Fundaments of Machine learning for and with engineering applications

Enrico Riccardi[1]

Department of Mathematics and Physics, University of Stavanger (UiS).[1]

Sep 23, 2025

1. Classification types
2. Agglomerative algorithms
3. k-means
4. Gaussian Mixture Model (GMM)
5. Self Organising Feature Mapping (SOFM)

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)

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)

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)

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.

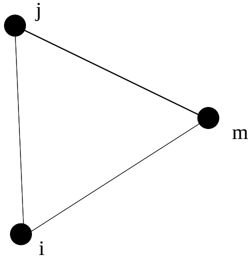
Proximity: Continuous variables

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

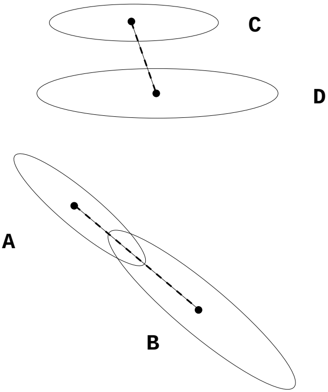
$$d_{ij}^{(M)} = \sum_{k=1}^{N} \| x_{ik} - x_{jk} \|$$

- Minkowski

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

Mahalanobis distance

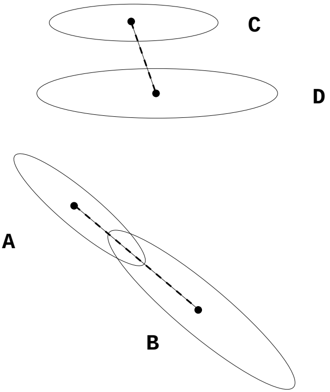
P.C. Mahalanobis invented in 1936 a distance measure which takes into consideration the covariance of a population when computing the distance between two vectors:



The Euclidean distance from C to D is shorter than A to B, but the Mahalanobis distance A-B is smaller than C-D because A and B are oriented along the same direction.

Mahalanobis distance

P.C. Mahalanobis invented in 1936 a distance measure which takes into consideration the covariance of a population when computing the distance between two vectors:



The Euclidean distance from C to D is shorter than A to B, but the Mahalanobis distance A-B is smaller than C-D because A and B are oriented along the same direction.

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.

Binary proximity measures

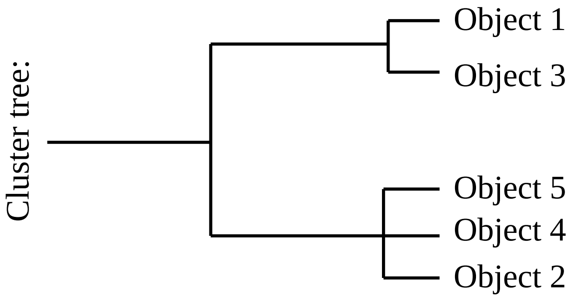
Types of binary proximity measures

Binary proximity measure	*Formula*
Matching coefficient	$d_{AB} = \frac{a+d}{a+b+c+d}$
Jackard	$d_{AB} = \frac{a}{a+b+c}$
Rogers and Tanimoto	$d_{AB} = \frac{a+d}{a+2(b+c)}$
Sokal and Sneath	$d_{AB} = \frac{a}{a+2(b+c)}$

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

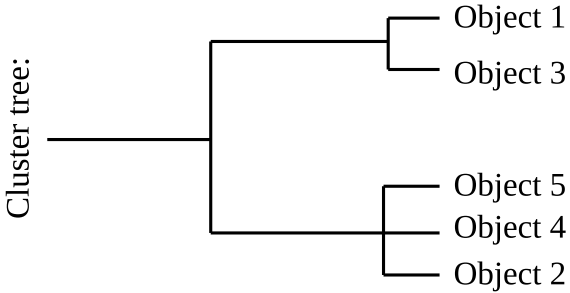
This can be summarised in a hierarchical cluster tree.



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

Distance between clusters

What is the distance between one cluster of objects to the next? The most common approaches are:

1. Single linkage
2. Complete linkage
3. Group average (unweighted pair group method average, UPGMA)
4. Ward's method

Cluster distances

Cluster distances

Single linkage

$$d_{AB} = \min(f_{i_A, j_B}), \forall i \in A, j \in B$$

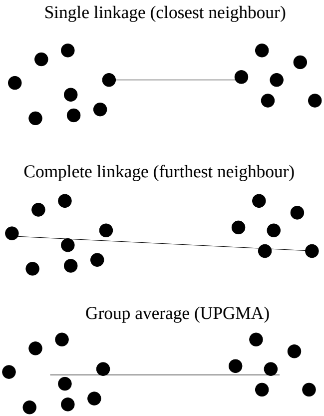
Complete linkage

$$d_{AB} = \max(f_{i_A, j_B}), \forall i \in A, j \in B$$

Group average (UPGMA)

$$d_{AB} = \frac{1}{nm} \sum_{i \in A} \sum_{j \in B} f_{i_A, j_B},$$

$$\forall i \in A, j \in B$$



Ward's method

Assume we have partitioned a set of objects into K clusters. For each cluster we can compute the total within-cluster error sum of squares:

$$E_m = \sum_{i=1}^{n_m} \sum_{j=1}^{M} \left[x_{ij}^{(m)} - \bar{x}_j^{(m)} \right]^2$$

where n_m is the number of objects in cluster m. $\bar{x}_j^{(m)}$ is the j'th variable for the mean vector of cluster m. M is the total number of variables.

Ward Equation

Now we sum all these E_m's for every cluster:

$$E_{tot}^{(0)} = \sum_{m=1}^{K} E_m$$

Assume we merge two clusters A and B so we are left with $K-1$ clusters. For the new clusters we compute $E_{tot}^{(1)}$ for the $K-1$ clusters.

The Ward criterion

The Ward criterion for merging A and B is that we want the number:

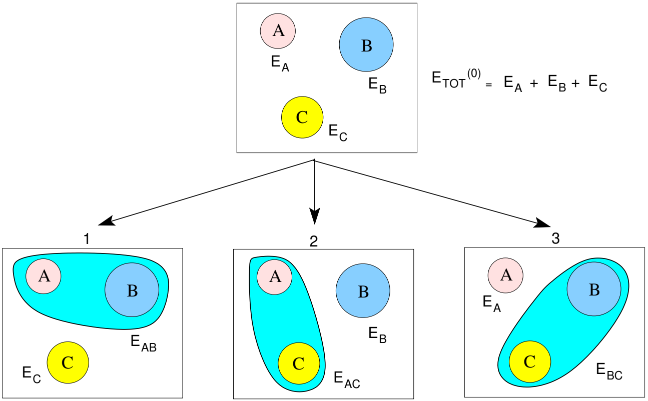
$$\phi = E_{tot}^{(1)} - E_{tot}^{(0)}$$

to be as small as possible. So in order to find this number we need to check ϕ for every possible merging of two clusters. In general there are

$$\frac{K(K-1)}{2}$$

number of such possible pairs of clusters. The pair A and B which gave the smallest number is chosen to be merged.

Example Ward's method



$E_{TOT}^{(0)} = E_A + E_B + E_C$

$E_{TOT,1}^{(1)} = E_{AB} + E_C$

$E_{TOT,2}^{(1)} = E_B + E_{AC}$

$E_{TOT,3}^{(1)} = E_A + E_{BC}$

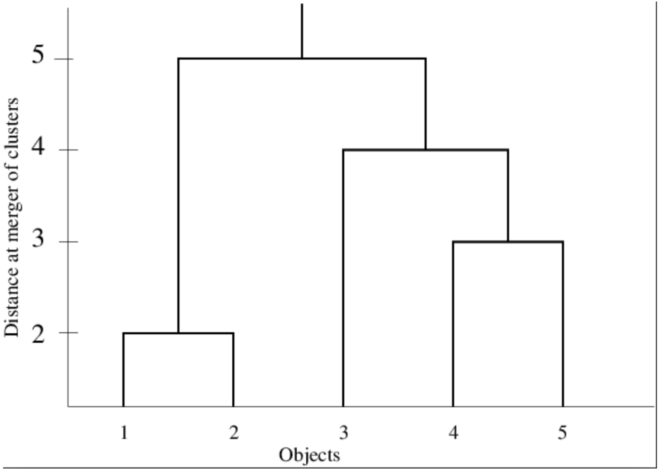
$\phi_1 = E_{TOT}^{(0)} - E_{TOT,1}^{(1)}$

$\phi_2 = E_{TOT}^{(0)} - E_{TOT,2}^{(1)}$

$\phi_3 = E_{TOT}^{(0)} - E_{TOT,3}^{(1)}$

Choose merging which gives smallest ϕ

Example dendrogram



k-means

It partitions n observations (x_1, x_2, ...) into a set of clusters (S_1, S_2, ...).

Each observation belongs to the cluster with the nearest mean.

$$arg_S min \sum_{i=1}^{k} ||x - \mu_i||^2$$

where μ_i is the centroid of cluster i

$$\mu_i = \frac{1}{|S_i|} \sum_{x \in S_i} x$$

k-means

It partitions n observations (x_1, x_2, ...) into a set of clusters (S_1, S_2, ...).

Each observation belongs to the cluster with the nearest mean.

$$arg_S\, min \sum_{i=1}^{k} ||\mathsf{x} - \mu_\mathsf{i}||^2$$

where μ_i is the centroid of cluster i

$$\mu_\mathsf{i} = \frac{1}{|S_i|} \sum_{\mathsf{x} \in S_i} \mathsf{x}$$

k-means logic

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.

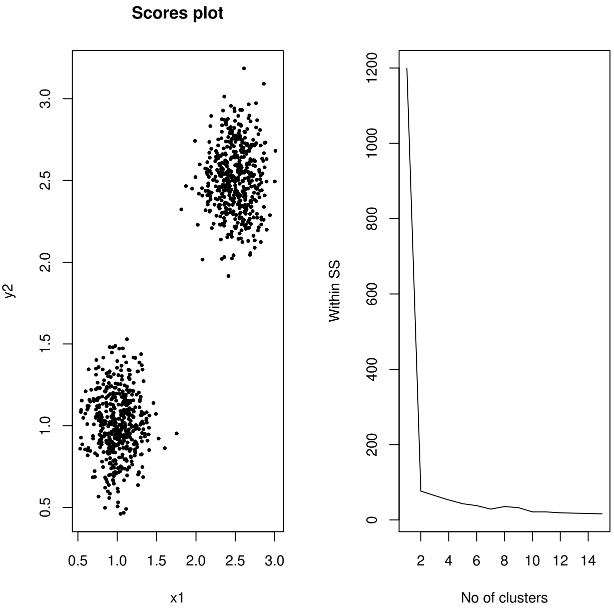
Optimal no of clusters

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.

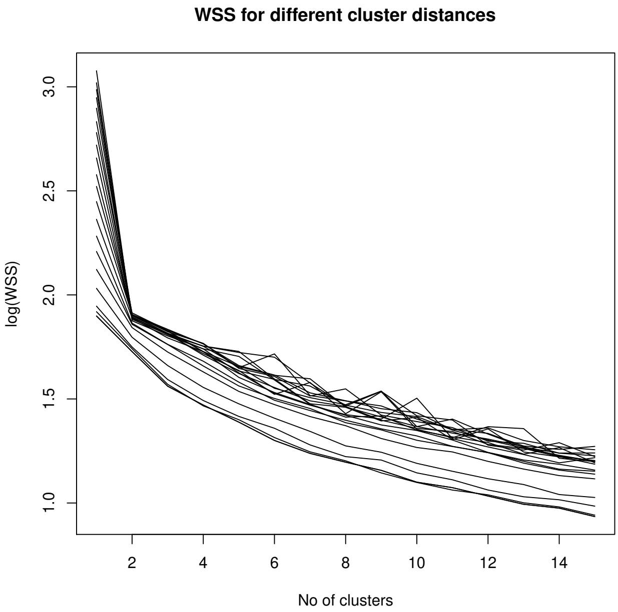
Optimal *n_clusters*

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 represented by a centre (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 represented by a centre (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 Models

Generate clusters that can overlap:

$$p(x) = \sum_{k=1}^{K} \theta_k \mathcal{N}(x|\mu_k, \sigma_k)$$

where

$$\sum_{i=1}^{K} \theta_i = 1$$

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

A sum of gaussian... easy! :)

Gaussian Mixture Models

Generate clusters that can overlap:

$$p(x) = \sum_{k=1}^{K} \theta_k \mathcal{N}(x|\mu_k, \sigma_k)$$

where

$$\sum_{i=1}^{K} \theta_i = 1$$

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

A sum of gaussian... easy! :)

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 depends on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationally 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 depends on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationally 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 depends on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationally 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 depends on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationally 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 depends on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationally 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 depends on initial guest.
- Interpretability of the results might be rather poor.
- High dimensionality limitations.
- Computationally 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.
- Anomaly Detection: By modelling the normal behaviour 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 colours or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Bioinformatics: GMMs are used in bioinformatics for tasks such as modelling gene expression data or protein structure. They can help in identifying biological patterns or clusters within the data that are not immediately apparent.
- Astronomy: GMMs are used in astronomy to classify celestial objects and to estimate the distribution of stars or galaxies based on their properties, such as brightness and colour.

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.
- Anomaly Detection: By modelling the normal behaviour 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 colours or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Bioinformatics: GMMs are used in bioinformatics for tasks such as modelling gene expression data or protein structure. They can help in identifying biological patterns or clusters within the data that are not immediately apparent.
- Astronomy: GMMs are used in astronomy to classify celestial objects and to estimate the distribution of stars or galaxies based on their properties, such as brightness and colour.

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.
- Anomaly Detection: By modelling the normal behaviour 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 colours or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Bioinformatics: GMMs are used in bioinformatics for tasks such as modelling gene expression data or protein structure. They can help in identifying biological patterns or clusters within the data that are not immediately apparent.
- Astronomy: GMMs are used in astronomy to classify celestial objects and to estimate the distribution of stars or galaxies based on their properties, such as brightness and colour.

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.
- Anomaly Detection: By modelling the normal behaviour 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 colours or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Bioinformatics: GMMs are used in bioinformatics for tasks such as modelling gene expression data or protein structure. They can help in identifying biological patterns or clusters within the data that are not immediately apparent.
- Astronomy: GMMs are used in astronomy to classify celestial objects and to estimate the distribution of stars or galaxies based on their properties, such as brightness and colour.

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.
- Anomaly Detection: By modelling the normal behaviour 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 colours or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Bioinformatics: GMMs are used in bioinformatics for tasks such as modelling gene expression data or protein structure. They can help in identifying biological patterns or clusters within the data that are not immediately apparent.
- Astronomy: GMMs are used in astronomy to classify celestial objects and to estimate the distribution of stars or galaxies based on their properties, such as brightness and colour.

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.
- Anomaly Detection: By modelling the normal behaviour 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 colours or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Bioinformatics: GMMs are used in bioinformatics for tasks such as modelling gene expression data or protein structure. They can help in identifying biological patterns or clusters within the data that are not immediately apparent.
- Astronomy: GMMs are used in astronomy to classify celestial objects and to estimate the distribution of stars or galaxies based on their properties, such as brightness and colour.

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.
- Anomaly Detection: By modelling the normal behaviour 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 colours or textures.
- Speech Recognition: GMMs have been used in speech recognition systems to model the distribution of audio features.
- Bioinformatics: GMMs are used in bioinformatics for tasks such as modelling gene expression data or protein structure. They can help in identifying biological patterns or clusters within the data that are not immediately apparent.
- Astronomy: GMMs are used in astronomy to classify celestial objects and to estimate the distribution of stars or galaxies based on their properties, such as brightness and colour.

Self Organising Feature Mapping (SOFM)

A technique invented by T. Kohonen in 1982. It is used for performing non-linear unsupervised classification. The results are presented as 2D maps where the different classes are distributed as political geographical map of the Earth. The map consists of a matrix of neurons that compete for the samples.

Typical application areas are:

1. Biological taxonomy
2. Chemistry
3. Image analysis
4. Geo-plotting

Self Organising Feature Mapping (SOFM)

A technique invented by T. Kohonen in 1982. It is used for performing non-linear unsupervised classification. The results are presented as 2D maps where the different classes are distributed as political geographical map of the Earth. The map consists of a matrix of neurons that compete for the samples.

Typical application areas are:

1. Biological taxonomy
2. Chemistry
3. Image analysis
4. Geo-plotting

SOFM algorithm

1. Initialize network by setting all weights to random numbers. However note that

$$\mathrm{w}_j(0) \neq \mathrm{w}_k(0), \; \forall k \neq j$$

$i \in 1, 2, \cdots, N$ where N is the number of nodes in the lattice

1. Take one sample x from the training (calibration) data set
2. What node i is most similar to $\mathbf{x}$? We look at $\|\mathbf{x} - \mathbf{w}_j\|$

for nodes $j \in [1, 2, \cdots, N]$

1. Adjust the connection weights:

$$\mathbf{w}_j(n+1) = \begin{cases} \mathbf{w}_j(n) + \eta(n)(\mathbf{x} - \mathbf{w}_j(n)) & \forall j \in \Lambda_i(n) \\ \mathbf{w}_j(n) & \text{otherwise} \end{cases}$$

$\Lambda_i(n)$ is the neighbourhood function centered around the winning node i. Both $\eta(n)$ and $\Lambda_i(n)$ vary dynamically during learning. $\Lambda_i(n)$ becomes smaller - a shrinking effect

SOFM map