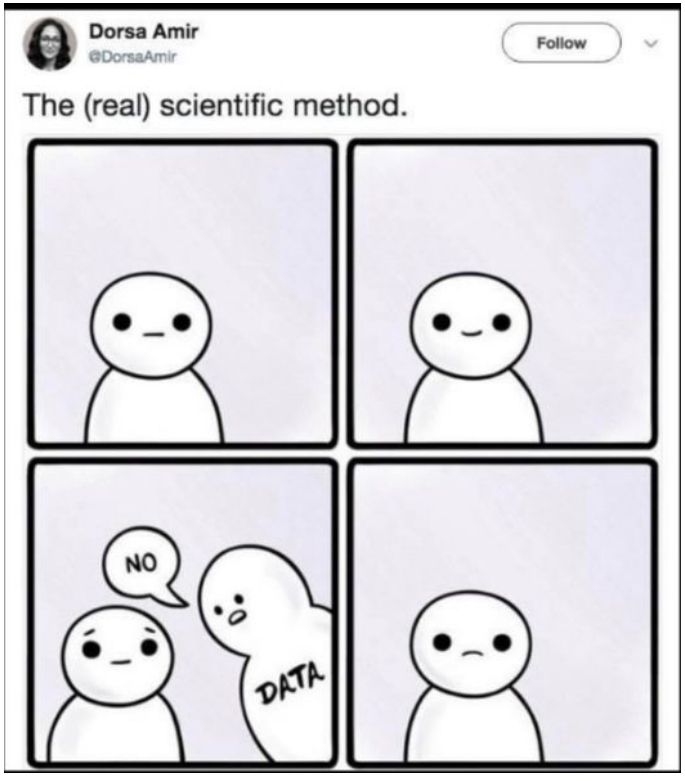
Applied statistics and Machine learning in Python with subsurface applications

Enrico Riccardi, University of Stavanger

Feb 29, 2024

Data


Dorsa Amir
@DorsaAmir
Follow
The (real) scientific method.
NO
DATA

Representation

A representation should **capture** the nature of the subject being studied.

Example: If you want to evaluate the 3D structure of a wind turbine, a set of descriptors an be:

1. Blade length
2. Turbine height
3. Geographical position
4. Output power
5. Wind direction

which are two decimal numbers, a 2d tuple, a 1D time series and a 2D time series (or 3D even).

Representation

A representation should **capture** the nature of the subject being studied.

Example: If you want to evaluate the 3D structure of a wind turbine, a set of descriptors an be:

1. Blade length
2. Turbine height
3. Geographical position
4. Output power
5. Wind direction

which are two decimal numbers, a 2d tuple, a 1D time series and a 2D time series (or 3D even).

Representation

A representation should **capture** the nature of the subject being studied.

Example: If you want to evaluate the 3D structure of a wind turbine, a set of descriptors an be:

1. Blade length
2. Turbine height
3. Geographical position
4. Output power
5. Wind direction

which are two decimal numbers, a 2d tuple, a 1D time series and a 2D time series (or 3D even).

Representation

A representation should **capture** the nature of the subject being studied.

Example: If you want to evaluate the 3D structure of a wind turbine, a set of descriptors an be:

1. Blade length
2. Turbine height
3. Geographical position
4. Output power
5. Wind direction

which are two decimal numbers, a 2d tuple, a 1D time series and a 2D time series (or 3D even).

Representation

A representation should **capture** the nature of the subject being studied.

Example: If you want to evaluate the 3D structure of a wind turbine, a set of descriptors an be:

1. Blade length
2. Turbine height
3. Geographical position
4. Output power
5. Wind direction

which are two decimal numbers, a 2d tuple, a 1D time series and a 2D time series (or 3D even).

Representation

A representation should **capture** the nature of the subject being studied.

Example: If you want to evaluate the 3D structure of a wind turbine, a set of descriptors an be:

1. Blade length
2. Turbine height
3. Geographical position
4. Output power
5. Wind direction

which are two decimal numbers, a 2d tuple, a 1D time series and a 2D time series (or 3D even).

Representation

A representation should **capture** the nature of the subject being studied.

Example: If you want to evaluate the 3D structure of a wind turbine, a set of descriptors an be:

1. Blade length
2. Turbine height
3. Geographical position
4. Output power
5. Wind direction

which are two decimal numbers, a 2d tuple, a 1D time series and a 2D time series (or 3D even).

Comparability

Same meaning **represenations** for different objects (inputs).

Discussion point!

How do we compare two wind turbines accounting for the 5 variables previously introduced?

Comparability

Same meaning **represenations** for different objects (inputs).

Discussion point!

How do we compare two wind turbines accounting for the 5 variables previously introduced?

Data properties

- All starts from data: what are data-properties?
- Are there such things as good data and bad data?

Life lesson (or exam question, same thing ;))

- Data DO NOT always have value.
- TRASH in TRASH out

Data properties

- All starts from data: what are data-properties?
- Are there such things as good data and bad data?

Life lesson (or exam question, same thing ;))

- Data DO NOT always have value.
- TRASH in TRASH out

Data properties

- All starts from data: what are data-properties?
- Are there such things as good data and bad data?

Life lesson (or exam question, same thing ;))

- Data DO NOT always have value.
- TRASH in TRASH out

Data properties

- All starts from data: what are data-properties?
- Are there such things as good data and bad data?

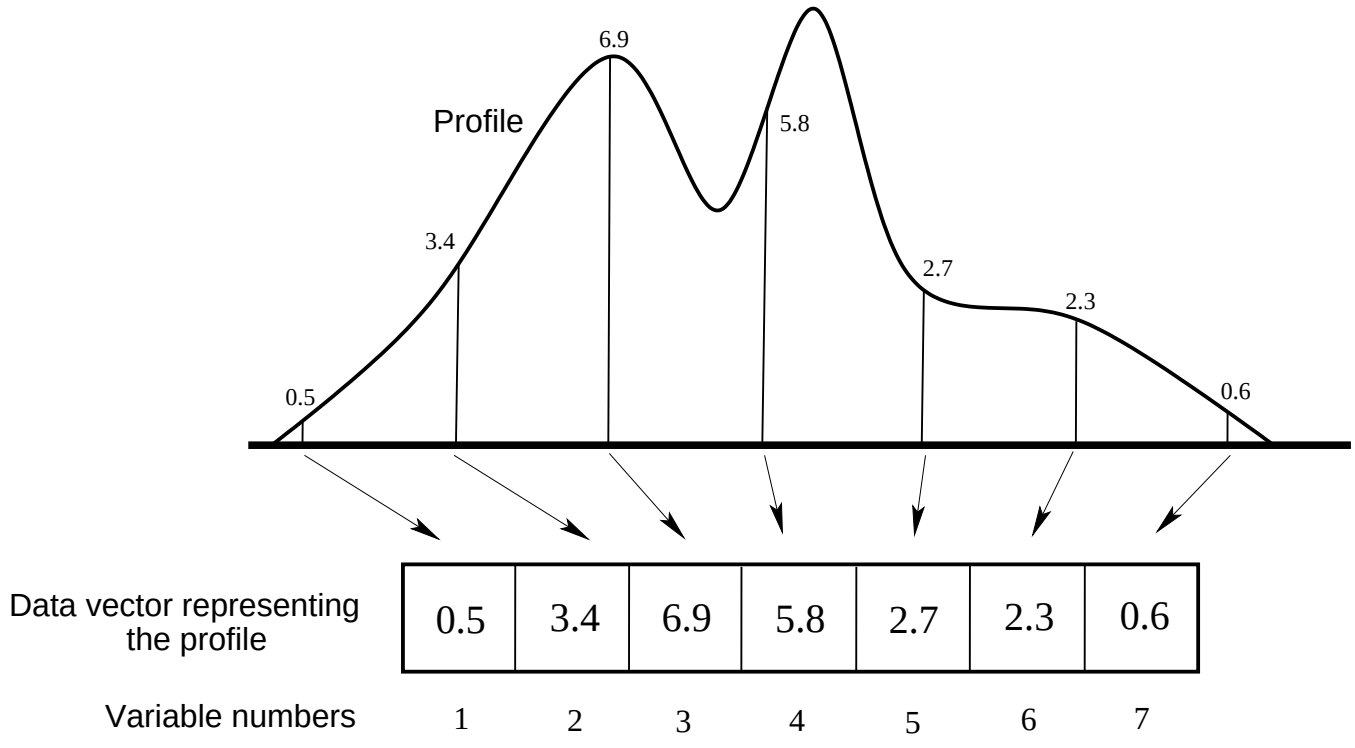
Life lesson (or exam question, same thing ;))

- Data DO NOT always have value.

- TRASH in TRASH out

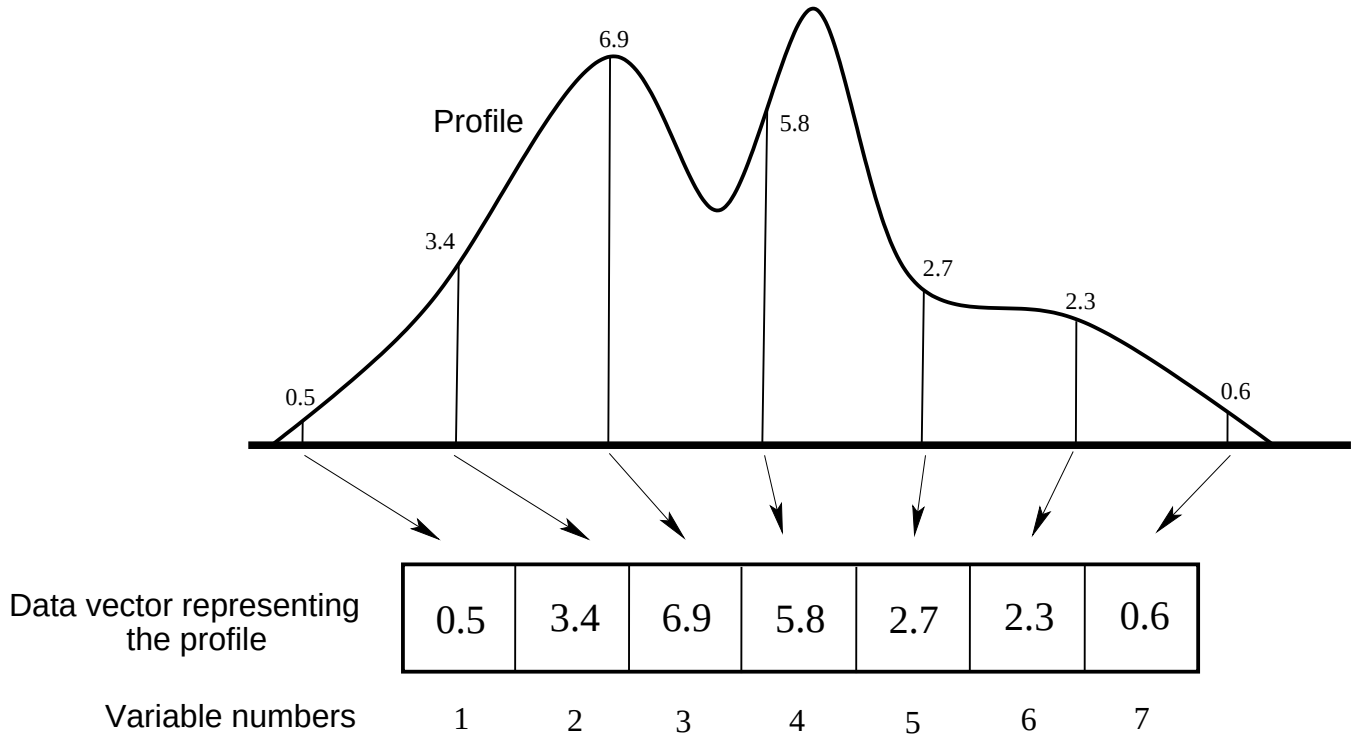
Sampling point representation (SPR)

- An intuitive way to represent curves and spectra is the **sampling point representation**.
- We sample at regular intervals where each sample point is represented by a variable



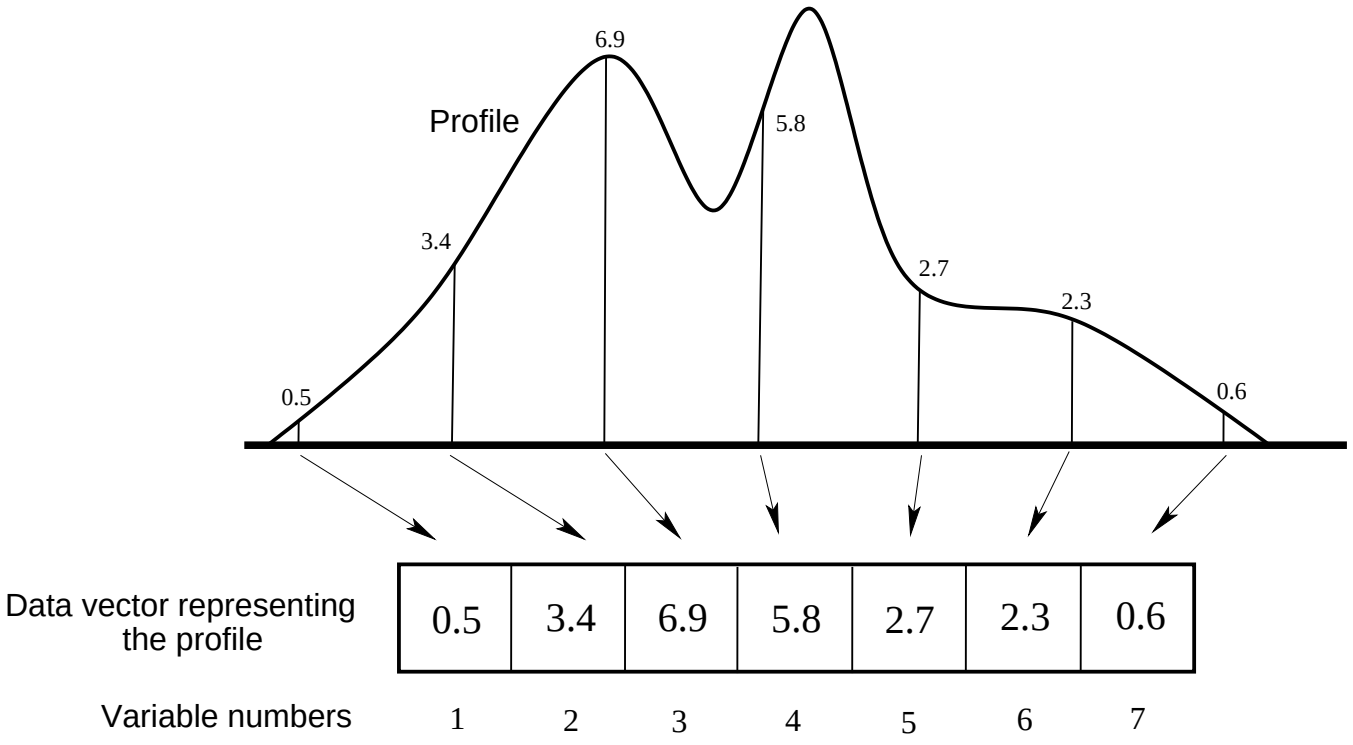
Sampling point representation (SPR)

- An intuitive way to represent curves and spectra is the **sampling point representation**.
- We sample at regular intervals where each sample point is represented by a variable



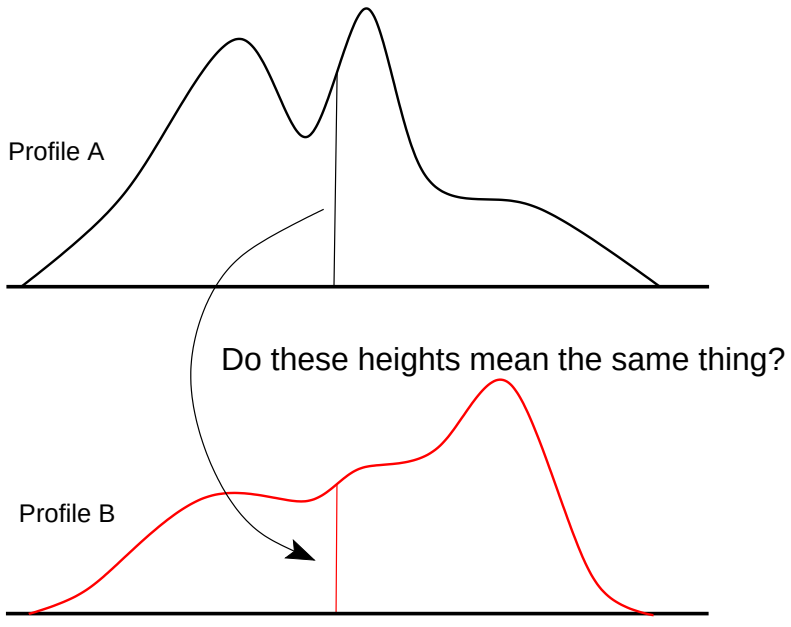
Sampling point representation (SPR)

- An intuitive way to represent curves and spectra is the **sampling point representation**.
- We sample at regular intervals where each sample point is represented by a variable



Sampling point representation (SPR)

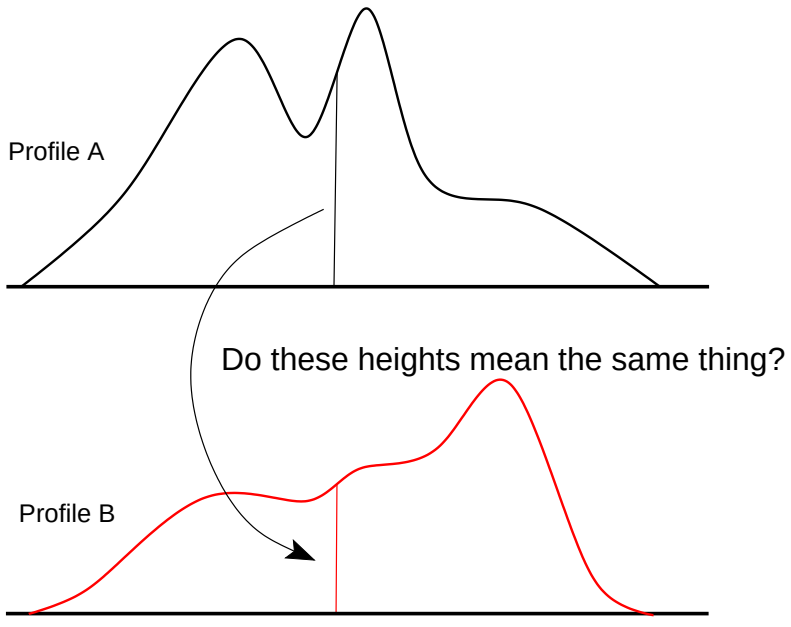
- SPR is useful until point *i* in a curve has the same meaning of the point *i* in another curve.



- Which parts of the profiles or shapes are comparable, i.e. have the same meaning?

Sampling point representation (SPR)

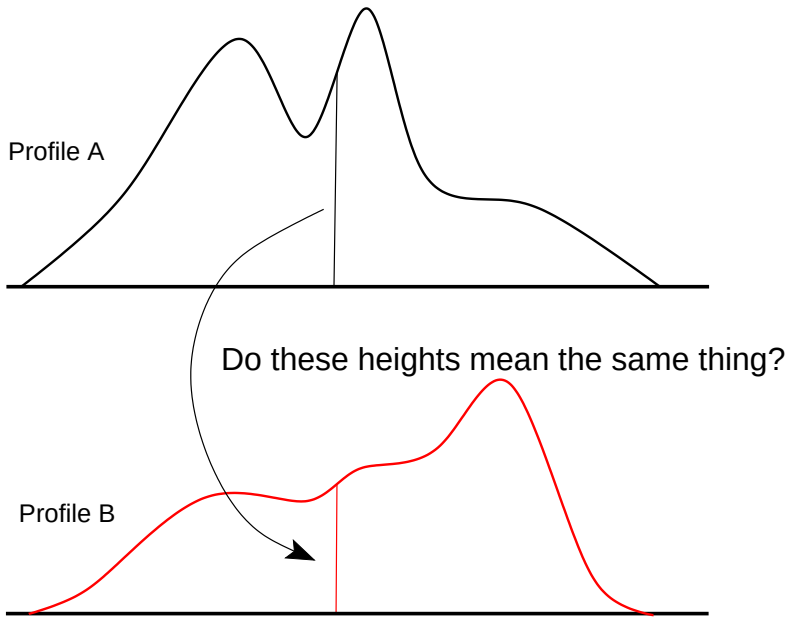
- SPR is useful until point *i* in a curve has the same meaning of the point *i* in another curve.



- Which parts of the profiles or shapes are comparable, i.e. have the same meaning?

Sampling point representation (SPR)

- SPR is useful until point *i* in a curve has the same meaning of the point *i* in another curve.



- Which parts of the profiles or shapes are comparable, i.e. have the same meaning?

Data structures

Given a representation, it is then needed to decide on a suitable **data structure** for the problem.

Definition

A data structure is a way of storing and organising data in a computer so that it can be used effectively.

Typical data structures used in data analysis are:

- Data points
- Arrays (vectors, matrices, N-mode (way) arrays)
- Graphs (trees)
- Databases

Data structures

Given a representation, it is then needed to decide on a suitable **data structure** for the problem.

Definition

A data structure is a way of storing and organising data in a computer so that it can be used effectively.

Typical data structures used in data analysis are:

- Data points
- Arrays (vectors, matrices, N-mode (way) arrays)
- Graphs (trees)
- Databases

Data structures

Given a representation, it is then needed to decide on a suitable **data structure** for the problem.

Definition

A data structure is a way of storing and organising data in a computer so that it can be used effectively.

Typical data structures used in data analysis are:

- Data points
- Arrays (vectors, matrices, N-mode (way) arrays)
- Graphs (trees)
- Databases

Data structures

Given a representation, it is then needed to decide on a suitable **data structure** for the problem.

Definition

A data structure is a way of storing and organising data in a computer so that it can be used effectively.

Typical data structures used in data analysis are:

- Data points
- Arrays (vectors, matrices, N-mode (way) arrays)
- Graphs (trees)
- Databases

Data structures

Given a representation, it is then needed to decide on a suitable **data structure** for the problem.

Definition

A data structure is a way of storing and organising data in a computer so that it can be used effectively.

Typical data structures used in data analysis are:

- Data points
- Arrays (vectors, matrices, N-mode (way) arrays)
- Graphs (trees)
- Databases

Data structures

Given a representation, it is then needed to decide on a suitable **data structure** for the problem.

Definition

A data structure is a way of storing and organising data in a computer so that it can be used effectively.

Typical data structures used in data analysis are:

- Data points
- Arrays (vectors, matrices, N-mode (way) arrays)
- Graphs (trees)
- Databases

Data structures

Given a representation, it is then needed to decide on a suitable **data structure** for the problem.

Definition

A data structure is a way of storing and organising data in a computer so that it can be used effectively.

Typical data structures used in data analysis are:

- Data points
- Arrays (vectors, matrices, N-mode (way) arrays)
- Graphs (trees)
- Databases

Workflow

Data has to be prepared with these steps in mind

1. Plan experiments: Use experimental design to set up experiments in a *systematic* way
2. Pre-processing: Is there systematic variation in the data which should be removed Can cross-checking/validation procedures be designed?
3. Examine the data: Look at data (tables and plots). Strange behaviours? Smooth behaviour? WARNING!
4. Define desired model outcomes (speed, accuracy, false positive/negatives rate)
5. Estimate and validate model: What do the results tell us? Is the generated model general (valid for future sampling)?
6. Apply model to unknown samples

Workflow

Data has to be prepared with these steps in mind

1. Plan experiments: Use experimental design to set up experiments in a *systematic* way
2. Pre-processing: Is there systematic variation in the data which should be removed Can cross-checking/validation procedures be designed?
3. Examine the data: Look at data (tables and plots). Strange behaviours? Smooth behaviour? WARNING!
4. Define desired model outcomes (speed, accuracy, false positive/negatives rate)
5. Estimate and validate model: What do the results tell us? Is the generated model general (valid for future sampling)?
6. Apply model to unknown samples

Workflow

Data has to be prepared with these steps in mind

1. Plan experiments: Use experimental design to set up experiments in a *systematic* way
2. Pre-processing: Is there systematic variation in the data which should be removed Can cross-checking/validation procedures be designed?
3. Examine the data: Look at data (tables and plots). Strange behaviours? Smooth behaviour? WARNING!
4. Define desired model outcomes (speed, accuracy, false positive/negatives rate)
5. Estimate and validate model: What do the results tell us? Is the generated model general (valid for future sampling)?
6. Apply model to unknown samples

Workflow

Data has to be prepared with these steps in mind

1. Plan experiments: Use experimental design to set up experiments in a *systematic* way
2. Pre-processing: Is there systematic variation in the data which should be removed Can cross-checking/validation procedures be designed?
3. Examine the data: Look at data (tables and plots). Strange behaviours? Smooth behaviour? WARNING!
4. Define desired model outcomes (speed, accuracy, false positive/negatives rate)
5. Estimate and validate model: What do the results tell us? Is the generated model general (valid for future sampling)?
6. Apply model to unknown samples

Workflow

Data has to be prepared with these steps in mind

1. Plan experiments: Use experimental design to set up experiments in a *systematic* way
2. Pre-processing: Is there systematic variation in the data which should be removed Can cross-checking/validation procedures be designed?
3. Examine the data: Look at data (tables and plots). Strange behaviours? Smooth behaviour? WARNING!
4. Define desired model outcomes (speed, accuracy, false positive/negatives rate)
5. Estimate and validate model: What do the results tell us? Is the generated model general (valid for future sampling)?
6. Apply model to unknown samples

Workflow

Data has to be prepared with these steps in mind

1. Plan experiments: Use experimental design to set up experiments in a *systematic* way
2. Pre-processing: Is there systematic variation in the data which should be removed Can cross-checking/validation procedures be designed?
3. Examine the data: Look at data (tables and plots). Strange behaviours? Smooth behaviour? WARNING!
4. Define desired model outcomes (speed, accuracy, false positive/negatives rate)
5. Estimate and validate model: What do the results tell us? Is the generated model general (valid for future sampling)?
6. Apply model to unknown samples

Workflow

Data has to be prepared with these steps in mind

1. Plan experiments: Use experimental design to set up experiments in a *systematic* way
2. Pre-processing: Is there systematic variation in the data which should be removed Can cross-checking/validation procedures be designed?
3. Examine the data: Look at data (tables and plots). Strange behaviours? Smooth behaviour? WARNING!
4. Define desired model outcomes (speed, accuracy, false positive/negatives rate)
5. Estimate and validate model: What do the results tell us? Is the generated model general (valid for future sampling)?
6. Apply model to unknown samples

Spatial and Temporal Data

Statistics is collecting, organising, and interpreting data

Spatial and temporal statistics is a branch of applied statistics that emphasises:

1. the geo context of the data
2. the spatial and time dependent relationship between data
3. the different relative value and precision of the data.

Spatial and Temporal Data

Statistics is collecting, organising, and interpreting data

Spatial and temporal statistics is a branch of applied statistics that emphasises:

1. the geo context of the data
2. the spatial and time dependent relationship between data
3. the different relative value and precision of the data.

Spatial and Temporal Data

Statistics is collecting, organising, and interpreting data

Spatial and temporal statistics is a branch of applied statistics that emphasises:

1. the geo context of the data
2. the spatial and time dependent relationship between data
3. the different relative value and precision of the data.

Spatial and Temporal Data

Statistics is collecting, organising, and interpreting data

Spatial and temporal statistics is a branch of applied statistics that emphasises:

1. the geo context of the data
2. the spatial and time dependent relationship between data
3. the different relative value and precision of the data.

Spatial and Temporal Data

Statistics is collecting, organising, and interpreting data

Spatial and temporal statistics is a branch of applied statistics that emphasises:

1. the geo context of the data
2. the spatial and time dependent relationship between data
3. the different relative value and precision of the data.

Actual data

The data matrix is an extremely common data structure.

$$X = \begin{bmatrix} 95 & 89 & 82 \\ 23 & 76 & 44 \\ 61 & 46 & 62 \\ 49 & 2 & 79 \end{bmatrix}$$

In python these can be saved as

- lists (vanilla python)
- numpy.arrays
- pandas dataframes

Actual data

The data matrix is an extremely common data structure.

$$X = \begin{bmatrix} 95 & 89 & 82 \\ 23 & 76 & 44 \\ 61 & 46 & 62 \\ 49 & 2 & 79 \end{bmatrix}$$

In python these can be saved as

- lists (vanilla python)
- numpy.arrays
- pandas dataframes

Actual data

The data matrix is an extremely common data structure.

$$X = \begin{bmatrix} 95 & 89 & 82 \\ 23 & 76 & 44 \\ 61 & 46 & 62 \\ 49 & 2 & 79 \end{bmatrix}$$

In python these can be saved as

- lists (vanilla python)
- numpy.arrays
- pandas dataframes

Actual data

The data matrix is an extremely common data structure.

$$X = \begin{bmatrix} 95 & 89 & 82 \\ 23 & 76 & 44 \\ 61 & 46 & 62 \\ 49 & 2 & 79 \end{bmatrix}$$

In python these can be saved as

- lists (vanilla python)
- numpy.arrays
- pandas dataframes

Nomenclature

There are different conventions. Commonly we will construct data matrix such that:

- Rows are called instances, objects or samples.
- Columns are called features, variables.

One can think of each row to be an experiment, and the rows its properties. Each row (experiment, object, sample, ...) is thus a list of values, one for property.

Note

Mathematically speaking, this is just a notation. As long as one keeps track and is consistent, columns can be used as rows and vice versa.

Nomenclature

There are different conventions. Commonly we will construct data matrix such that:

- Rows are called instances, objects or samples.
- Columns are called features, variables.

One can think of each row to be an experiment, and the rows its properties. Each row (experiment, object, sample, ...) is thus a list of values, one for property.

Note

Mathematically speaking, this is just a notation. As long as one keeps track and is consistent, columns can be used as rows and vice versa.

Nomenclature

There are different conventions. Commonly we will construct data matrix such that:

- Rows are called instances, objects or samples.
- Columns are called features, variables.

One can think of each row to be an experiment, and the rows its properties. Each row (experiment, object, sample, ...) is thus a list of values, one for property.

Note

Mathematically speaking, this is just a notation. As long as one keeps track and is consistent, columns can be used as rows and vice versa.

Nomenclature

There are different conventions. Commonly we will construct data matrix such that:

- Rows are called instances, objects or samples.
- Columns are called features, variables.

One can think of each row to be an experiment, and the rows its properties. Each row (experiment, object, sample, ...) is thus a list of values, one for property.

Note

Mathematically speaking, this is just a notation. As long as one keeps track and is consistent, columns can be used as rows and vice versa.

Nomenclature

There are different conventions. Commonly we will construct data matrix such that:

- Rows are called instances, objects or samples.
- Columns are called features, variables.

One can think of each row to be an experiment, and the rows its properties. Each row (experiment, object, sample, ...) is thus a list of values, one for property.

Note

Mathematically speaking, this is just a notation. As long as one keeps track and is consistent, columns can be used as rows and vice versa.

A quick example

Environmental measurements of rivers. The features (properties) can be:

- pH
- Temperature
- Concentration of pollutants
- Flow rate
- water speed

The experiments/observations/sample can be:

- Po
- Danube
- Rio delle Amazzoni
- Sjoa
- Atna

A quick example

Environmental measurements of rivers. The features (properties) can be:

- pH
- Temperature
- Concentration of pollutants
- Flow rate
- water speed

The experiments/observations/sample can be:

- Po
- Danube
- Rio delle Amazzoni
- Sjoa
- Atna

Data into Machine Learning algorithms: normalisation

Different measurements can lead to different data distribution and numerical values.

When fed to numerical recipes, data is just a set of numbers.

For faster computations, data can be **normalized**: i.e. rescaled to the same numerical interval.

Normalisation

The procedure simplifies the numerical recipes work, but it also loses information

As it is not possible to complete sample any distribution, data might be disproportionately handled depending upon data collection (i.e. this can be an error source).

Classification types

There are two main types of classification methods for analysis of a set of objects stored in matrix X:

Unsupervised classification

- Only the X data is used
- "Natural" classes/clusters/groupings in X are discovered

Supervised classification

- We know the class/group/cluster membership of every object/sample
- Class information is stored in an Y matrix

Classification types

There are two main types of classification methods for analysis of a set of objects stored in matrix X:

Unsupervised classification

- Only the X data is used
- "Natural" classes/clusters/groupings in X are discovered

Supervised classification

- We know the class/group/cluster membership of every object/sample
- Class information is stored in an Y matrix

Classification types

There are two main types of classification methods for analysis of a set of objects stored in matrix X:

Unsupervised classification

- Only the X data is used
- "Natural" classes/clusters/groupings in X are discovered

Supervised classification

- We know the class/group/cluster membership of every object/sample
- Class information is stored in an Y matrix

Classification types

Several approaches can be found in classification tasks:

Unsupervised classification

- Principal component analysis (PCA)
- Agglomerative (hierarchical) cluster analysis.
- k-means cluster analysis
- Fuzzy c-means cluster analysis
- Self organising feature maps (SOFM)

Supervised classification

- Linear discriminant analysis (LDA)
- k-nearest neighbours (kNN)
- Discriminant partial least squares
- Soft independent modelling of class analogies (SIMCA)
- Support vector machine (SVM)

Classification types

Several approaches can be found in classification tasks:

Unsupervised classification

- Principal component analysis (PCA)
- Agglomerative (hierarchical) cluster analysis.
- k-means cluster analysis
- Fuzzy c-means cluster analysis
- Self organising feature maps (SOFM)

Supervised classification

- Linear discriminant analysis (LDA)
- k-nearest neighbours (kNN)
- Discriminant partial least squares
- Soft independent modelling of class analogies (SIMCA)
- Support vector machine (SVM)

Classification types

Several approaches can be found in classification tasks:

Unsupervised classification

- Principal component analysis (PCA)
- Agglomerative (hierarchical) cluster analysis.
- k-means cluster analysis
- Fuzzy c-means cluster analysis
- Self organising feature maps (SOFM)

Supervised classification

- Linear discriminant analysis (LDA)
- k-nearest neighbours (kNN)
- Discriminant partial least squares
- Soft independent modelling of class analogies (SIMCA)
- Support vector machine (SVM)

Limitations of unsupervised classification

- Do "natural" clusters in a data set exist and/or have any meaning?
- First we must have a definition of what is a cluster. To do this we

must define what we mean by similar or dissimilar objects.

- Objects that are close have low dissimilarity and high similarity.

A metric system is required.

Limitations of unsupervised classification

- Do "natural" clusters in a data set exist and/or have any meaning?
- First we must have a definition of what is a cluster. To do this we

must define what we mean by similar or dissimilar objects.

- Objects that are close have low dissimilarity and high similarity.

A metric system is required.

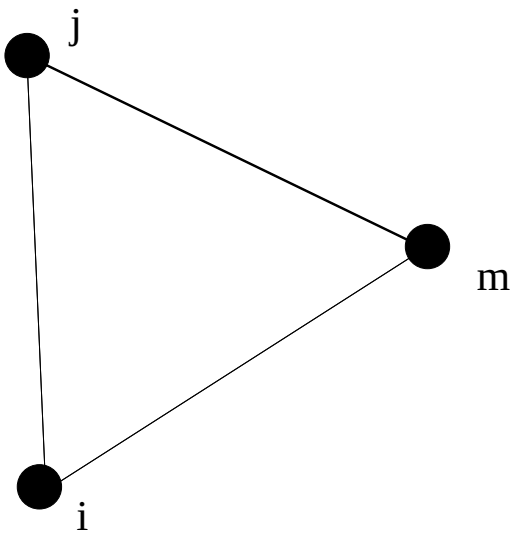
Proximity: Continuous variables

Triangle inequality

Considering a vectors in an N-dimensional space, to be a distance it must satisfy the triangle inequality:

$$d_{ij} + d_{im} \geq d_{jm}$$

If also $d_{jj} = 0$, if $i = j$ and $d_{jj} - d_{ii} = 0$, then we call it a *metric*.



Proximity: Continuous measures

Common metrics:

- Euclidean.

$$d_{ij}^{(E)} = \left[\sum_{k=1}^{N} (x_{ik} - x_{jk})^2 \right]^{\frac{1}{2}}$$

- Manhattan

$$d_{ij}^{(M)} = \sum_{k=1}^{N} \|x_{ik} - x_{jk}\|$$

- Minkowski

$$d_{ij}^{(M(p))} = \left[\sum_{k=1}^{N} (x_{ik} - x_{jk})^p \right]^{\frac{1}{p}}$$

Proximity: Categorical variables

- Many applications consist of binary vectors, typical is "yes" and "no" answers to a lot of tests
- It is tempting to use distance between binary vectors to signify distance. However that is *by far* not optimal.

Lets look at an example:

- $\mathbf{v}_1 = [1\ 1\ 0\ 0\ 0\ 0]$
- $\mathbf{v}_2 = [0\ 0\ 1\ 1\ 0\ 0]$
- $\mathbf{v}_3 = [1\ 1\ 1\ 1\ 1\ 1]$

It makes NO SENSE to compute the Euclidean distances between these vectors

Proximity: Categorical variables

- Many applications consist of binary vectors, typical is "yes" and "no" answers to a lot of tests
- It is tempting to use distance between binary vectors to signify distance. However that is *by far* not optimal.

Lets look at an example:

- $\mathbf{v}_1 = [1\ 1\ 0\ 0\ 0\ 0]$
- $\mathbf{v}_2 = [0\ 0\ 1\ 1\ 0\ 0]$
- $\mathbf{v}_3 = [1\ 1\ 1\ 1\ 1\ 1]$

It makes NO SENSE to compute the Euclidean distances between these vectors

Proximity: Categorical variables: Binary matching

	Object B value 1	*Object B* value 0
Object A value 1	a	b
Object A value 0	c	d

Example

$\mathbf{a} = [0\ 0\ 0\ 1]$

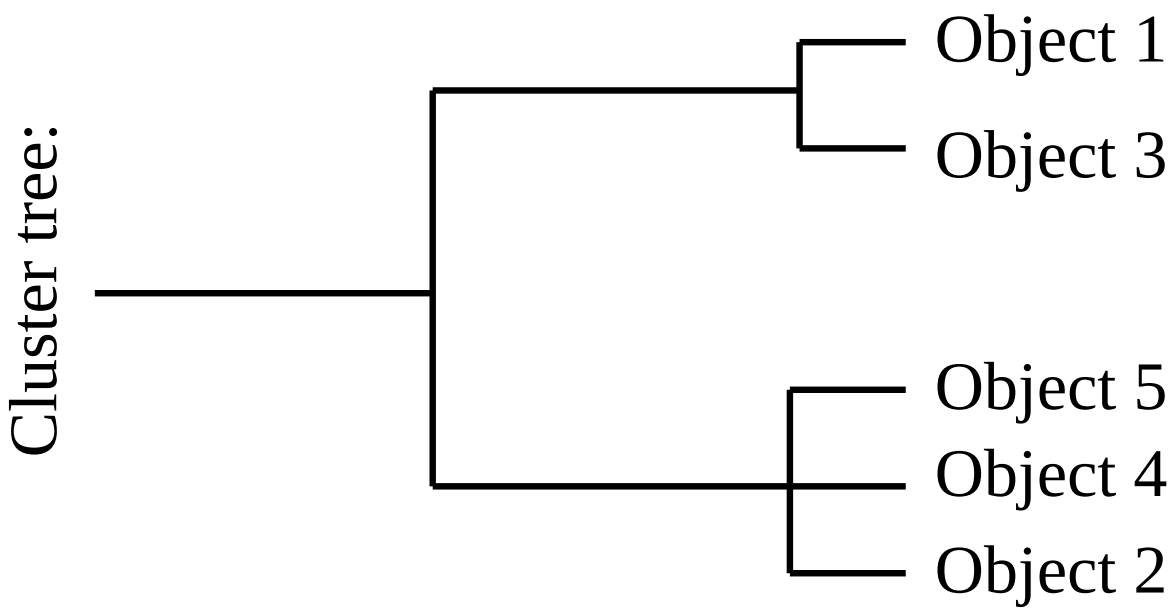
$\mathbf{b} = [1\ 1\ 0\ 1]$

- $c = 2$: two places where A has 0 and B has 1.
- $d = 1$: one place where A and B are equal to 0.
- $a = 1$: one place where A and B are equal to 1.

Agglomerative algorithms

Agglomerative cluster analysis "clumps" objects together according to a definition of similarity or dissimilarity. The objects are merged progressively into larger clusters until only one cluster remains which consists of all the objects in the data set

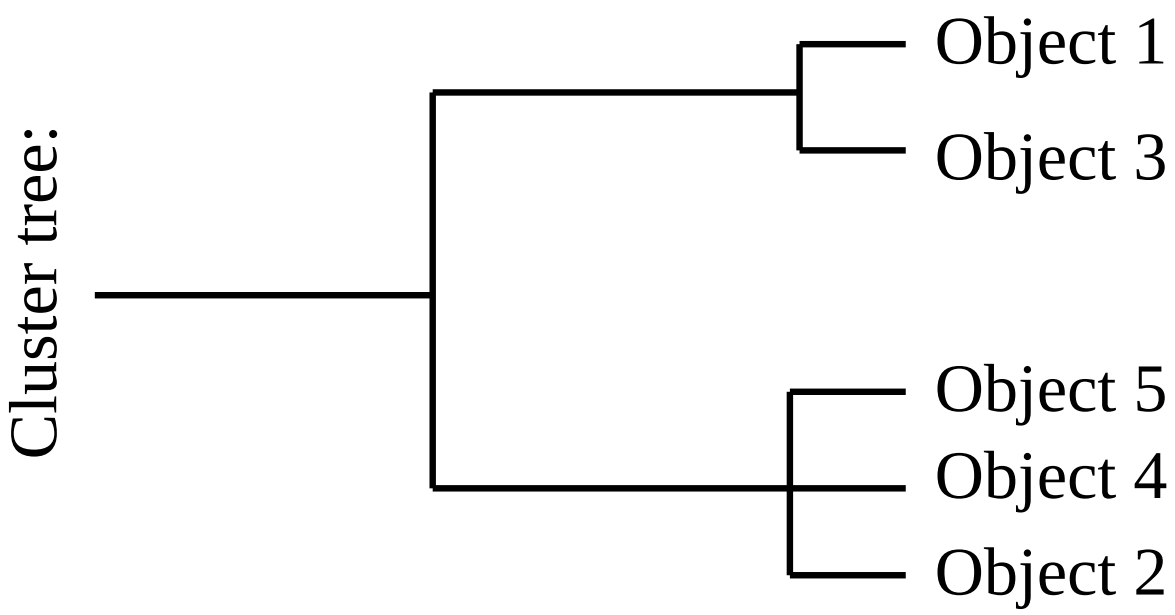
This can be summarised in a hierarchical cluster tree.



Agglomerative algorithms

Agglomerative cluster analysis "clumps" objects together according to a definition of similarity or dissimilarity. The objects are merged progressively into larger clusters until only one cluster remains which consists of all the objects in the data set

This can be summarised in a hierarchical cluster tree.



Clumping objects

One of the simpliest iterative approaches for unsupervised clustering is:

n_clusters = n_datapoints

1. WHILE no. clusters > 1
2. Find smallest distance between clusters A and B
3. Merge clusters A and B
4. Define a new cluster (AB)
5. Distance matrix between all clusters
6. ENDWHILE

Unsupervised learning

Unsupervised learning, a term that resonates with the autonomy of machine intelligence, operates on the principle of identifying patterns and structures in datasets without labelled responses.

This branch of machine learning is distinguished by its lack of explicit guidance, where algorithms are tasked with uncovering hidden structures from unlabeled data.

The most common clustering strategies are :

- filtering
- clustering
- dimensionality reduction
- association learning

Unsupervised learning

Unsupervised learning, a term that resonates with the autonomy of machine intelligence, operates on the principle of identifying patterns and structures in datasets without labelled responses.

This branch of machine learning is distinguished by its lack of explicit guidance, where algorithms are tasked with uncovering hidden structures from unlabeled data.

The most common clustering strategies are :

- filtering
- clustering
- dimensionality reduction
- association learning

Application of unsupervised learning

It is a bit of a holy grail: a computer that finds patterns without guidance. (Yes, it doesn't work, most of the time)

Still, it has been shown efficient for:

- Computer vision
- Anomaly detection
- Exploratory data analysis

Main challenge

The right result is quite undefined, Uncertain goal.

Application of unsupervised learning

It is a bit of a holy grail: a computer that finds patterns without guidance. (Yes, it doesn't work, most of the time)

Still, it has been shown efficient for:

- Computer vision
- Anomaly detection
- Exploratory data analysis

Main challenge

The right result is quite undefined, Uncertain goal.

Goals

Finding a pattern in data does not mean to find something useful.

Relevant is not the same as causal.

Correlation is not causation.

As we do not have a 'truth', or a supervision, we can look for data-patterns by

- grouping strategy according to some properties analogy
- reducing variance within groups.

Note: these approaches will (almost) always converge or provide information. Which is a better outcome, strategy or model parameter can be, at the end, only evaluated with domain expertise.

Goals

Finding a pattern in data does not mean to find something useful.

Relevant is not the same as causal.

Correlation is not causation.

As we do not have a 'truth', or a supervision, we can look for data-patterns by

- grouping strategy according to some properties analogy
- reducing variance within groups.

Note: these approaches will (almost) always converge or provide information. Which is a better outcome, strategy or model parameter can be, at the end, only evaluated with domain expertise.

Goals

Finding a pattern in data does not mean to find something useful.

Relevant is not the same as causal.

Correlation is not causation.

As we do not have a 'truth', or a supervision, we can look for data-patterns by

- grouping strategy according to some properties analogy
- reducing variance within groups.

Note: these approaches will (almost) always converge or provide information. Which is a better outcome, strategy or model parameter can be, at the end, only evaluated with domain expertise.

Common approaches

The number of unsupervised approach is tremendous, yet, the most common are the following:

- k-means (rigid clustering)
- Gaussian mixed model (soft clustering)
- Hierarchical clustering (grouping into trees/dendrograms)
- Principal component analysis (PCA) (Principal components identifications,)
- Auto-encoders (data compression)

We will discuss a few of them.

Common approaches

The number of unsupervised approach is tremendous, yet, the most common are the following:

- k-means (rigid clustering)
- Gaussian mixed model (soft clustering)
- Hierarchical clustering (grouping into trees/dendrograms)
- Principal component analysis (PCA) (Principal components identifications,)
- Auto-encoders (data compression)

We will discuss a few of them.

Common approaches

The number of unsupervised approach is tremendous, yet, the most common are the following:

- k-means (rigid clustering)
- Gaussian mixed model (soft clustering)
- Hierarchical clustering (grouping into trees/dendrograms)
- Principal component analysis (PCA) (Principal components identifications,)
- Auto-encoders (data compression)

We will discuss a few of them.

Common approaches

The number of unsupervised approach is tremendous, yet, the most common are the following:

- k-means (rigid clustering)
- Gaussian mixed model (soft clustering)
- Hierarchical clustering (grouping into trees/dendrograms)
- Principal component analysis (PCA) (Principal components identifications,)
- Auto-encoders (data compression)

We will discuss a few of them.

Common approaches

The number of unsupervised approach is tremendous, yet, the most common are the following:

- k-means (rigid clustering)
- Gaussian mixed model (soft clustering)
- Hierarchical clustering (grouping into trees/dendrograms)
- Principal component analysis (PCA) (Principal components identifications,)
- Auto-encoders (data compression)

We will discuss a few of them.

Common approaches

The number of unsupervised approach is tremendous, yet, the most common are the following:

- k-means (rigid clustering)
- Gaussian mixed model (soft clustering)
- Hierarchical clustering (grouping into trees/dendrograms)
- Principal component analysis (PCA) (Principal components identifications,)
- Auto-encoders (data compression)

We will discuss a few of them.

k-means cluster analysis

With the raise of Machine Learning, one of the most popular approach is k-means.

The algorithm consists of:

1. Select the number of clusters $K \leq K_{max}$ to look for
2. Start by creating K random cluster centres $\mathbf{m}_k$
3. For each object $\mathbf{x}_j$ assign it to the cluster center it is nearest to
4. Re-compute center points $\mathbf{m}_k$ for the new clusters and re-iterate towards convergence

This procedure minimises the within-cluster variance

Optimal no of clusters

In k-means cluster analysis we assume a **true** number of clusters.

To estimate the optimal no. of clusters K^* from data we may do as follows:

1. Compute k means for $K \in [1, 2, \cdots, K_{max}]$
2. Compute the mean within cluster variance W_K for each selection of $K \in [1, K_{max}]$
3. The variances $[W_1, W_2, \cdots, W_{max}]$ generally decrease with increasing K. This will even be the case for an independent test set such that cross-validation cannot be used.

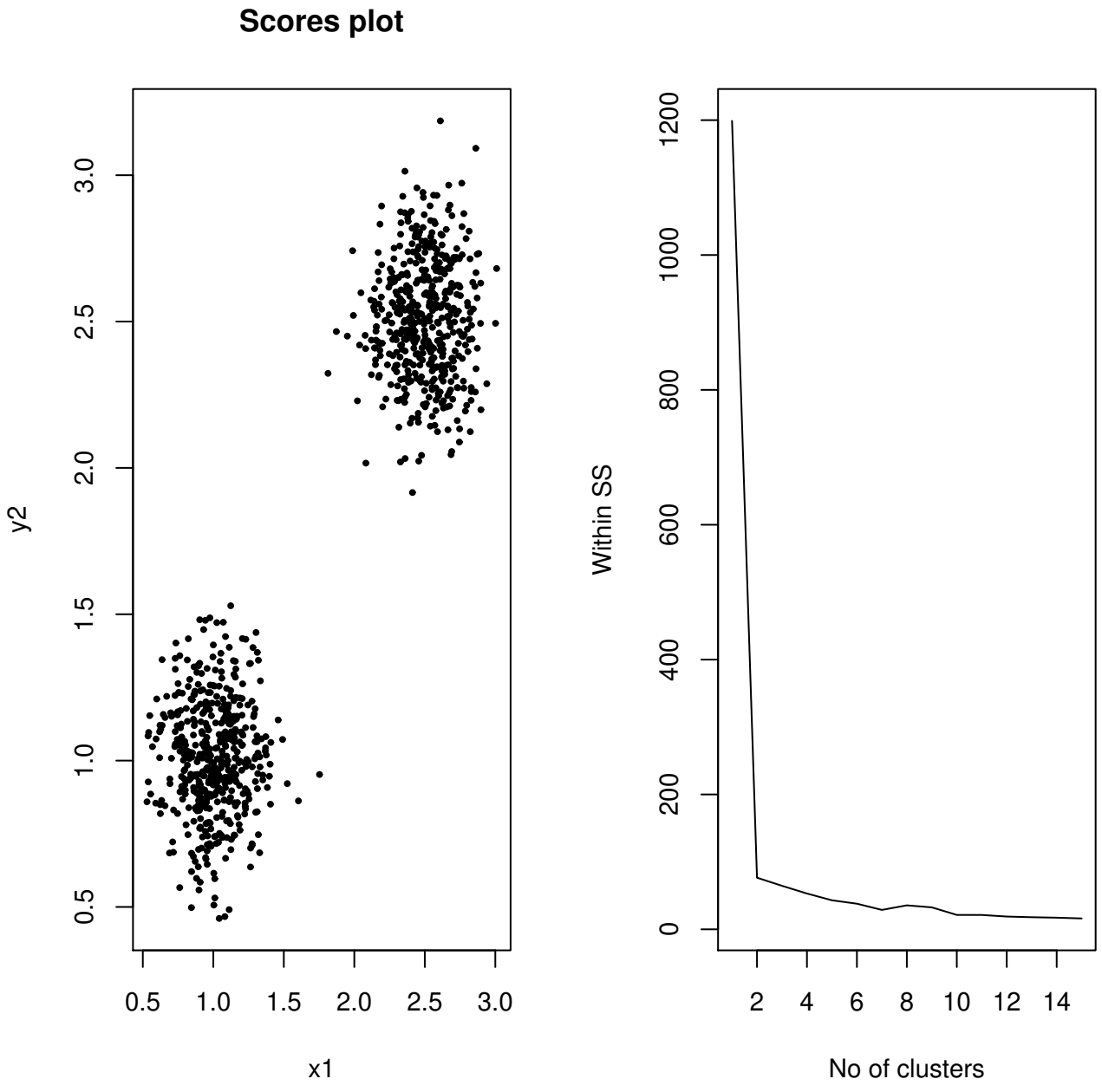
Optimal no of clusters

1. Intuitively, when $K < K^*$ we expect that an additional cluster will lower the within cluster variance: $W_{K+1} \ll W_K$.
2. When $K > K^*$ the decrease of the variance will be less evident.

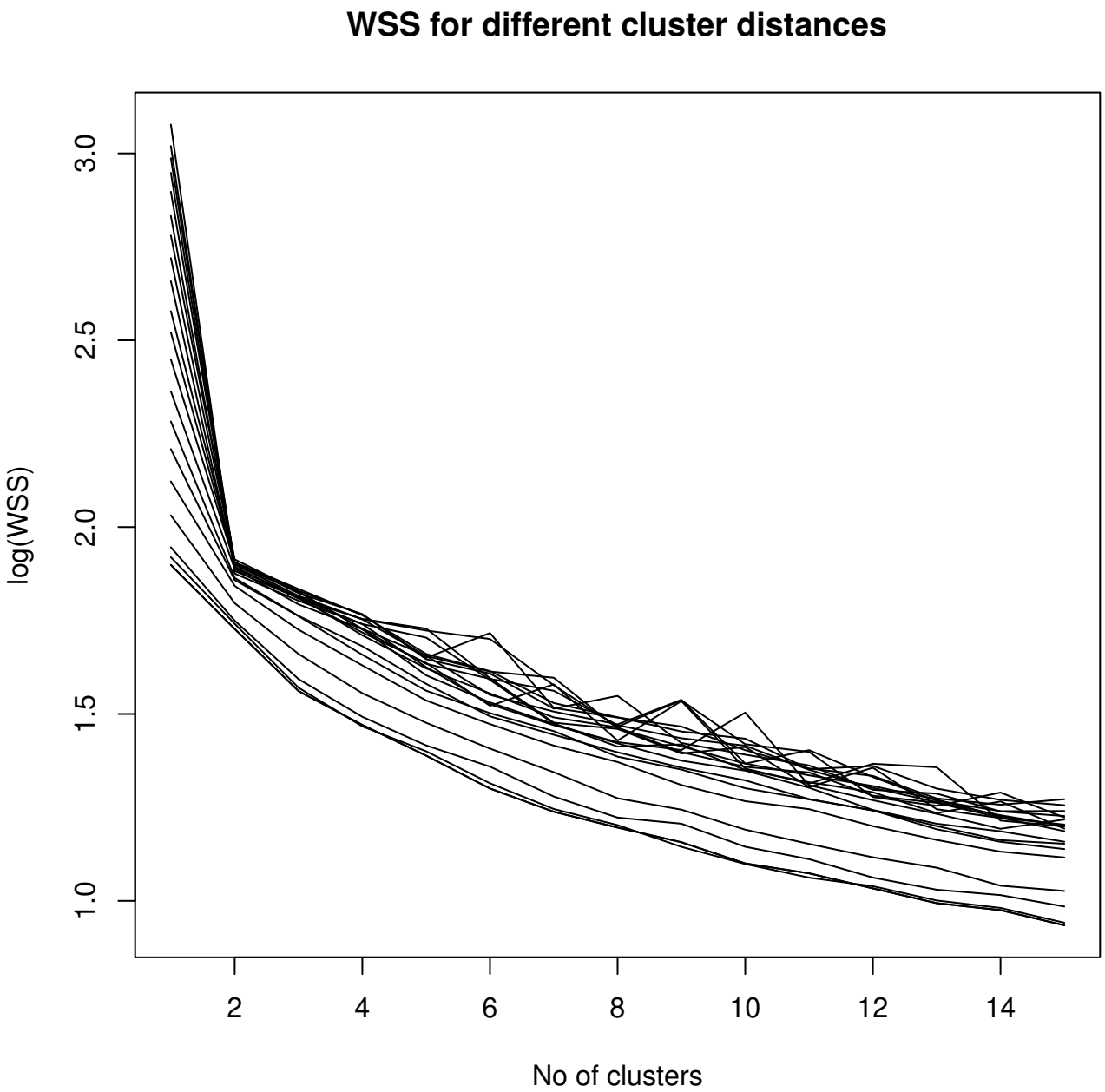
Optimal *n_clusters*

This means there will be flattening of the W_j curve. A sharp drop in the variance may be used to identify the optimal no. of clusters.

Optimal no of clusters, example



Optimal no of clusters, example



Gaussian Mixture Model (GMM)

Gaussian Mixture is a probabilistic model that tries to split the data into clusters.

Each cluster is rapresented by a center (its mean) and a Gaussian distribution around it with different variance for the different dimensions.

To each datapoint, a belonging probability to each cluster is assigned.

Assumption

Data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters.

Gaussian Mixture Model (GMM)

Gaussian Mixture is a probabilistic model that tries to split the data into clusters.

Each cluster is rapresented by a center (its mean) and a Gaussian distribution around it with different variance for the different dimensions.

To each datapoint, a belonging probability to each cluster is assigned.

Assumption

Data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters.

Limitations

The method is rather powerful, but it has quite significant drawbacks:

- The method can be rather unstable and not converge for 'poor' data.
- Iterative procedure is usually needed and convergence dependend on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationaly expensive

Limitations

The method is rather powerful, but it has quite significant drawbacks:

- The method can be rather unstable and not converge for 'poor' data.
- Iterative procedure is usually needed and convergence dependend on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationaly expensive

Limitations

The method is rather powerful, but it has quite significant drawbacks:

- The method can be rather unstable and not converge for 'poor' data.
- Iterative procedure is usually needed and convergence dependend on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationaly expensive

Limitations

The method is rather powerful, but it has quite significant drawbacks:

- The method can be rather unstable and not converge for 'poor' data.
- Iterative procedure is usually needed and convergence dependend on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationaly expensive

Limitations

The method is rather powerful, but it has quite significant drawbacks:

- The method can be rather unstable and not converge for 'poor' data.
- Iterative procedure is usually needed and convergence dependend on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationaly expensive

Limitations

The method is rather powerful, but it has quite significant drawbacks:

- The method can be rather unstable and not converge for 'poor' data.
- Iterative procedure is usually needed and convergence dependend on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationaly expensive

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM usage example

GMM is used in a large span of fields:

- Clustering: GMMs can identify clusters with different shapes and sizes due to their probabilistic nature, making them suitable for more complex datasets.
- Density Estimation: GMMs can model the probability distribution of a dataset
- Anomaly Detection: By modeling the normal behavior of data through its distribution, GMMs can be used to detect outliers or anomalous events.
- Image Segmentation: In computer vision, GMMs can be used for image segmentation, where the goal is to partition an image into segments based on the colors or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Pattern Classification: GMMs can serve as a soft-clustering method, providing probabilistic class memberships for data points, allowing the identification of pattern classification tasks where an instance may belong to multiple classes with

GMM model development

$$p(x) = \sum_{k=1}^{K} w_k f_k(x, \theta_k)$$

where

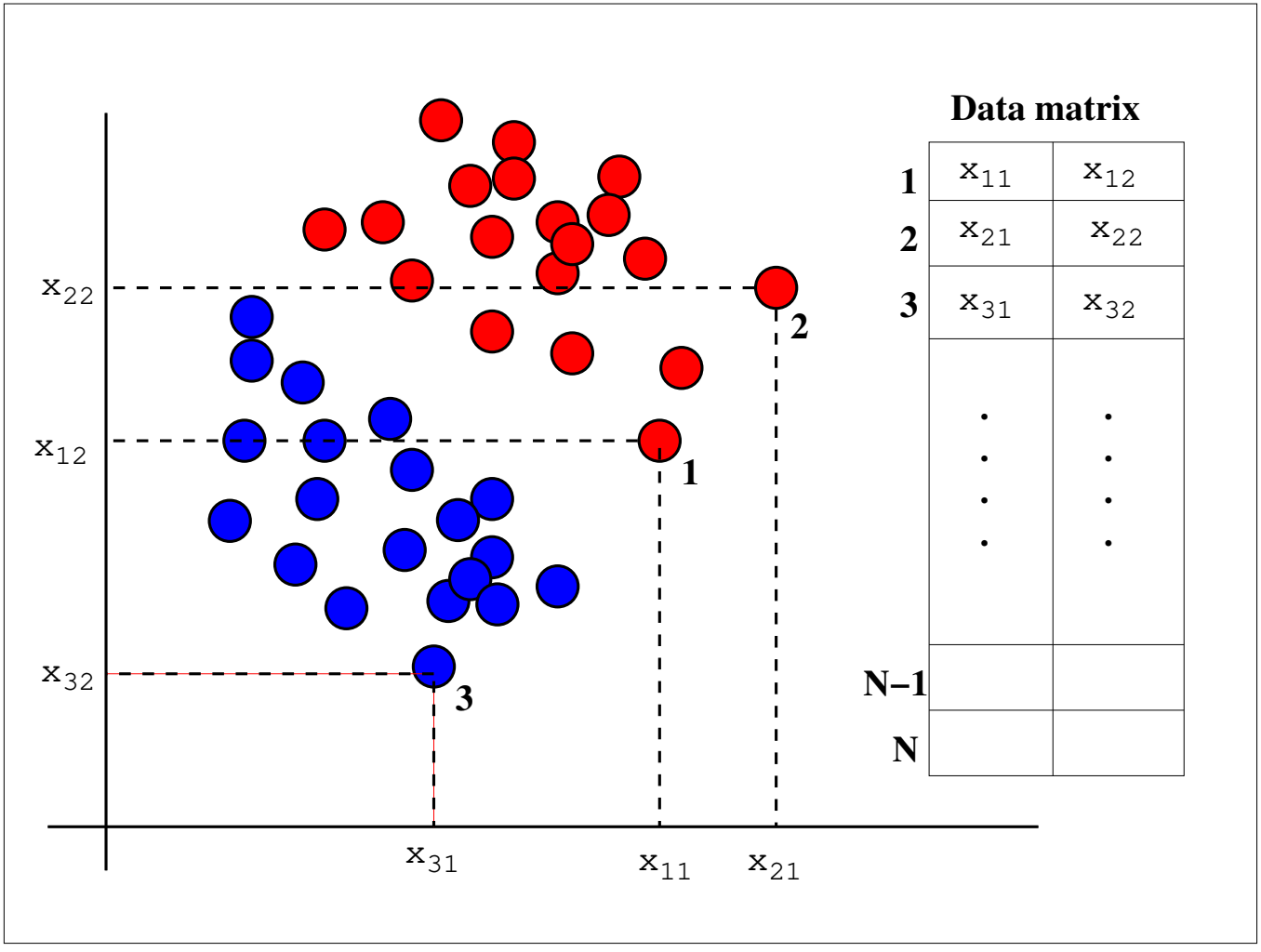
- $p(x)$ is the overall density or mass function of the mixture model.
- K is the number of distibutions to be considered
- $g_k(x, \theta_k)$ is the density mass function of the k-th component
- θ_k are the parameters for the $g_k()$ distribution

and where a Gaussian is:

$$g(x) = \frac{1}{\sigma\sqrt{(2\pi)}} exp(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2})$$

PCA

Central in any data analysis is the use of a *data matrix*



PCA base

Question:

How can we understand the information contained in such a data matrix?

- The geometry of the "object cloud" in N dimensions is used to understand relations

Geometrical insights

Plotting provides **geometrical insights** and observation of **hidden data structures** and **patterns**

PCA base

Question:

How can we understand the information contained in such a data matrix?

- The geometry of the "object cloud" in N dimensions is used to understand relations

Geometrical insights

Plotting provides **geometrical insights** and observation of **hidden data structures** and **patterns**

Correlated variables

Problem

How can we use plotting if we have more than 3 variables?

Solution

1. Seek for **correlated variables** in the data matrix
2. Seek for latent variable
3. Give up (use AI)

Correlated variables

Problem

How can we use plotting if we have more than 3 variables?

Solution

1. Seek for **correlated variables** in the data matrix
2. Seek for latent variable
3. Give up (use AI)

Correlated variables

- Correlated variables contain approximately the same information.
- Several correlated variables suggests:
  - the same **phenomenon** is manifested in different way
  - an **underlying phenomenon** more fundamental exists

Let's assume the latter:

Linear combinations

Assuming the latent variables to be a **linear combinations** of the original variables, i.e.:

$$\mathrm{LV} = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n$$

Correlated variables

- Correlated variables contain approximately the same information.
- Several correlated variables suggests:
  - the same **phenomenon** is manifested in different way
  - an **underlying phenomenon** more fundamental exists

Let's assume the latter:

Linear combinations

Assuming the latent variables to be a **linear combinations** of the original variables, i.e.:

$$\mathrm{LV} = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n$$

Latent variables

A new coordinate system

1. Latent variables are based on creating a new coordinate system based on linear combination of the original variables
2. Objects are projected from a higher dimensional

data space onto this new (lower dimensional) coordinate system

1. The new coordinate system improves interpretation and prediction.

Principal component analysis (PCA) can automatically create useful latent variables

PCA

Originally invented in 1901 by Karl Pearson and re-invented several times. The PCA method is also referred to as:

1. Singular value decomposition (numerical analysis)
2. Karhunen-Loeve expansion (electric engineering)
3. Eigenvector analysis (physical sciences)
4. Hotelling transform (image analysis/statistics)
5. Correspondence analysis (double scaled version of PCA)

Karl Pearson

PCA

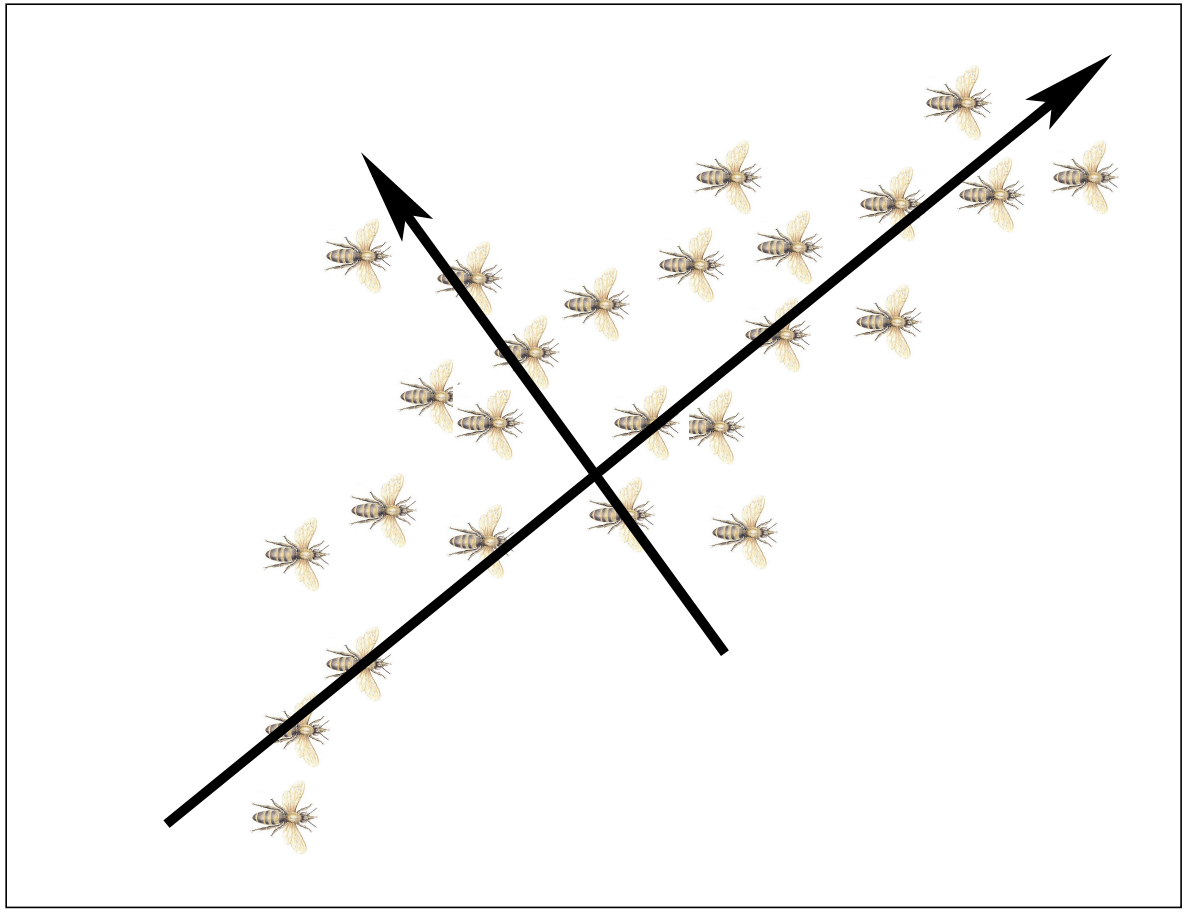
Goals of PCA:

1. Simplification.
2. Data reduction and data compression
3. Modeling
4. Outlier detection
5. Variable selection
6. Classification
7. Prediction
8. ... world peace ...

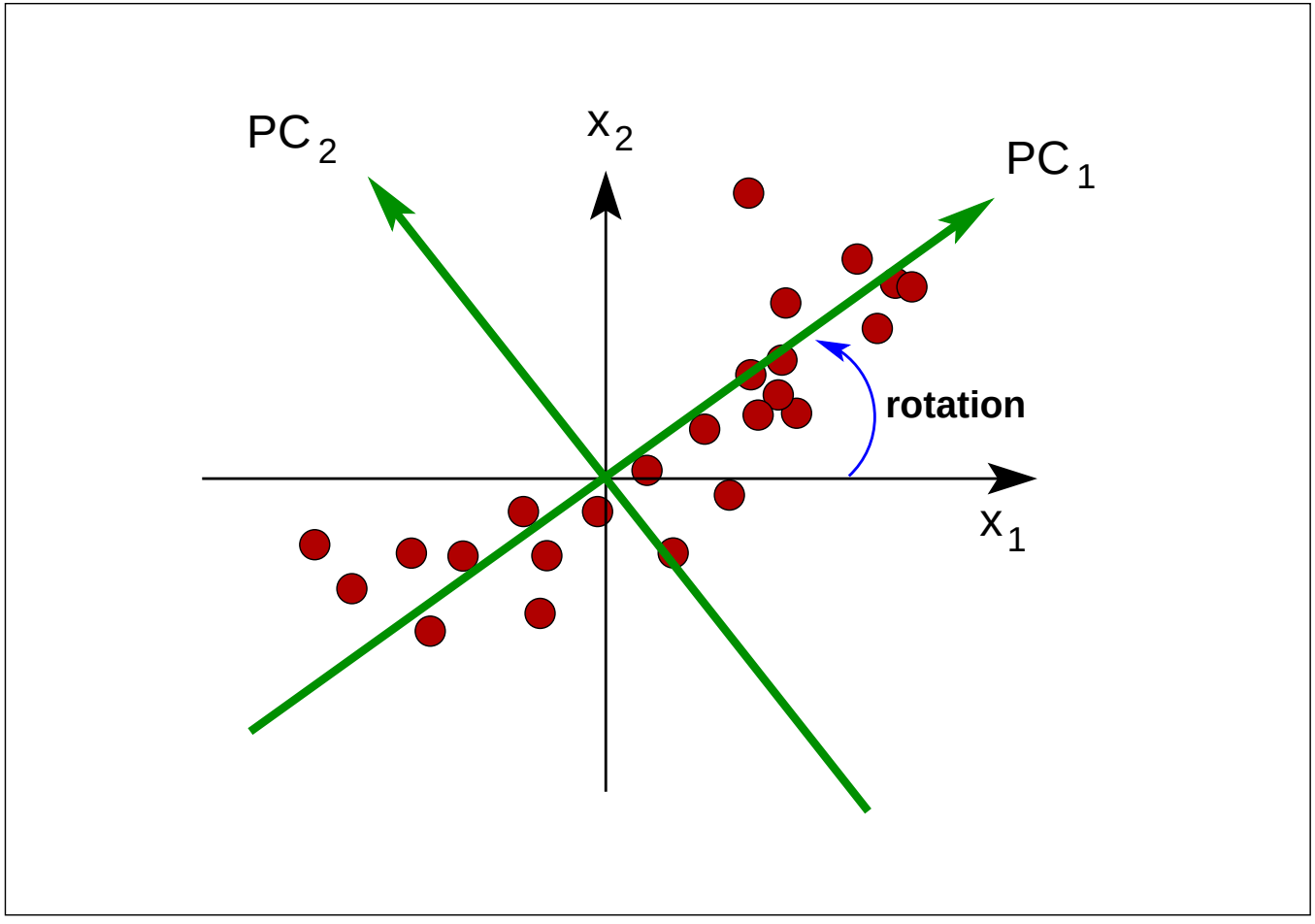
PCA

Rotation of the coordinate system

In PCA the original coordinate system is rotated such that the new latent variable axes point in the direction of max variance

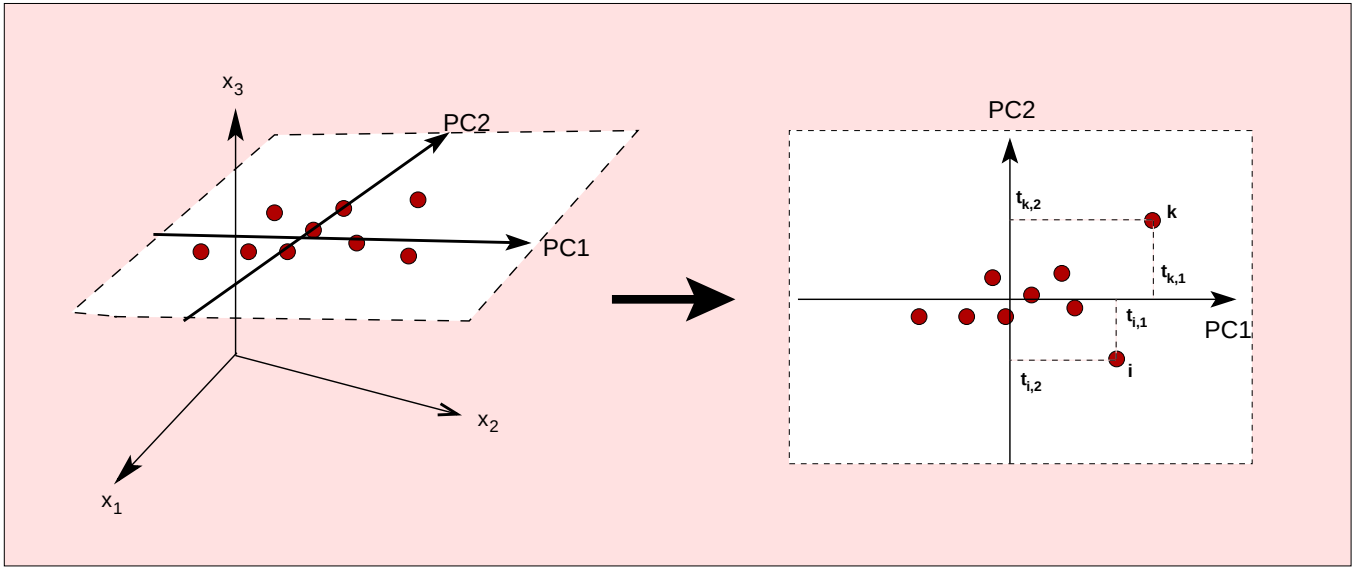


Rotation of the coordinate system

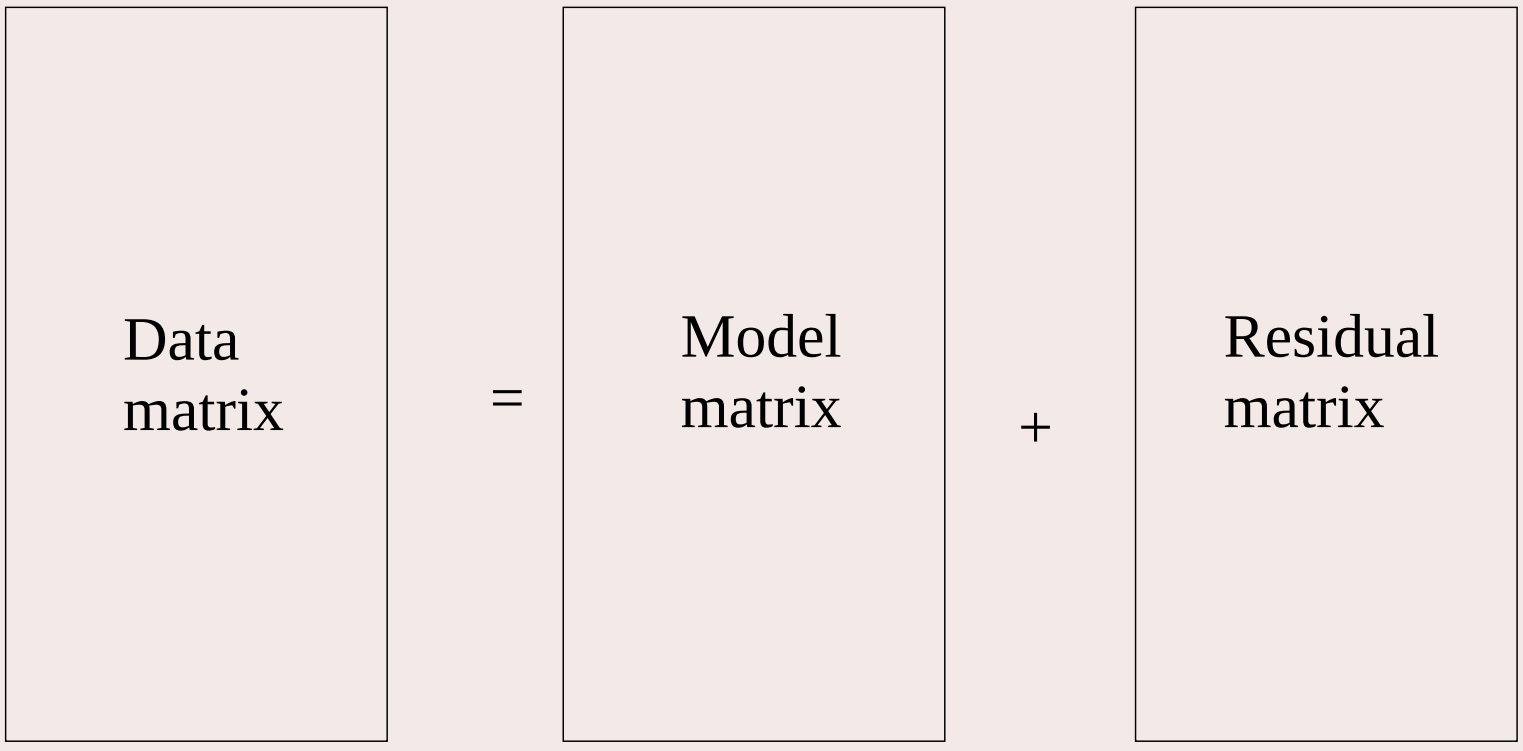


Scores are new coordinates

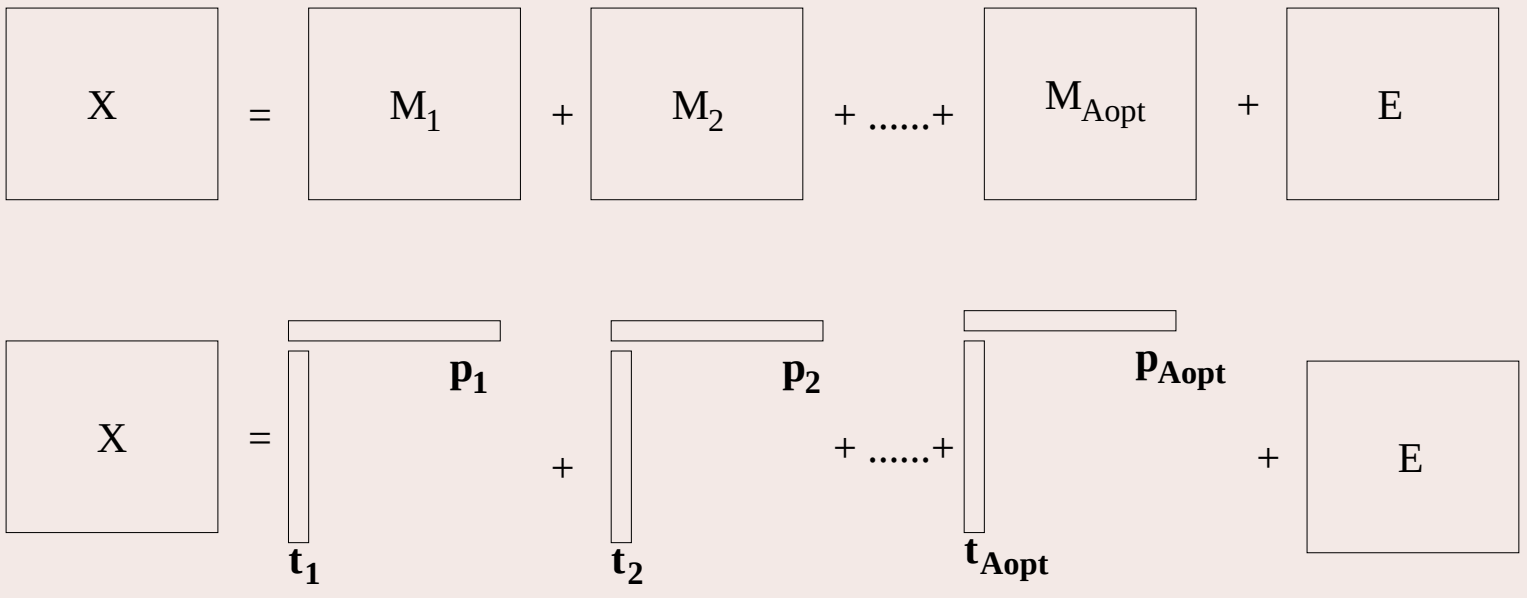
Scores are the coordinates of objects in the **new** coordinate system



Data = Model + Noise



PCA Model



Residual variance plot