WELCOME

Introduction to Machine Learning

DBDA.X408.(33)

Instructor:

Bill Chen

UCSC Silicon Valley Extension
PROFESSIONAL EDUCATION

UCSC Silicon Valley Extension

E: xch375@ucsc.edu

Week 4

Feature Engineering.

ML Tooling: Python

PyTorch

TensorFlow

Data understanding using plots

Feature Reduction

A linear or non-linear transform on the original feature space.

- Unsupervised methods
  - Principal Component Analysis (PCA)
  - Independent Component Analysis (ICA)
  - Autoencoder
- Supervised methods
  - Linear Discriminant Analysis (LDA)

PCA (Principal component analysis)

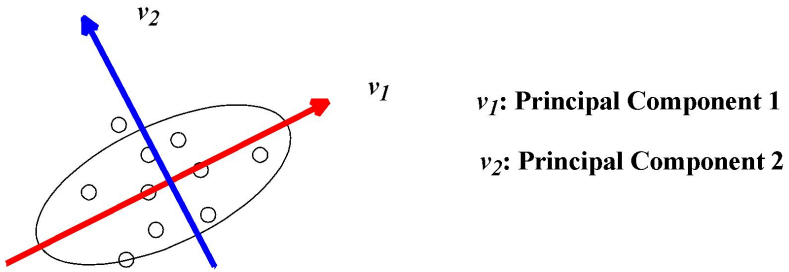
PCA is to calculate the principal components for data analytics, where only the a few of principal components are selected in terms of a predefined number K of features.

- It is commonly used for dimensionality reduction by projecting each data point onto only the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible.
- The first principal component can equivalently be defined as a direction that maximizes the variance of the projected data.
- The i^{th} principal component can be taken as a direction orthogonal to the $i-1$ principal components, which maximizes the variance of the projected data.

PCA (Principal component analysis)

The goal of PCA is to reduce the dimensionality of the data while preserving the variation present in the dataset as much as possible.

- The axes have been rotated to new principal components such that:
  - Principal component 1 has the highest variance.
  - Principal component 2 has the next highest variance, and so on.



PCA (Principal component analysis)

Principal Components (PCs): orthogonal vectors that are ordered by the fraction of the total information (variation) in the corresponding directions.

PCs can be found as the “best” eigenvectors of the covariance matrix of the data points.

PCs are linear least squares fits to samples, each orthogonal to the previous PCs:

- First PC is a minimum distance fit to a vector in the original feature space
- Second PC is a minimum distance fit to a vector in the plane perpendicular to the first PC

Mathematics for PCA

Covariance Matrix

$$\boldsymbol{\mu_x} = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_d \end{bmatrix} = \begin{bmatrix} E(x_1) \\ \vdots \\ E(x_d) \end{bmatrix}$$

$$\boldsymbol{\Sigma} = E[(\boldsymbol{x} - \boldsymbol{\mu_x})(\boldsymbol{x} - \boldsymbol{\mu_x})^T]$$

Estimating covariance matrix from data points $\{x^{(i)}\}_{i=1}^{N}$:

$$\widehat{\boldsymbol{\Sigma}} = \frac{1}{N}(\widetilde{\boldsymbol{X}}^T \widetilde{\boldsymbol{X}})$$

$$\widetilde{\boldsymbol{X}} = \begin{bmatrix} \boldsymbol{x}^{(1)} - \widehat{\boldsymbol{\mu}} \\ \vdots \\ \boldsymbol{x}^{(N)} - \widehat{\boldsymbol{\mu}} \end{bmatrix} \qquad \widehat{\boldsymbol{\mu}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}^{(i)}$$

Mean-centered data

Calculating PCA

Input: $N \times d$ data matrix X (each row contains a d dimensional data points)

- $\mu = \frac{1}{N}\sum_{i=1}^{N} x^{(i)}$ **// calculate the mean of data**
- $\tilde{X} \leftarrow$ Mean values of data points is subtracted from rows of data.

$$\tilde{X} = \begin{bmatrix} x^{(1)} - \hat{\mu} \\ \vdots \\ x^{(N)} - \hat{\mu} \end{bmatrix}$$

Mean-centered data

- $\sum = \frac{1}{N}\tilde{X}^T\tilde{X}$ (Covariance matrix) **// calculate the covariance of data**
- Calculate eigenvalues and eigenvectors of the covariance matrix
- Pick eigenvectors corresponding to the largest eigenvalues based on a predefined dimensions K and put them in the columns of $e = [e_1, \ldots, e_{d'}]$
  - $K < d$
- $X' = e^T\tilde{X}$ (Obtaining PCs)

Output: X' (PCs)

An Example of PCA application

House Information

#	House Price	Area
1	10	9
2	2	3
3	1	2
4	7	6.5
5	3	2.5

- Two features of house data
  - House Price (Million of dollars)
  - House Area

An Example of PCA application

Calculate $\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}^{(i)}$

#	House Price	Area
1	10	9
2	2	3
3	1	2
4	7	6.5
5	3	2.5

$$\boldsymbol{\mu}_{\text{House Price}} = \frac{10 + 2 + 1 + 7 + 3}{5} = 4.6$$

$$\boldsymbol{\mu}_{\text{Area}} = \frac{9 + 3 + 2 + 6.5 + 2.5}{5} = 4.6$$

An Example of PCA application

Calculate $\widetilde{X} \leftarrow$ Mean values of data points is subtracted from rows of data.

#	House Price	Area
1	10 - 4.6	9 - 4.6
2	2 - 4.6	3 - 4.6
3	1 - 4.6	2 - 4.6
4	7 - 4.6	6.5 - 4.6
5	3 - 4.6	2.5 - 4.6

#	House Price	Area
1	5.4	4.4
2	-2.6	-1.6
3	-3.6	-2.6
4	2.4	1.9
5	-1.6	-2.1

An Example of PCA application

Calculate $\sum = \frac{1}{N}\tilde{X}^T\tilde{X}$ (Covariance matrix)

#	House Price	Area
1	5.4	4.4
2	-2.6	-1.6
3	-3.6	-2.6
4	2.4	1.9
5	-1.6	-2.1

$$A = \begin{pmatrix} 5.4 \\ -2.6 \\ -3.6 \\ 2.4 \\ -1.6 \end{pmatrix} \qquad B = \begin{pmatrix} 4.4 \\ -1.6 \\ -2.6 \\ 1.9 \\ -2.1 \end{pmatrix}$$

$$\sum = \begin{pmatrix} Var(A) & Cov(A,B) \\ Cov(A,B) & Var(B) \end{pmatrix} = \begin{matrix} Cov(A,A) & Cov(A,B) \\ Cov(A,B) & Cov(B,B) \end{matrix} = \frac{1}{5}\begin{pmatrix} A \cdot A & A \cdot B \\ A \cdot B & B \cdot B \end{pmatrix} = \frac{1}{5}\begin{pmatrix} 57.2 & 45.2 \\ 45.2 & 36.7 \end{pmatrix}$$

An Example of PCA application

Calculate eigenvalues λ and eigenvectors e of the covariance matrix with **singular value decomposition (SVD).**

#	House Price	Area
1	5.4	4.4
2	-2.6	-1.6
3	-3.6	-2.6
4	2.4	1.9
5	-1.6	-2.1

$$\Sigma = \begin{pmatrix} -0.78 & -0.62 \\ -0.62 & 0.78 \end{pmatrix} \begin{pmatrix} 18.66 & 0 \\ 0 & 0.12 \end{pmatrix} \begin{pmatrix} -0.78 & -0.62 \\ -0.62 & 0.78 \end{pmatrix}$$

$$e_1 = \begin{pmatrix} -0.78 \\ -0.62 \end{pmatrix}. \quad e_2 = \begin{pmatrix} -0.62 \\ 0.78 \end{pmatrix}. \quad \lambda = \begin{pmatrix} 18.66 \\ 0.12 \end{pmatrix}$$

An Example of PCA application

Pick eigenvectors corresponding to the largest eigenvalues and put them in the columns of $e = [e_1, \ldots, e_{d'}]$

#	House Price	Area
1	5.4	4.4
2	-2.6	-1.6
3	-3.6	-2.6
4	2.4	1.9
5	-1.6	-2.1

$$e_1 = \begin{pmatrix} -0.78 \\ -0.62 \end{pmatrix}. \quad e_2 = \begin{pmatrix} -0.62 \\ 0.78 \end{pmatrix}.$$

$$e = (e_1, e_2) = \begin{pmatrix} -0.78 & -0.62 \\ -0.62 & 0.78 \end{pmatrix}.$$

An Example of PCA application

Obtain PCs $X' = eT\tilde{X}$

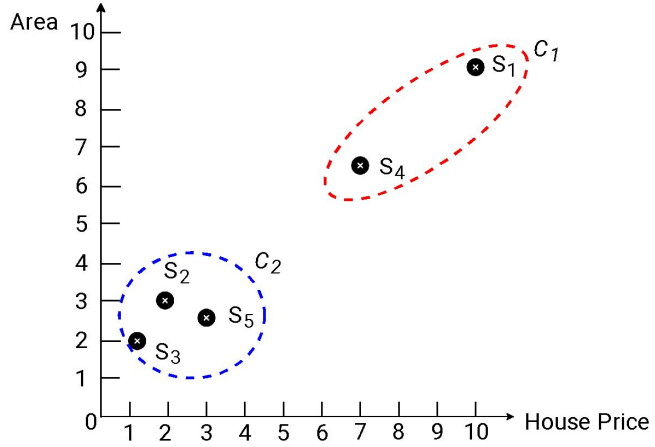
$$\tilde{X} = \begin{pmatrix} 5.4 & 4.4 \\ -2.6 & -1.6 \\ -3.6 & -2.6 \\ 2.4 & 1.9 \\ -1.6 & -2.1 \end{pmatrix}$$

$$e = (e_1, e_2) = \begin{pmatrix} -0.78 & -0.62 \\ -0.62 & 0.78 \end{pmatrix}.$$

$$-6.94 = -0.78 \times 5.4 + 4.4 \times (-0.62)$$
$$0.084 = -0.62 \times 5.4 + 0.78 \times 4.4$$

#	PC1	PC2
1	-6.94	0.084
2	3.02	0.364
3	4.42	0.204
4	-3.05	-0.006
5	2.55	-0.646

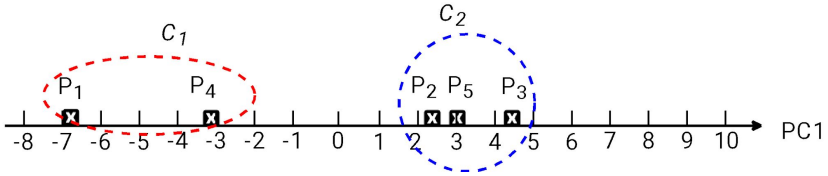
PCA for clustering



Point	House Price	Area
S_1	10	9
S_2	2	3
S_3	1	2
S_4	7	6.5
S_5	3	2.5

PCA

Map 2D to 1D



Point	PC1
P_1	-6.94
P_2	3.02
P_3	4.42
P_4	-3.05
P_5	2.55

Unsupervised Learning

Supervised Learning vs Unsupervised Learning

Supervised Learning

Data: (x, y)

Goal: Learn function to map, discover patterns in the data that relate data attributes with a target (class) attribute.

Examples: Classification, Regression, Object detection,

Object tracking, etc.

Unsupervised Learning

Data: x

Goal: Learn the hidden or intrinsic structure of the data

Examples: Clustering, dimensionality reduction,

etc.

Clustering

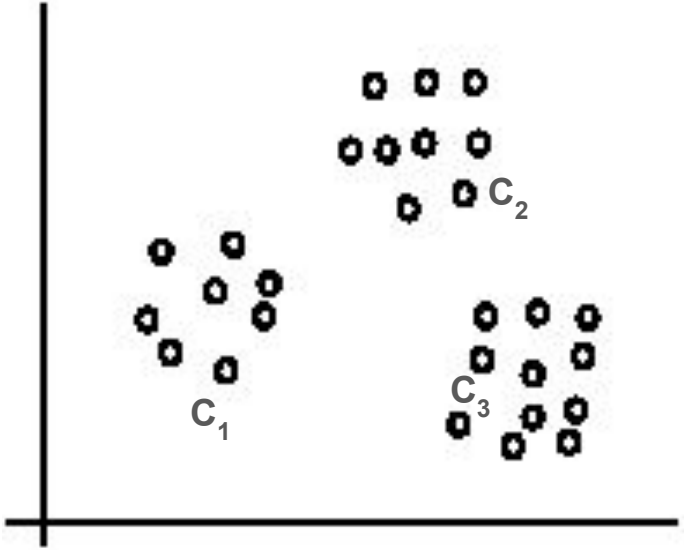
Clustering is a technique for finding similarity groups in data, called clusters. i.e.,

- It groups data instances that are similar to (near) each other in one cluster and data instances that are very different (far away) from each other into different clusters.

Clustering is often called an unsupervised learning task

- No class values

Clustering is often considered synonymous with unsupervised learning.



Applications Based on Clustering

Example 1: groups people of similar sizes together to make “small”, “medium” and “large” T-Shirts.

- Tailor-made for each person: too expensive
- One-size-fits-all: does not fit all.

Example 2: In marketing, segment customers according to their similarities

- To do targeted marketing.

Example 3: Group a collection of text documents according to their content similarities,

- To produce a topic hierarchy

In fact, clustering is one of the most utilized data mining techniques.

- It has a long history, and used in almost every field, e.g., medicine, psychology, botany, sociology, biology, archeology, marketing, insurance, libraries, etc.

Aspects of Clustering

A clustering algorithm

- Partitional clustering
- Hierarchical clustering
- …

A distance (similarity, or dissimilarity) function

Clustering quality

- Inter-clusters distance => maximized
- Intra-clusters distance => minimized

The quality of a clustering result depends on the algorithm, the distance function, and the application.

Summary

Clustering (Unsupervised Learning) has a long history and is still active.

- There are a huge number of clustering algorithms
- More are still coming every year.

Clustering is hard to evaluate, but very useful in practice.

Clustering is highly application dependent and to some extent subjective.

K-means

K-means Clustering

K-means is a partitional clustering algorithm

Let the set of data points (or instances) D be $\{x_1, x_2, \ldots, , \ldots \ x_i, x_n\}$, where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ir})$ is a vector in a real-valued space $X \subseteq R^r$, and r is the number of attributes (dimensions) in the data.

The k-means algorithm partitions the given data into k clusters.

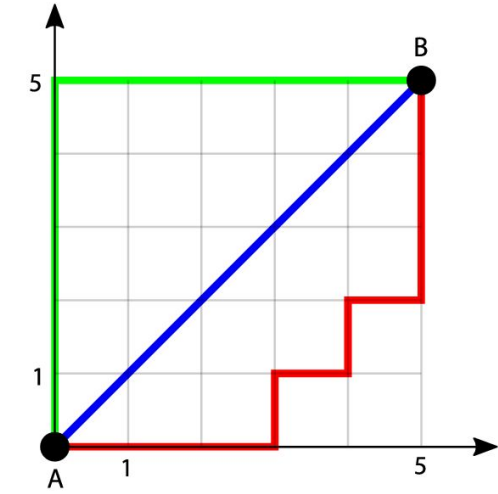
- Each cluster has a cluster center, called centroid.
- k is specified by the user

Distance Functions

Key to clustering. “similarity” and “dissimilarity” can also commonly used terms.

There are numerous distance functions for

- Different types of data
- Different specific applications



Most commonly used functions are

- Euclidean distance and $dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{ir} - x_{jr})^2}$
- Manhattan (city block) distance $dist(\mathbf{x}_i, \mathbf{x}_j) = | x_{i1} - x_{j1} | + | x_{i2} - x_{j2} | + ... + | x_{ir} - x_{jr} |$

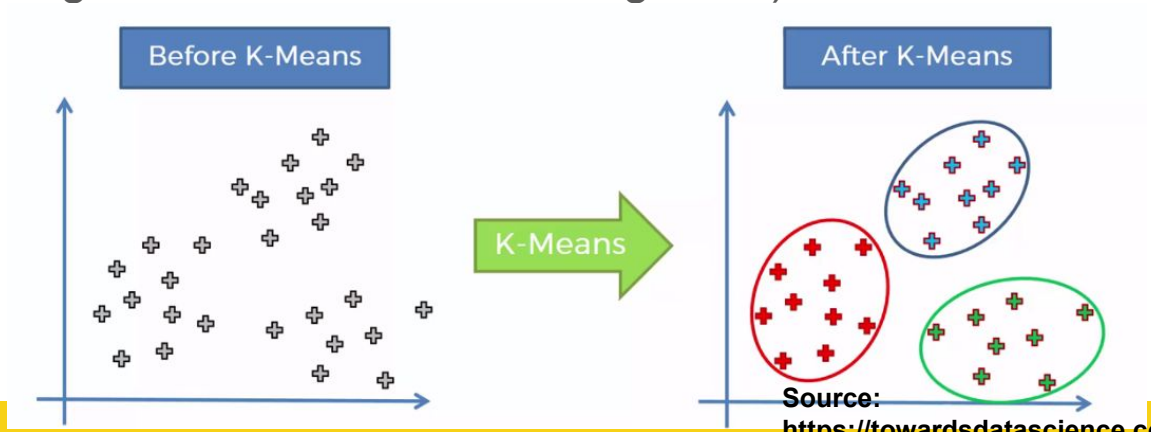
They are special cases of Minkowski distance. h is positive integer

$$dist(\mathbf{x}_i, \mathbf{x}_j) = ((x_{i1} - x_{j1})^h + (x_{i2} - x_{j2})^h + ... + (x_{ir} - x_{jr})^h)^{\frac{1}{h}}$$

Algorithm

Given k, the k-means algorithm works as follows:

1. Randomly choose k data points (seeds) to be the initial centroids, cluster centers
2. Assign each data point to the closest centroid
3. Re-compute the centroids using the current cluster memberships.
4. If a convergence criterion is not met, go to 2).



Source: https://towardsdatascience.com/k-means-clustering-identifying-f-r-i-e-n-d-s-in-the-world-of-strangers-695537505d

Stopping/Convergence Criterion

No (or minimum) re-assignments of data points to different clusters,

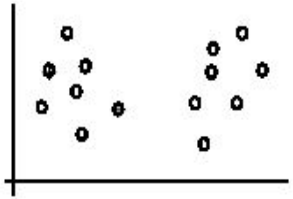
No (or minimum) change of centroids, or

Minimum decrease in the sum of squared error (SSE),

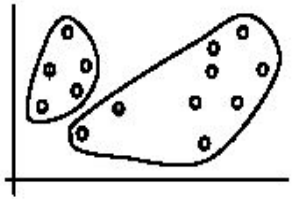
- C_i is the j^{th} cluster, m_j is the centroid of cluster C_j (the mean vector of all the data points in C_j), and $dist(x, m_j)$ is the distance between data point x and centroid m_j.

$$SSE = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} dist(\mathbf{x}, \mathbf{m}_j)^2$$

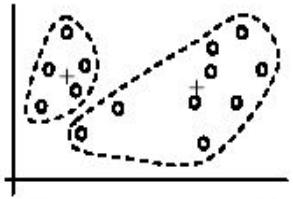
An Example of KMeans Clustering



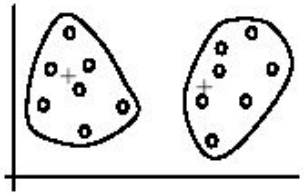
(A). Random selection of k centers



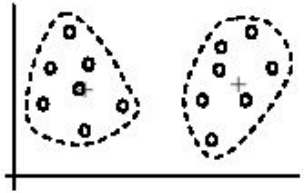
Iteration 1: (B). Cluster assignment



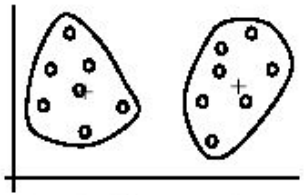
(C). Re-compute centroids



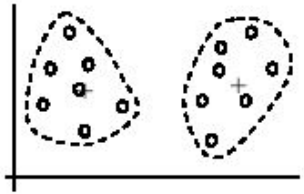
Iteration 2: (D). Cluster assignment



(E). Re-compute centroids



Iteration 3: (F). Cluster assignment



(G). Re-compute centroids

Strengths and Weakness of KMeans

Strength

Simple: easy to understand and to implement

Efficient: Time complexity: O(tkn),

- where n is the number of data points,
- k is the number of clusters, and
- t is the number of iterations.
-

Since both k and t are small, k-means is considered a linear algorithm

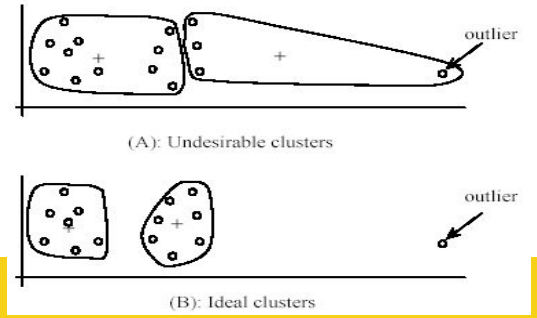
The most popular clustering algorithm

Weakness

The user needs to specify k.

The algorithm is sensitive to outliers

- Outliers are data points that are very far away from other data points.
- Outliers could be errors in the data recording or some special data points with very different values.



Strengths and Weakness of KMeans

Weakness

Processing outliers

Remove some data points in the clustering process that are much further away from the centroids than other data points.

- To be safe, we may want to monitor these possible outliers over a few iterations and then decide to remove them.

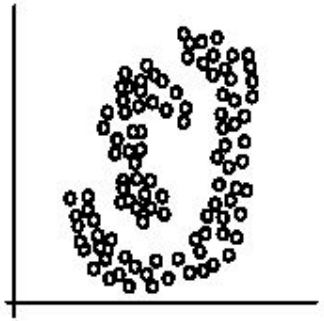
Perform random sampling.

- Since in sampling we only choose a small subset of the data points, the chance of selecting an outlier is very small.
- Assign the rest of the data points to the clusters by distance or similarity comparison, or classification

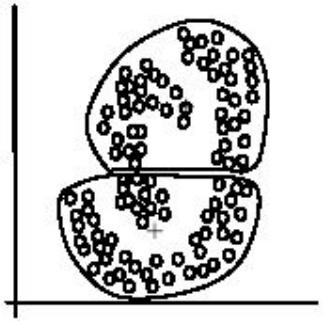
Strengths and Weakness of KMeans

Weakness

The k-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres).



(A): Two natural clusters



(B): *k*-means clusters

KMeans Summary

Despite weaknesses, k-means is still the most popular algorithm due to its simplicity, efficiency.

No clear evidence that any other clustering algorithm performs better in general

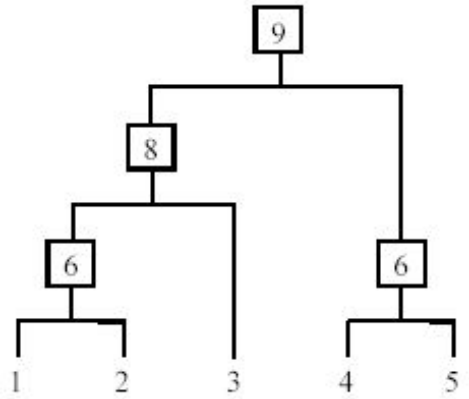
- although they may be more suitable for some specific types of data or applications.

Comparing different clustering algorithms is a difficult task.

- No one knows the correct clusters!

Hierarchical Clustering

Produce a nested sequence of clusters, a tree, also called Dendrogram.



Types of Hierarchical Clustering

Agglomerative (bottom up) clustering: It builds the tree from the bottom level, and

- Merges the most similar (or nearest) pair of clusters
- Stops when all the data points are merged into a single cluster (i.e., the root cluster).

Divisive (top down) clustering: It starts with all data points in one cluster, the root.

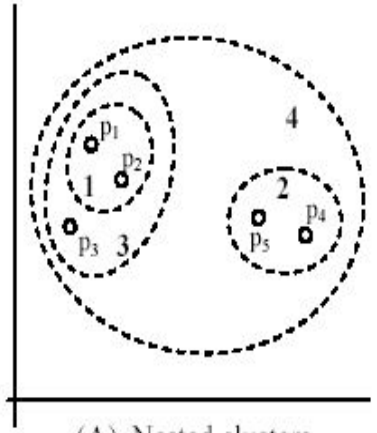
- Splits the root into a set of child clusters. Each child cluster is recursively divided further
- Stops when only singleton clusters of individual data points remain, i.e., each cluster with only a single point

Agglomerative Clustering

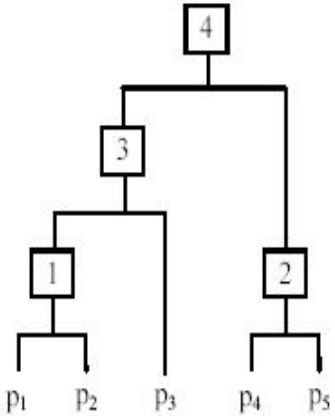
It is more popular than divisive methods.

The basic ideas are below:

1. At the beginning, each data point forms a cluster (also called a node).
2. Merge nodes/clusters that have the least distance.
3. Go on merging
4. Eventually all nodes belong to one cluster



(A). Nested clusters



(B) Dendrogram

Hard to Evaluate Clustering Performance

The quality of a clustering is very hard to evaluate since we do not know the correct clusters

Some methods are used:

- User inspection
- Study centroids
- For text documents, one can read some documents in clusters.

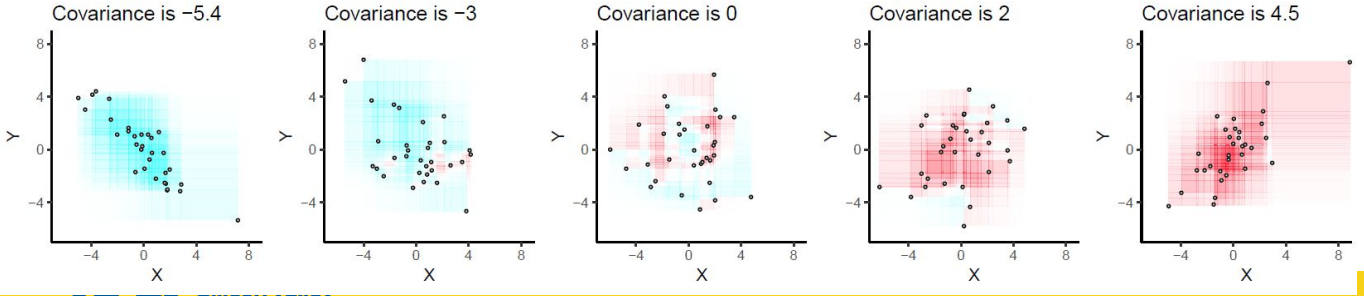
Gaussian Mixture Modelling (GMM)

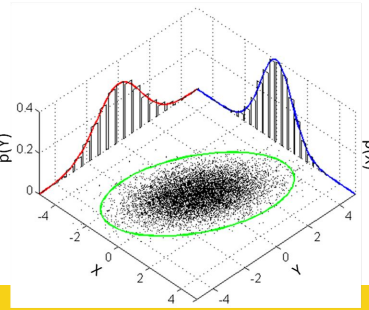
Gaussian Mixture Modelling (GMM)

For k-means, the biggest limitation is that each cluster has the same diagonal covariance matrix.

Density of the data is not considered, most of the data are distributions.

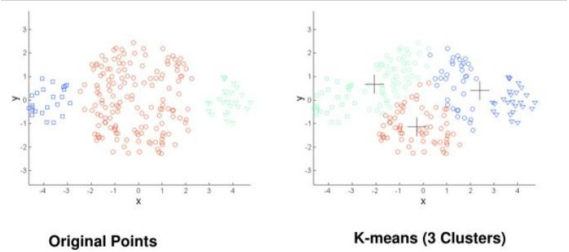
These produce spherical clusters that are quite inflexible in terms of the types of distributions they can model.

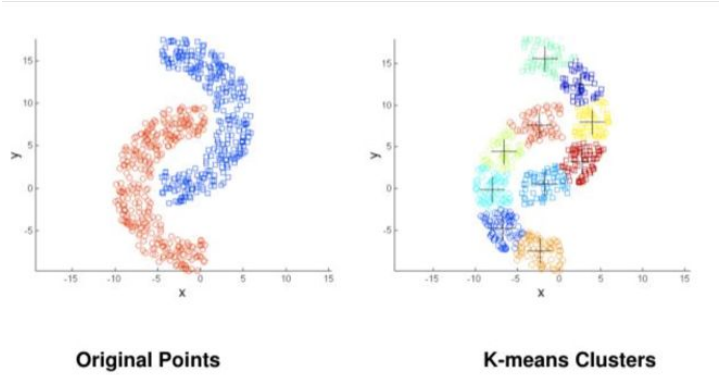




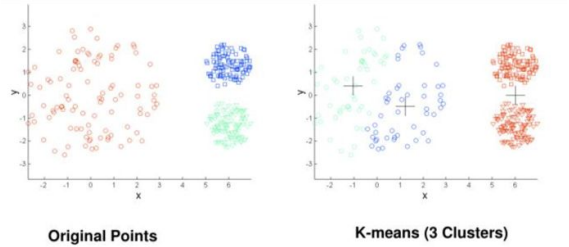
Limitation of Kmeans

Different sizes of data distribution.



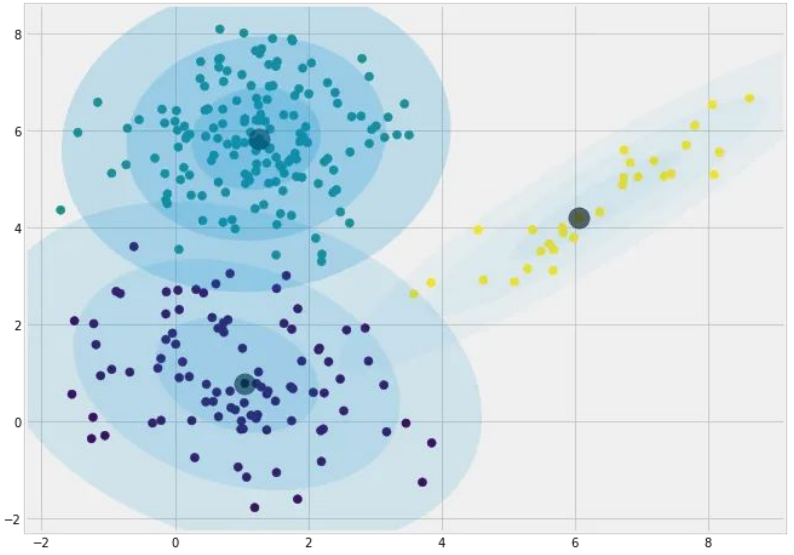


Different density of data distribution.



Solution: smaller cluster.
Or algo mentioned earlier.

GMM made simple(ish)

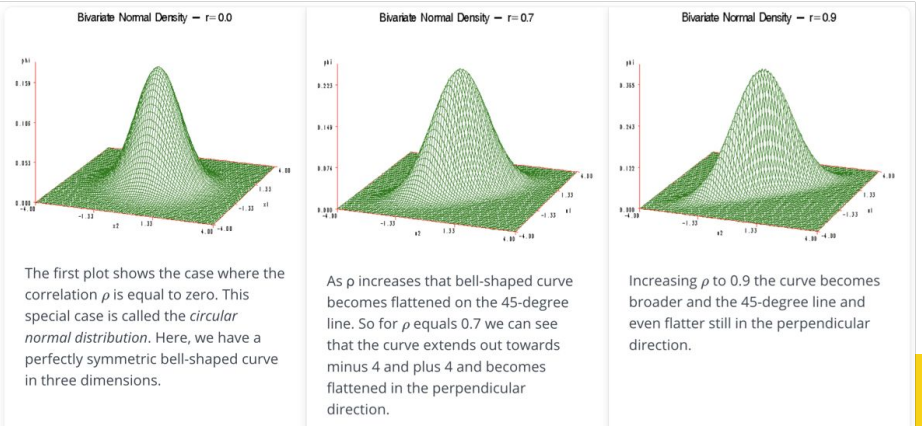


At its simplest, GMM is also a type of clustering algorithm. As its name implies, each cluster is modelled according to a different Gaussian distribution.

This flexible and probabilistic approach to modelling the data means that rather than having hard assignments into clusters like k-means, we have soft assignments.

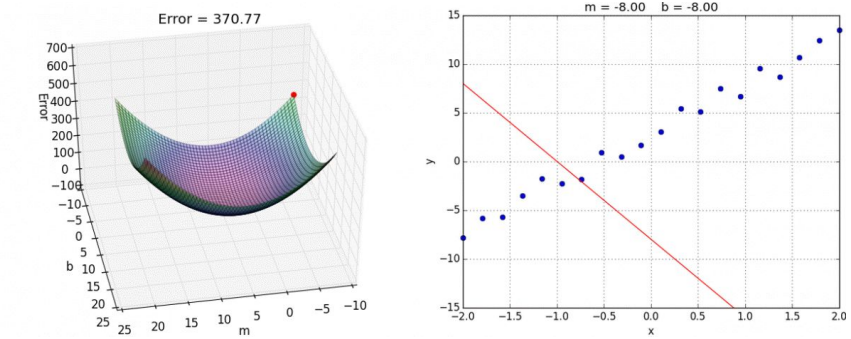
This means that each data point could have been generated by any of the distributions with a correspondi probability.

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim N\left[\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}\right]$$



Bivariate Normal Distribution

Maximum-likelihood Estimation (MLE)



We want to estimate parameter θ

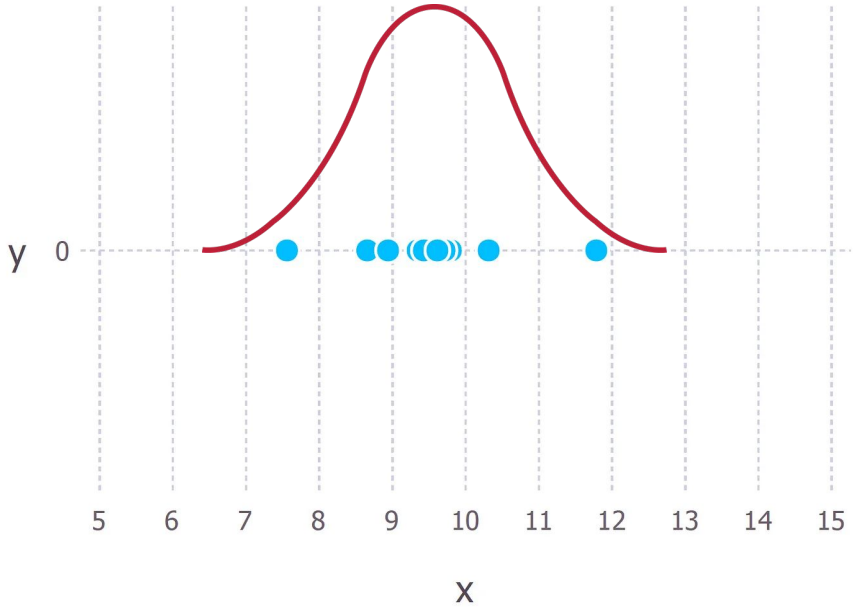
Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$

Maximum-likelihood Estimation (MLE)

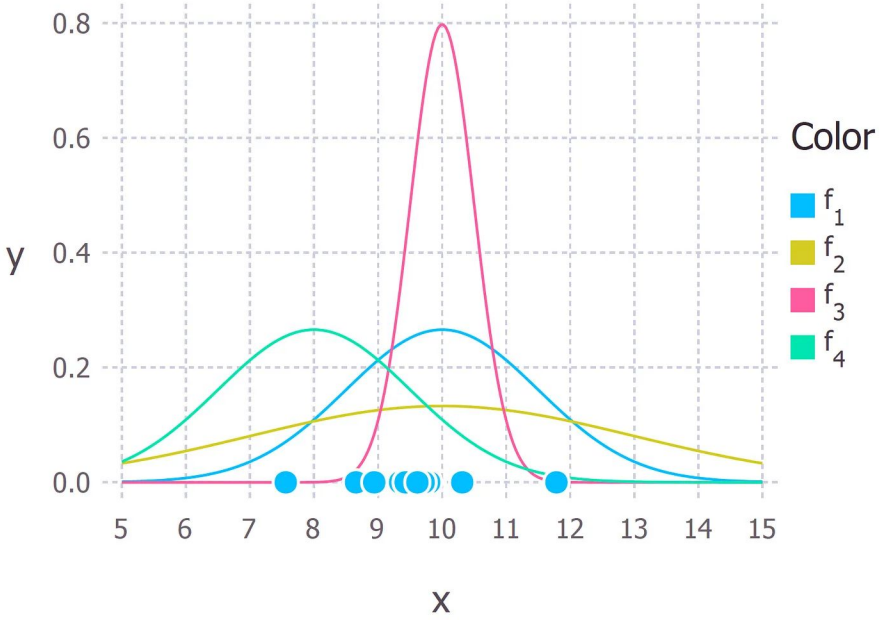


Normal Distribution:

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

$\mu =$ Mean
$\sigma =$ Standard Deviation
$\pi \approx 3.14159\cdots$
$e \approx 2.71828\cdots$

Maximum-likelihood Estimation (MLE)



Normal Distribution:

$$P(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

$$P(9, 9.5, 11;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(9-\mu)^2}{2\sigma^2}\right) \times \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(9.5-\mu)^2}{2\sigma^2}\right) \times \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(11-\mu)^2}{2\sigma^2}\right)$$

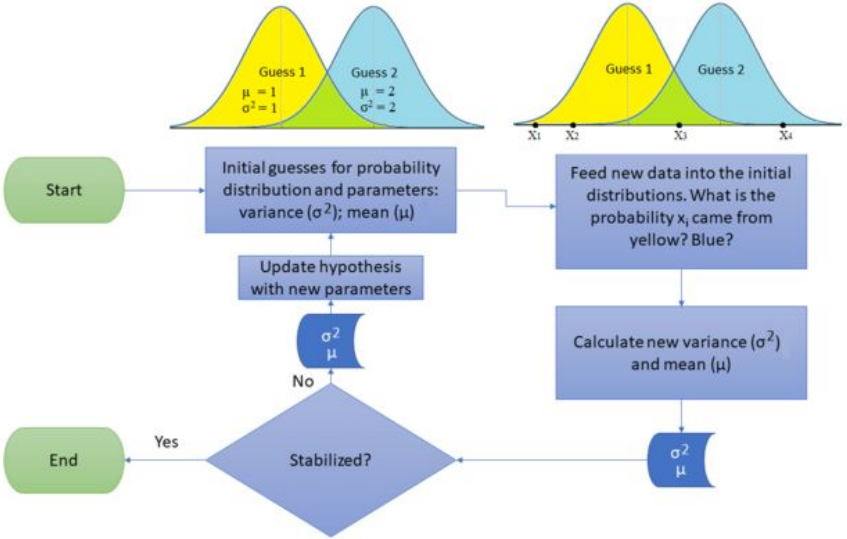
We just have to figure out the values of μ and σ that results in giving the maximum value of the above expression.

Expectation Maximisation (EM) Algorithm

How can we estimate this type of model?

The EM algorithm finds maximum-likelihood estimates for model parameters when you have incomplete data.

- The “E-Step” finds probabilities for the assignment of data points, based on a set of hypothesized probability density functions;
- The “M-Step” updates the original hypothesis with new data. The cycle repeats until the parameters stabilize.



EM Algo