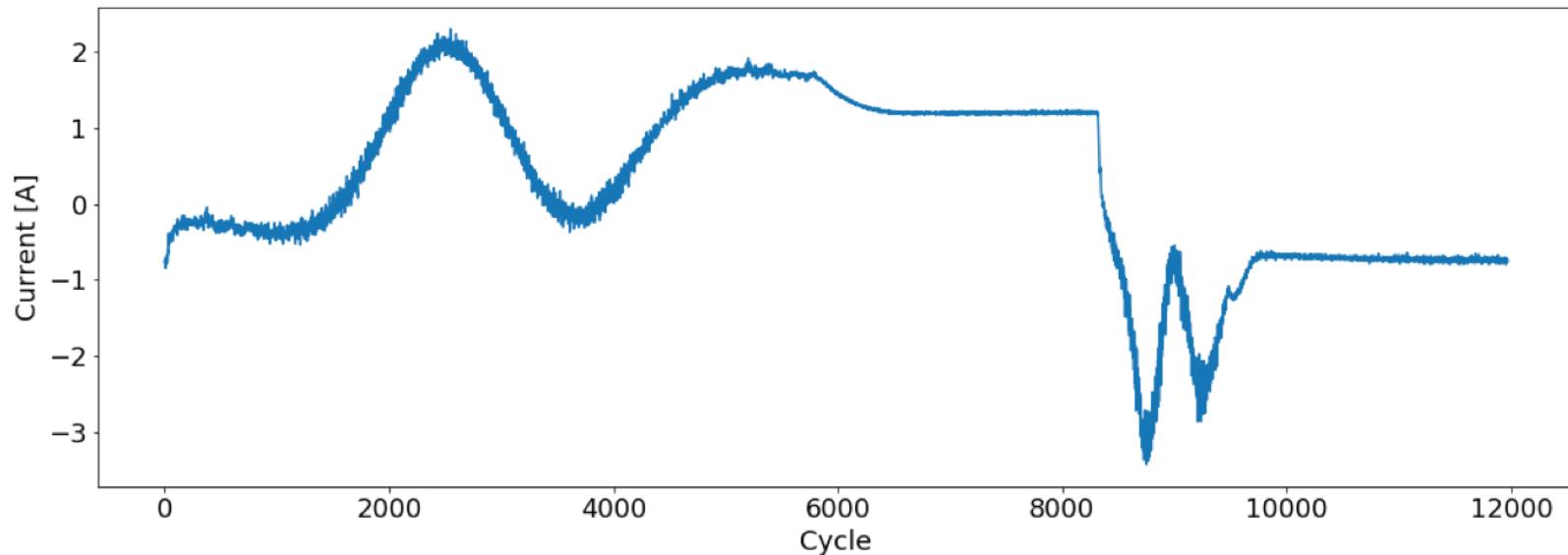
- Erratic or residual fluctuations
  - Caused by **unpredictable** influences
    - Influences are not regular and also they do not repeat in a specific pattern
  - This component usually represents “**noise**” in the time series

- **Discrete** time series
  - Numerical sequence of data obtained at regular time intervals
    - e.g. time intervals can be annually, quarterly, monthly, weekly, daily, hourly.

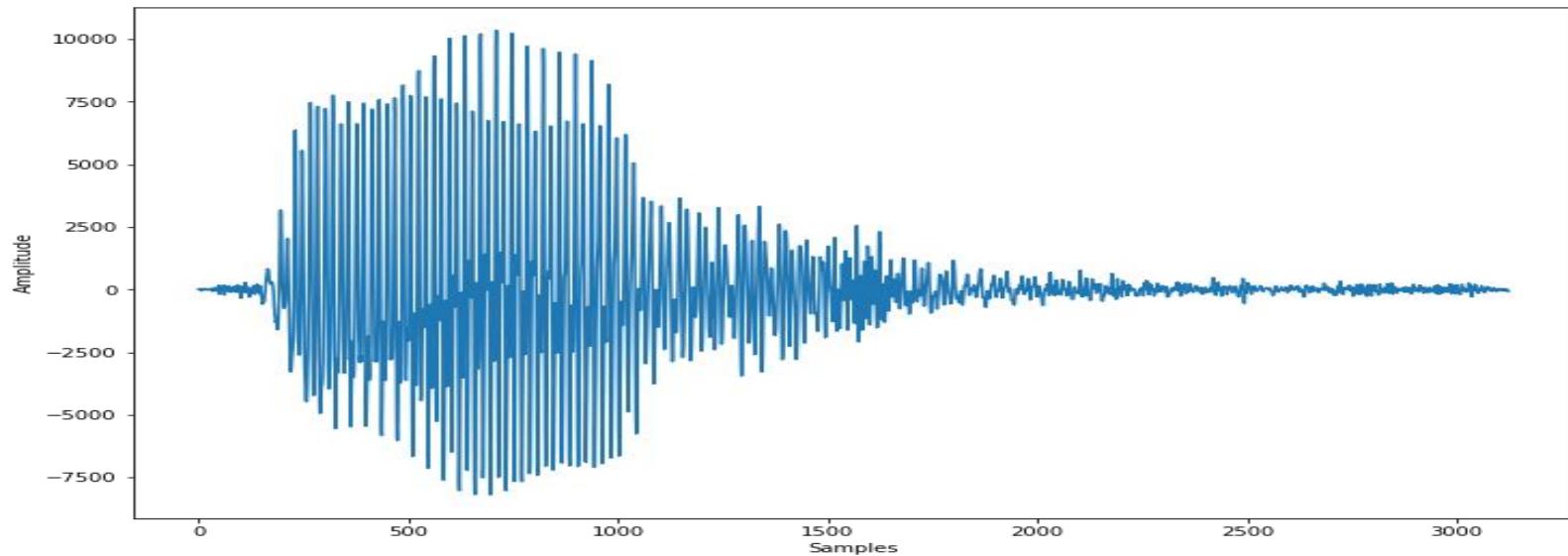


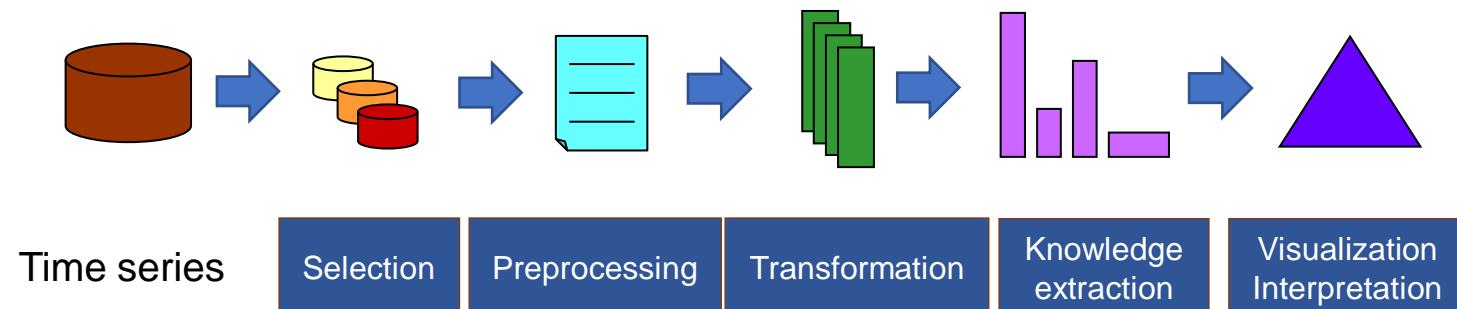
# Continuous Time Series

- **Continuous** time series
  - observations are measured at every instance of time
- Example
  - The plot shows the current (Ampere) trend of a robotic arm over time.
  - Robot Cycle duration: about 24 s
  - Sampled every 2 ms (around 11,972 samples)



- Example of **continuous** time series
  - Audio signal
  - The speaker said numbers from 0 to 9
  - Classification task: classifier the number said by the speaker





- In the preprocessing step
  - A time series **alignment** technique might be required
    - e.g., padding technique
  - In case of multivariate problem, **correlated time series** should be **identified** and **removed**
    - Correlation-based approach
    - Domain-driven knowledge
    - Mixed approach
- Transformation
  - Feature engineering
  - Feature embedding

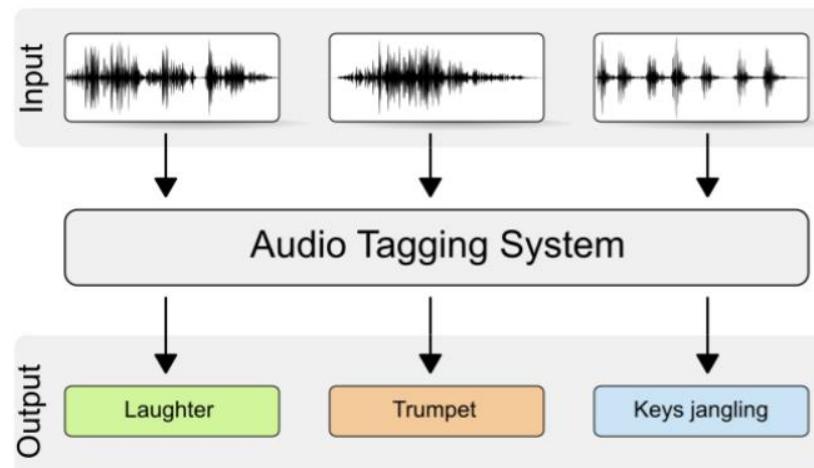
# Analytics tasks: Classification vs Forecasting

- Many algorithms can be applied to address classification and forecasting tasks
  - Machine learning algorithms such as
    - Random forest classifier/regressor
    - SVM
    - Neural networks, etc.
  - Statistical approaches
    - e.g., ARIMA (Autoregressive integrated moving average) models
- Given an analytics goal different methods can be exploited
  - The algorithm selection is driven by
    - Application requirements: accuracy, human-readable model, scalability, noise and outlier management
    - The complexity of the analytics task

- Based on the analytics goal common neural network architectures can be used to analyze time series
  - Classification task
    - CNN (Convolutional Neural Network)
  - Forecasting task
    - RNN (Recurrent Neural Network)

# Convolutional Neural Networks

- VGGish is a convolutional neural network to extract the relevant features from audio signals
  - The inputs of the network are log mel spectrogram audios
  - The output is an audio embedding
    - It can be used for further analytics tasks like classification



- RNN for time series forecasting
  - Connections between nodes form a directed graph along a temporal sequence
    - This allows the network to exhibit temporal dynamic behavior.
  - RNNs can use their internal state memory to process sequences of inputs

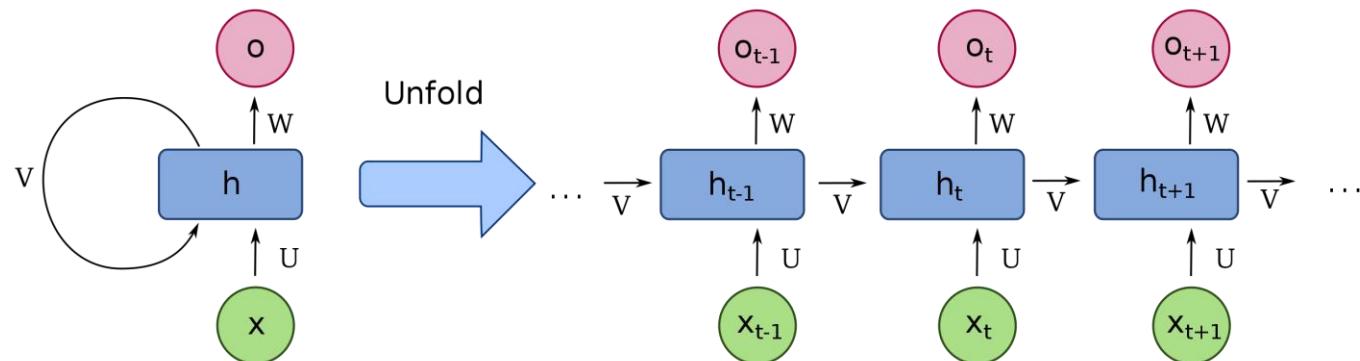
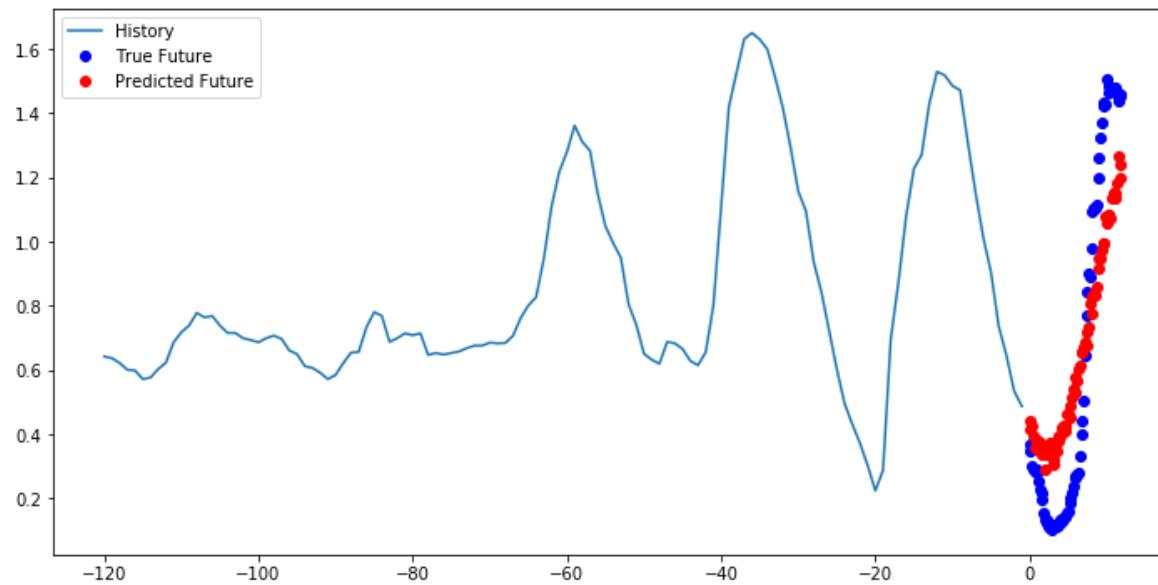


Diagram of a single unit RNN

# Recurrent Neural Networks

- RNN for time series forecasting in the context of meteorological data
- 72 predicted values



[https://www.tensorflow.org/tutorials/structured\\_data/time\\_series](https://www.tensorflow.org/tutorials/structured_data/time_series)

# Strong and weak points of Artificial Neural Networks

PoliTo

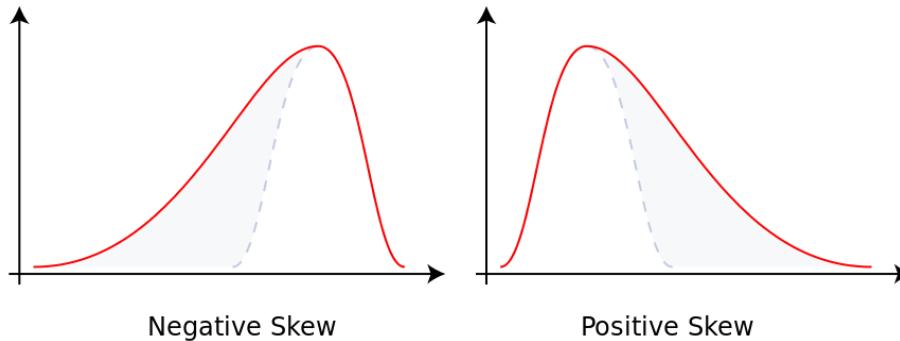
DB  
M  
G

- Main advantages
  - Powerful algorithms to train accurate models
  - Able to deal with different complex analytics tasks
- Main drawbacks
  - A huge amount of data is required for the training phase
  - Training is an heavy task in terms of both computational time and hardware resources
  - The feature learning step is hidden in the network
    - The user is not able to understand the key features driving the prediction task
    - Some specific analytics tasks might require an ad-hoc feature engineering step to better characterize the input data and train more accurate models

- Feature computation over a time series
  - Basic statistics
    - Min,Max,Mean,Standard Deviation
  - Indices
    - Kurtosis
    - Skewness
  - Time series summarization
    - Percentile
    - Joint approach based on CDF + percentile
    - Technique based on Derivate + CDF + percentile
  - Linear Regression
- Different combinations of features can be evaluated
- Correlated features are identified and removed
- Selected features model the time series under analysis
- Selected features will feed the next analytics tasks

- Given a time series
  - Minimum value
  - Maximum value
  - Mean value
  - Number of samples
  - Standard deviation
  - ...

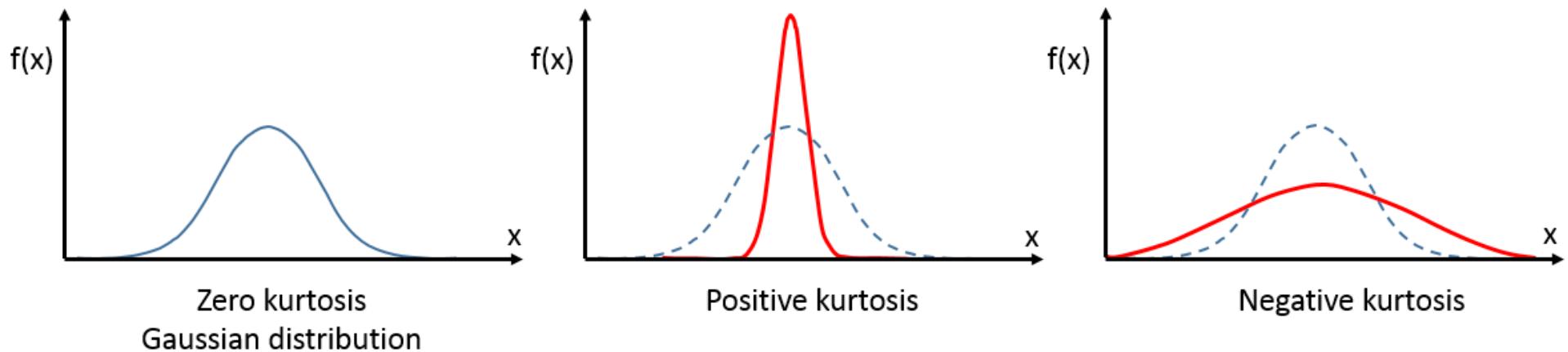
- **Skewness** is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean.



$$\gamma_1 = E\left[\left(\frac{X - \mu}{\sigma}\right)^3\right] = \frac{\mu_3}{\sigma^3} = \frac{E[(X - \mu)^3]}{(E[(X - \mu)^2])^{3/2}} = \frac{\kappa_3}{\kappa_2^{3/2}}$$

- where  $\mu$  is the mean,  $\sigma$  is the standard deviation,  $E$  is the expectation operator,  $\mu_3$  is the third central moment, and  $\kappa_t$  are the  $t$ -th cumulants.

- **kurtosis** is a measure of the "tailedness" of the probability distribution of a real-valued random variable.

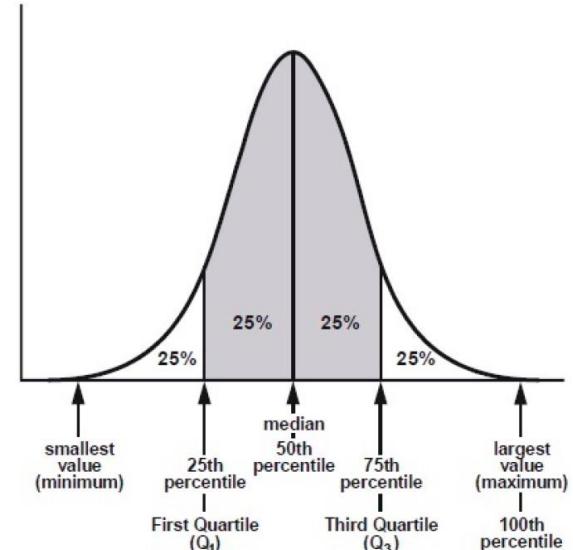


$$\text{Kurt}[X] = E \left[ \left( \frac{X - \mu}{\sigma} \right)^4 \right] = \frac{E[(X - \mu)^4]}{(E[(X - \mu)^2])^2} = \frac{\mu_4}{\sigma^4},$$

- where  $\mu_4$  is the fourth central moment and  $\sigma$  is the standard deviation.
- A very common choice is  $\kappa$ , which is fine as long as it is clear that it does not refer to a cumulant.
- Other choices include  $\gamma_2$ , to be similar to the notation for skewness, although sometimes this is instead reserved for the excess kurtosis.

# Time Series Summarization

- **Percentile** indicates the value below which a given percentage of observations in a group of observations falls
- Representing a time series through percentiles allow representing the entire distribution
  - Selecting the four percentile
  - Selecting the ten percentiles
    - selected **10 percentiles**: 10, 20, 30, 40, 50, 60, 70, 80, 90, 99
    - remove outliers by removing the **last** percentile of the distribution
- The temporal sequence is lost
- The percentiles are the **features** describing the time series



- **The Cumulative Distribution Function** of a real-valued random Variable  $X$  is the function given by

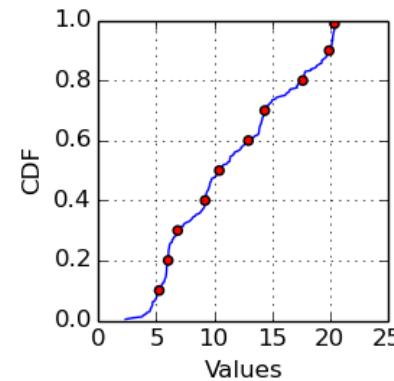
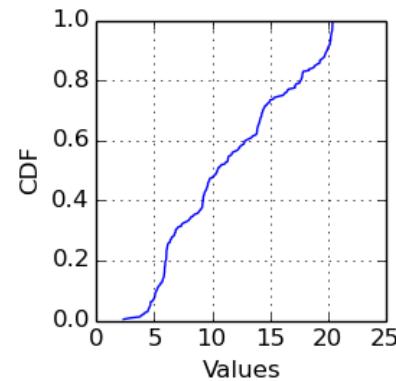
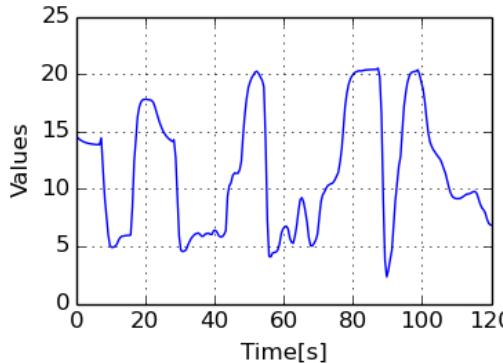
$$F_x(x) = P(X \leq x)$$

- where the right-hand side represents the probability that the random variable  $X$  takes on a value less than or equal to  $x$ . The probability that  $X$  lies in the semi-closed interval  $(a, b]$ , where  $a < b$ , is therefore

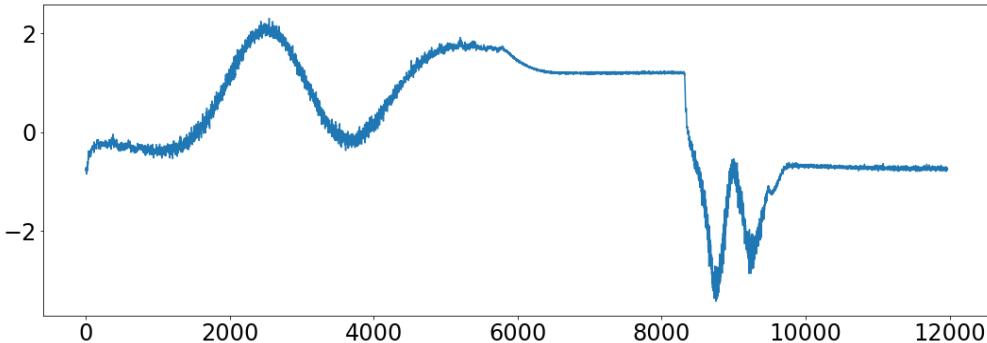
$$P(a < X \leq b) = F_x(b) - F_x(a)$$

# Time Series Summarization

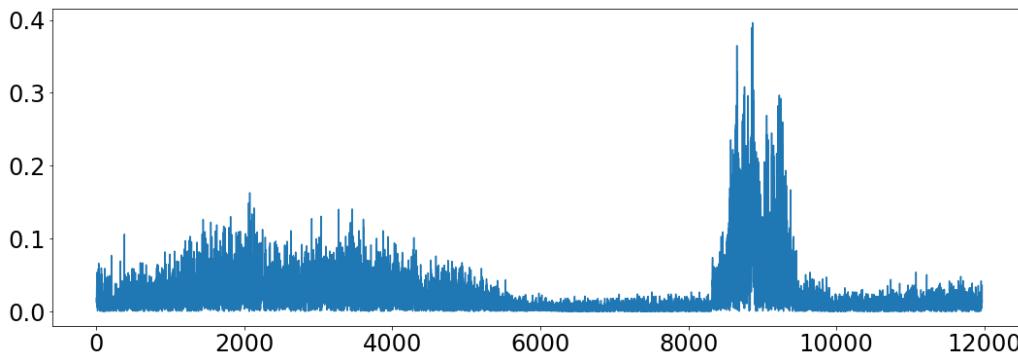
- Compute the Cumulative Distribution Function (CDF) to represent time series samples
  - To compute a reliable CDF at least 100 samples are required
- Summarize the CDF with a few **percentiles**
- The percentiles are the **features** describing the time series



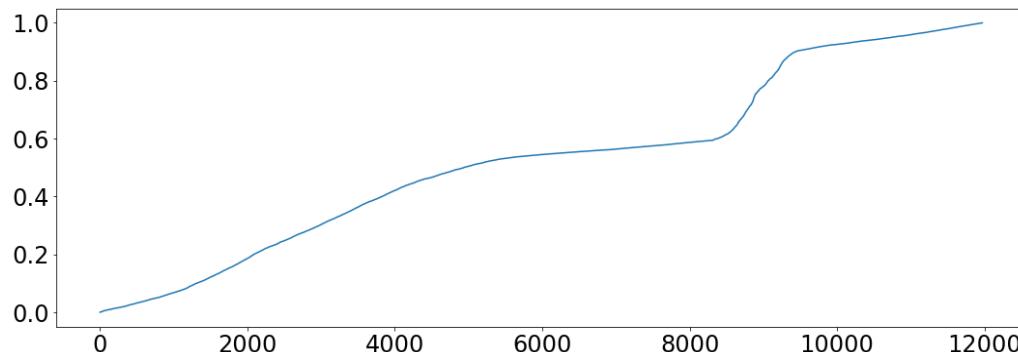
# Derivate + CDF + Percentile



- Current (Ampere) of a robotic arm



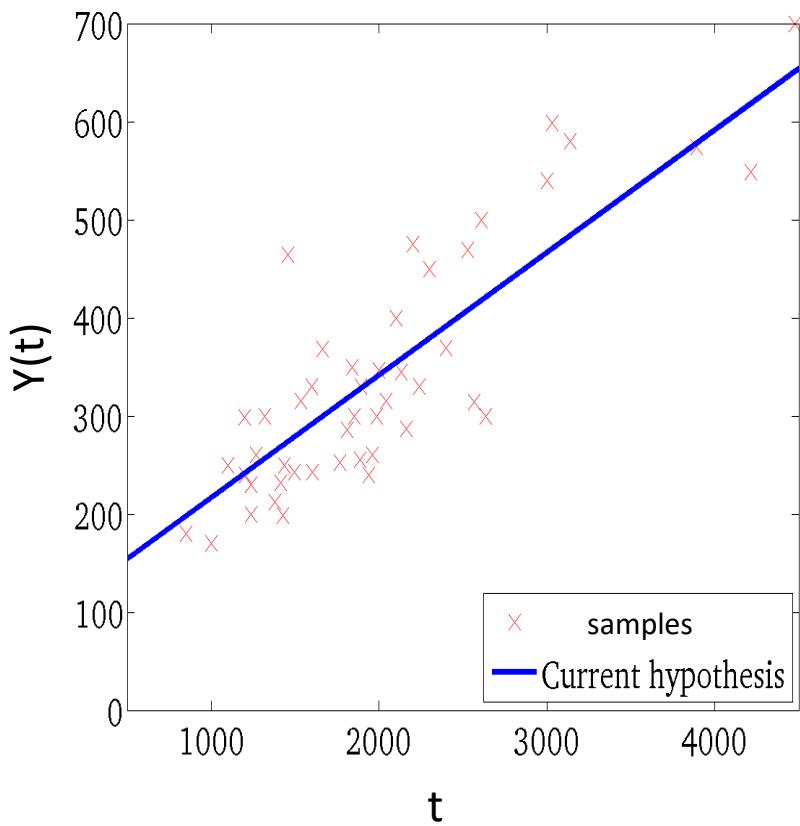
- Derivate



- CDF
- Summarize the CDF with a few **percentiles**

# Linear Regression

$$Y(t) = \beta_0 + \beta_1 t + \varepsilon$$



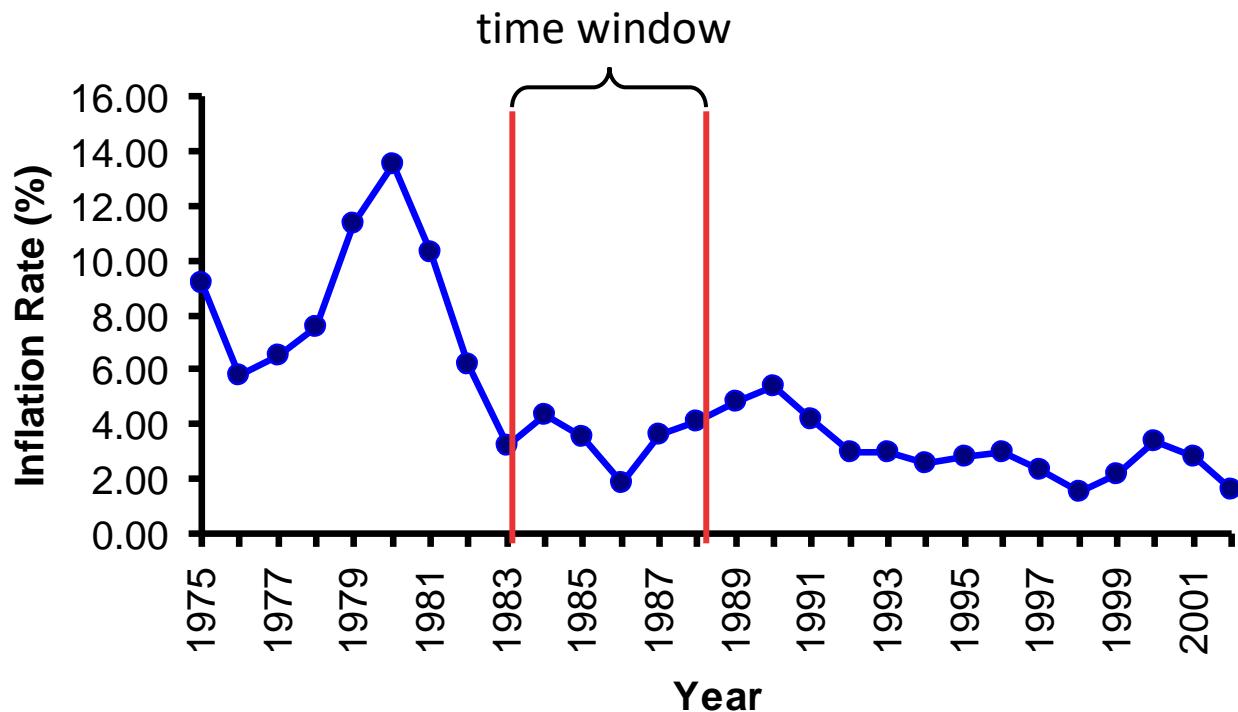
- $\beta_0$ : The **intercept** represents the estimated value of  $y$  when  $t$  assumes 0
  - $\beta_0$  is the portion of  $y$  not explained by  $t$
- $\beta_1$ : the **slope** measures the estimated change in the  $y$  value as for every one-unit change in  $t$ 
  - The average value of a  $t$  change

- Feature engineering can be calculated on the entire time series
  - e.g. on a signal representing the current consumption
    - Statistics on the entire robot cycle can be useful to characterize the overall time series trend
- Feature engineering can be also calculated in local parts of the time series
  - Time windows
  - For each time window, statistical features summarize the local time series trend

- Time windows are defined by
  - **Window length:** size (in time units) of each window
    - *Domain-driven* (e.g. parts of the speech, actions of a mechanical arm)
    - *Data-driven* (e.g. time windows on seasonal features of a signal)
  - **Window shift:** window position with respect to contiguous windows
    - *Not-overlapped (jumping)*: all windows are independent and do not share any data
    - *Overlapped (sliding)*: two consecutive windows share a portion of data

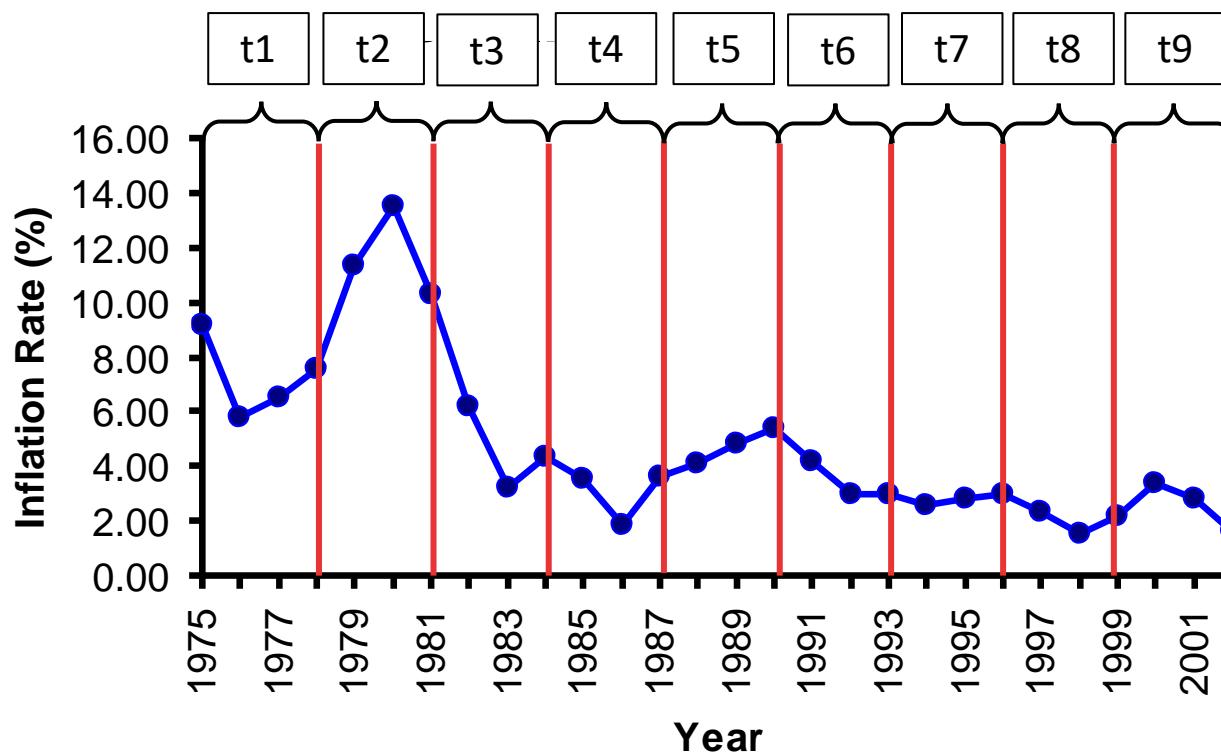
# Time Window

- A time-window is a fixed interval of time when the data stream is processed for query and mining purposes



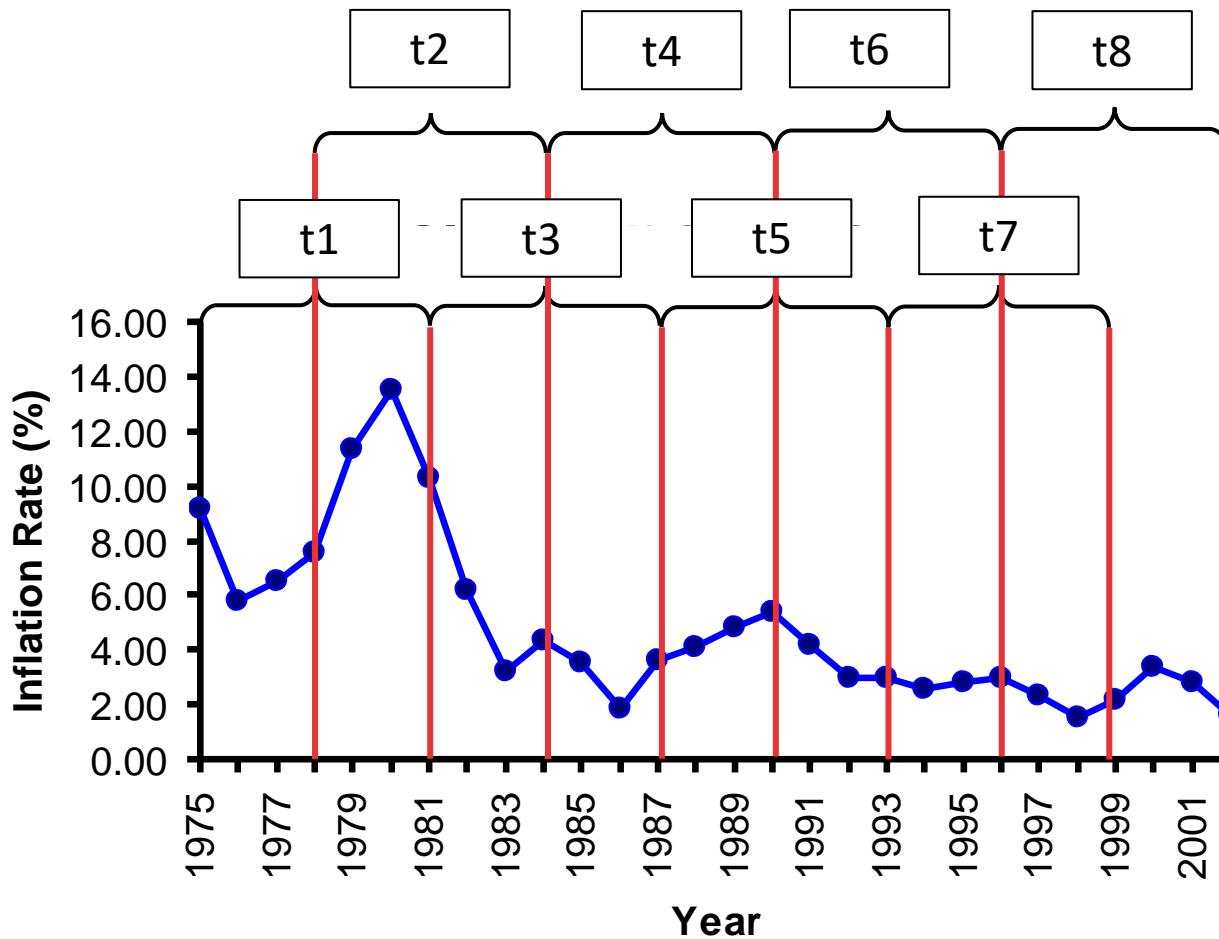
# Time Window Not-overlapped

All windows are independent and do not share any data



# Time Window Overlapped

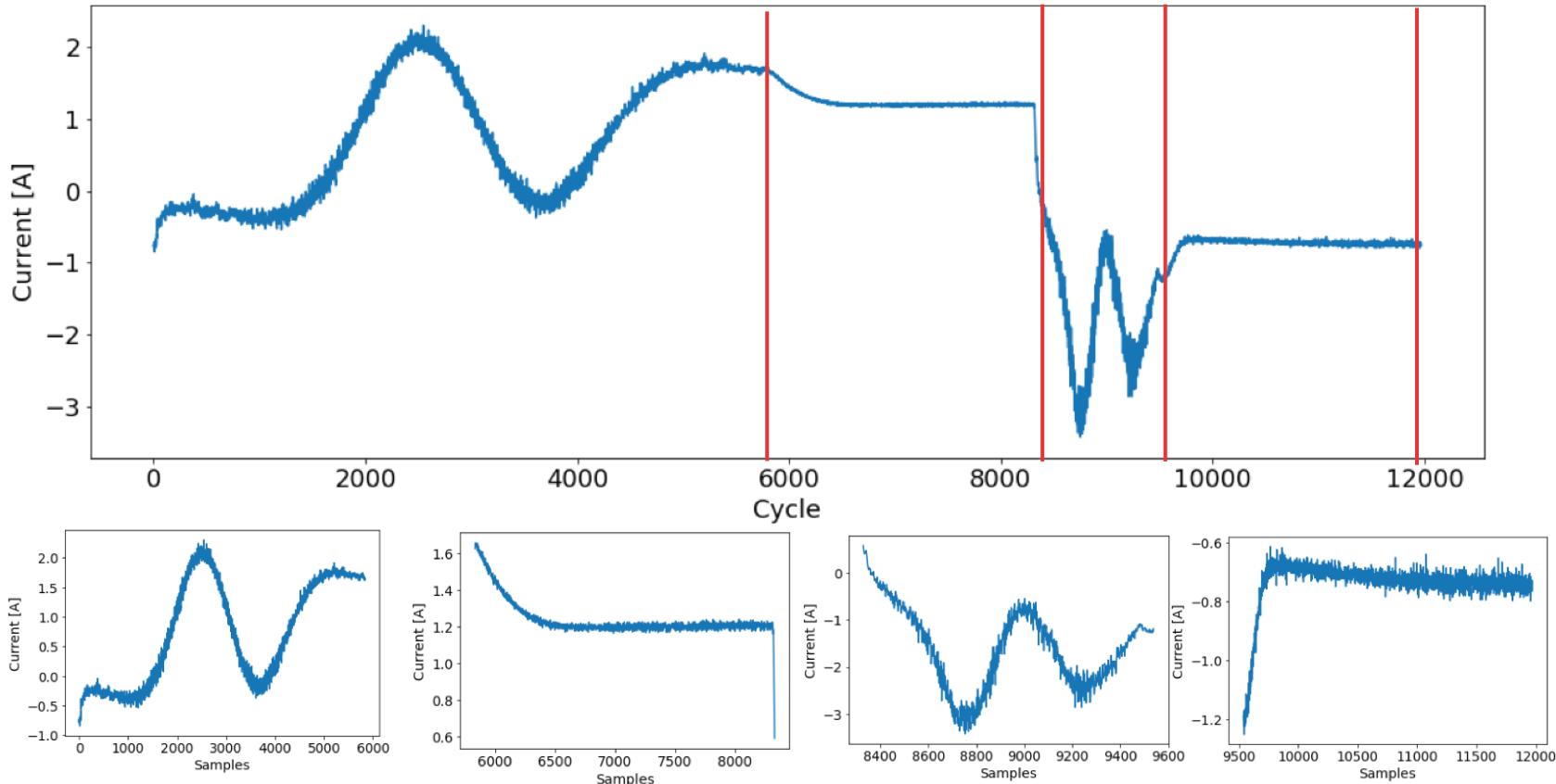
Two consecutive windows share a portion of data



# Time window: an example

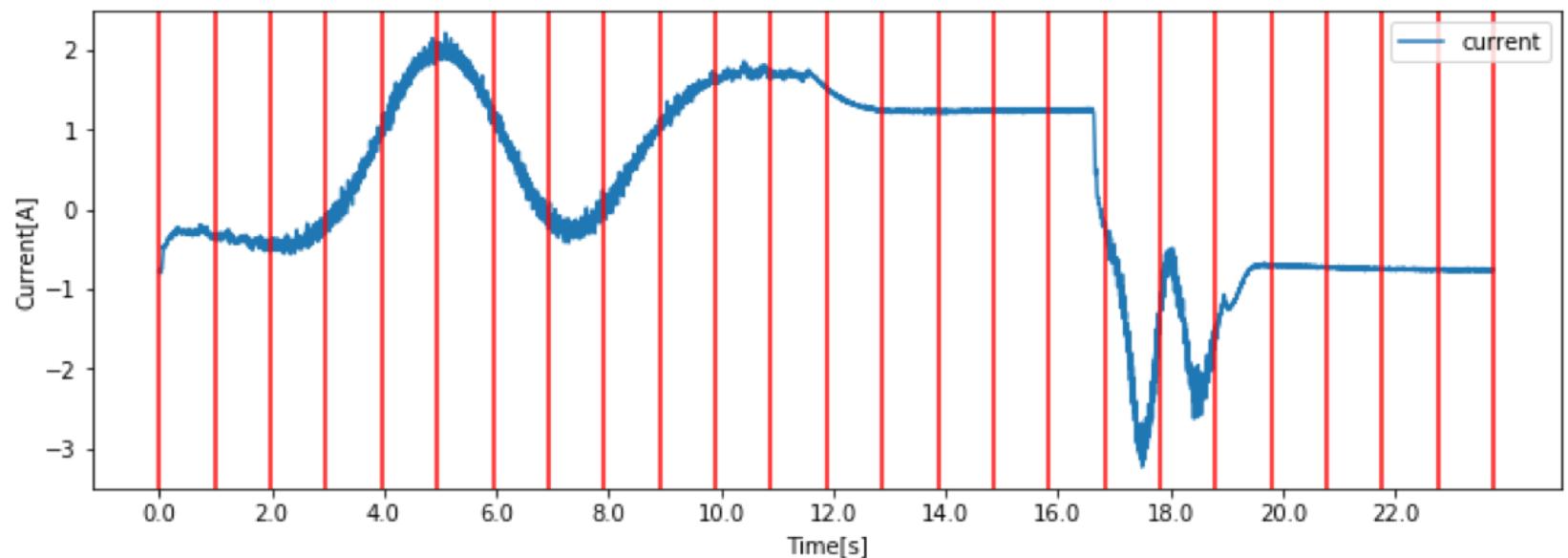
## ■ Domain Driven

The plot shows the current (Ampere) trend of a robotic arm over time.



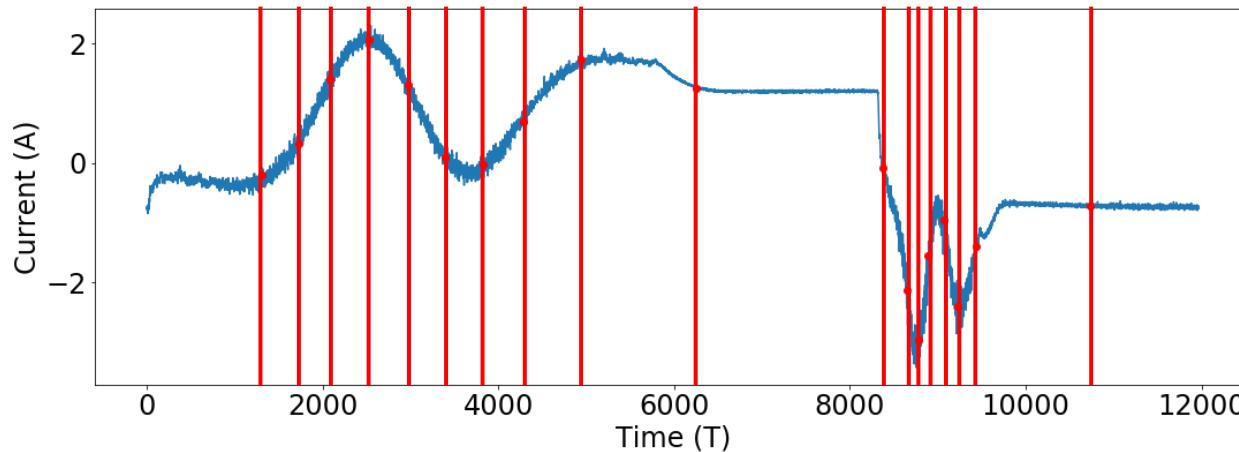
# Time window: an example

- Data Driven

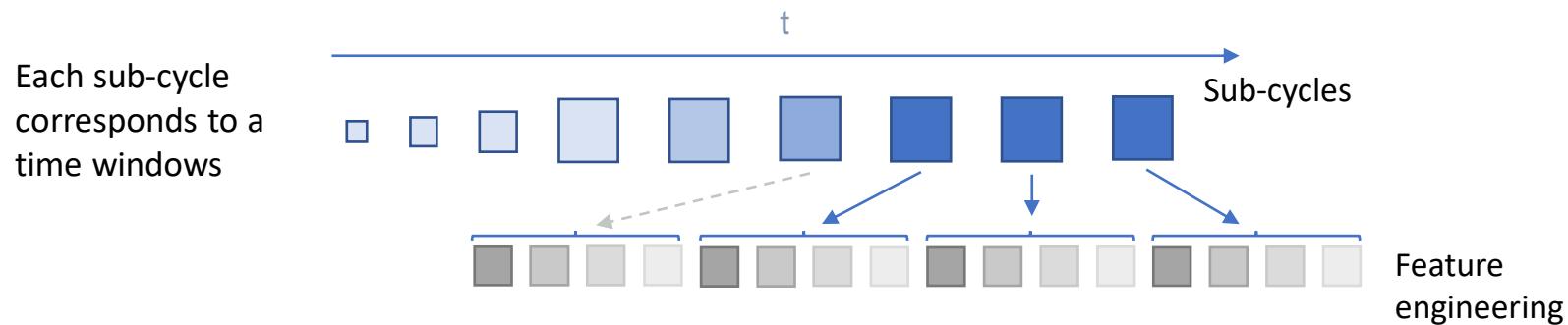
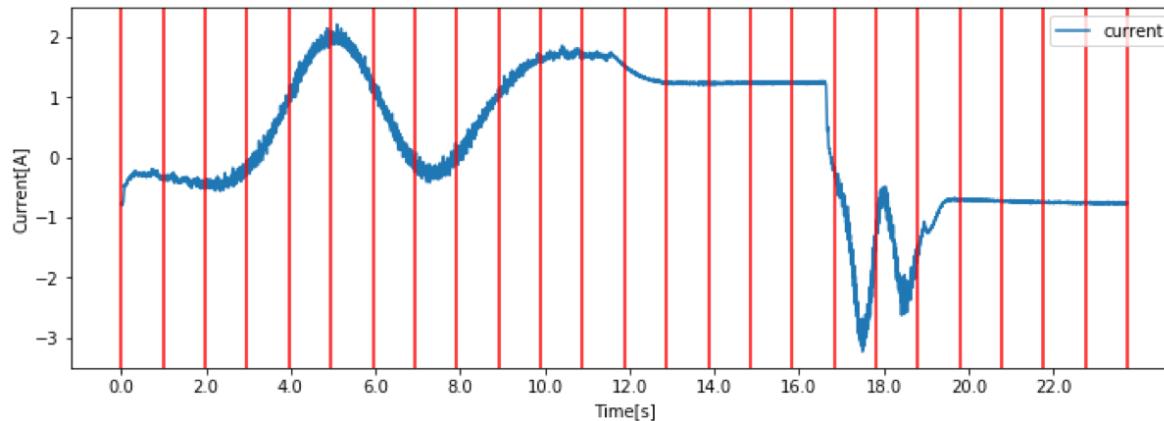


# Time window: an example

- Longer time window in those parts where the time series is more stable
- Shorter time window in those parts where the time series varies most



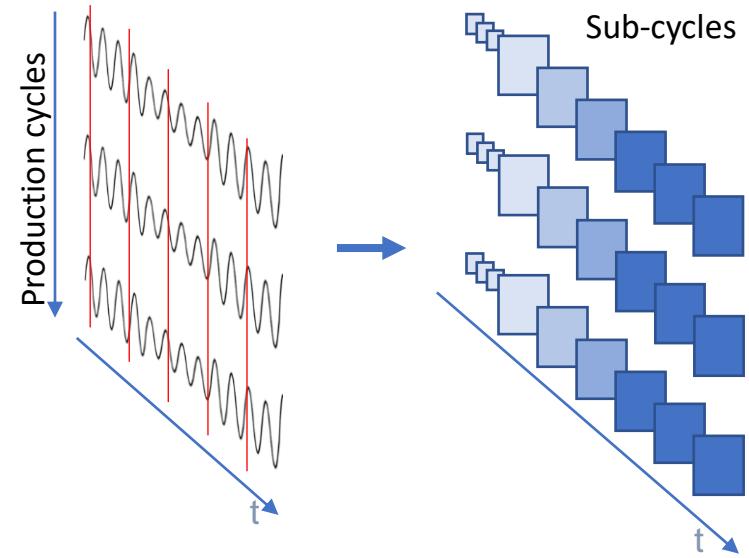
# Time Series summarization



The time series trend can be captured through the features extracted from each sub-cycle of each time series

# Time Series Aggregation

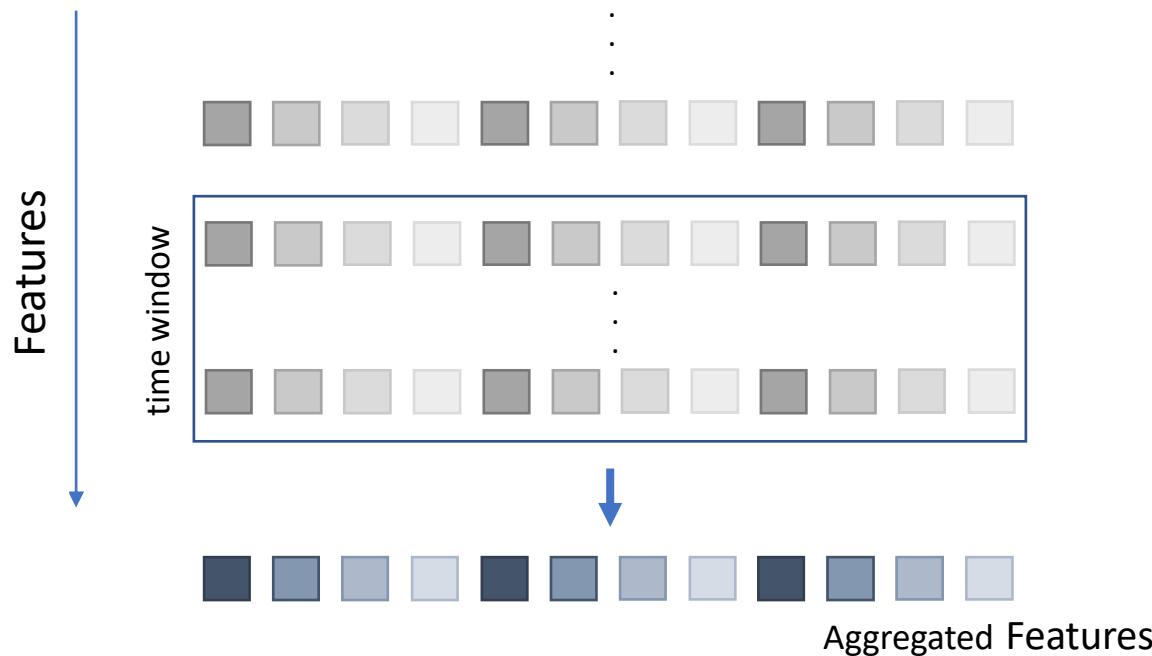
- In slowly-degrading environments single time-series (cycle) predictions have a too short horizon.
- To deal with long horizon prediction
  - The multi-cycle time-based aggregation step could be based on time series aggregation over a time window.



Cycles divided into time window to extract the variability of each sub-cycle

# Time Series Aggregation

- The main characteristics of the window is captured through feature computation over a time window of features
- The feature aggregation preserves the meaning of the time series, keeping the process transparent.
  - Different feature computation can be exploited

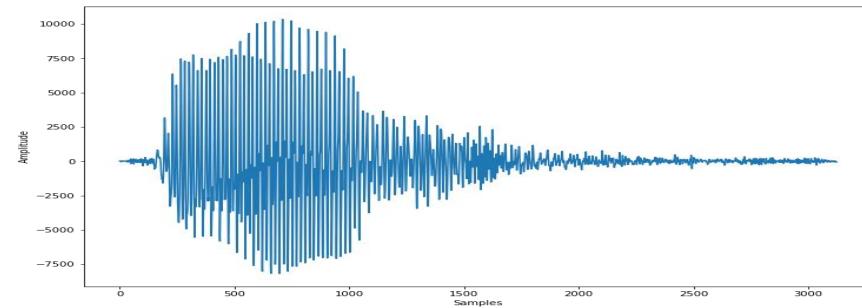


# Feature selection and removal

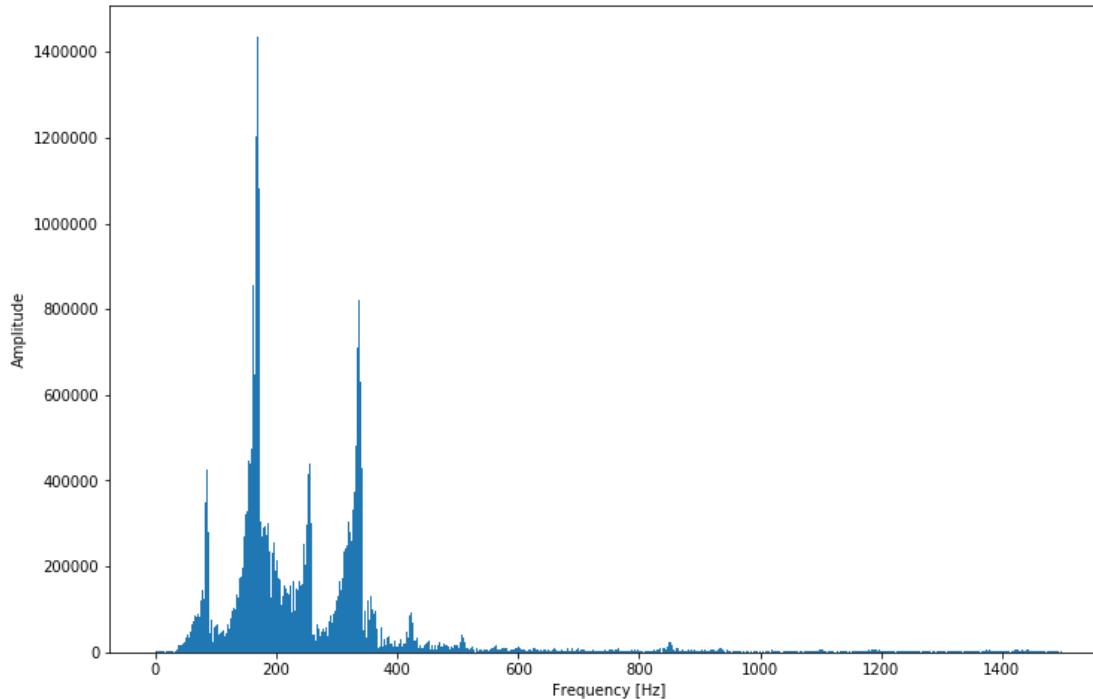
- In case of a large number of features modeling the time series, **some of them** might provide **redundant information**.
- Feature selection and removal simplifying the model computation
  - improving the model performance
  - Enhancing the model interpretation (i.e., better explainability of the dependent variables)
- Feature selection based on correlation-based approach
  - Features highly-correlated with other features could be discarded from the analysis
  - having dependence or association in any statistical relationship, whether causal or not

- In some cases it may be useful to analyse a signal in the frequency domain.
  - e.g., audio, video, etc...
- The Fourier transformation can transform a time series in the frequency domain

# Time series in frequency domain



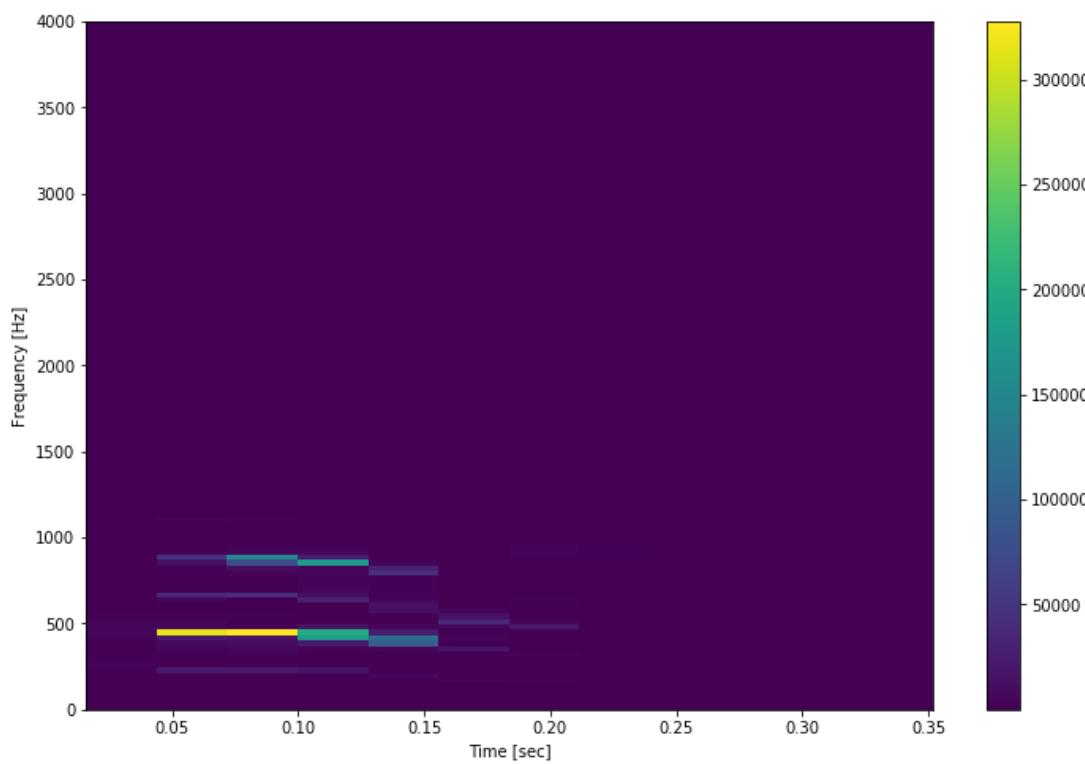
- Audio Signal in time domain



- Audio Signal in the frequency domain through the Fourier Transformation

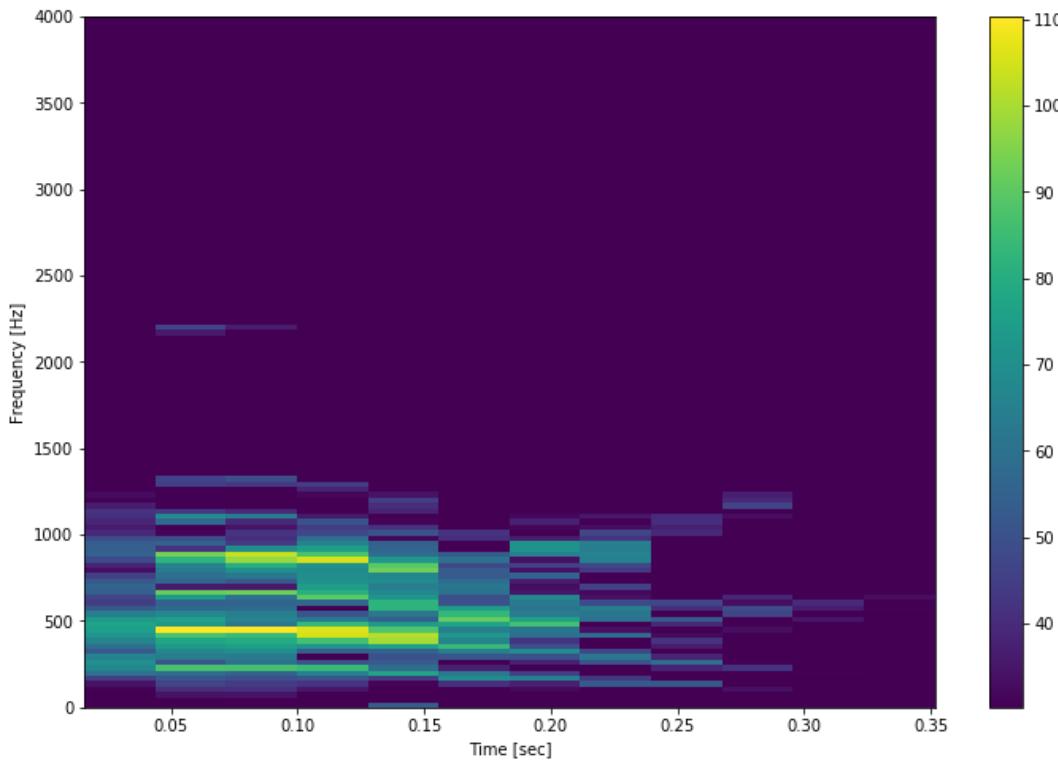
- To analyse an audio signal in the frequency domain the spectrograms are usually used
- A **spectrogram** is a visual representation of the spectrum of frequencies of a time series as it varies with time.
  - In the case of audio, spectrograms are sometimes called **sonographs**, **voiceprints**, or **voicegrams**.

# Time series in frequency domain

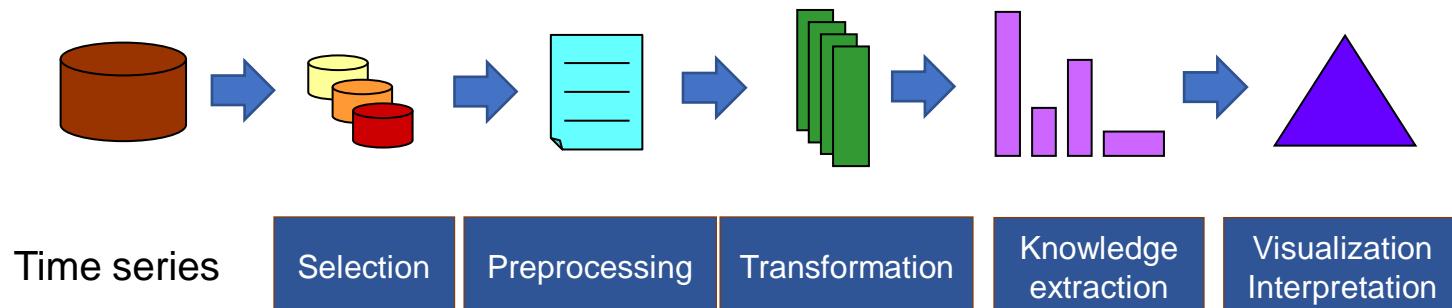


- In the **spectrogram** the colour intensity corresponds to the signal amplitude.
- If the amplitude is linear, it is difficult to identify the components because the audio follows logarithmic trends
- A data transformation is needed

# Time series in frequency domain



- In this plot the amplitude has been transformed from linear to logarithmic in order to give more emphasis to musical, tonal relationships



- Knowledge extraction
  - Different algorithms can be exploited to address the analytics tasks
  - Selected features feed the knowledge extraction algorithm
- Visualization and interpretation
  - Help the domain expert correctly understand the extracted knowledge items to effectively support the decision-making process