Data Mining

Data Preparation

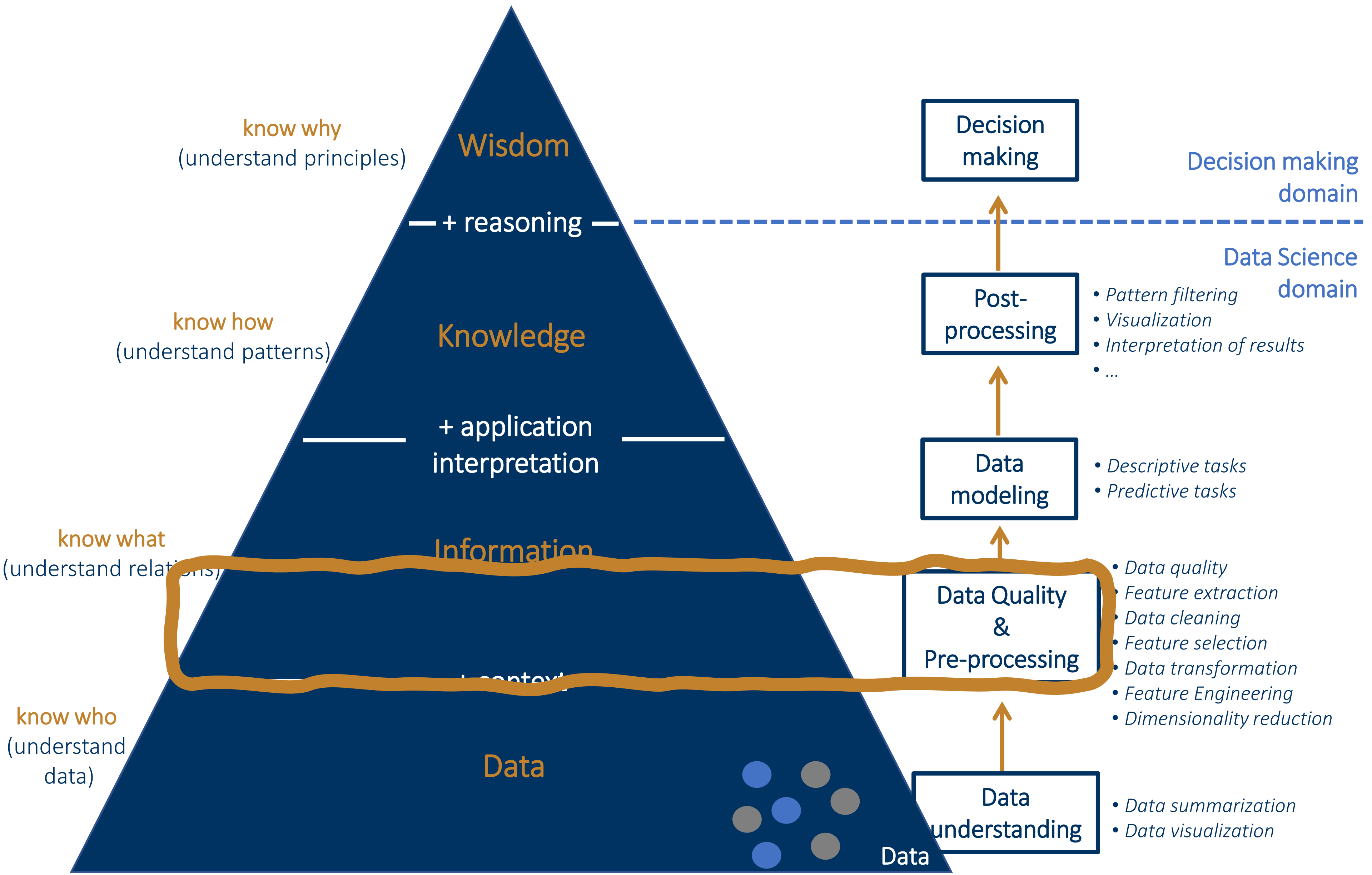
Raquel Sebastião

Departamento de Eletrónica, Telecomunicações e Informática

Universidade de Aveiro

raquel.sebastiao@ua.pt

2022/2023



know why
(understand principles)
Wisdom
+ reasoning
know how
(understand patterns)
Knowledge
+ application
interpretation
know what
(understand relations)
Information
+ context
know who
(understand
data)
Data
Data
Decision making
Decision making domain
Data Science domain
Post-processing
• Pattern filtering
• Visualization
• Interpretation of results
• ...
Data modeling
• Descriptive tasks
• Predictive tasks
Data Quality & Pre-processing
• Data quality
• Feature extraction
• Data cleaning
• Feature selection
• Data transformation
• Feature Engineering
• Dimensionality reduction
Data understanding
• Data summarization
• Data visualization

Contents

- Data Quality
- Data Pre-processing
  - Feature extraction
  - Data integration
  - Data cleaning
  - Feature transformation
  - Feature Engineering
  - Data reduction
- Summary

Data quality

Poor data quality **negatively affects effective** data analysis

Example: a classification model for detecting client's loan risks is built using poor data

- Some credit-worthy candidates are denied loans
- More loans are given to individuals that default

Data quality

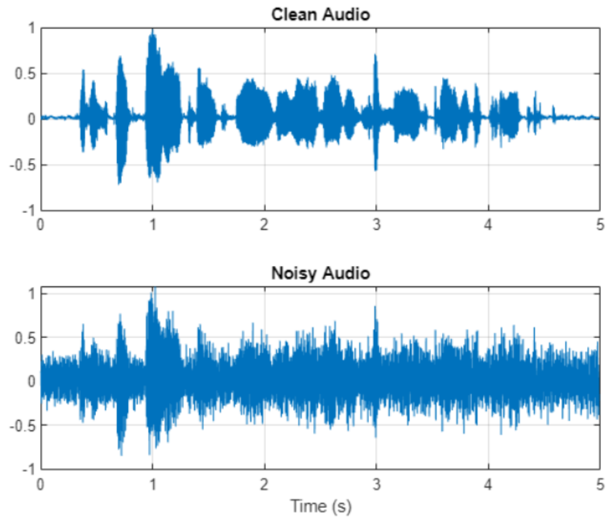
- What kinds of data quality problems?
- How can we detect problems with the data?
- What can we do about these problems?

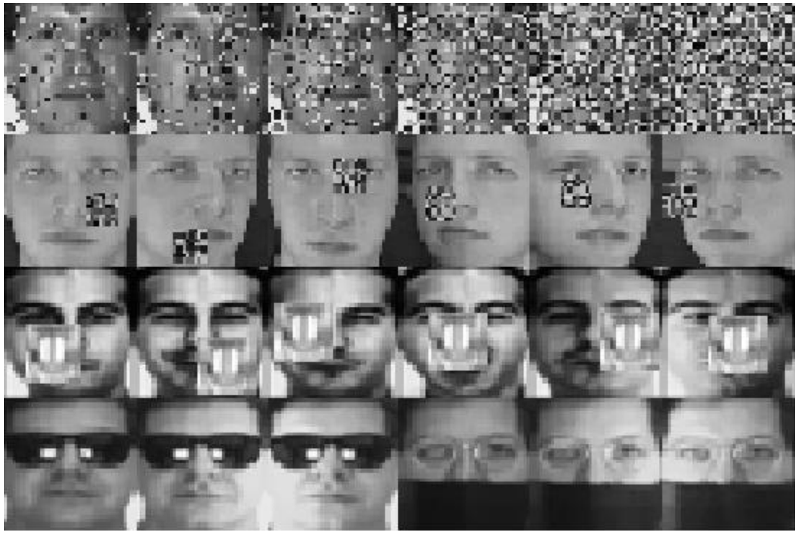
Examples of data quality problems

- Missing values
- Duplicate data
- Noise and outliers
- Wrong data
- Fake data
- Inconsistent across different data sources

Data quality: noise

- For objects, noise is an extraneous object
- For attributes, noise refers to modification of original values
  - distortion of a person’s voice when talking on a poor phone
  - “snow” on television screen





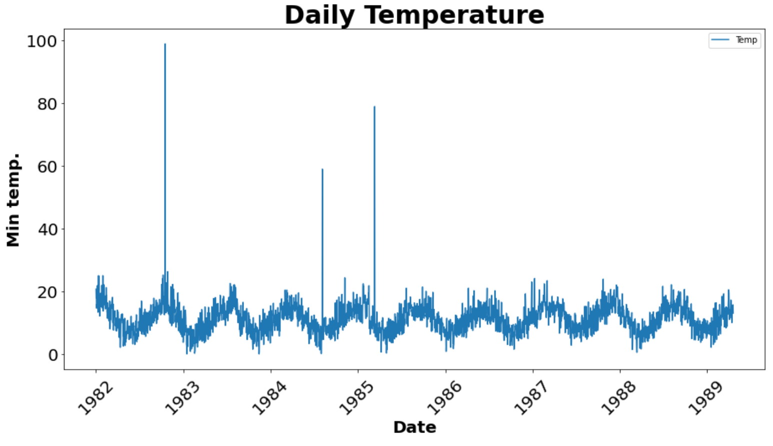
Some corrupted face images from the Yale dataset

Data quality: outliers

> "An outlier is a point that deviates so much from the other data points as to arouse suspicions that it was generated by a different mechanism" (Hawkins, 1980)"
> Hawkins, 1980

- **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
    - Min. temperature values above 50ºC



Data quality: outliers

"An outlier is a point that deviates so much from the other data points as to arouse suspicions that it was generated by a different mechanism" (Hawkins, 1980)"
Hawkins, 1980

- **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
  - Min. temperature values above 50ºC

- **Case 2:** Outliers are the goal of our analysis
  - Credit card fraud
  - Intrusion detection

Data quality: missing values

- Information was not collected
  - Missing value is related to unobserved data of the variable
    - people decline to give their age and weight
- Attributes may not be applicable to all cases
  - Missing value is related to observed data, not to unobserved data
    - annual income is not applicable to children

Data quality: missing values

Missing data may be due to

- Equipment malfunction
- Incongruent with other recorded data and thus deleted
- Data were not entered due to misunderstanding
- Certain data may not be considered important at the time of entry
- Did not register history or changes of the data

Data quality: duplicates

- Data set may include data objects that are duplicates, or almost duplicates of one another
  - Major issue when merging data from heterogeneous sources
- Examples:
  - Same person with multiple email addresses
- Necessary a process of dealing with duplicate data issues
  - When should duplicate data not be removed?

Data quality: inconsistent data

Typical when the data is available from different sources in different formats

- Examples
  - Person’s name may be spelled out differently in different sources
    - John Smith, J. Smith, Smith J.
  - Person's height should not be negative
  - Same object: Attribute ***country*** = ‘United States’ & attribute ***city*** = ‘Shanghai’

Contents

Data pre-processing: what and why?

- Extremely important
- Time-consuming

Steps carried out before any further analysis of the available data

- Data can come from several sources (in different formats)
- Data sets may have unknown attributes values
- Many data mining methods are sensitive to the scale and/or the type of attributes
- The need to create new attributes to achieve data analysis goals
- The need to select representative subsets of data, as the data set may be too large for some methods to be applicable

Data pre-processing: major tasks

- **Feature extraction**
- **Data integration**
- **Data cleaning**
- **Feature transformation**
  - Aggregation
  - Scaling
  - Discretization
  - Binarization
- **Feature Engineering**
- **Data reduction**
  - Numerosity reduction
  - Dimensionality reduction
    - Feature selection
    - Singular Value Decomposition (SVD)
    - Principal Component Analysis (PCA)
    - Linear Discriminant Functions

Data pre-processing: feature extraction

Extract features from raw data

… features need to be extracted for processing

- Text to categorical and numeric data
- Time Series to discrete sequence data
- Time Series to numeric data
- Discrete sequence to numeric data
- Spatial to numeric data
- Graphs to numeric data

- Sensor data
- Image data
- Web logs
- Network traffic
- Document data

Data pre-processing: feature extraction

Extract features from raw data

… features need to be extracted for processing

- sensor data
  - large volume of low-level signals associated with date/time attributes
- image data
  - very high-dimensional data that can be represented by pixels, color histograms, etc.
- web logs
  - text in a prespecified format with both categorical and numerical attributes
- network traffic
  - network packets information
- document data
  - raw and unstructured data

Data pre-processing: data integration

Redundant and **inconsistent** data occur often when integration of multiple datasets

- The same attribute or object may have different names in different datasets
- An attribute may be a “derived” from another attribute or set of attributes in another table
  - ***Age*** = “42”, ***Birthday*** = “03/07/1980”

Careful integration of the data may help reduce/avoid redundancies and inconsistencies

Data pre-processing: data cleaning

Data in the Real World Is Dirty

- Lots of potentially incorrect data
  - e.g., instrument faulty, human or computer error, and transmission error

Poor data quality negatively affects data processing tasks and impacts performance of the models

Data pre-processing: data cleaning

Poor data quality negatively affects data processing tasks and impacts performance of the models

- Handle **missing values**
- Deal with **duplicate** data
- Smooth **noisy** data
  - In columns (e.g., features): due to sensing errors
  - In rows: extraneous object
- Identify or remove **outliers**
  - In columns (e.g., features): univariate statistics (can be noise)
  - In rows (might be the goal of analysis)
- Resolve **inconsistencies**
  - Some easy to detect. For instance: person's height should not be negative
  - Correction of inconsistencies requires redundant or additional information

Data pre-processing: data cleaning

Data in the Real World Is Dirty

Ultimate goal

- Make the data set tidy
  - each value belongs to an attribute and an object
  - each attribute contains all values of a certain property measured across all objects
  - each object contains all values of the attribute measured for the respective case
- These properties lead to **data tables**
  - each row represents an object
  - each column represents an attribute measured for each object

Data cleaning: handling missing values

- Information was not collected
- Features/attributes may not be applicable to all cases

Main Strategies to Handle missing values

- Elimination
  - Eliminating Rows (e.g., objects)
  - Eliminating Columns (e.g., features)
- Imputation: substituting missing by
  - mean, median (numerical feature)
  - mode (categorical feature)
  - linear interpolation of nearby values in time and/or space
- Ignore the missing value during analysis
  - methods inherently designed to work robustly with missing values

Data cleaning: handling missing values

Consider the following "data tables" with missing values (marked- ?)

A1	A2	A3	A4	A5
		?		
		?		
		?		
		?		

A1	A2	A3	A4	A5
?		?		
		?		?

A1	A2	A3	A4	A5
		?		
	?			
			?	
				?

- Select the best strategy to handle the missing data
- Advantages and Disadvantages

Data cleaning: handling incorrect values

- Inconsistent detection
  - Data integration techniques
- Domain knowledge
  - Data auditing: by analyzing data to identify features' ranges or discover constraints/rules that specify the relationships across different features
- Data-centric methods
  - Statistical-based methods to detect outliers

Data pre-processing

"At the end of the day, some machine learning projects succeed and some fail.
What makes the difference?
Easily the most **important** factor is the **features used**."
Pedro Domingos, in "A Few Useful Things to Know about Machine Learning".
DOI:10.1145/2347736.2347755

- **Feature transformation**
  - Aggregation
  - Scaling
  - Discretization
  - Binarization
- Feature Engineering
- Data reduction

Data pre-processing: feature transformation

Map the entire set of values of a given feature to a new set of replacement values such that each old value can be identified with one of the new values

Why it may be useful?

- two features (e.g., age, salary) with very different scales
- Any aggregation function (e.g., Euclidean distance) computed on the set of objects, will be dominated by the feature of larger magnitude

Some common strategies:

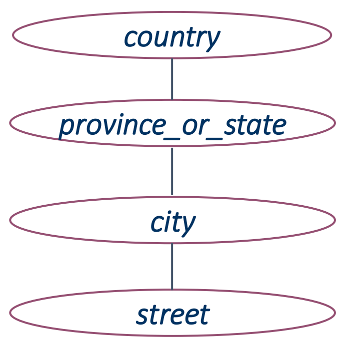
- Aggregation
- Scaling
- Discretization
- Binarization

Feature transformation: aggregation

Combining two or more **features** (or *objects*) into a single **features** (or *object*)

Goals

- More “stable” data - aggregated data tends to have less variability
- Data reduction - reduce the number of attributes (or objects)
  - **Products** aggregated into **categories** (grocery, electronics, clothing, toys, ...)
  - **Days** aggregated into **weeks**, **months**, or **years**



15 distinct values

365 distinct values

3567 distinct values

674 339 distinct values

Feature transformation: scaling

What?

Techniques to adjust to differences among attributes in terms of frequency of occurrence, mean, variance, range

- Representing all numerical (integer or real) features in the same scale

Why?

Due to the sensitivity of aggregation functions (e.g., Euclidean distance) to the scale/magnitude of the input values

Feature scaling is important to:

- PCA, LDA, kNN, LR, SVM, Neural Networks, k-Means, Regression, ...

Feature scaling is not important (not distance-based)

- Naïve-Bayes and Decision trees

Feature transformation: scaling

Min-Max scaling (range-based scaling) *

$$\tilde{x}_i = \frac{x_i - min}{max - min}$$

- min and max mininum and maximum, respectively, of feature $\mathbf{x_i}$
- Scaled values lie int the range [0,1]

- Outliers are not correctly handled
  - if an erroneous age value of 800 is registered instead of 80, most of the values will be in the range [0;0.1]

* Just scales the data

Feature transformation: scaling

Standardization (z-score normalization)*

$$\tilde{\mathrm{x}}_i = \frac{\mathrm{x}_i - \mu_i}{\sigma_i}$$

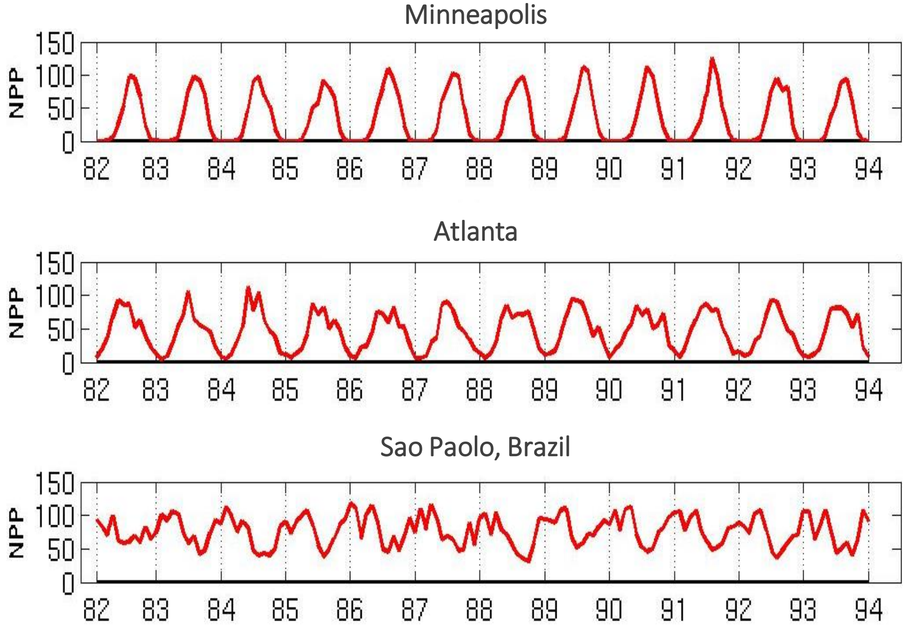
- $\boldsymbol{\mu_i}$ and $\boldsymbol{\sigma_i}$ are the mean and standard deviation, respectively, of feature $\mathbf{x_i}$
- values are scaled s.t. μ_i=0 and $\boldsymbol{\sigma}_\mathrm{i}$=1
- Scaled values, typically, lie in the range [-3 , 3] under a normal distribution assumption

* More than scaling, it changes the distribution of the data

Feature transformation: scaling

Time-series

- Adjust differences in terms of mean, variance and range
- Take out unwanted, common signal, e.g., seasonality



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

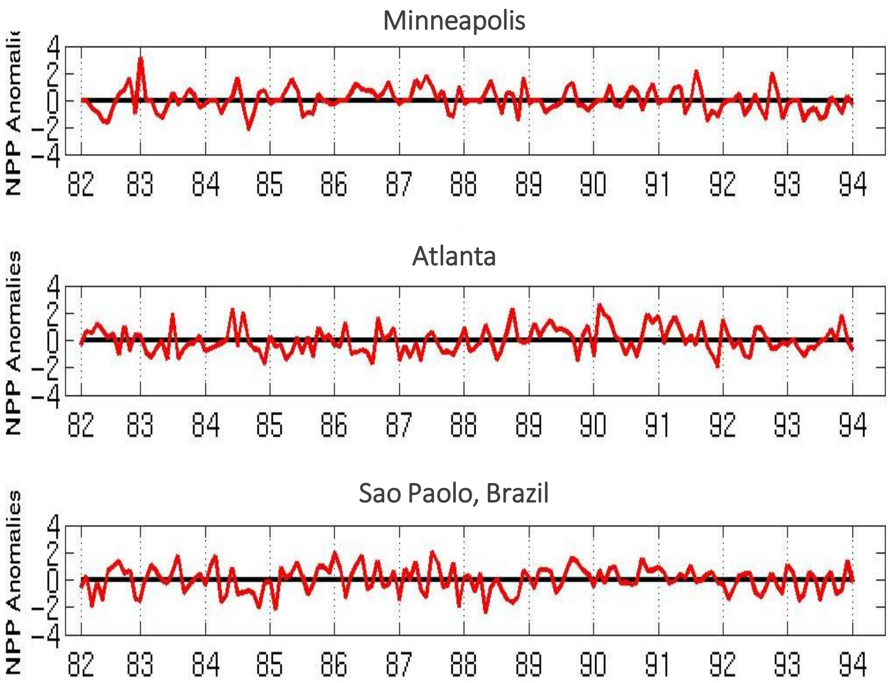
Correlations between time series

	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

Feature transformation: scaling

Time-series

- Adjust differences in terms of mean, variance and range
- Take out unwanted, common signal, e.g., seasonality



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

Feature transformation: discretization

Discretization: converting a continuous feature into an ordinal feature

- A potentially infinite number of values are mapped into a small number of categories

Unsupervised discretization: find breaks in the data values

- **Equal-width:** divides the range into equal-width intervals
  - it may be affected by the presence of outliers
  - Skew data is not correctly handled
- **Equal-frequency:** divides the range into intervals with the same number of values
  - it can generate ranges with very different amplitudes
  - Good data scaling
- **Clustering** approaches

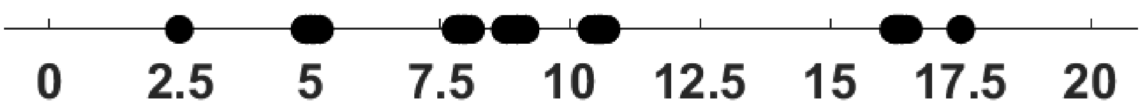
Supervised discretization: use class labels to divide data values

- Decision-tree analysis, target correlation analysis

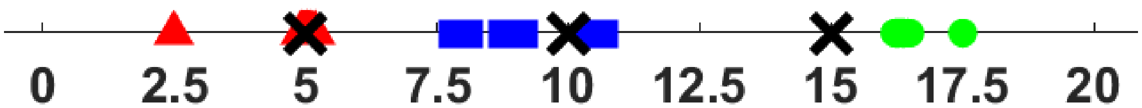
Feature transformation: discretization

Examples (**unsupervised** discretization)

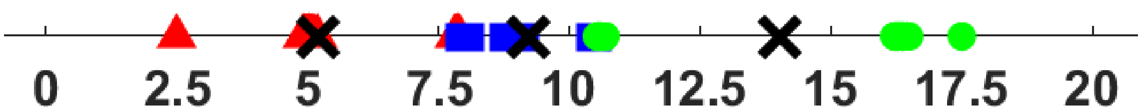
Data consists of four groups of points and two outliers



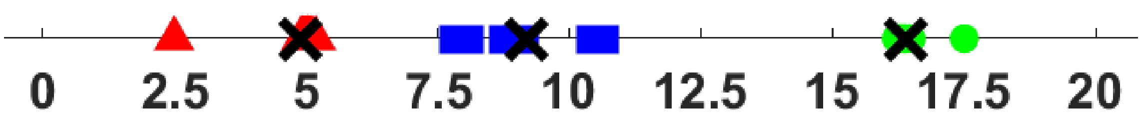
Equal interval width approach to obtain 3 values



Equal frequency approach to obtain 3 values



K-means approach to obtain 3 values



Feature transformation: binarization

Binarization maps a categorical nominal feature into one or more binary features (numeric)

- Binarization: if the feature has only 2 possible nominal values, it can be transformed into 1 binary feature
  - survived: yes/no -> survived: 1/0
- One-Hot Encoding: creates a binary variable for each category

Hair color	*Hair_brown*	*Hair_blond*	*Hair_black*
Brown	1	0	0
Blond	0	1	0
Black	0	0	1

Disadvantages: the number of features and sparsity on data increase

Feature transformation: ordinal features

Transformation of ordinal features

- Map the features into an interval [0,1]
- Map the features into integers
  - *Examples:*
    - *{ Good, Very Good and Excellent } → {0, 1, 2}*
    - Likert scale applied by social sciences

      {Extremely Unlikely (1) to Extremely Likely (7) } → {1, 2 . . . , 7}

Categorical ordinal

Sizes	A
Very Small	1
Small	2
Medium	3
Large	4
Very Large	5

- Ordinal Encoder: Encode categorical features as an integer array, allows to specify the order of the categories

Data pre-processing

"At the end of the day, some machine learning projects succeed and some fail.
What makes the difference?
Easily the most **important** factor is the **features used**."
Pedro Domingos, in "A Few Useful Things to Know about Machine Learning".
DOI:10.1145/2347736.2347755

- Feature transformation
- Feature Engineering
  - Creation of features
- Data reduction

Data pre-processing: feature engineering

Creation of features maps the entire set of values of given features to a new set of replacement values such that each old value can be identified with one of the new values

- The process of using domain knowledge of the data to create features that might help when solving the problem.
- New features that can capture the important information in a data set much more efficiently than the original features.

Feature engineering: creation of features

Express known relationships between existing variables

- create ratios and proportions like credit card sales per person
- the average web session duration per user, frequency of access, ...

Example: features X_1: distance and X_2: duration

create speed $X_3 = X_1 / X_2$

Express known case dependencies

- Create features using the information about case dependencies relationships (time, space, space-time)

Feature engineering: creation of features

Express known case dependencies

Time series

- create feature that represent relative values instead of absolute values, so to avoid trend effects.

$$y_t = \frac{x_t - x_{t-1}}{x_{t-1}}$$

- Time Delay Embedding - create features whose values are the value of the same variable in previous time steps
  - Standard tools will be able to model the time relationships

X_{t-3}	X_{t-2}	X_{t-1}	X_t
X_{t_1}	X_{t_2}	X_{t_3}	X_{t_4}
X_{t_2}	X_{t_3}	X_{t_4}	X_{t_5}
…			
$X_{t_{n-3}}$	$X_{t_{n-2}}$	$X_{t_{n-1}}$	X_{t_n}

- Similar “tricks” can be done with space and space-time dependencies

Data pre-processing

"At the end of the day, some machine learning projects succeed and some fail.
What makes the difference?
Easily the most **important** factor is the **features used**."
Pedro Domingos, in "A Few Useful Things to Know about Machine Learning". DOI:10.1145/2347736.2347755

- Feature transformation
- Feature Engineering
- **Data reduction**
  - Numerosity reduction
  - Dimensionality reduction
    - Feature selection
    - Singular Value Decomposition (SVD)
    - Principal Component Analysis (PCA), Kernel PCA
    - Linear Discriminant Functions

Data pre-processing: data reduction

- Numerosity reduction
- Dimensionality reduction
  - Feature selection
  - Singular Value Decomposition (SVD)
  - Principal Component Analysis (PCA)
  - Kernel PCA
  - Linear Discriminant Functions

Data reduction: numerosity reduction

What?

- Obtain a reduced representation of the data set
  - **much smaller** in volume but yet produces *almost* the same analytical results

Why?

- A data set may store terabytes of data
  - Complex analysis may take a very long time to run on the complete data set

Methods

- Regression and Log-Linear Models
- Histograms, clustering, sampling
- Data cube aggregation
- Data compression

Numerosity reduction: sampling

What?

Obtaining a **smaller data set** to represent the whole **data set of size *N***

Why?

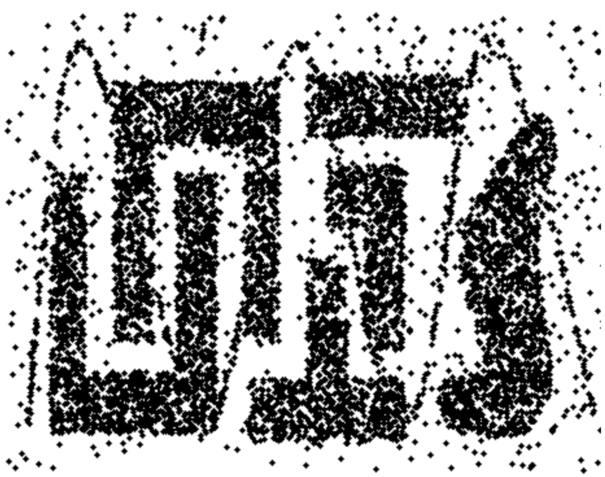
- Processing the entire set of data of interest is too expensive or time consuming
- To allow a mining algorithm to run in complexity that is potentially sub-linear to the size of the data

It is often used for both the preliminary investigation of the data and the final data analysis

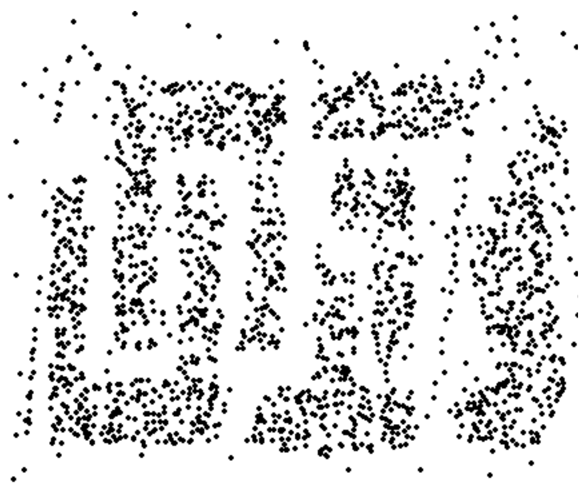
Numerosity reduction: sampling

Key principle: Choose a **representative** subset of the data

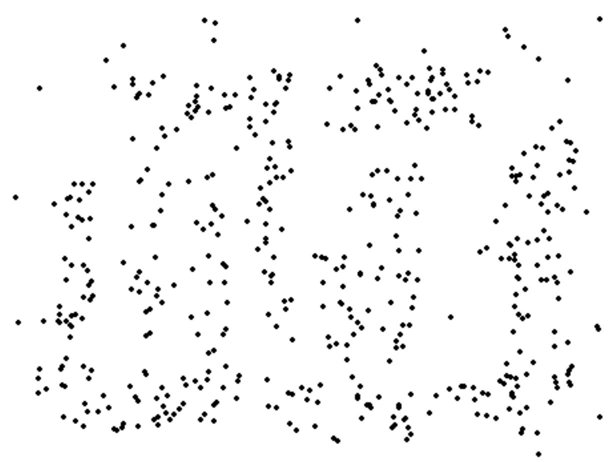
- using a sample will work almost as well as using the entire data set, if the sample is representative
- a sample is representative if it has approximately the same properties (of interest) as the original set of data



8000 points



2000 Points



500 Points

Numerosity reduction: types of sampling

Simple random sampling

- Equal probability of selecting any particular object
- **Sampling without replacement**
  - Once an object is selected, it is removed from the population
- **Sampling with replacement**
  - A selected object is not removed from the population
  - The same object can be picked up more than once

Stratified sampling

- Split the data into several partitions
- Draw random samples from each partition

(proportionally, i.e., approximately the same percentage of the data)

Incremental sampling

Data reduction: dimensionality reduction

Key questions

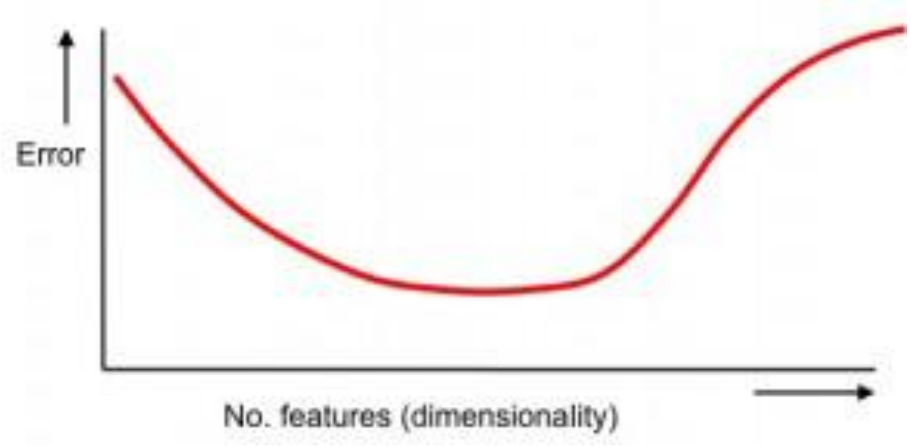
- How many features are required?
- Is there a point where we have too many features?
- How do we know beforehand which features will work best?
- What happens when there is feature redundance/correlation?

Dimensionality reduction: The curse of dimensionality

- When dimensionality of feature space increases, the number of possible combinations of feature values increases exponentially
- The data becomes increasingly sparse in the space that it occupies
- We may assume that the more details (features) of the object we collect, the better description of the situation we have at hand
  - Counter-intuitively, it is not valid

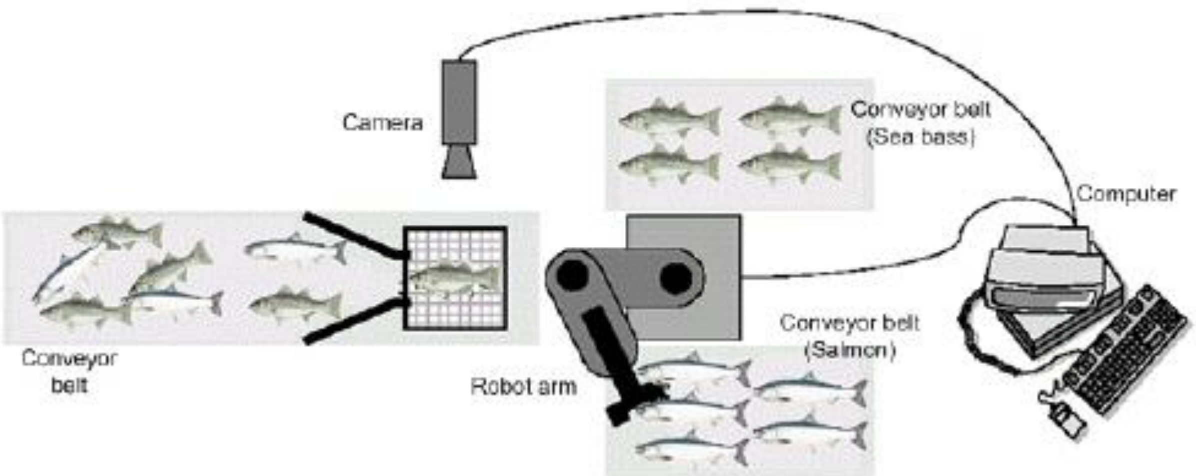
Dimensionality reduction: The curse of dimensionality

- There is a certain point after which adding new details becomes useless, and moreover, they may work against your model.
- In very high dimensional data many data mining algorithms do not work effectively.



- For example, distance between points, which is critical to some algorithms, becomes less meaningful.

Dimensionality reduction: Toy example sorting fish



...

Feature extraction

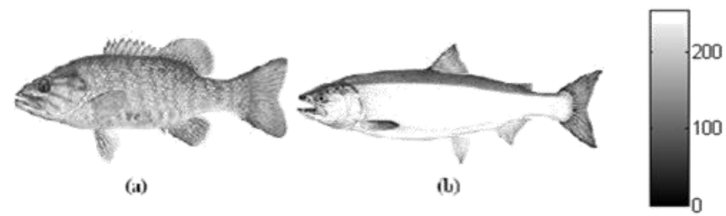
Classification

Sea bass **Salmon**

GOAL: A decision is made by processing the image of a single fish taken by the camera

Dimensionality reduction: Toy example sorting fish

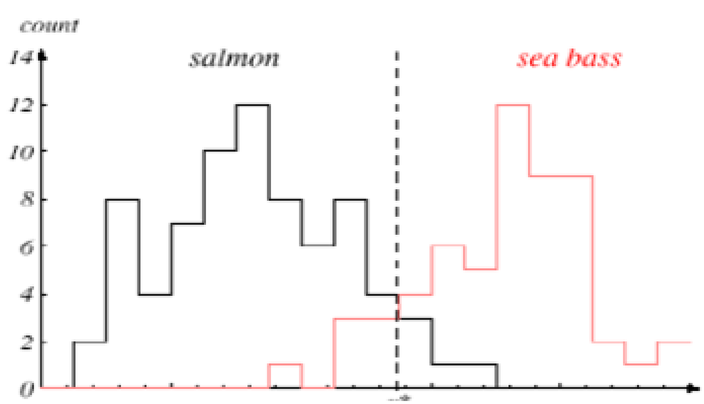
one feature



Intensity of the two fish images are in different ranges:

salmon is typically darker

Histogram of intensity values of a set of fishes

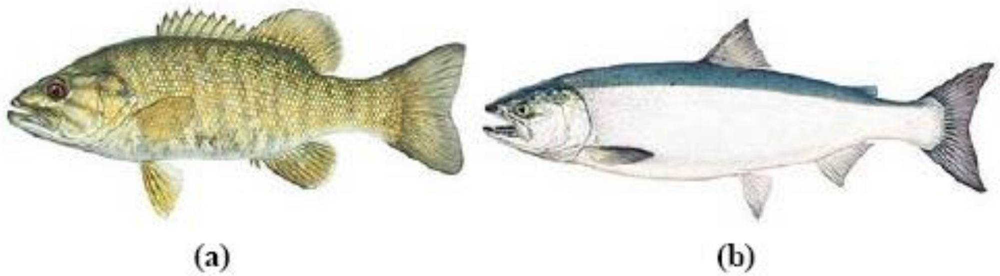


With histogram:

- GOAL of learning: estimate a threshold (model)
- Future decisions are made based on the learned threshold value

Dimensionality reduction: Toy example sorting fish

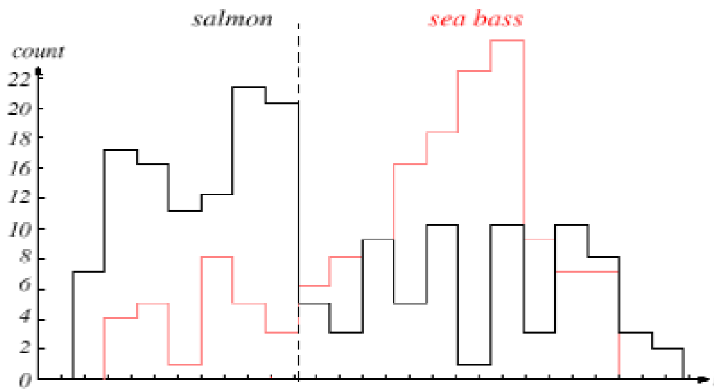
two features



Length of the two fish images is different:

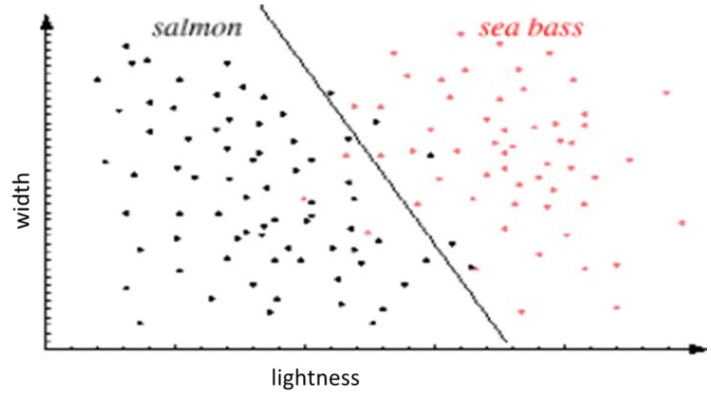
<u>sea bass is typically wider</u>

Histogram of the length of a set of fishes



Scatter plot:

values of the two features of the data set



In the 2D feature space: the two categories occupy distinct regions

With scatter plot:

- GOAL of Learning: find out the separation surface
- Future decisions are made based on the learned surface

Dimensionality reduction: Toy example sorting fish

The two features obviously separate the classes much better than one alone. This suggests adding a third feature. And a fourth feature. And so on.

Key questions

- How many features are required?
- Is there a point where we have too many features?
- How do we know beforehand which features will work best?
- What happens when there is feature redundance/correlation?

Dimensionality reduction

Purpose:

- Avoid **curse of dimensionality**
- Help **eliminate irrelevant** features or reduce noise
- **Reduce** amount of **time** and **memory** required by data mining algorithms
- Allow data to be more **easily visualized**
- May increase **interpretability** (e.g., avoiding huge DT)

- **How**
  - Feature selection
  - Singular Value Decomposition (SVD)
  - Principal Components Analysis (PCA), Kernel PCA
  - Linear Discriminant Analysis (LDA)

Dimensionality reduction: feature selection

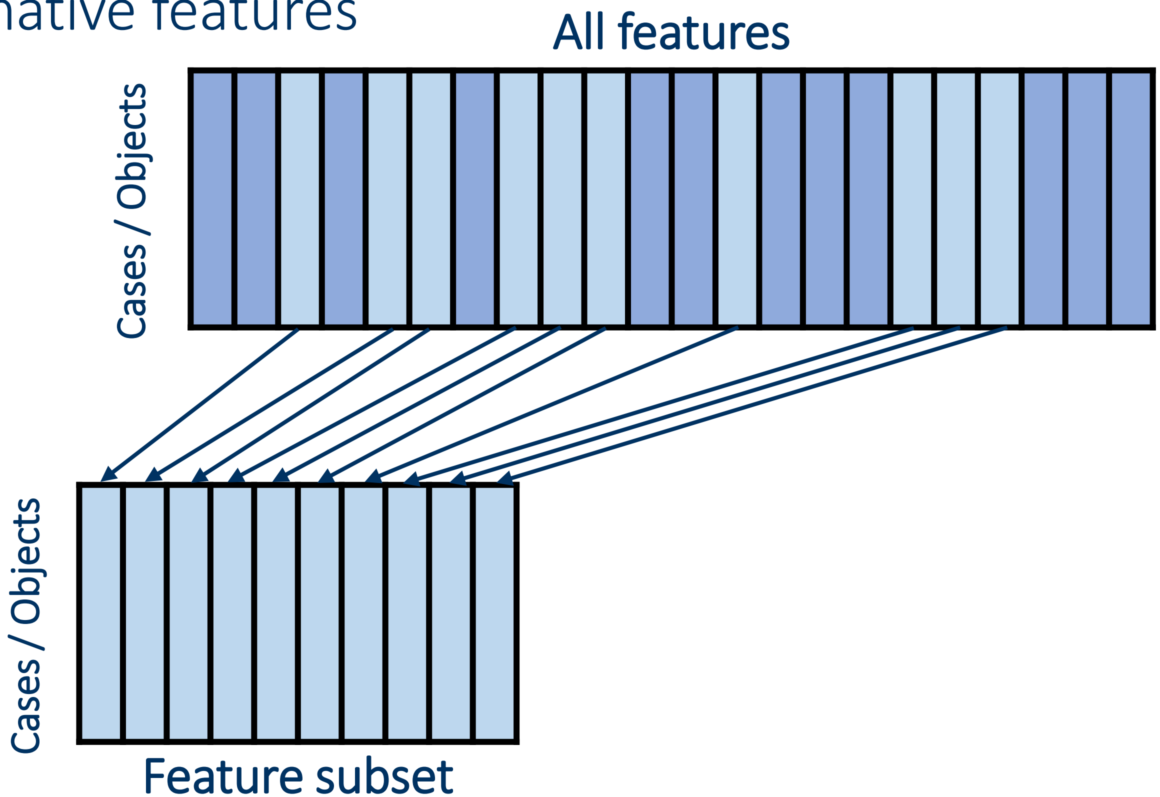
- Discard Redundant Features
  - Duplicate much or all the information in one or more other attributes
    - Example: purchase price of a product and the amount of sales tax paid

- Discard Irrelevant features
  - Do not contain useful information for the data mining task at hand
  - Example: students' ID is often irrelevant to the task of predicting grades

Dimensionality reduction: feature selection

WHY?

- to achieve dimension reduction
- to construct more accurate classification models
- to find the more informative features



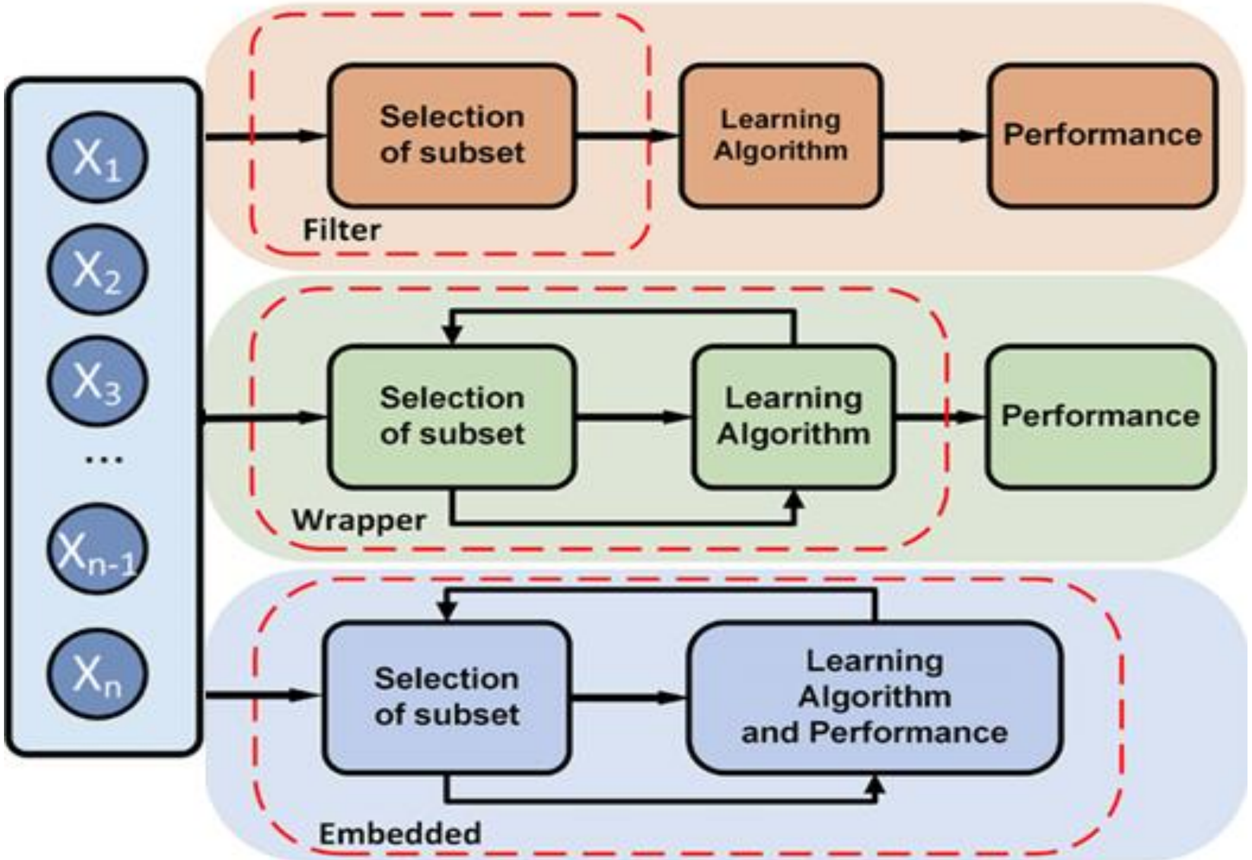
Dimensionality reduction: feature selection

Feature selection methods

- **Unsupervised**
  - Filters
    - Drop incomplete features
    - Drop features with near-zero variance
    - Drop feature based on pairwise-correlation
- **Supervised**
  - Filters
    - Feature score
  - Wrappers
    - Forward selection
    - Backward selection
    - Recursive selection
  - Embedded
    - Decision tree
    - Lasso

Dimensionality reduction: feature selection

Supervised feature selection methods



Computational Diagnostic Techniques for Electrocardiogram Signal Analysis. Liping Xie Zilong, Li Yihan Zhou, Jiaxin Zhu. ***Sensors***. 2020 **(adapted)**

Dimensionality reduction: feature selection

Filter methods

- Selects a subset of variables independently of the classification model
  - Removing features with low variance (rank by cut-off)
  - Pairwise-correlation-based (rank by cut-off)
  - Ranking features by relevancy measure, depending on relationship with the target

Wrapper methods

- Selects a subset of variables taking into account the classification
  - Search for optimal subset of features

Embedded methods

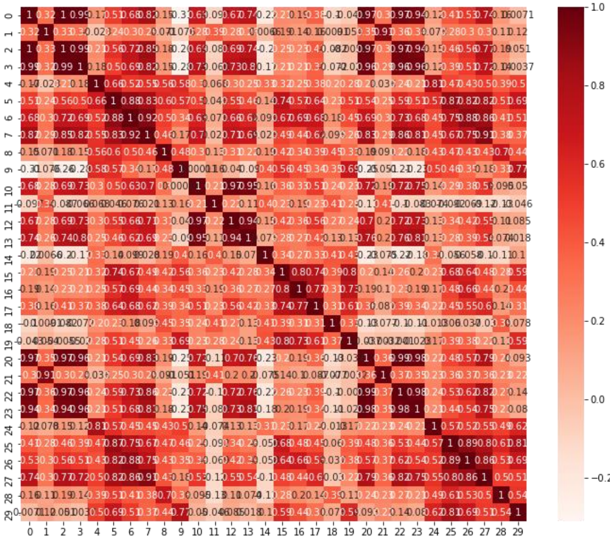
- The feature selection method is built in the classification model (or rather its training algorithm) itself (e.g. decision trees)

Feature selection: filter methods

Unsupervised feature selection methods

Filter methods: Non-iterative process

- Selects a subset of variables independently of the classification model
  - Drop incomplete features (e.g., with great number of missing values)
  - Drop features with near-zero variance (rank by cut-off)
  - Drop feature based on pairwise-correlation (rank by cut-off)
    - Eliminate one feature of a pair if correlation coefficient is larger than a threshold
    - The user chooses threshold (usually larger than 0.8)



Feature selection: filter methods

Supervised feature selection methods

Filter methods: Non-iterative process

- Selects a subset of features independently of the ML algorithm
  - Ranking features by **relevancy** measure, depending on relationship with the target
  - Select the features which are highly dependent on the target
  - Applied in parallel to all features providing scores

Select k best features based on a **relevance** measure to rank the features

- F-test, chi-square, mutual-information

All features → Filter → Feature subset $\xrightarrow{\text{ML alg}}$ Model

Feature selection: filter methods

Ranking features by **relevancy** measure, depending on relationship with the target

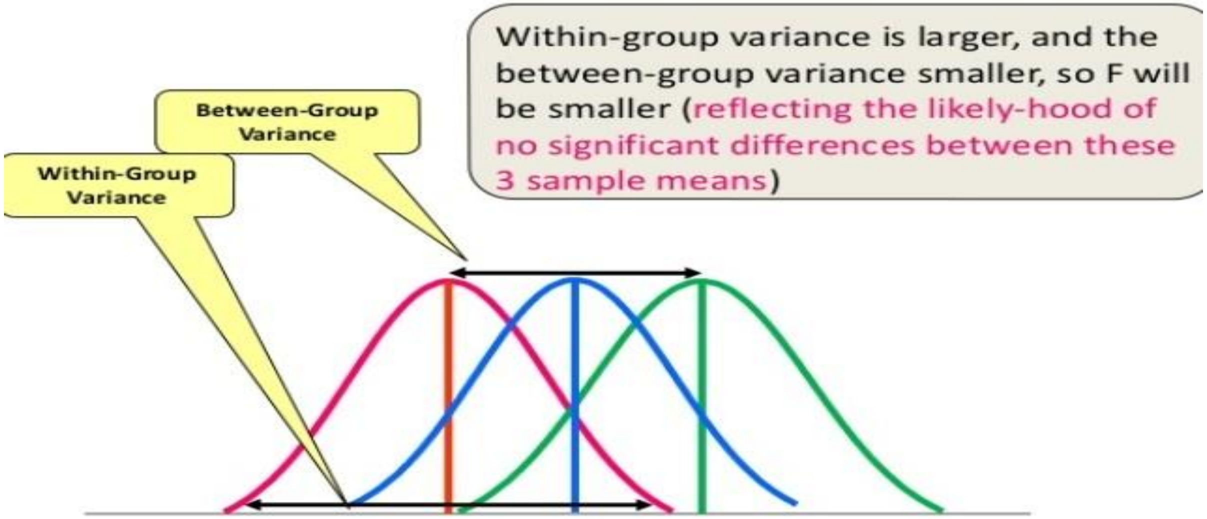
- Select the features which are highly dependent on the target
- Applied in parallel to all features providing scores

Numeric feature: Hypothesis test (measures if the mean of one feature is equal for all groups/classes of the target)

- $F-value$ or $t-value$ are used to rank the features
  - select the K features corresponding to K largest F-value

Feature selection: filter methods

Example: F-value qualitative interpretation



Selection Procedure:

- The groups: the categorical variable to be predicted on a classification task
- Calculate F value for all features
- Choose the K features corresponding to K largest F-value

Feature selection: filter methods

Ranking features by **relevancy** measure, depending on relationship with the target

- Select the features which are highly dependent on the target
- Applied in parallel to all features providing scores

Categorical feature: Hypothesis test (measures if target and feature are independent)

- $\chi^2 - value$ is used to rank the features
  - select the K features corresponding to K largest chi-square values

Feature selection: filter methods

Ranking features by **relevancy** measure, depending on relationship with the target

- Select the features which are highly dependent on the target
- Applied in parallel to all features providing scores

Disadvantages

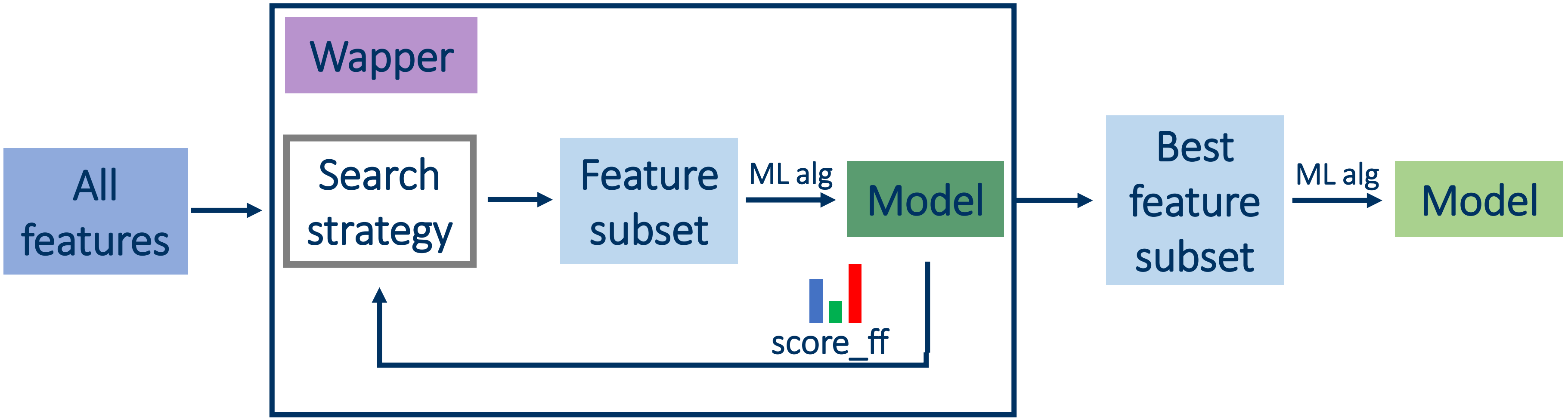
- Fail to recognize that a feature is important in combination with another variable
- Select a group of variables that are dependent and carry similar (or the same) information about the class label
- Often causes overfitting

Feature selection: wrapper methods

Supervised feature selection methods

Wrapper methods: Wrapper around learner

- Select features
- Evaluate learner (e.g., cross-validation)



Feature selection: wrapper methods

Supervised feature selection methods

Wrapper methods: Wrapper around learner

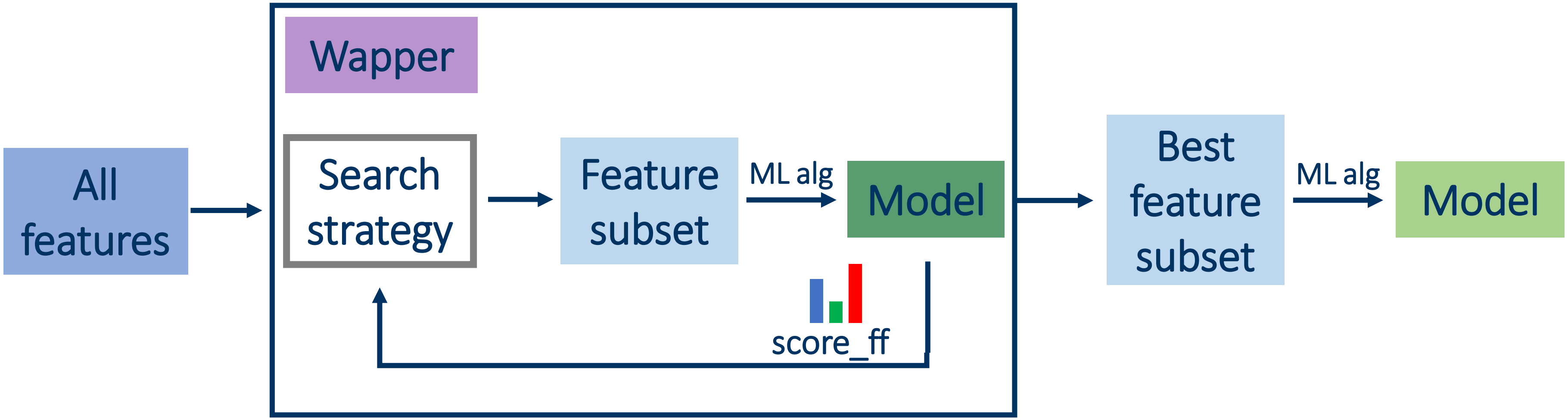
- Select features
- Evaluate learner (e.g., cross-validation)
  - Iterative procedure: select 1 attribute, remove, repeat / select 1, add, repeat

  or
  - Recursive procedure: attributes are recursively removed from current set

- Several subsets of features are generated and tested on the particular model

Linear SVM and tree-based learners are often used

Feature selection: wrapper methods



Disadvantages

- Learner-dependent (selection for specific learner)
- Expensive
  - Greedy search: O(k^2) for k attributes
  - When using a prior ranking (only find cut-off): O(k)

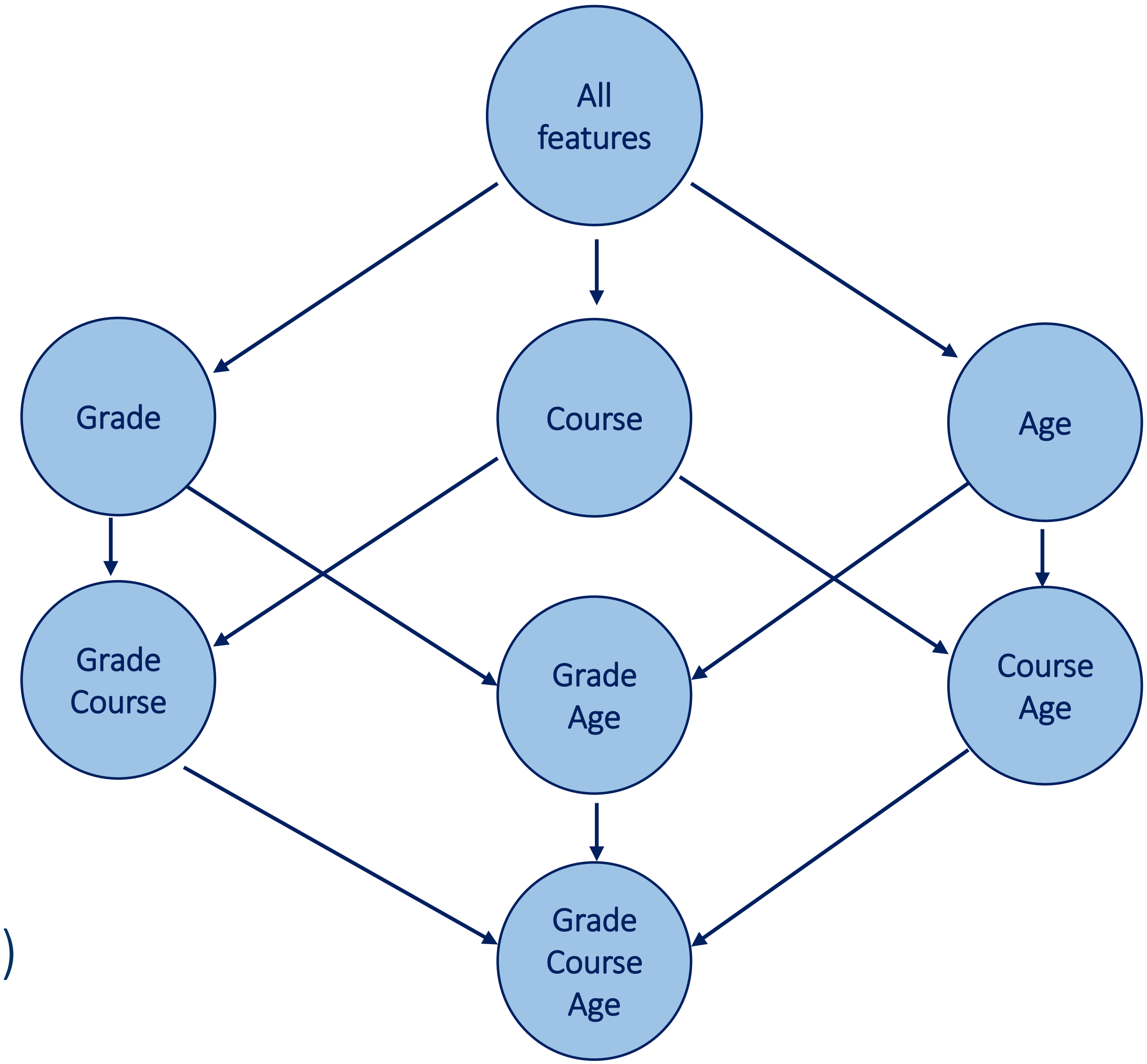
Feature selection: wrapper methods

Example

Greedy search

Forward selection
(add one, select best)

Backward elimination
(remove one, select best)



Feature selection: embedded methods

Supervised feature selection methods

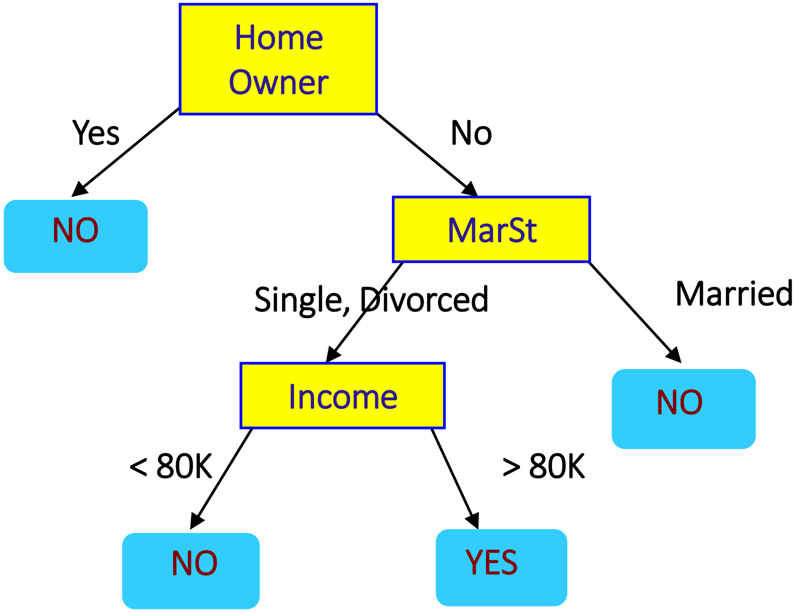
Embedded methods

- The classification method includes feature selection
  - models that show importance of features
    - Tree-based learners

Training Data

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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

Model: Decision Tree



(**Introduction to Data Mining**, Tan et al.)

Dimensionality reduction: feature selection

Filter vs Wrapper Methods

- Filter Methods
  - faster, as they do not involve training the models
  - use statistical methods for evaluation of a subset of features
  - fail to recognize importance of combined features
  - select features that carry similar information about the target

- Wrapper Methods
  - computationally more expensive
  - use model performance estimation strategies
  - provide the best subset of features
  - ML algorithm dependent

Dimensionality reduction

Instead of selecting features, we can replace by “new” features

- A new (smaller) set of features where most of the "information“ on the problem is still expressed.
- Sometimes, the correlation among the features is not perfect (redundant) but there may exist significant dependencies.

Dimensionality reduction

Main methods:

- Singular Value Decomposition (SVD)
- Principal Component Analysis (PCA)
- Kernel PCA
- Linear Discriminant Functions
- Others: supervised and non-linear techniques

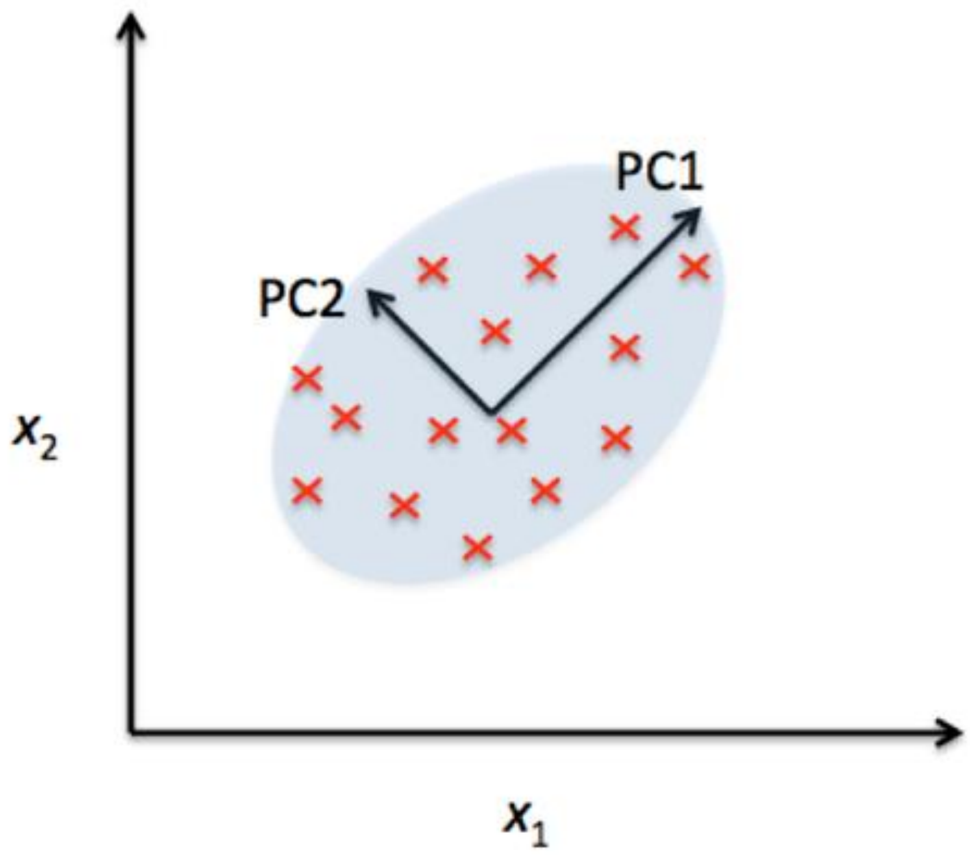
Dimensionality reduction: PCA

Principal Component Analysis (PCA)

- Unsupervised data reduction technique
- Reduces high dimensions into low dimension subspace
- Projects the data points onto new axes such that these new components **carry most of the essential information** of all the features
- These new components are a **linear combination of all the features** and the components thus formed are nothing but the **eigenvectors** which are now called the **Principal components**
- The eigenvalue corresponding to each of these eigenvectors will tell us about **how much variation** in the data has been captured by that **particular Principal Component**
- Principal components are **orthogonal to each other** and are **uncorrelated that increases maximum variance**. As they are uncorrelated it solves the problem of multicollinearity

Dimensionality reduction: PCA

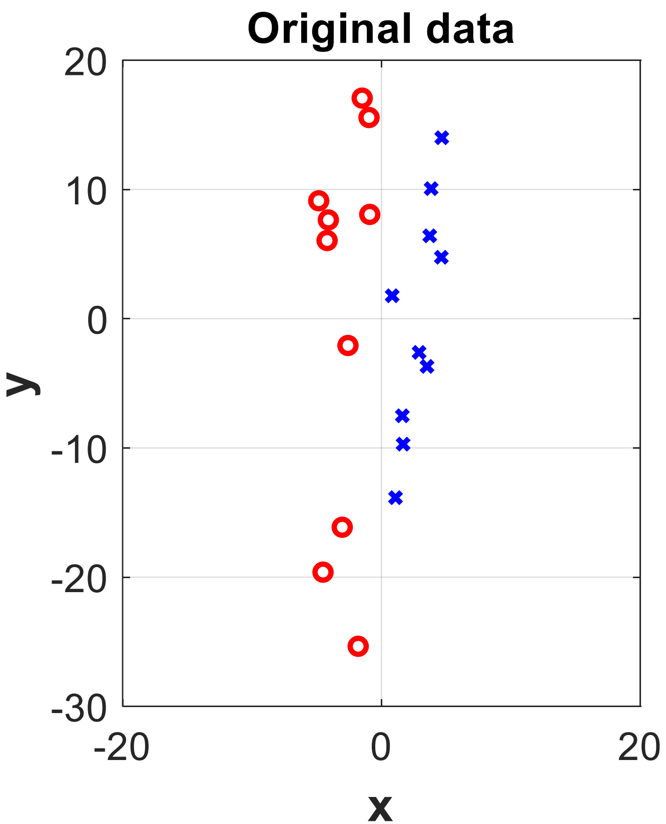
Principal Component Analysis (PCA)



Dimensionality reduction: PCA

Principal Component Analysis (PCA)

Target categories/classes are not taken into consideration



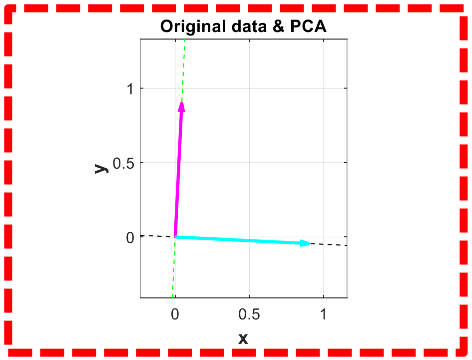
Dimensionality reduction: PCA

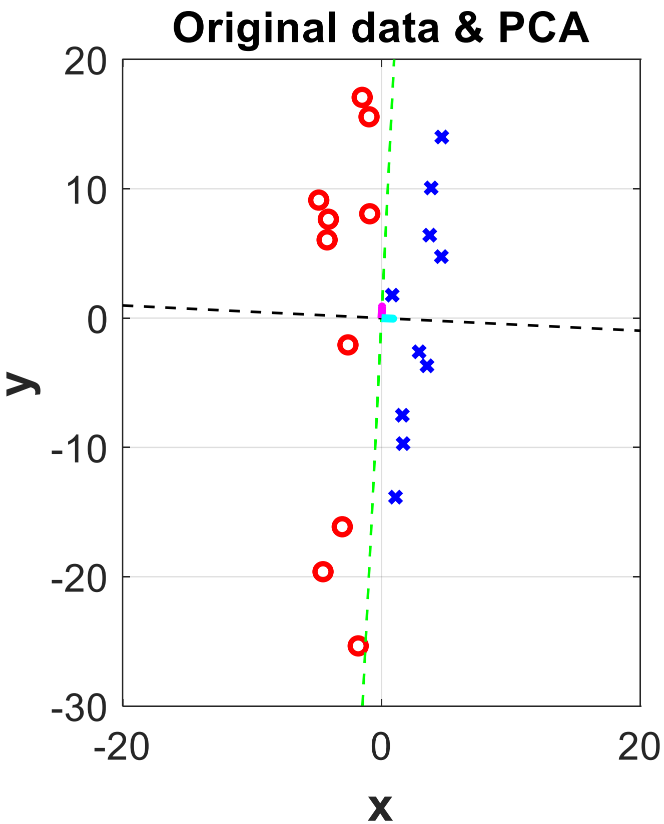
Principal Component Analysis (PCA)

Eigenvectors

ev1 = [0.0488   0.9988]

ev2 = [0.9988   -0.0488]

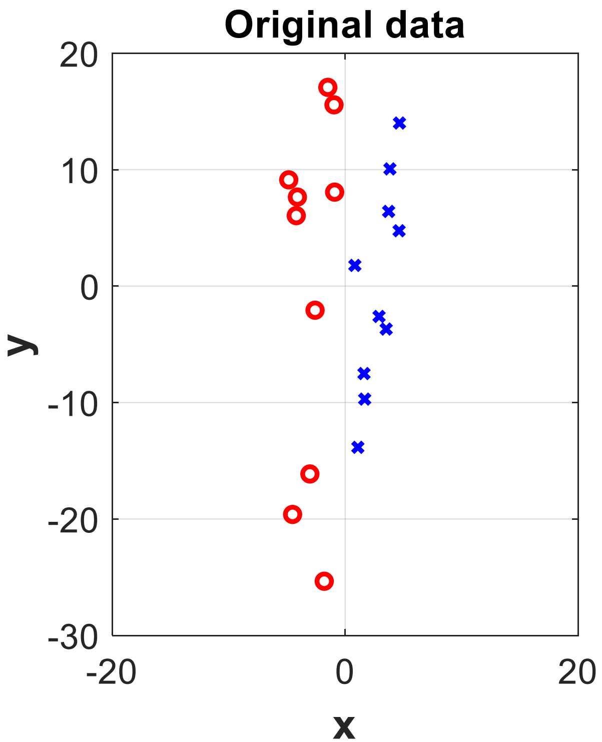


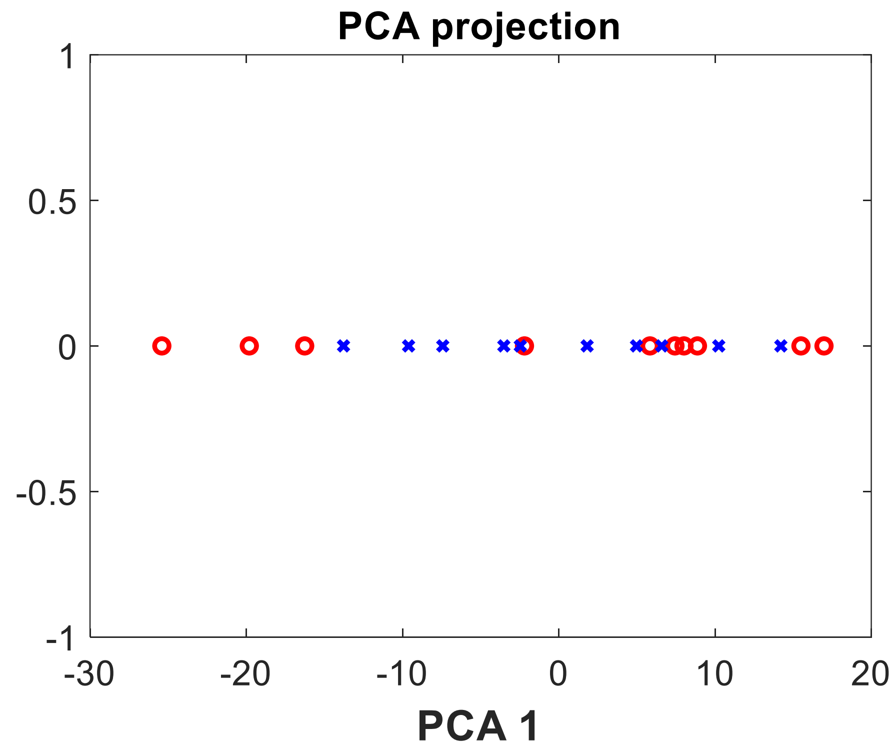


Dimensionality reduction: PCA

Principal Component Analysis (PCA)

PC1 = [0.0488   0.9988]



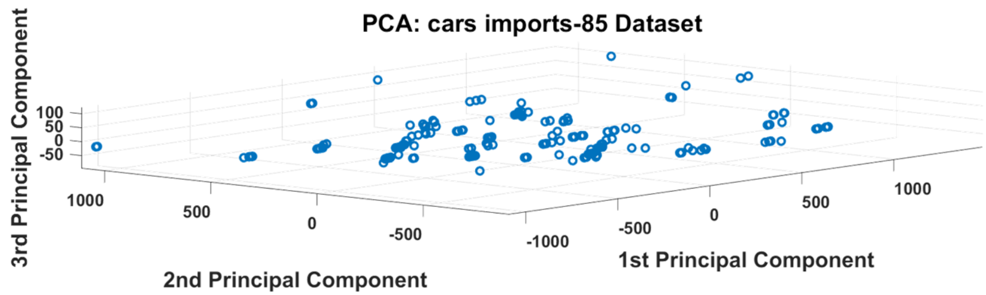


Dimensionality reduction: PCA example

Principal Component Analysis (PCA)

Data matrix

- 13 features
- 205 objects



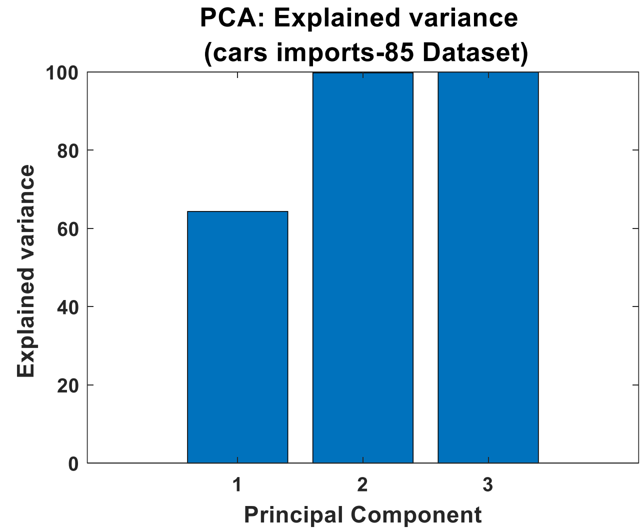
Explained variance

PC1: ~64.3%

PC2: ~35.4%

PC3: ~0.15%

remain: (...)



The first three components explain 99.95% of all variability

Dimensionality reduction: PCA

Principal Component Analysis (PCA)

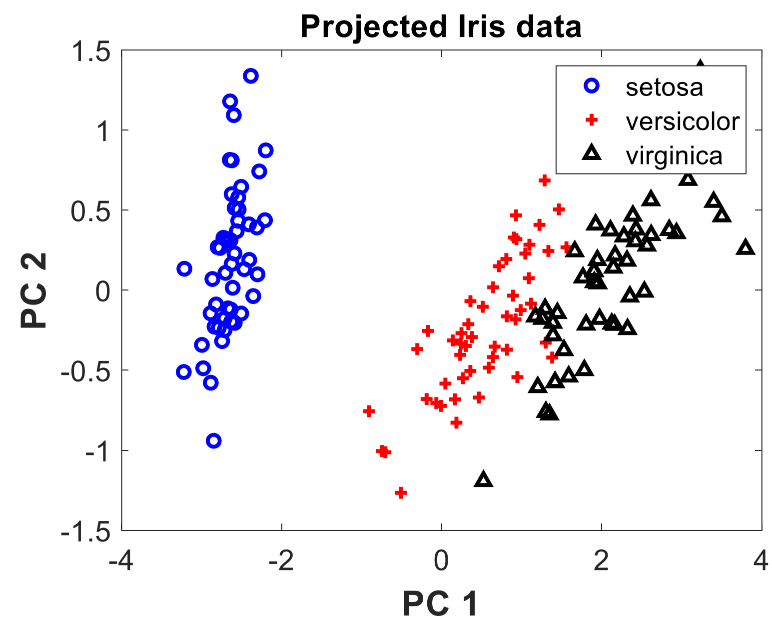
- Find a **first linear combination** that **better** captures the **variability** in the data
- Move to the **second linear combination** to try to capture the variability not explained by the first one
- Continue until the set of new variables explains **most of the variability** (common values are 80% to 95%)

	PC 1	PC2	PC 3	PC 4
Sepal_Length	0.361	0.657	-0.582	0.315
Sepal_Width		0.730	0.598	-0.320
Petal_Length	0.857	-0.173		-0.480
Petal_Width	0.358		0.546	0.754

PC1 = 0.361 x Sepal_Length

+ 0.857 x Petal_Length

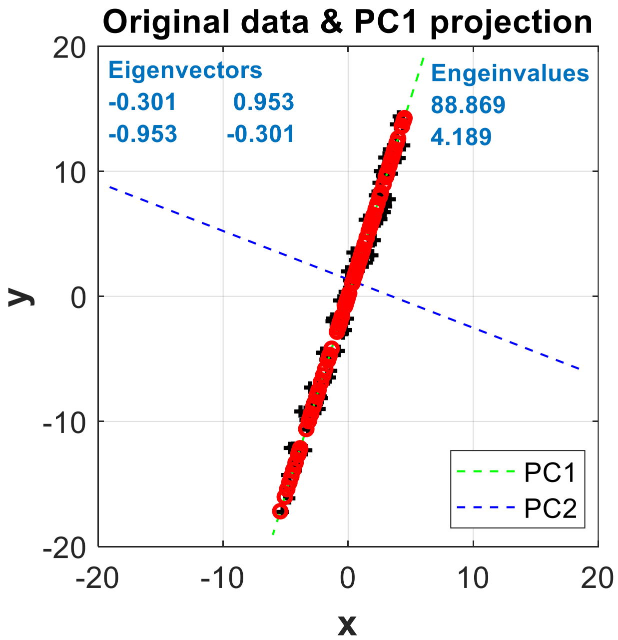
+ 0.358 x Peatl_Width



Dimensionality reduction: PCA model

The eigenvectors form a new basis

- Basis vector model for the data: the eigenvectors $\mathbf{U}=[\mathbf{u_1}, \ldots,\mathbf{u}_D]$
- $\mathbf{u}_i$ is related to the entry (i, i) of the diagonal matrix $\mathbf{\Lambda}$ with eigenvalues: $\boldsymbol{\lambda}_i$



Basis vector model $\mathbf{u_1}$ adapted to the spread of the data

Dimensionality reduction: PCA model

The PCA model is the D x D eigenvector matrix $\mathbf{U}$

- SVD decomposition of centered data matrix $\mathbf{Z}$
- Eigendecomposition of the scatter matrix or covariance matrix or kernel matrix

The eigenvalues (or singular values) allow to select the **columns of eigenvector** matrix ($\mathbf{U}$) to project data $\mathbf{Z}$

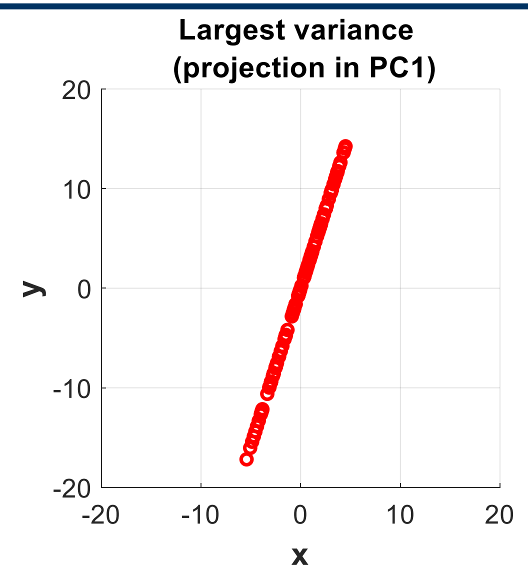
Dimensionality reduction: PCA projections

Projection

Data (the rows of data matrix) can be projected onto m^{th} eigenvector $\mathbf{u}_m$

$$\mathbf{P}_m = \mathbf{Z}\,\mathbf{u}_m$$

- $\mathbf{P}_m$ projections

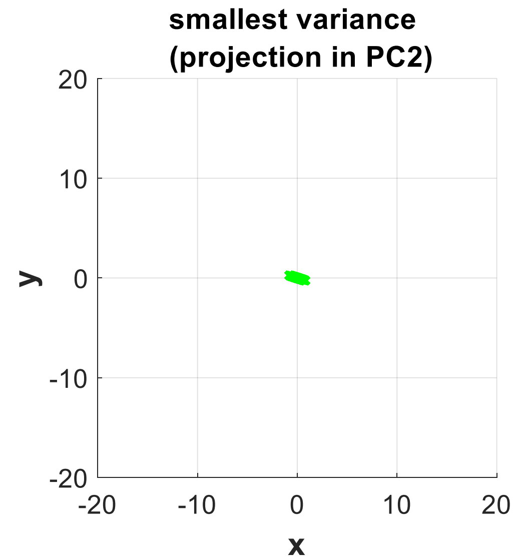


Reconstruction

With projections $\mathbf{P}_m$ and performing

$$\tilde{\mathbf{Z}} = \mathbf{P}_m\,\mathbf{u}_m^T$$

Is possible to reconstruct to the original dimension, e.g, an approximation $\tilde{\mathbf{Z}}$ original centered values $\mathbf{Z}$



Dimensionality reduction: PCA projections

PCA MODEL: assuming that

- **Eigenvalue matrix** (or singular value matrix) has diagonal entries (i, i) ordered in **decreasing order**:

$$\boldsymbol{\lambda_1 > \lambda_2 > \ \dots > \lambda_L > \ \dots > \ \lambda_D}$$

- The columns of eigenvector matrix $\mathbf{U}$ is formed with **D** eigenvectors
  - *i*th column is related with the corresponding eigenvalue $\boldsymbol{\lambda_i}$

Dimension reduction occurs by **projecting** the data onto the **first *L* columns** of $\mathbf{U}$

- forming a *D x L* matrix $\mathbf{U}_{\boldsymbol{L}}$

Note: user assigns *L* or denes a criterium (like explained variance)to calculate *L*

Dimensionality reduction: PCA projections

Considering the ordered eigenvalues

$$\boldsymbol{\lambda_1 > \lambda_2 > \ \dots > \lambda_L > \ \dots > \ \lambda_D}$$

The criterium Explained variance

$$\frac{\sum_{i=1}^{L} \lambda_i}{\sum_{i=1}^{D} \lambda_i} \mathbf{x\,100} \geq threshold$$

- where ***D*** is the total number of non-zero eigenvalues, then
  - user defines the *threshold* (common values are 80% to 95%)
  - and ***L*** is calculated according the required threshold

Dimensionality reduction: PCA – number of PC

Dimension reduction

- Using the **first L** principal directions

$$\mathbf{P} = \mathbf{Z}\,\mathbf{U}_L$$

  - $\mathbf{P}$ is a ***N x L*** matrix **->** new representation of the data with **small dimension**
    - The new feature vector has only L (< D) entries

Recovering **to the original dimension**

$$\tilde{\mathbf{Z}} = \mathbf{P}\,\mathbf{U}_L^T$$

- The recovery data can be **compared** with the original:
  - *Square error or mean square error (related with **discarded** eigenvalues)*

Dimensionality reduction: LDA

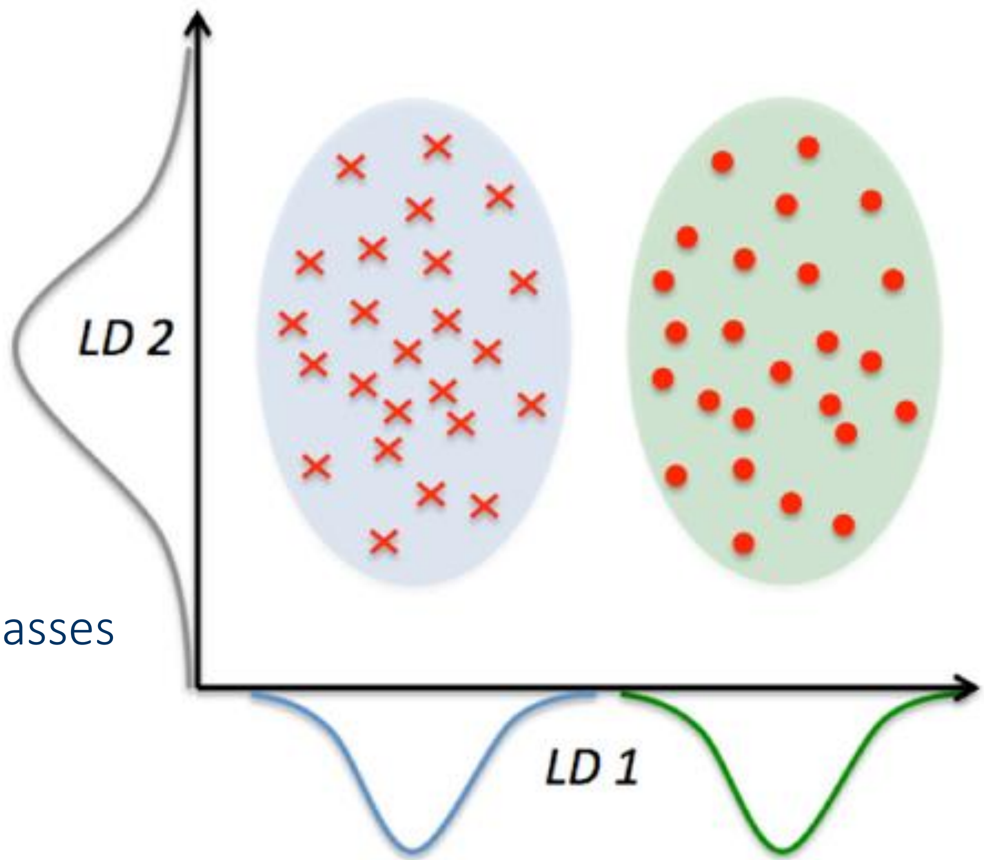
Linear Discriminant Analysis (LDA)

- Supervised data reduction technique
- Reduces high dimensions into low dimension subspace
  - dimension = #classes - 1
- Projects the data points onto new axes:
  - It **maximizes** the distance **between the means of each category** (maximizes separability among class categories)
  - It **minimizes** the **variation within each category**
- These new components are a **linear combination of all the features** that separates two or more categories of objects
- The resulting combination may be used as a linear classifier

But there are certain assumptions Linear Discriminant Analysis makes on the data set…

Dimensionality reduction: LDA

Linear Discriminant Analysis (LDA)



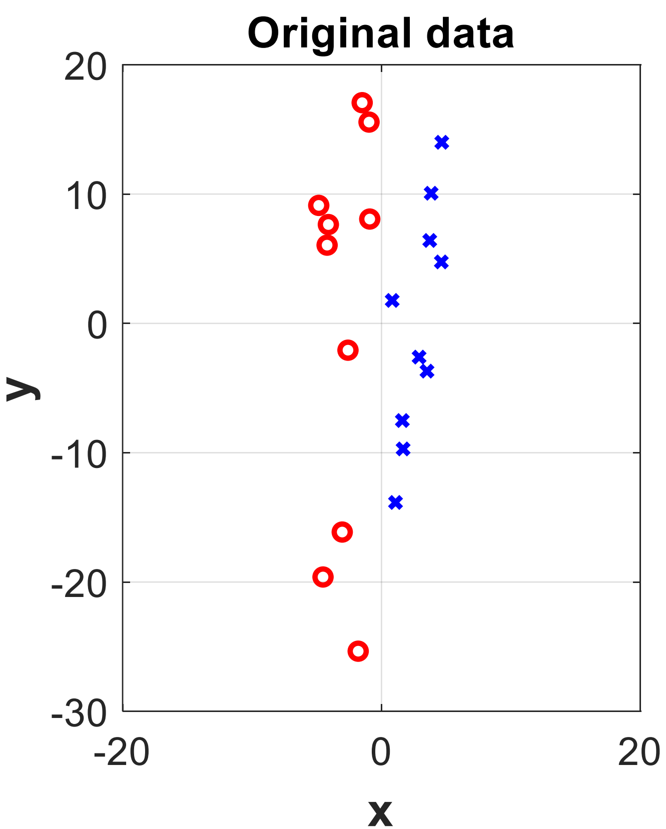
LDA makes **assumptions**

about normally distributed classes

and equal class covariances

Dimensionality reduction: LDA

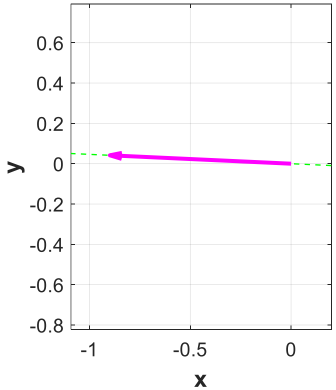
Linear Discriminant Analysis (LDA)

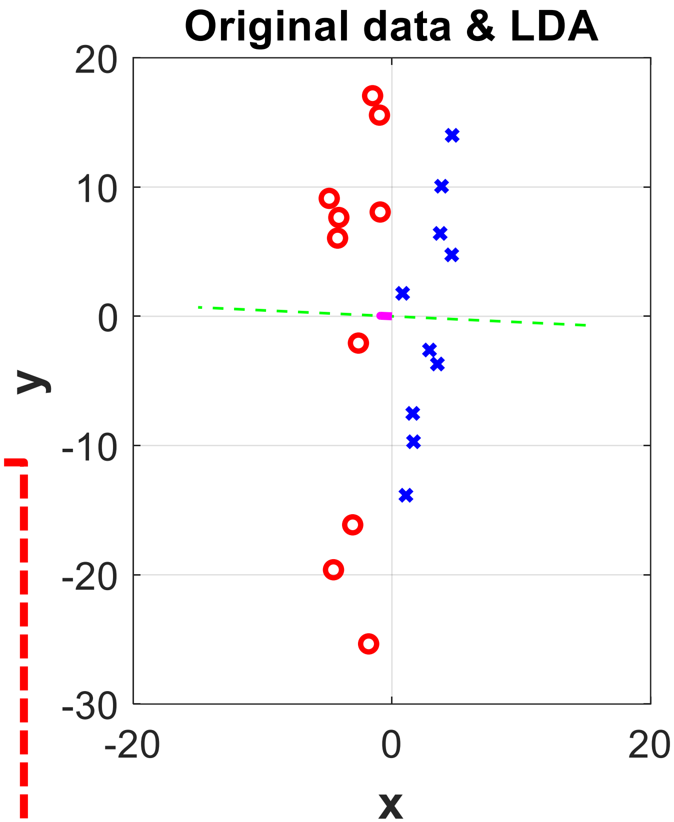


Dimensionality reduction: LDA

Linear Discriminant Analysis (LDA)

PC = [-0.9989   0.0462]

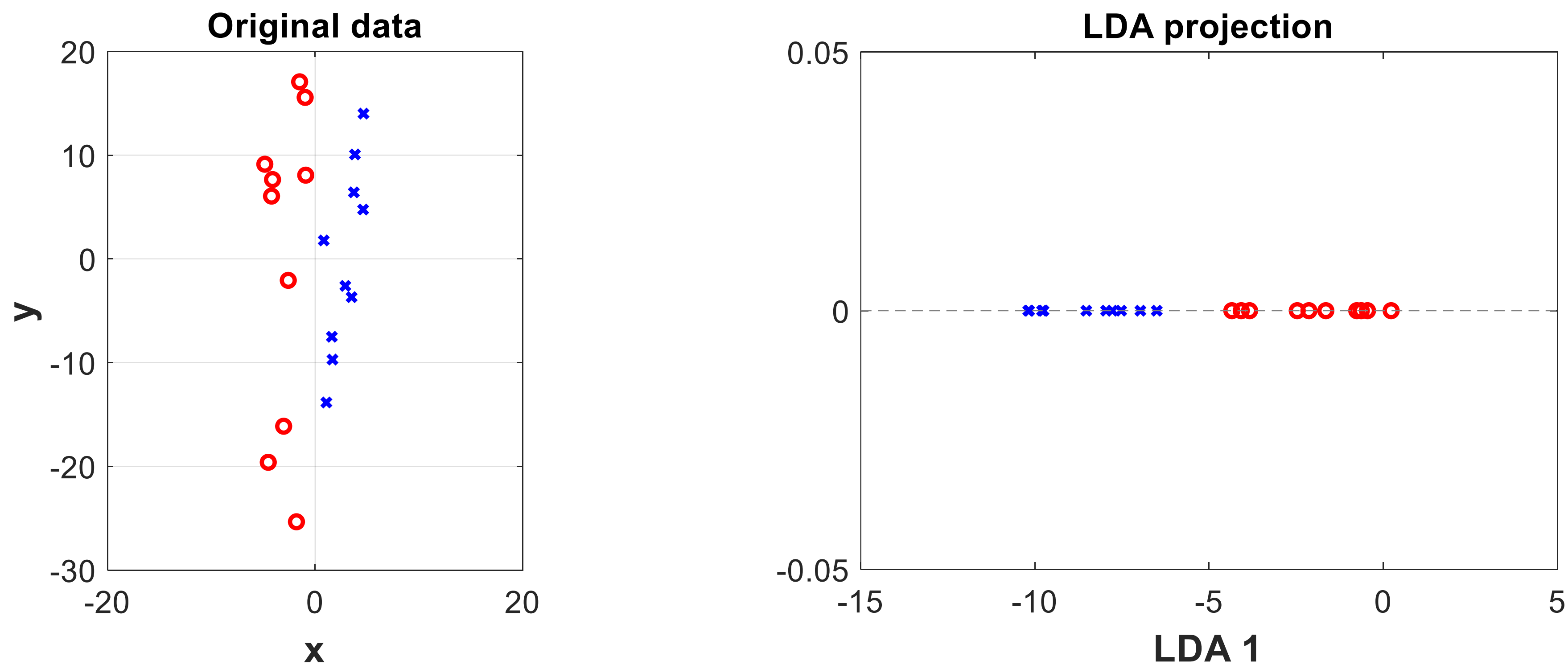




Dimensionality reduction: LDA

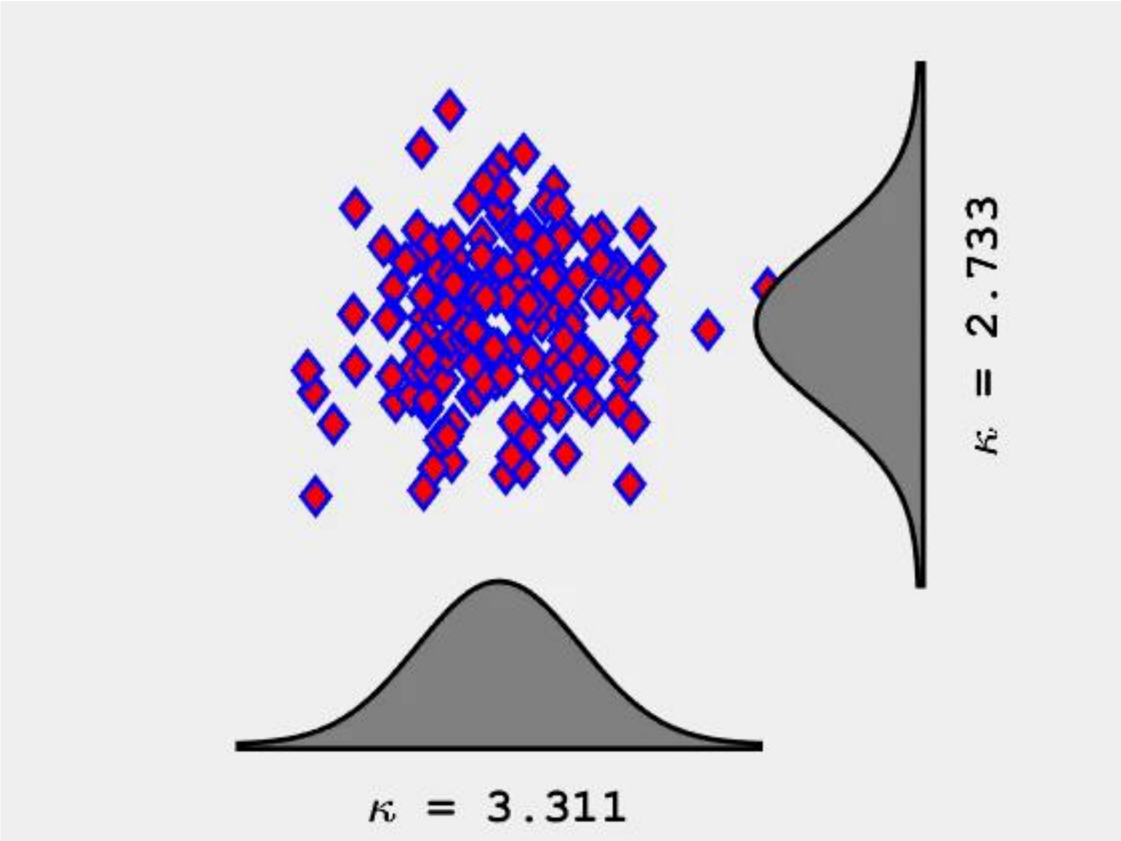
Linear Discriminant Analysis (LDA)

LDA1 = [-0.9989   0.0462]



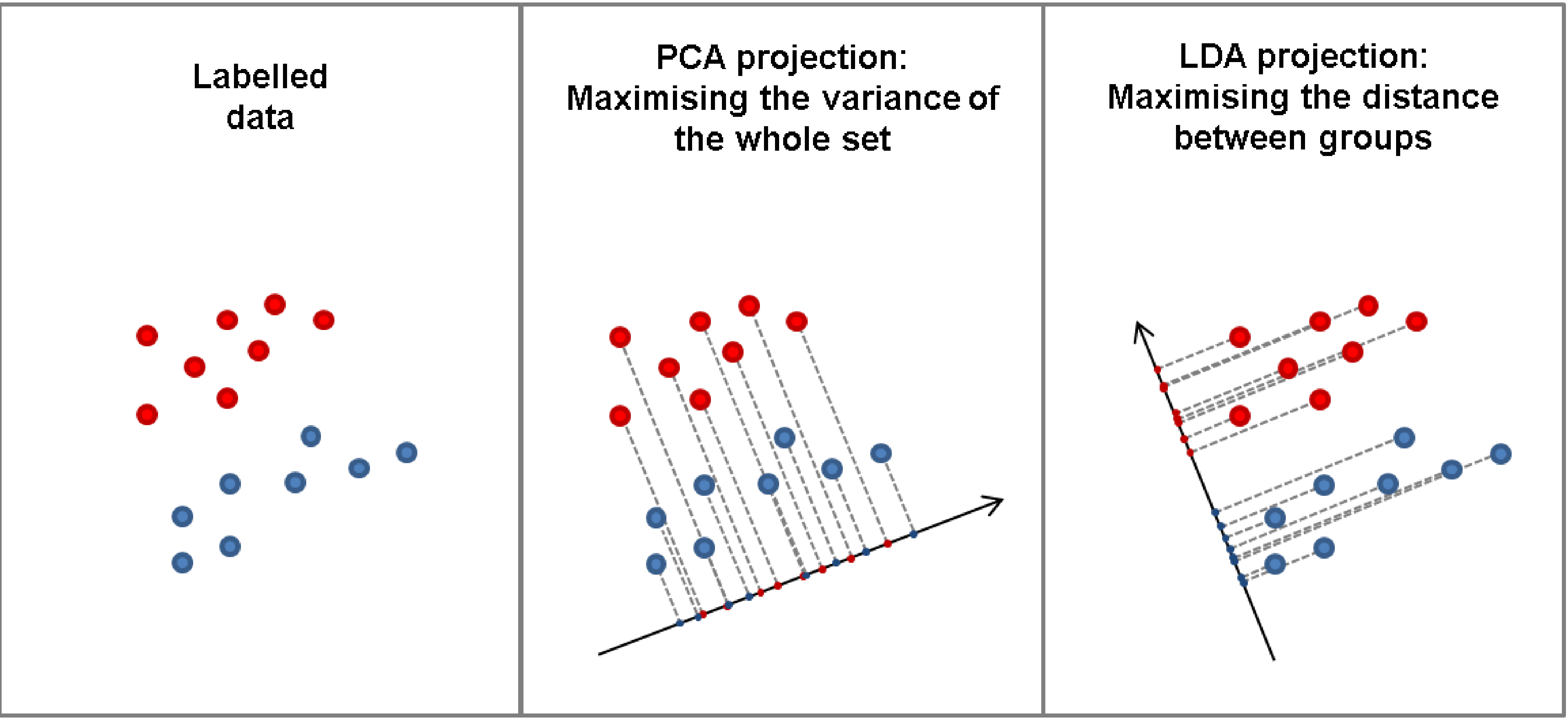
Dimensionality reduction: LDA

Linear Discriminant Analysis (LDA)



https://towardsdatascience.com/interesting-projections-where-pca-fails-fe64ddca73e6

Dimensionality reduction: PCA vs LDA



Contents

Summary

- Data quality
- Data pre-processing
  - Feature extraction
  - Data integration
  - Data cleaning
  - Feature transformation
    - Aggregation
    - Scaling
    - Discretization
    - Binarization
  - Feature Engineering
  - Data reduction
    - Numerosity reduction
    - Dimensionality reduction

Homework

- Assignment II (see eLearning)
  - Data Preparation:
    - Hands on: Handling categoric
    - Hands on: Handling missing
    - Hands on: Data Preprocessing

Bibliography

Introduction to Data Mining, Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar, *Pearson*, 2019 (chap 2)

Data Mining, the Textbook, Charu C. Aggarwal, *Springer*, 2015 (chap 2)

https://towardsdatascience.com/smarter-ways-to-encode-categorical-data-for-machine-learning-part-1-of-3-6dca2f71b159

https://sebastianraschka.com/faq/docs/lda-vs-pca.html