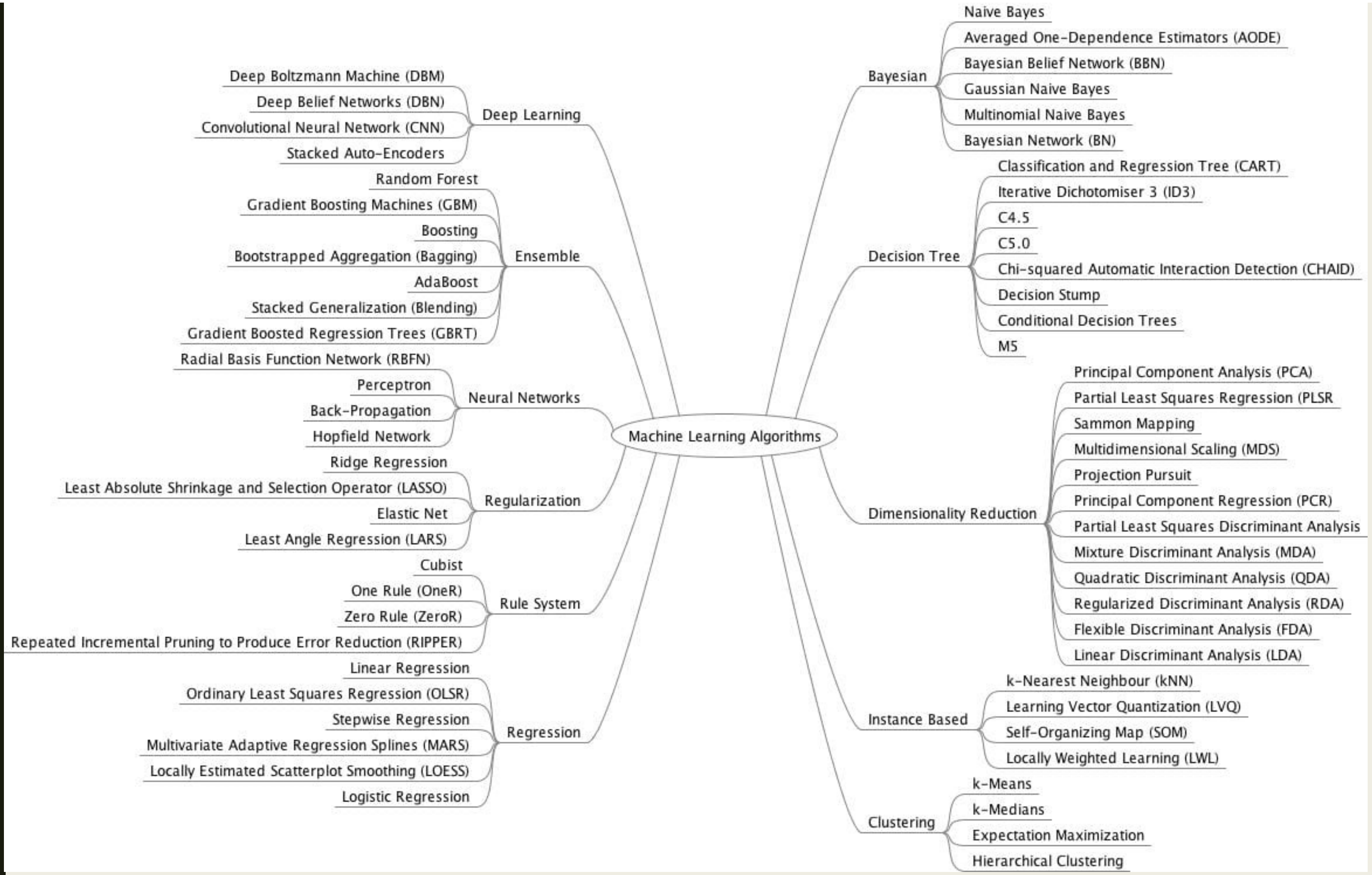
UNIT V - CLUSTERING

Dr.Vani V

LECTURE 30

Introduction to Clustering, Mixture Densities, K-Means Clustering

Dr.Vani V


Machine Learning Algorithms
Deep Learning
Deep Boltzmann Machine (DBM)
Deep Belief Networks (DBN)
Convolutional Neural Network (CNN)
Stacked Auto-Encoders
Ensemble
Random Forest
Gradient Boosting Machines (GBM)
Boosting
Bootstrapped Aggregation (Bagging)
AdaBoost
Stacked Generalization (Blending)
Gradient Boosted Regression Trees (GBRT)
Neural Networks
Radial Basis Function Network (RBFN)
Perceptron
Back-Propagation
Hopfield Network
Regularization
Ridge Regression
Least Absolute Shrinkage and Selection Operator (LASSO)
Elastic Net
Least Angle Regression (LARS)
Rule System
Cubist
One Rule (OneR)
Zero Rule (ZeroR)
Repeated Incremental Pruning to Produce Error Reduction (RIPPER)
Regression
Linear Regression
Ordinary Least Squares Regression (OLSR)
Stepwise Regression
Multivariate Adaptive Regression Splines (MARS)
Locally Estimated Scatterplot Smoothing (LOESS)
Logistic Regression
Bayesian
Naive Bayes
Averaged One-Dependence Estimators (AODE)
Bayesian Belief Network (BBN)
Gaussian Naive Bayes
Multinomial Naive Bayes
Bayesian Network (BN)
Decision Tree
Classification and Regression Tree (CART)
Iterative Dichotomiser 3 (ID3)
C4.5
C5.0
Chi-squared Automatic Interaction Detection (CHAID)
Decision Stump
Conditional Decision Trees
M5
Dimensionality Reduction
Principal Component Analysis (PCA)
Partial Least Squares Regression (PLSR
Sammon Mapping
Multidimensional Scaling (MDS)
Projection Pursuit
Principal Component Regression (PCR)
Partial Least Squares Discriminant Analysis
Mixture Discriminant Analysis (MDA)
Quadratic Discriminant Analysis (QDA)
Regularized Discriminant Analysis (RDA)
Flexible Discriminant Analysis (FDA)
Linear Discriminant Analysis (LDA)
Instance Based
k-Nearest Neighbour (kNN)
Learning Vector Quantization (LVQ)
Self-Organizing Map (SOM)
Locally Weighted Learning (LWL)
Clustering
k-Means
k-Medians
Expectation Maximization
Hierarchical Clustering

Source: Machine Learning Mastery: https://machinelearningmastery.com/wp-content/uploads/2021/03/MachineLearningAlgorithms.jpg?__s=hbkixgpvleiclesIspeo&utm_source=drip&utm_medium=email&utm_campaign=MMLA+Mini-Course&utm_content=Machine+Learning+Algorithms+Mind-Map+and+Mini-Course

Unit -V (T2-Chapter 7;T1- Chapters 14)

Unsupervised Learning-Clustering : Introduction, Mixture Densities, *K*-means Clustering, Expectation-Maximization Algorithm, Mixtures of Latent Variable Models, Supervised Learning after Clustering, Hierarchical Clustering, Choosing the Number of Clusters

Parametric & Semiparametric Approaches

- Parametric Approach:
  - *Assumption : the sample comes from a known distribution.*
  - *If assumption is weak, use a **semiparametric approach***
- Semiparametric Approach:
  - ***Allows a mixture of distributions** to be used for estimating the input sample.*
- **Clustering methods allow learning the mixture parameters from data.**

Parametric & Semiparametric Approaches

- Parametric Approach:
  - *The advantage of any parametric approach is that given a model, the problem reduces to the estimation of a small number of parameters, which, in the case of density estimation, are the sufficient statistics of the density.*
  - *For example: the mean and covariance in the case of Gaussian densities.*
  - *used quite frequently*
  - *Assume rigid parametric model may be a source of bias in many applications where this assumption does not hold.*
  - *Assuming Gaussian density corresponds to assuming that the sample, for example, instances of a class, forms one single group in the d-dimensional space*

  *Therefore, **we need more flexible models.***

Parametric & Semiparametric Approaches

- In many applications, the sample is not one group; there may be several groups
  - *Example 1: Optical Character Recognition:*
    - two ways of writing the digit 7 the American writing is ‘7’, European writing style has a horizontal bar in the middle.
    - In such a case, when the sample contains examples from both continents, the class for the digit 7 should be represented as the disjunction of two groups.
    - **If each of these groups can be represented by a Gaussian, the class can be represented by a mixture of two Gaussians,** one for each writing style.

Parametric & Semiparametric Approaches

- In many applications, the sample is not one group; there may be several groups
  - *Example 2: Speech Recognition*
    - where the same word can be uttered in different ways, due to different pronunciation, accent, gender, age, and so forth. Thus, when there is not a single, universal prototype, all these different ways should be represented in the density to be statistically correct.

This approach is called **semiparametric density estimation**, as density estimation assume a parametric model for each group in the sample.

Semiparametric Density Estimation

- Parametric: Assume a single model for $p\ (\boldsymbol{x} \mid C_i)$
- Semiparametric: $p\ (\boldsymbol{x} \mid C_i)$ is a mixture of densities

  Multiple possible explanations/prototypes:

  Different handwriting styles, accents in speech
- Nonparametric: No model; data speaks for itself

Mixture Densities

$$p(\mathbf{x}) = \sum_{i=1}^{k} p(\mathbf{x} \mid G_i) P(G_i)$$

where G_i the components/groups/clusters,

$P(G_i)$ mixture proportions (priors),

$p(\boldsymbol{x} \mid G_i)$ component densities

The number of components, k, is a hyperparameter and should be specified before.

Gaussian mixture where $p(\boldsymbol{x}|G_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, and parameters $\Phi = \{P(G_i), \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{k}$

unlabeled sample $X=\{\boldsymbol{x}^t\}_t$ (unsupervised learning)

Mixture Densities

- When the assumption is the component densities obey a parametric model, then estimate only their parameters.
- If the component densities are multivariate Gaussian then
  - *We have $p(\mathbf{x} | G_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, and parameters $\Phi = \{P(G_i), \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{k}$ are the parameters that should be estimated from the unlabeled sample $X=\{\mathbf{x}^t\}_t$ (unsupervised learning)*

Mixture Densities

- Parametric classification is a bona fide mixture model where
  - *groups, G_i , correspond to classes, C_i ,*
  - *component densities $p(x|G_i)$ correspond to class densities $p(x|C_i)$, and*
  - *$P(G_i)$ correspond to class priors, $P(C_i)$:*

$$p(\mathbf{x}) = \sum_{i=1}^{K} p(\mathbf{x}|C_i)P(C_i)$$

Mixture Densities

- In this supervised case,
  - *we are given the labels, namely, which instance belongs to which class (component).*
  - *When sample is given $X = \{x^t, r^t\}_{t=1}^N$, where $r_i^t = 1$ if $x^t \in C^i$ and 0 otherwise*
- When each class is Gaussian distributed, **estimates for the means and covariances** are
- found using maximum likelihood separately for each class:

$$\hat{P}(C_i) = \frac{\sum_t r_i^t}{N} \quad \mathbf{m}_i = \frac{\sum_t r_i^t \mathbf{x}^t}{\sum_t r_i^t}$$

$$\mathbf{S}_i = \frac{\sum_t r_i^t (\mathbf{x}^t - \mathbf{m}_i)(\mathbf{x}^t - \mathbf{m}_i)^T}{\sum_t r_i^t}$$

Classes vs. Clusters

- Supervised: X = { $\boldsymbol{x}^t, \boldsymbol{r}^t$ }$_t$
- Classes C_i $i=1,...,K$

$$p(\mathbf{x}) = \sum_{i=1}^{K} p(\mathbf{x} \mid C_i) P(C_i)$$

where $p(\boldsymbol{x} \mid C_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$

- $\Phi = \{P(C_i), \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{K}$

$$\hat{P}(C_i) = \frac{\sum_t r_i^t}{N} \quad \mathbf{m}_i = \frac{\sum_t r_i^t \mathbf{x}^t}{\sum_t r_i^t}$$

$$\mathbf{S}_i = \frac{\sum_t r_i^t (\mathbf{x}^t - \mathbf{m}_i)(\mathbf{x}^t - \mathbf{m}_i)^T}{\sum_t r_i^t}$$

- Unsupervised : X = { $\boldsymbol{x}^t$ }$_t$
- Clusters G_i $i=1,...,k$

$$p(\mathbf{x}) = \sum_{i=1}^{k} p(\mathbf{x} \mid G_i) P(G_i)$$

where $p(\boldsymbol{x} \mid G_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$

- $\Phi = \{P(G_i), \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^{k}$

Labels, $\boldsymbol{r}_i^t$?

Clustering…

A cluster is a subset of data which are similar.

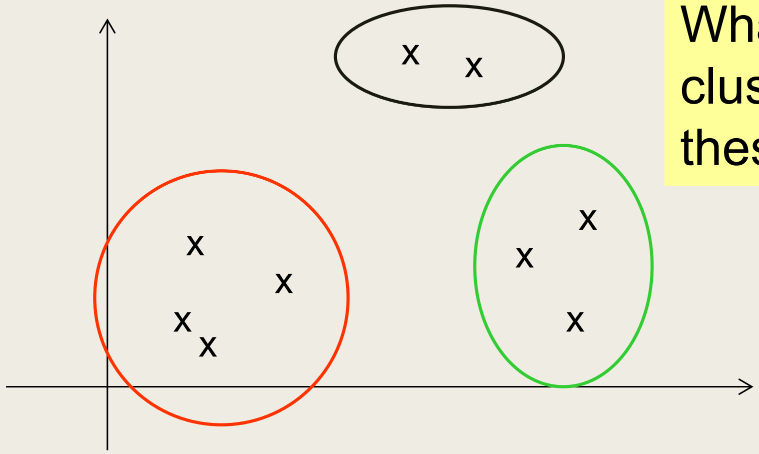
Clustering (also called unsupervised learning) is the process of dividing a dataset into groups such that the members of each group are as similar (close) as possible to one another, and different groups are as dissimilar (far) as possible from one another.

Clustering can uncover previously undetected relationships in a dataset.

There are many applications for cluster analysis. For example, in business, cluster analysis can be used to discover and characterize customer segments for marketing purposes and in biology, it can be used for classification of plants and animals given their features.

Clustering...

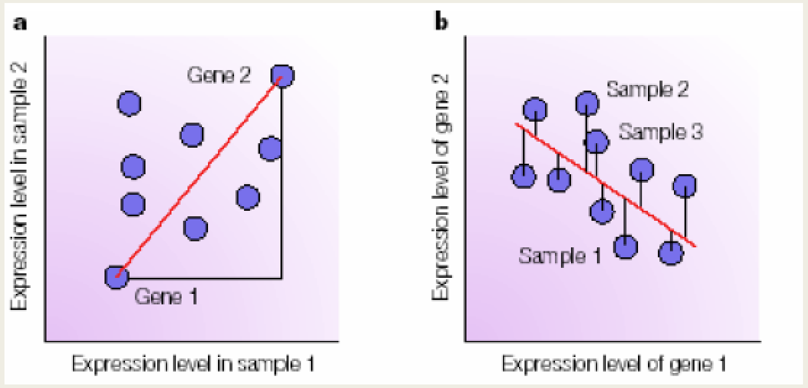
- The goal of clustering is to
  - *group data points that are close (or* ***similar****) to each other*
  - *identify such groupings (or clusters) in an* ***unsupervised*** *manner*
    - Unsupervised: no information is provided to the algorithm on which data points belong to which clusters
- Example



What should the clusters be for these data points?

Measuring Similarity

- When trying to group together objects that are similar, we need:
  1. ***Distance Metric*** –
     which define the meaning of similarity/dissimilarity

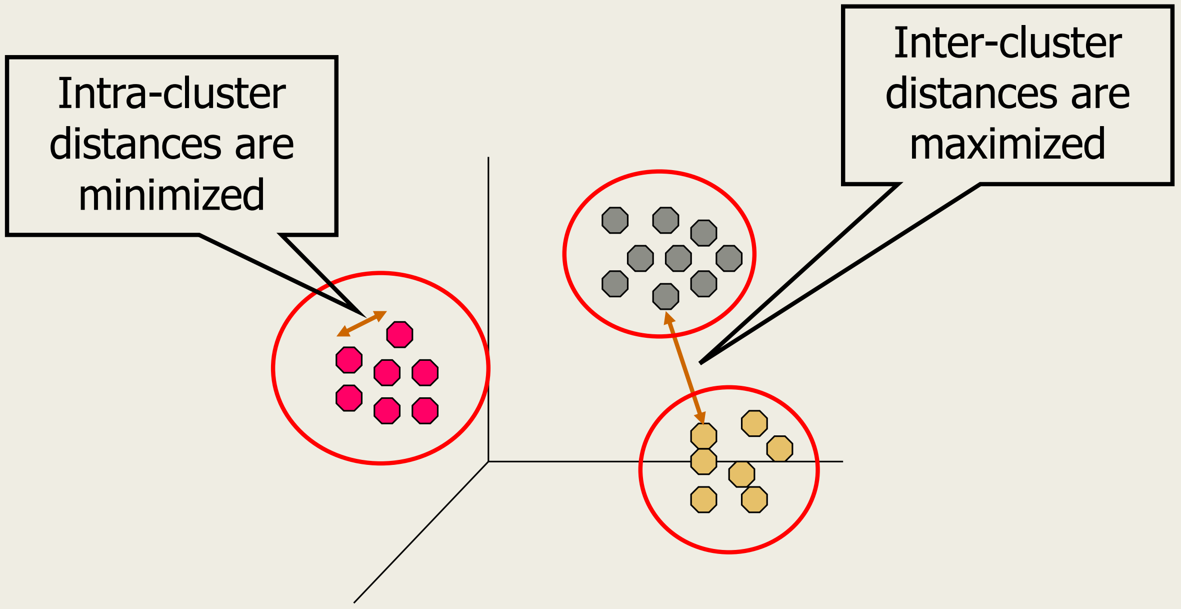


a) Two conditions and n genes b) Two genes and n conditions

What Is Good Clustering?

- A <u>good clustering</u> method will produce high quality clusters with
  - *high <u>intra-class</u> similarity*
  - *low <u>inter-class</u> similarity*
- The <u>quality</u> of a clustering result depends on both the similarity measure used by the method and its implementation.
- The <u>quality</u> of a clustering method is also measured by its ability to discover some or all of the <u>hidden</u> patterns.

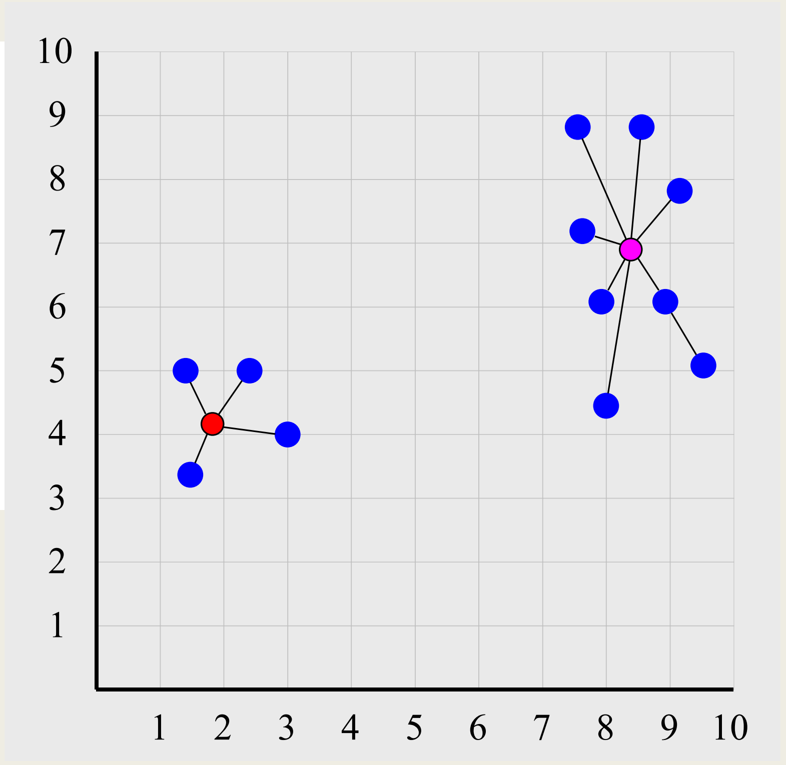
Intra-cluster and Inter-cluster distances


Intra-cluster distances are minimized
Inter-cluster distances are maximized

Squared Error

$$se_{K_i} = \sum_{j=1}^{m} \|t_{ij} - C_k\|^2$$

$$se_K = \sum_{j=1}^{k} se_{K_j}$$



Objective Function

Major Types of Clustering Algorithms

- Partitioning:

  Partition the database into k clusters which are represented by representative objects of them

- Hierarchical:

  Decompose the database into several levels of partitioning which are represented by dendrogram

- Density-based:

  Based on connectivity and density functions

k-Means Clustering

- Let us say we have an image that is stored with 24 bits/pixel and can have up to 16 million colors. Assume we have a color screen with 8 bits/pixel that can display only 256 colors. We want to find the best 256 colors among all 16 million colors such that the image using only the 256 colors in the palette looks as close as possible to the original image.
- This is **color quantization where we map from high to lower resolution.**
- In general case, the aim is to map from a continuous space to a discrete space; this vector process is called vector quantization.

k-Means Clustering

- Find *k* reference vectors (prototypes/codebook vectors/codewords) which best represent data
- Reference vectors, $\mathbf{m}_j$, $j = 1,\dots,k$
- Use nearest (most similar) reference:

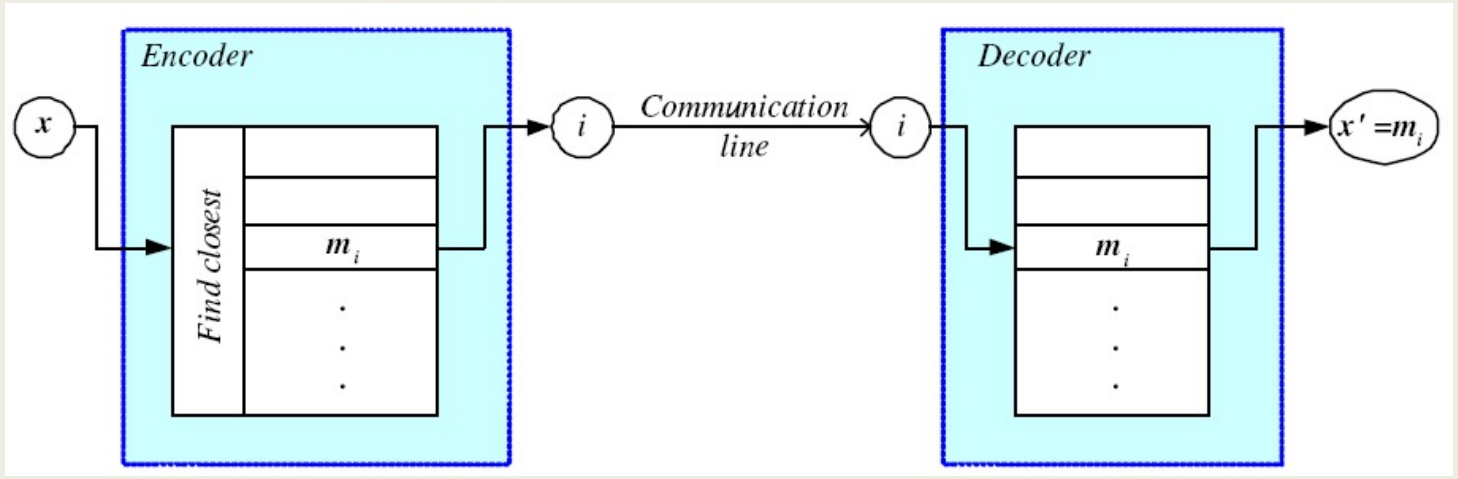
$$\left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_j \left\|\mathbf{x}^t - \mathbf{m}_j\right\|$$

- Reconstruction error

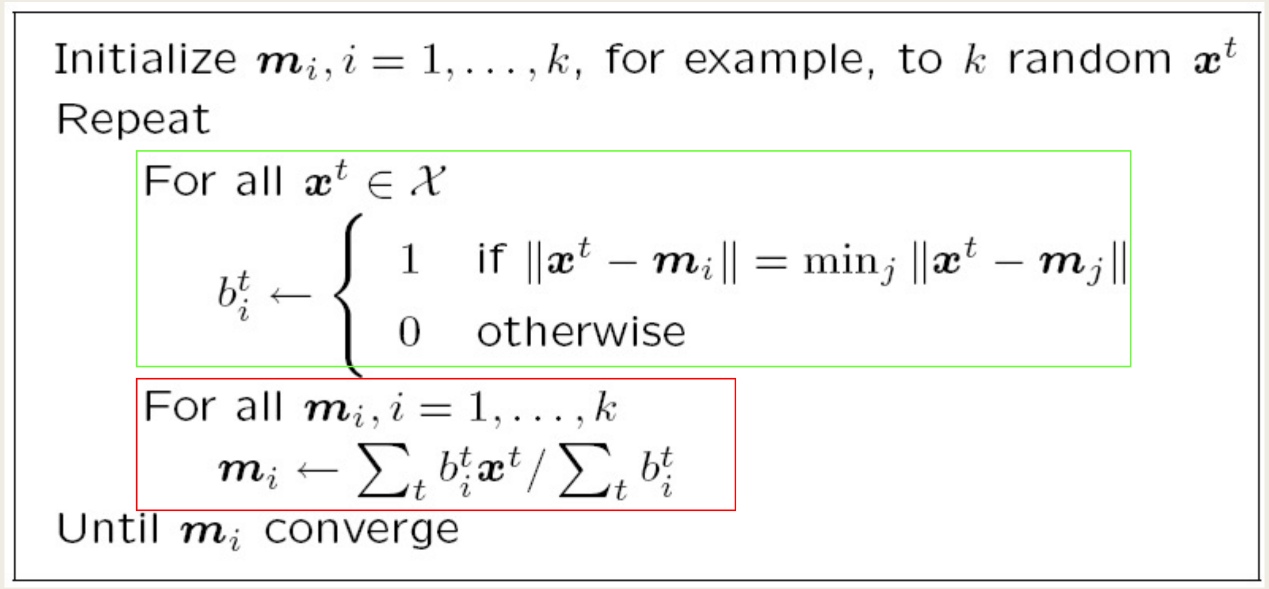
$$E\left(\{\mathbf{m}_i\}_{i=1}^{k} \middle| \mathcal{X}\right) = \sum_t \sum_i b_i^t \left\|\mathbf{x}^t - \mathbf{m}_i\right\|$$

$$b_i^t = \begin{cases} 1 & \text{if } \left\|\mathbf{x}^t - \mathbf{m}_i\right\| = \min_j \left\|\mathbf{x}^t - \mathbf{m}_j\right\| \\ 0 & \text{otherwise} \end{cases}$$

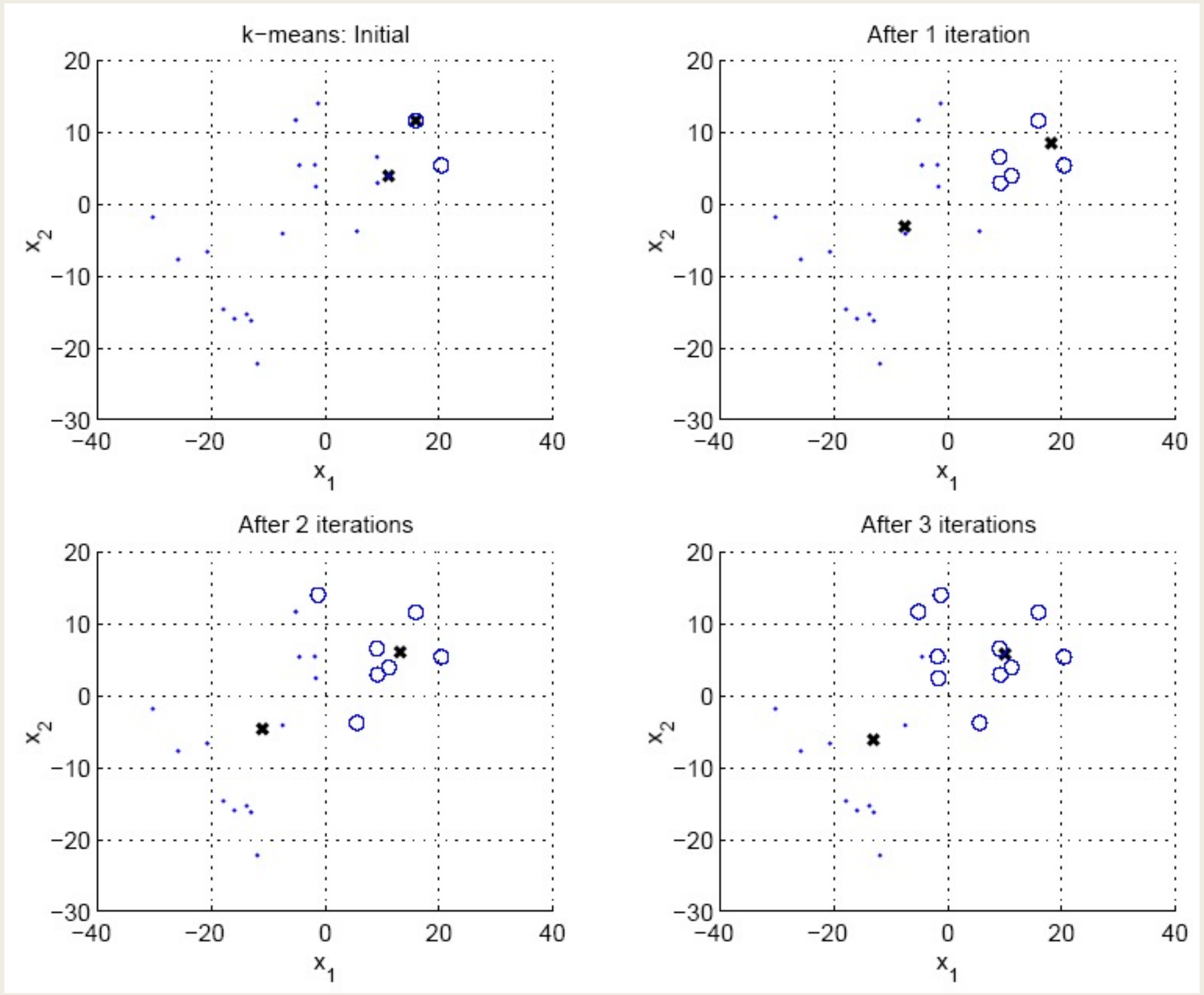
Encoding/Decoding



k-means Clustering



```
Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to $k$ random $\boldsymbol{x}^t$
Repeat
    For all $\boldsymbol{x}^t \in \mathcal{X}$
        $b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$
    For all $\boldsymbol{m}_i, i = 1, \ldots, k$
        $\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$
Until $\boldsymbol{m}_i$ converge
```



k−means: Initial
After 1 iteration
After 2 iterations
After 3 iterations
x_1
x_2

Example 1:…

- Suppose we want to group the visitors to a website using just their age (a one-dimensional space) as follows:

15,15,16,19,19,20,20,21,22,28,35,40,41,42,43,44,60,61,65

Initial clusters:

Centroid (C1) = 16 [16]

Centroid (C2) = 22 [22]

Iteration 1:

C1 = 15.33 [15,15,16]

C2 = 36.25 [19,19,20,20,21,22,28,35,40,41,42,43,44,60,61,65]

Iteration 2:

C1 = 18.56 [15,15,16,19,19,20,20,21,22]

C2 = 45.90 [28,35,40,41,42,43,44,60,61,65]

Example 1:

Iteration 3:

C1 = 19.50 [15,15,16,19,19,20,20,21,22,28]

C2 = 47.89 [35,40,41,42,43,44,60,61,65]

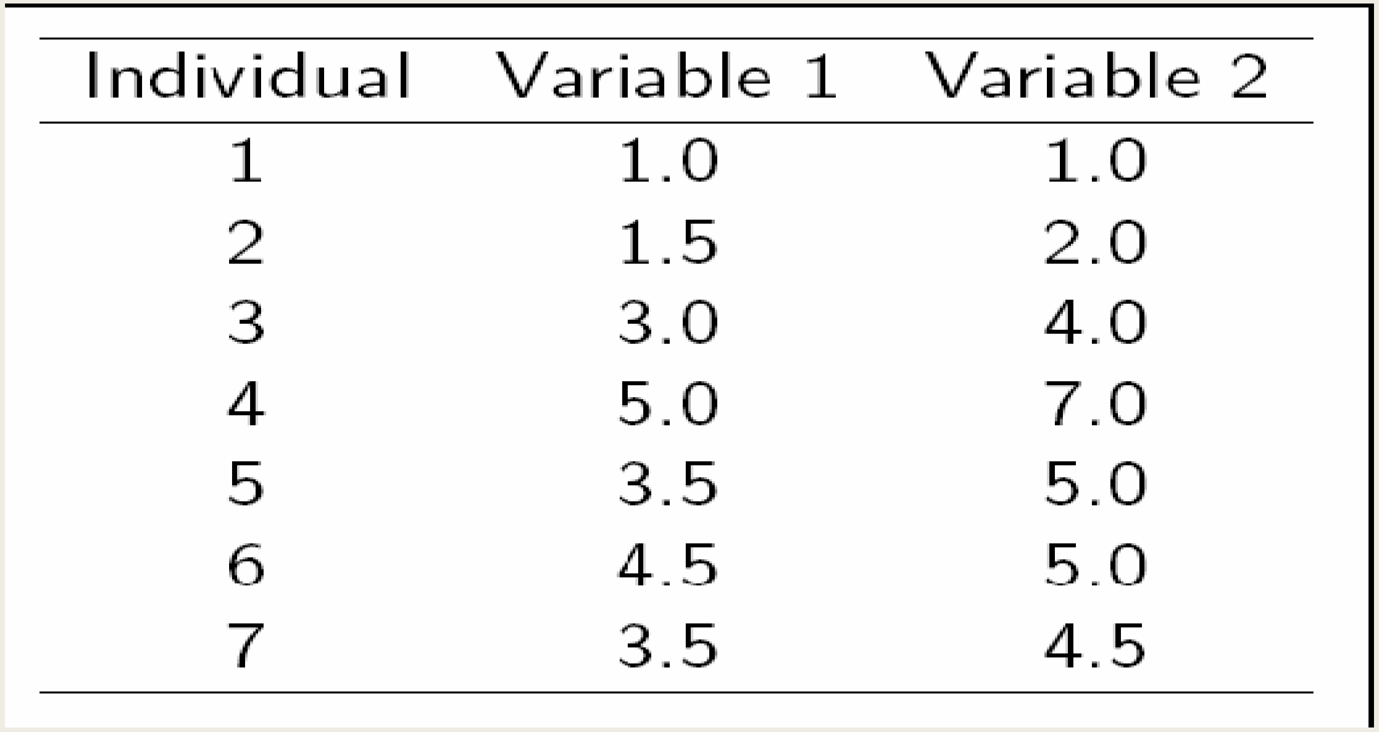
Iteration 4:

C1 = 19.50 [15,15,16,19,19,20,20,21,22,28]

C2 = 47.89 [35,40,41,42,43,44,60,61,65]

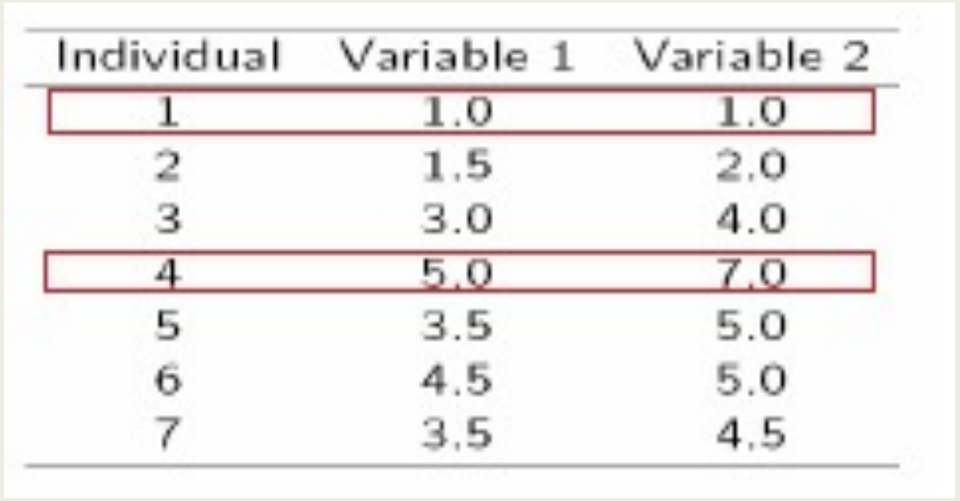
No change between iterations 3 and 4 has been noted. By using clustering, 2 groups have been identified 15-28 and 35-65. The initial choice of centroids can affect the output clusters, so the algorithm is often run multiple times with different starting conditions in order to get a fair view of what the clusters should be.

Example 2

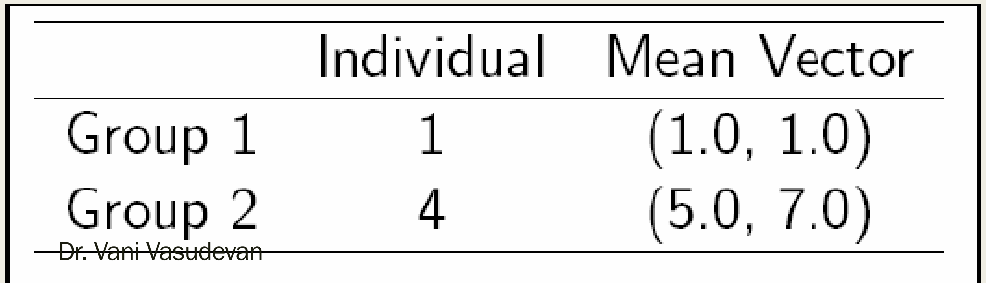


Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

Step 1:
Initialization: Randomly we choose following two centroids (k=2) for two clusters.In this case the 2 centroid are: m1=(1.0,1.0) and m2=(5.0,7.0).



Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5



	Individual	Mean Vector
Group 1	1	(1.0, 1.0)
Group 2	4	(5.0, 7.0)

Step 2:

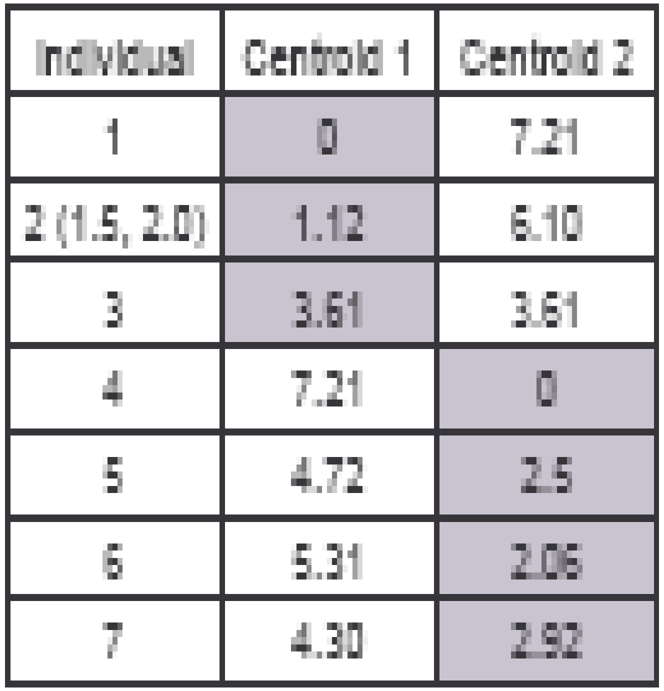
- Thus, we obtain two clusters containing:

  {1,2,3} and {4,5,6,7}.
- Their new centroids are:

$$m_1=\left(\frac{1}{3}(1.0+1.5+3.0),\frac{1}{3}(1.0+2.0+4.0)\right)=(1.83,2.33)$$

$$m_2=\left(\frac{1}{4}(5.0+3.5+4.5+3.5),\frac{1}{4}(7.0+5.0+5.0+4.5)\right)$$

$$=(4.12,5.38)$$



Individual	Centroid 1	Centroid 2
1	0	7.21
2 (1.5, 2.0)	1.12	6.10
3	3.61	3.61
4	7.21	0
5	4.72	2.5
6	5.31	2.06
7	4.30	2.92

$$d(m_1,2)=\sqrt{|1.0-1.5|^2+|1.0-2.0|^2}=1.12$$

$$d(m_2,2)=\sqrt{|5.0-1.5|^2+|7.0-2.0|^2}=6.10$$

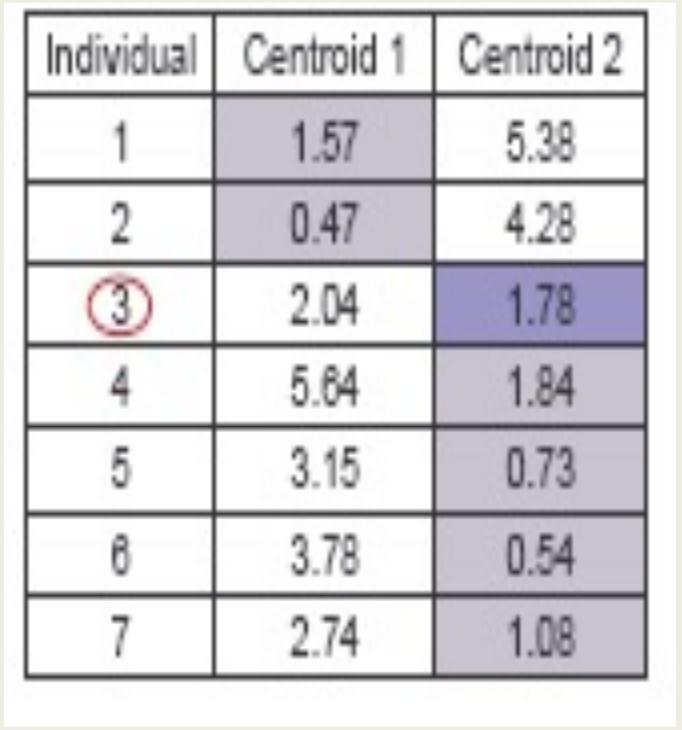
Step 3:

- Now using these centroids we compute the Euclidean distance of each object, as shown in table.

- Therefore, the new clusters are:

  {1,2} and {3,4,5,6,7}

- Next centroids are: m1=(1.25,1.5) and m2 = (3.9,5.1)



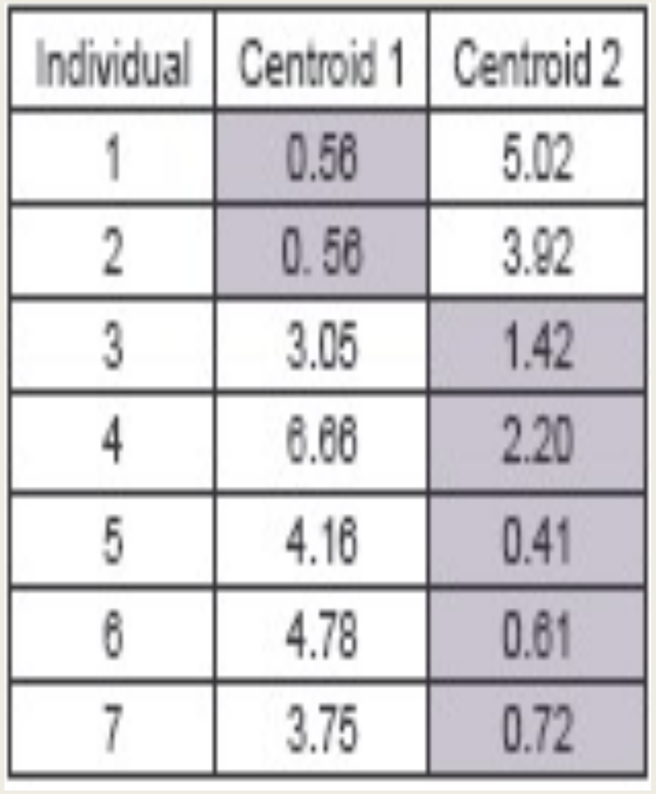
Individual	Centroid 1	Centroid 2
1	1.57	5.38
2	0.47	4.28
3	2.04	1.78
4	5.64	1.84
5	3.15	0.73
6	3.78	0.54
7	2.74	1.08

- <u>Step 4 </u>:

  The clusters obtained are:

  {1,2} and {3,4,5,6,7}

- Therefore, there is no change in the cluster.
- Thus, the algorithm comes to a halt here and final result consist of 2 clusters {1,2} and {3,4,5,6,7}.



Individual	Centroid 1	Centroid 2
1	0.56	5.02
2	0. 56	3.92
3	3.05	1.42
4	6.66	2.20
5	4.16	0.41
6	4.78	0.61
7	3.75	0.72

LECTURE 31

Exercises on SVM & K-Means Clustering

Dr.Vani V

Exercise-1

- Consider the two-dimensional data set shown below, compute the parameters of the decision boundary w1, w2 and bias .

x_1	x_2	y	Lagrange Multiplier
0.3858	0.4687	1	65.5261
0.4871	0.611	–1	65.5261
0.9218	0.4103	–1	0
0.7382	0.8936	–1	0
0.1763	0.0579	1	0
0.4057	0.3529	1	0
0.9355	0.8132	–1	0
0.2146	0.0099	1	0

$$\mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i,$$

$$y_i(w \cdot x + b) - 1 = 0$$

Exercise-2

Consider the XOR problem where there are four training points:

$(1,1,-),(1,0,+),(0,1,+),(0,0,-)$.

Transform the data into the following feature space:

$\Phi = (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2)$.

Find the maximum margin linear decision boundary in the transformed space.

Exercise-3

- Consider the 1-dimensional data set with 10 data points {1, 2, 3, . . . 10}. Show three iterations of the k-means algorithms when k = 2, and the random seeds are initialized to {1,2}. Repeat the problem with random seeds {2,9}. How did the different choice of the seed set affect the quality of the results?
- **Use Manhattan Distance Measure**

Exercise-4

- For the given initial set of three clusters (k=3), using k-means algorithm, show when the clusters converge?

Note: Use Manhattan Distance

C1 = {(1,3),(3,6),(3,5)}
C2 = {(5,3),(6,7),(2,2)}
C3 = {(6,5),(3,1),(2,3)}

LECTURE 32

Expectation-Maximization (EM) Algorithm

Dr.Vani V

Expectation-Maximization (EM)

- In k-means, we approached clustering as the problem of finding codebook vectors that minimize the total reconstruction error.
- EM approach is probabilistic and considers the component density parameters that maximize the likelihood of the sample.
- Using the mixture model

$$p(\mathbf{x}) = \sum_{i=1}^{k} p(\mathbf{x} | G_i) P(G_i)$$

the **log likelihood** given the sample $\mathcal{X} = \{x^t\}_t$ is

Expectation-Maximization (EM)

- Log likelihood with a mixture model

$$\mathcal{L}(\Phi \mid \mathcal{X}) = \log \prod_t p(\mathbf{x}^t \mid \Phi) = \sum_t \log \sum_{i=1}^{k} p(\mathbf{x}^t \mid G_i) P(G_i)$$

where Φ includes the priors $P(G_i)$ and also the sufficient statistics of the component densities $p(x^t|G_i)$.

- Assume hidden variables *z*, which when known, make optimization much simpler
- Complete likelihood, $L_c(\Phi \mid X,Z)$, in terms of ***x*** and ***z***
- Incomplete likelihood, $L(\Phi \mid X)$, in terms of ***x***

Expectation-Maximization (EM)

- The Expectation-Maximization (EM) algorithm is used in maximum likelihood estimation where the problem involves two sets of random variables of which one, **X, is observable and the other, Z, is hidden.**
- The goal of the algorithm is to find the parameter vector Φ that maximizes the likelihood of the observed values of X, L(Φ|X). But in cases where this is not feasible, we associate the extra hidden variables Z and express the underlying model using both, to maximize the likelihood of the joint distribution of X and Z, the complete likelihood Lc(Φ|X,Z).
- Since the Z values are not observed, we cannot work directly with the complete data likelihood Lc ;
- Instead, we work with its expectation, $\mathcal{Q}$ given X and the current parameter values Φ^l, where l indexes iteration. This is the expectation (E) step of the algorithm.
- Then in the maximization (M) step, we look for the new parameter values, Φ^{l+1}, that maximize this.

Worthy Reads: 1. https://www.geeksforgeeks.org/ml-expectation-maximization-algorithm/
2. https://machinelearningmastery.com/expectation-maximization-em-algorithm/
3. https://medium.com/@chloebee/the-em-algorithm-explained-52182dbb19d9

E- and M-steps

- Iterate the two steps

1. E-step: Estimate *z* given X and current Φ
2. M-step: Find new Φ' given *z*, X, and old Φ.

$$\text{E-step}: \mathcal{Q}\left(\Phi \mid \Phi^l\right) = E\left[\mathcal{L}_C(\Phi \mid \mathcal{X}, \mathcal{Z}) \mid \mathcal{X}, \Phi^l\right]$$

$$\text{M-step}: \Phi^{l+1} = \arg\max_{\Phi} \mathcal{Q}\left(\Phi \mid \Phi^l\right)$$

An increase in Q increases incomplete likelihood

$$\mathcal{L}\left(\Phi^{l+1} \mid \mathcal{X}\right) \geq \mathcal{L}\left(\Phi^l \mid \mathcal{X}\right)$$

EM in Gaussian Mixtures

- $z^t_i = 1$ if $\boldsymbol{x}^t$ belongs to G_i, 0 otherwise (labels $\boldsymbol{r}^t_i$ of supervised learning); assume $p(\boldsymbol{x}|G_i) \sim N(\boldsymbol{\mu}_i, \Sigma_i)$
- E-step:

$$E\left[z_i^t \middle| \mathcal{X}, \Phi^l\right] = \frac{p\left(\mathbf{x}^t \mid G_i, \Phi^l\right) P(G_i)}{\sum_j p\left(\mathbf{x}^t \mid G_j, \Phi^l\right) P(G_j)}$$

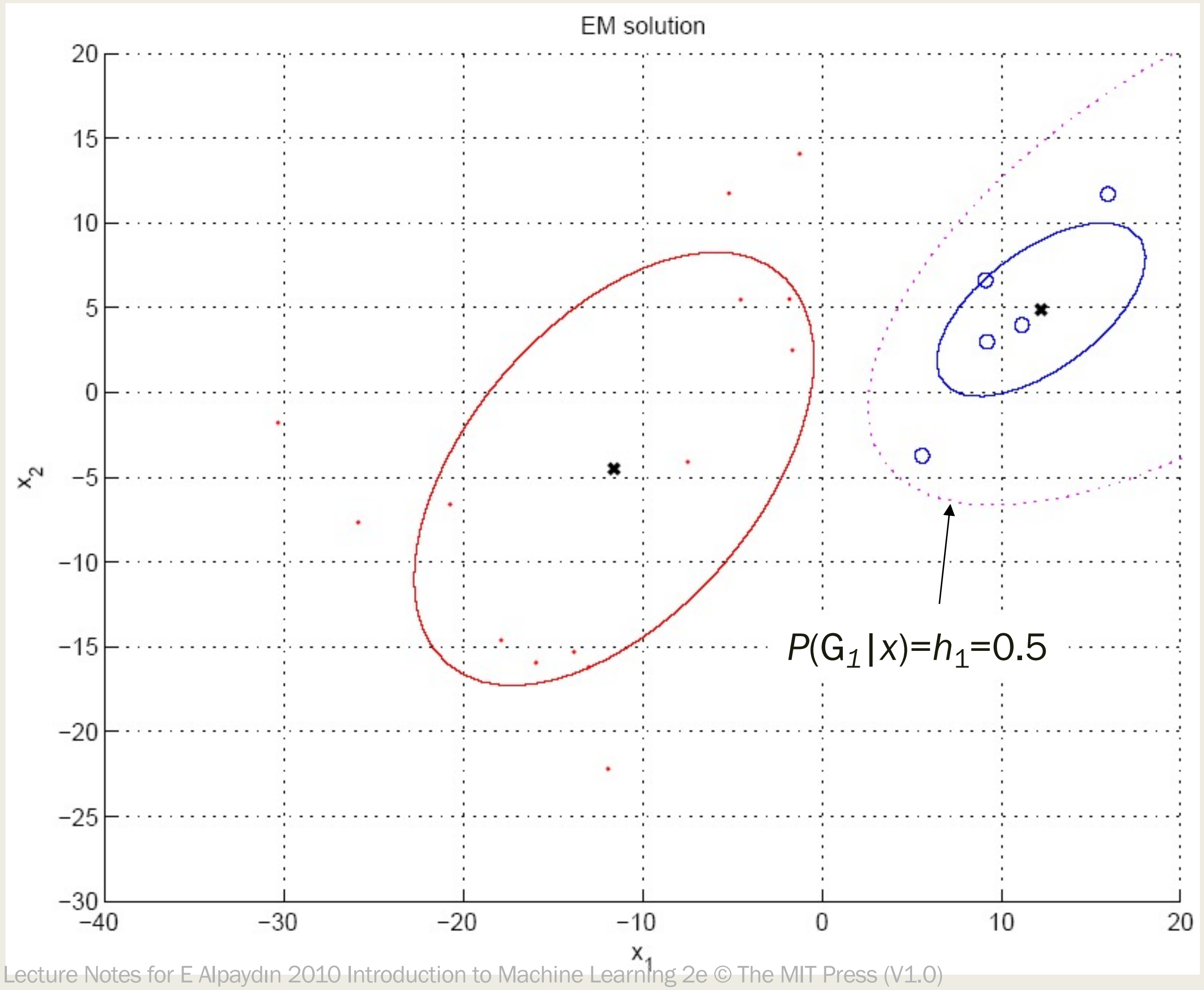
$$= P\left(G_i \mid \mathbf{x}^t, \Phi^l\right) \equiv h_i^t$$

- M-step:

$$P(G_i) = \frac{\sum_t h_i^t}{N} \qquad \mathbf{m}_i^{l+1} = \frac{\sum_t h_i^t \mathbf{x}^t}{\sum_t h_i^t}$$

Use estimated labels in place of unknown labels

$$\mathbf{S}_i^{l+1} = \frac{\sum_t h_i^t \left(\mathbf{x}^t - \mathbf{m}_i^{l+1}\right)\left(\mathbf{x}^t - \mathbf{m}_i^{l+1}\right)^T}{\sum_t h_i^t}$$



$P(G_1|x)=h_1=0.5$

Mixtures of Latent Variable Models

- Regularize clusters

1. Assume shared/diagonal covariance matrices
2. Use PCA/FA to decrease dimensionality: Mixtures of PCA/FA

$$p(\mathbf{x}_t \mid G_i) = \mathcal{N}\left(\mathbf{m}_i, \mathbf{V}_i \mathbf{V}_i^T + \boldsymbol{\psi}_i\right)$$

Can use EM to learn $\mathbf{V}_i$ (Ghahramani and Hinton, 1997; Tipping and Bishop, 1999)

After Clustering

- Dimensionality reduction methods find correlations between features and group features
- Clustering methods find similarities between instances and group instances
- Allows knowledge extraction through

    number of clusters,

    prior probabilities,

    cluster parameters, i.e., center, range of features.

  Example: CRM, customer segmentation

Clustering as Preprocessing

- Estimated group labels h_j (soft) or b_j (hard) may be seen as the dimensions of a new k dimensional space, where we can then learn our discriminant or regressor.
- Local representation (only one b_j is 1, all others are 0; only few h_j are nonzero) vs

  Distributed representation (After PCA; all z_j are nonzero)

Mixture of Mixtures

- In classification, the input comes from a mixture of classes (supervised).
- If each class is also a mixture, e.g., of Gaussians, (unsupervised), we have a mixture of mixtures:

$$p(\mathbf{x} | C_i) = \sum_{j=1}^{k_i} p(\mathbf{x} | G_{ij}) P(G_{ij})$$

$$p(\mathbf{x}) = \sum_{i=1}^{K} p(\mathbf{x} | C_i) P(C_i)$$

Hierarchical Clustering

- Cluster based on similarities/distances
- Distance measure between instances $\mathbf{x}^r$ and $\mathbf{x}^s$

Minkowski (L_p) (Euclidean for p = 2)

$$d_m(\mathbf{x}^r, \mathbf{x}^s) = \left[\sum_{j=1}^{d} (x_j^r - x_j^s)^p\right]^{1/p}$$

City-block distance

$$d_{cb}(\mathbf{x}^r, \mathbf{x}^s) = \sum_{j=1}^{d} \left|x_j^r - x_j^s\right|$$

Agglomerative Clustering

- Start with *N* groups each with one instance and merge two closest groups at each iteration
- Distance between two groups G$_i$ and G$_j$:
  - *Single-link:*

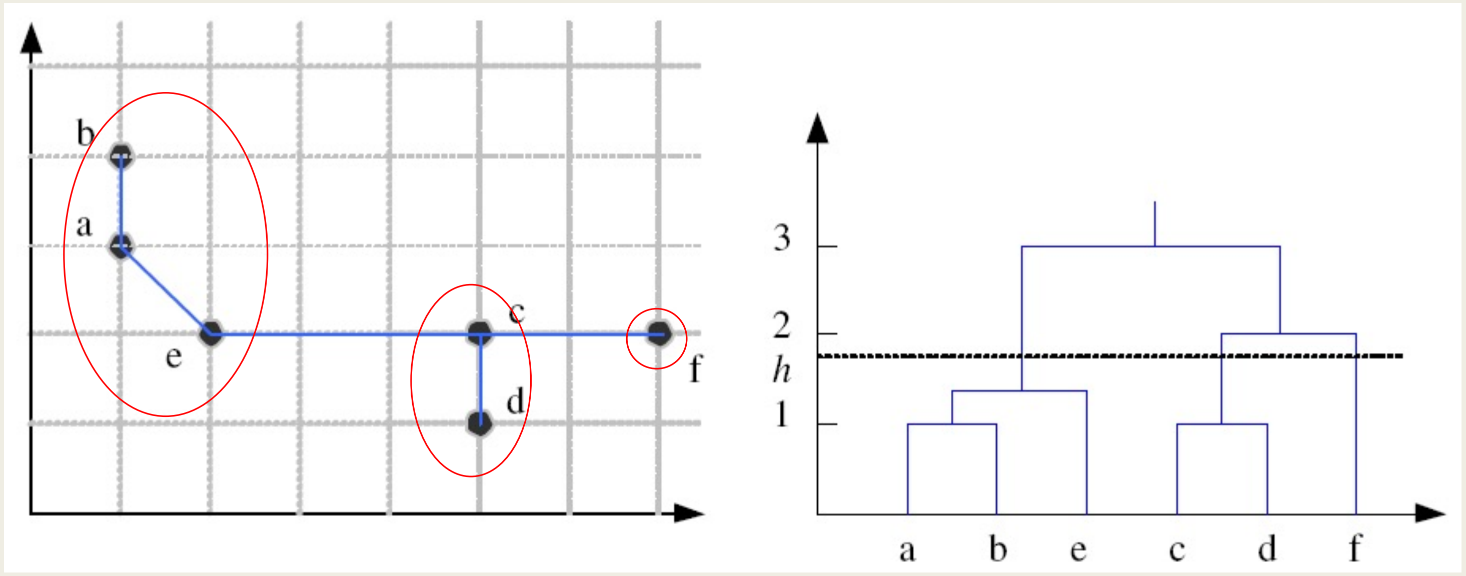
$$d(G_i, G_j) = \min_{\mathbf{x}^r \in \mathcal{G}_i, \mathbf{x}^s \in \mathcal{G}_j} d(\mathbf{x}^r, \mathbf{x}^s)$$

  - *Complete-link:*

$$d(G_i, G_j) = \max_{\mathbf{x}^r \in \mathcal{G}_i, \mathbf{x}^s \in \mathcal{G}_j} d(\mathbf{x}^r, \mathbf{x}^s)$$

  - *Average-link, centroid*

Example: Single-Link Clustering



Dendrogram

Choosing *k*

- Defined by the application, e.g., image quantization
- Plot data (after PCA) and check for clusters
- Incremental (leader-cluster) algorithm: Add one at a time until “elbow” (reconstruction error/log likelihood/intergroup distances)
- Manually check for meaning