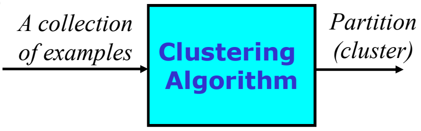
Clustering

Clustering

- Process of grouping a set of examples (samples)
- Clustering generates a partition consisting of cohesive groups or clusters from given collection of examples (samples)



A collection of examples

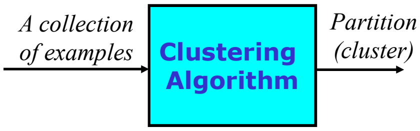
Clustering Algorithm

Partition (cluster)

- For example:
  - Grouping students in a class based on gender
  - Grouping students in a class based the month of birth
  - Grouping the students based on the place of sitting

Clustering

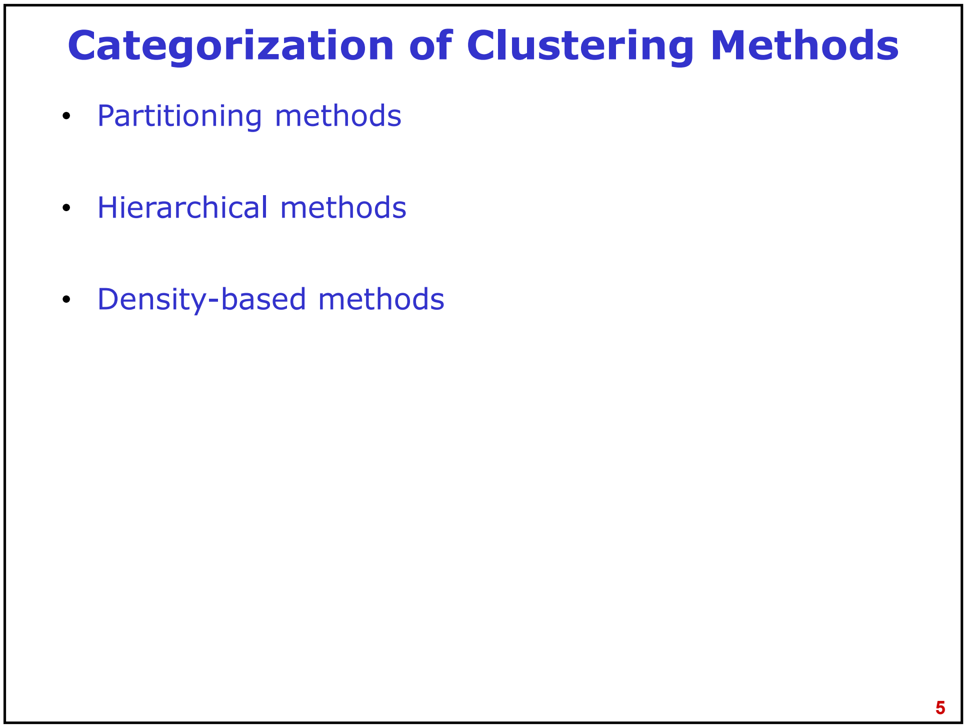
- Process of grouping a set of examples
- Clustering generates a partition consisting of cohesive groups or clusters from given collection of examples



- The examples to be clustered are either labelled or unlabelled
  - Algorithms which cluster labelled examples:
    - Supervised clustering
    - Classification: Learning by examples
  - Algorithms which cluster unlabelled examples:
    - Unsupervised clustering
    - Do not rely on predefined classes
    - Learning by observation, rather than learning by examples.

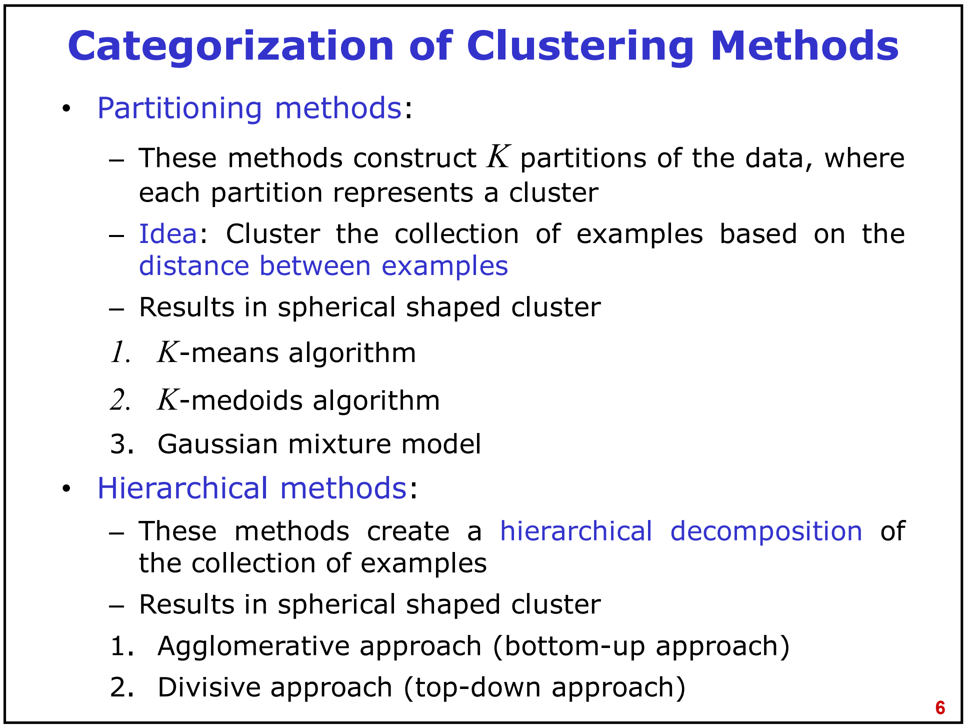
Clustering

- Clustering is a two step process
  - Step1: Partition the collection of examples (clustering)
    - Learning by observation (training phase)
    - Group the collection of examples into finite number of clusters such that the examples that are similar to one another within the same cluster and are dissimilar to examples in other clusters
    - Obtaining cluster labels
    - Unsupervised learning: Do not rely on predefined classes and class-labelled training examples
  - Step2: Assign cluster labels to examples
    - Testing phase



Categorization of Clustering Methods

- Partitioning methods
- Hierarchical methods
- Density-based methods



Categorization of Clustering Methods

- Partitioning methods:
  - These methods construct K partitions of the data, where each partition represents a cluster
  - Idea: Cluster the collection of examples based on the distance between examples
  - Results in spherical shaped cluster
  1. K-means algorithm
  2. K-medoids algorithm
  3. Gaussian mixture model
- Hierarchical methods:
  - These methods create a hierarchical decomposition of the collection of examples
  - Results in spherical shaped cluster
  1. Agglomerative approach (bottom-up approach)
  2. Divisive approach (top-down approach)

Categorization of Clustering Methods

- Density-based methods:
  - These methods cluster collection of examples based on the notion of density
  - General idea: To continue growing the given cluster as long as density (number of examples) in the neighbourhood exceeds some threshold
  - Example:
    - DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Partitioning Method based Clustering

Classical Portioning Methods

- Centroid-based technique:
  - Partition the collection of examples into K clusters based on the distance between examples
  - Cluster similarity is measured in regard to the sample mean of the examples within a cluster
  - Cluster centroid or center of gravity: Sample mean value of the examples within a cluster
  - Cluster center is used to represent the cluster
  - Example: K-means algorithm
- Representative object-based technique:
  - Actual example is considered to represent the cluster
  - One representative example per cluster
  - Example: K-medoids algorithm

K-Means Clustering Algorithm

- Dividing the data into K groups or partitions
- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$ and K
- Target: Partition the set $\mathcal{D}$ into K clusters (disjoint subsets), $\{\mathcal{D}_k\}_{k=1}^{K}$
  - Each of the clusters is associated with centers, $\boldsymbol{\mu}_k$, $k=1, 2, \ldots, K$
  - Come up with the centers of clusters
  - Cluster center acts as a cluster representative
- Euclidean distance with center of a cluster can be used as a measure of dissimilarity

K-Means Clustering Algorithm: Training Phase

- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$ and K

1. Initialize the cluster center, $\boldsymbol{\mu}_k$, $k=1, 2, \ldots, K$ using randomly selected K data points in $\mathcal{D}$
2. Assign each data point $\mathbf{x}_n$ to cluster center k^*

$$k^* = \arg\min_{k} \left\| \mathbf{x}_n - \boldsymbol{\mu}_k \right\|^2$$

Squared Euclidian distance

3. Update $\boldsymbol{\mu}_k$, $k=1, 2, \ldots, K$: Re-compute $\boldsymbol{\mu}_k$ after assigning all the data points.

$$\hat{\boldsymbol{\mu}}_k = \frac{\sum_{\mathcal{D}_k} \mathbf{x}_n}{N_k}$$

$\mathcal{D}_k$: Data for cluster k

N_k: Number of examples in cluster k

4. Repeat the steps 2 and 3 until the convergence

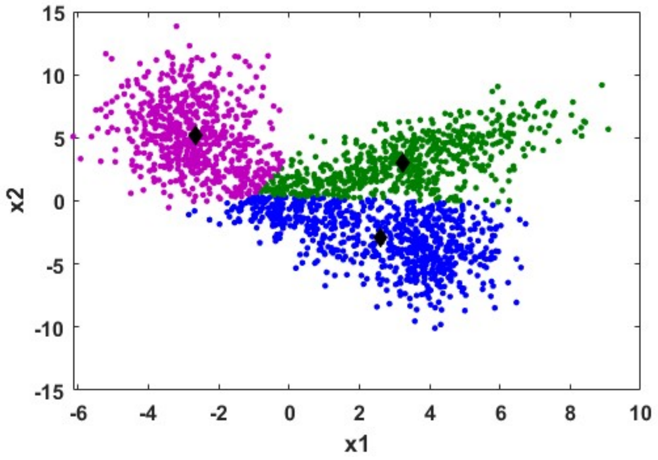
K-Means Clustering Algorithm: Training Phase

- Convergence criteria:
  - No change in the cluster assignment **OR**
  - The difference between the distortion measure (J) in the successive iteration falls below the threshold
    - Distortion measure (J) : Sum of the squares of the distance of each example to its assigned cluster center

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \left\| \mathbf{x}_n - \boldsymbol{\mu}_k \right\|^2$$

z_{nk} is 1 if $\mathbf{x}_n$ belongs to cluster k, otherwise 0

Illustration of K-Means Clustering



$K = 3$

- Boundary between the cluster is linear
- Hard clustering: Each example must belong to exactly one group

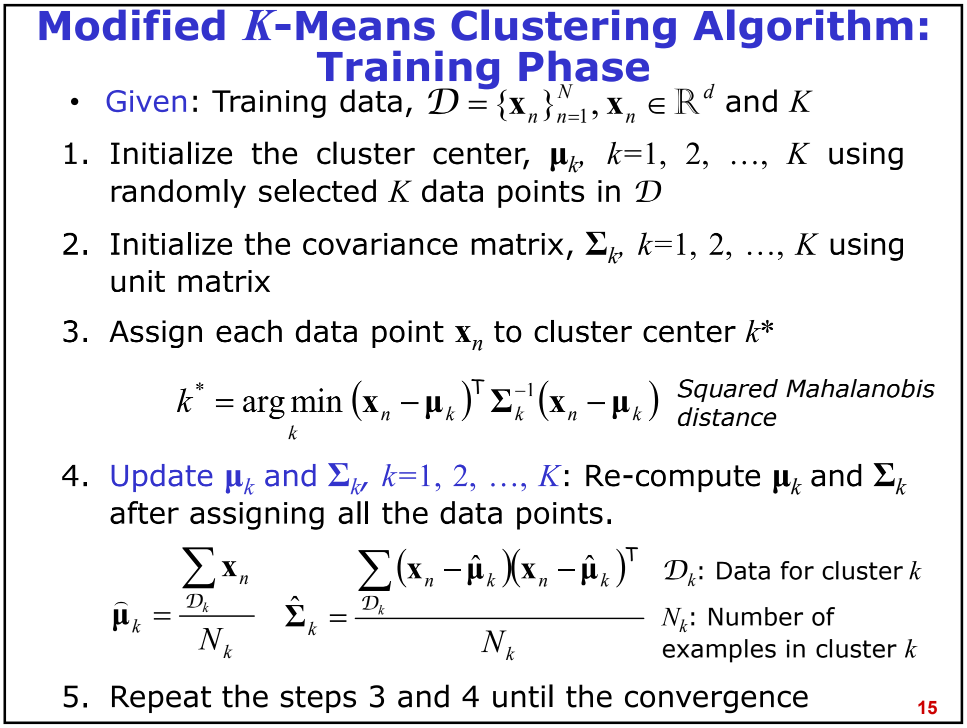
Modified K-Means Clustering Algorithm

- Dividing the data into K groups or partitions
- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$ and K
- Target: Partition the set $\mathcal{D}$ into K clusters (disjoint subsets), $\{\mathcal{D}_k\}_{k=1}^{K}$
  - Each of the clusters is associated with centers, $\boldsymbol{\mu}_k$, k=1, 2, …, K
  - ***Better representative for a cluster***
  - Come up with the centers of clusters and variance & covariance (covariance matrix)
  - Cluster center and covariance matrix act as cluster representatives
  - For d=2

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} E[x_1] \\ E[x_2] \end{bmatrix}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix} \quad \boldsymbol{\Sigma} = \begin{bmatrix} E\left[(x_1-\mu_1)^2\right] & E[(x_1-\mu_1)(x_2-\mu_2)] \\ E[(x_2-\mu_2)(x_1-\mu_1)] & E\left[(x_2-\mu_2)^2\right] \end{bmatrix}$$

- Mahalanobis distance with cluster representatives can be used as a measure of dissimilarity



Modified K-Means Clustering Algorithm: Training Phase

- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$ and K

1. Initialize the cluster center, $\boldsymbol{\mu}_k$, $k=1, 2, \ldots, K$ using randomly selected K data points in $\mathcal{D}$
2. Initialize the covariance matrix, $\boldsymbol{\Sigma}_k$, $k=1, 2, \ldots, K$ using unit matrix
3. Assign each data point $\mathbf{x}_n$ to cluster center k^*

$$k^* = \arg\min_k \left(\mathbf{x}_n - \boldsymbol{\mu}_k\right)^{\mathsf{T}} \boldsymbol{\Sigma}_k^{-1} \left(\mathbf{x}_n - \boldsymbol{\mu}_k\right)$$

Squared Mahalanobis distance

4. Update $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$, $k=1, 2, \ldots, K$: Re-compute $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ after assigning all the data points.

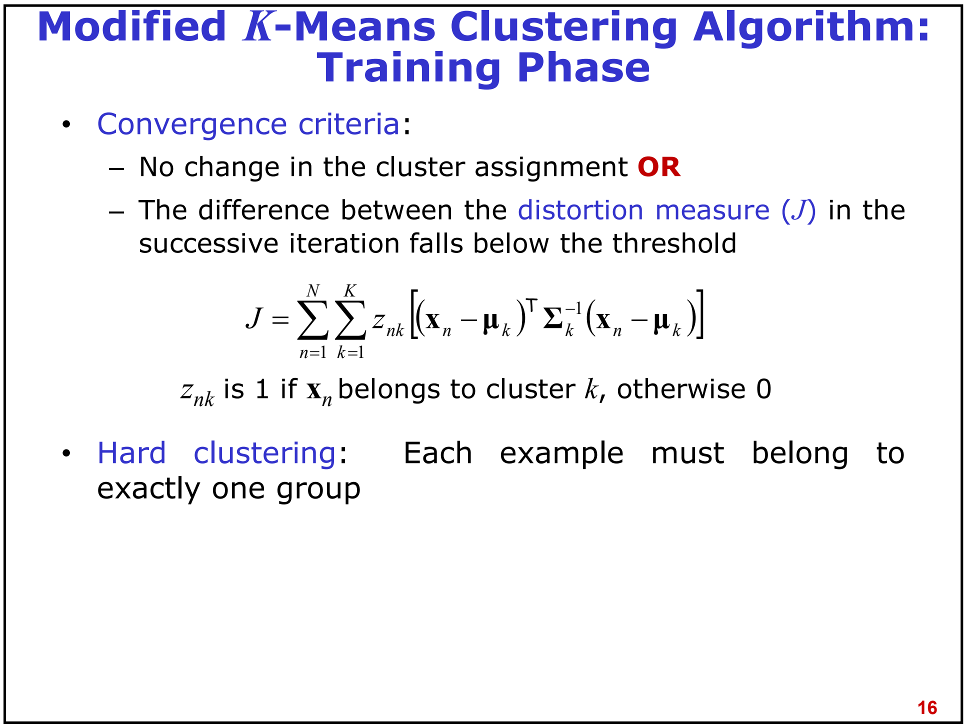
$$\hat{\boldsymbol{\mu}}_k = \frac{\sum_{\mathcal{D}_k} \mathbf{x}_n}{N_k} \qquad \hat{\boldsymbol{\Sigma}}_k = \frac{\sum_{\mathcal{D}_k} \left(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k\right)\left(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k\right)^{\mathsf{T}}}{N_k}$$

$\mathcal{D}_k$: Data for cluster k

N_k: Number of examples in cluster k

5. Repeat the steps 3 and 4 until the convergence

15



Modified K-Means Clustering Algorithm: Training Phase

- Convergence criteria:
  - No change in the cluster assignment **OR**
  - The difference between the distortion measure (J) in the successive iteration falls below the threshold

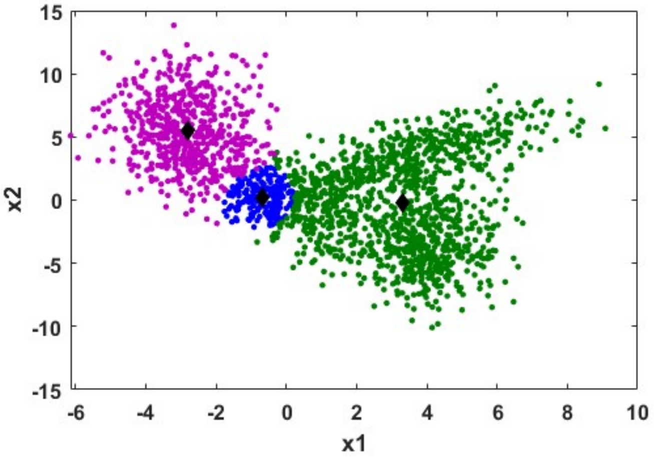
$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \left[\left(\mathbf{x}_n - \boldsymbol{\mu}_k\right)^{\mathsf{T}} \boldsymbol{\Sigma}_k^{-1} \left(\mathbf{x}_n - \boldsymbol{\mu}_k\right)\right]$$

z_{nk} is 1 if $\mathbf{x}_n$ belongs to cluster k, otherwise 0

- Hard clustering: Each example must belong to exactly one group

16

Illustration of *K*-Means Clustering



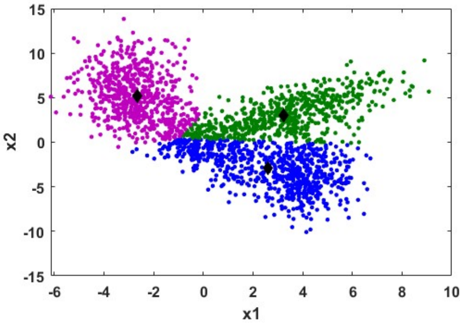
K = 3

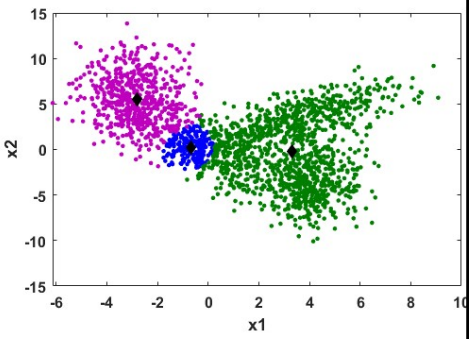
- Boundary between the cluster is quadratic
- Hard clustering: Each example must belong to exactly one group

17

Illustration of *K*-Means Clustering

K = 3



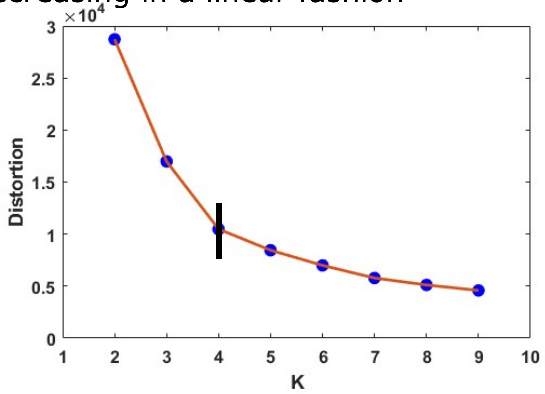


Measure of Dissimilarity: ***Euclidian Distance*** ***Mahalanobis Distance***

18

Elbow Method to Choose K

- Determine the distortion measure for different values of K
- Plot the K vs Distortion
- Optimal number of clusters: Select the value of K at the "elbow" i.e. the point after which the distortion start decreasing in a linear fashion



19

K-Means Clustering Algorithm: Training Phase

- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$ and K

1. Initialize the cluster center, $\boldsymbol{\mu}_k$, $k=1, 2, \ldots, K$ using randomly selected K data points in $\mathcal{D}$
2. Assign each data point $\mathbf{x}_n$ to cluster center k*

$$k^* = \arg\min_k \left\|\mathbf{x}_n - \boldsymbol{\mu}_k\right\|^2$$ *Squared Euclidian distance*

3. Update $\boldsymbol{\mu}_k$, $k=1, 2, \ldots, K$: Re-compute $\boldsymbol{\mu}_k$ after assigning all the data points.

$$\widehat{\boldsymbol{\mu}}_k = \frac{\sum_{\mathcal{D}_k} \mathbf{x}_n}{N_k}$$

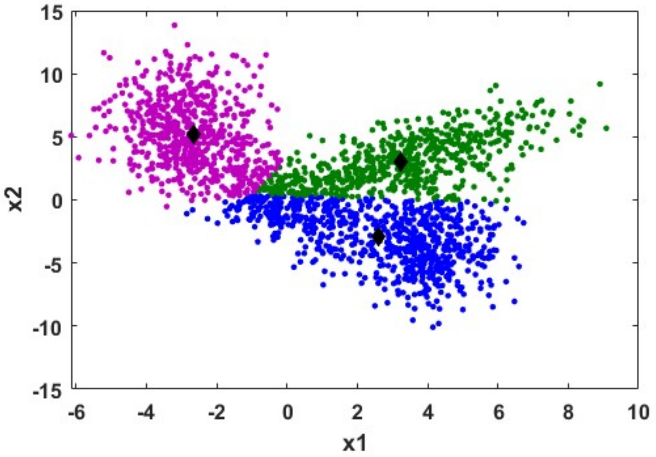
$\mathcal{D}_k$: Data for cluster k

N_k: Number of examples in cluster k

4. Repeat the steps 2 and 3 until the convergence

20

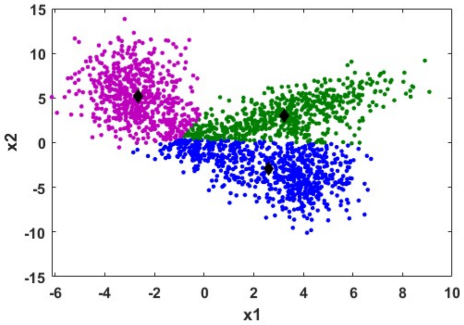
Illustration of K-Means Clustering



$K = 3$

- Boundary between the cluster is linear
- Hard clustering: Each example must belong to exactly one group

Soft Clustering



- Soft clustering: Each example belong to each group with some probability
  - Fuzzyness at the boundary of the clusters
- Gaussian mixture model (GMM) is one of the soft clustering techniques
- GMM can be seen as similar to K-means clustering
- Each cluster is represented as Gaussian density

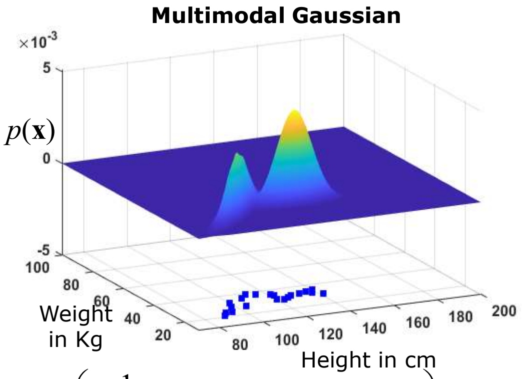
Gaussian Mixture Model (GMM)

- Data is considered to have multiple clusters and each cluster is an Gaussian distribution
- Given: Training data having N samples

$$\mathcal{D}=\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, \ldots, \mathbf{x}_N\}, \quad \mathbf{x}_n \in \mathbb{R}^d$$

- GMM is a linear superposition of multiple (K) *Gaussian components*:

$$p(\mathbf{x}) = \sum_{k=1}^{K} w_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$



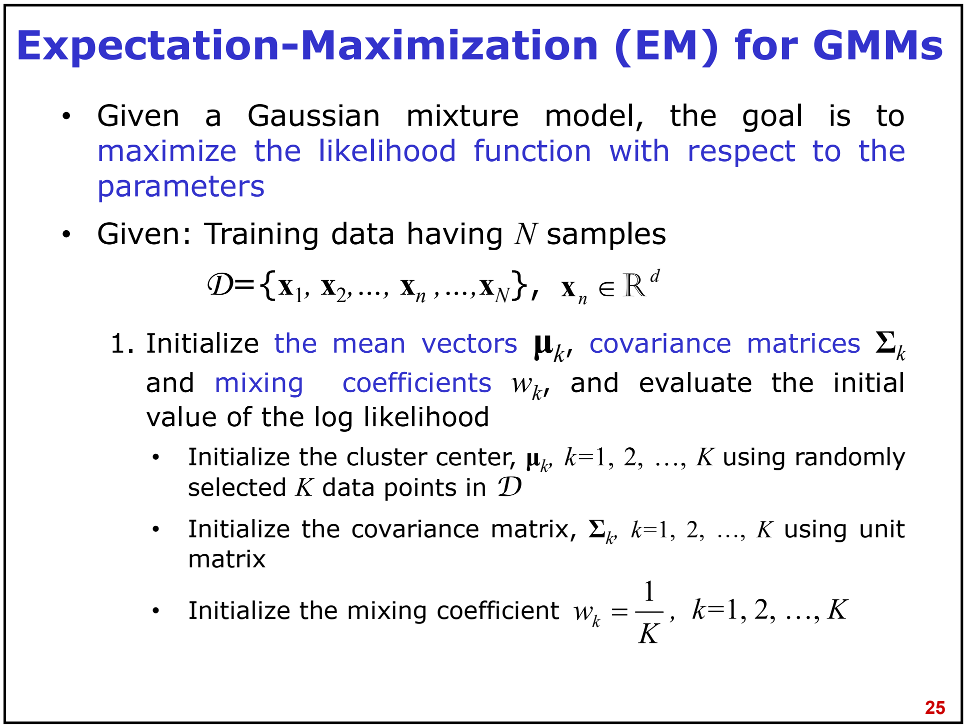
$$\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}_k|^{1/2}} \exp\left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)^{\mathsf{T}} \boldsymbol{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \right)$$

Gaussian Mixture Model (GMM)

- GMM is a linear superposition of multiple Gaussians:

$$p(\mathbf{x}) = \sum_{k=1}^{K} w_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- For a d-dimensional feature vector representation of data, the parameters of GMM are
  - Mixture coefficients, w_k, $k = 1, 2, \ldots, K$
    - *Mixture weight or Strength of each clusters (or mixtures or modes)*
    - Property: $\sum_{k=1}^{K} w_k = 1$
  - d-dimensional mean vector, $\boldsymbol{\mu}_k$, $k = 1, 2, \ldots, K$
  - dxd size covariance matrices, $\boldsymbol{\Sigma}_k$, $k = 1, 2, \ldots, K$
- Training process objective:
  - Partition the data into K groups
  - To estimate the parameters of the each cluster in GMM



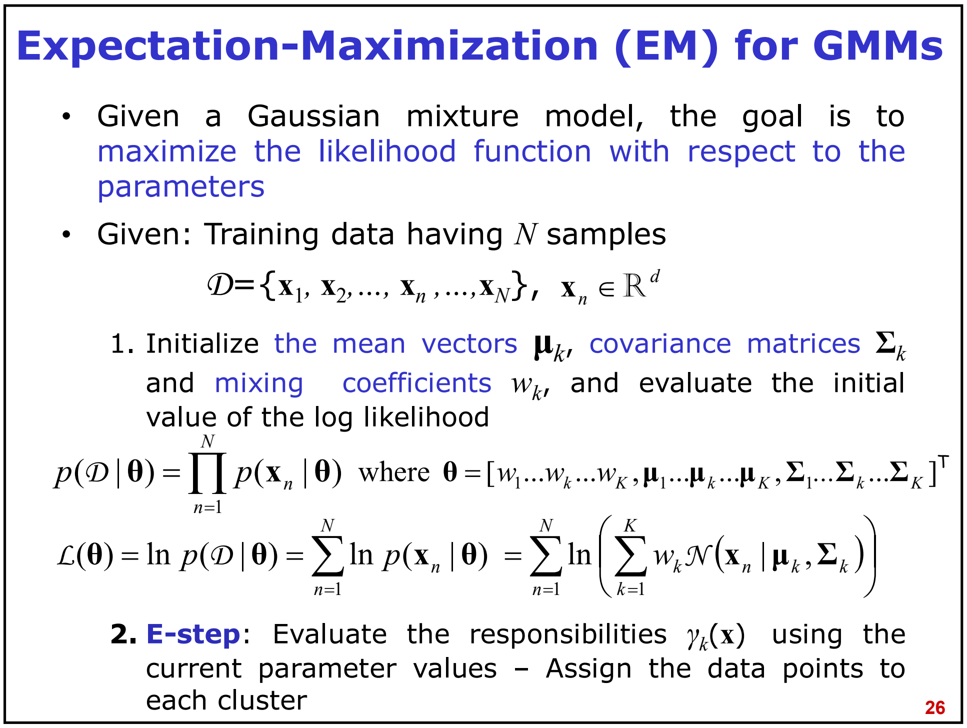
Expectation-Maximization (EM) for GMMs

- Given a Gaussian mixture model, the goal is to maximize the likelihood function with respect to the parameters
- Given: Training data having N samples

$$\mathcal{D}=\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, \ldots, \mathbf{x}_N\}, \quad \mathbf{x}_n \in \mathbb{R}^d$$

1. Initialize the mean vectors $\boldsymbol{\mu}_k$, covariance matrices $\boldsymbol{\Sigma}_k$ and mixing coefficients w_k, and evaluate the initial value of the log likelihood
   - Initialize the cluster center, $\boldsymbol{\mu}_k$, $k=1, 2, \ldots, K$ using randomly selected K data points in $\mathcal{D}$
   - Initialize the covariance matrix, $\boldsymbol{\Sigma}_k$, $k=1, 2, \ldots, K$ using unit matrix
   - Initialize the mixing coefficient $w_k = \dfrac{1}{K}$, $k=1, 2, \ldots, K$

25



Expectation-Maximization (EM) for GMMs

- Given a Gaussian mixture model, the goal is to maximize the likelihood function with respect to the parameters
- Given: Training data having N samples

$$\mathcal{D}=\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, \ldots, \mathbf{x}_N\}, \quad \mathbf{x}_n \in \mathbb{R}^d$$

1. Initialize the mean vectors $\boldsymbol{\mu}_k$, covariance matrices $\boldsymbol{\Sigma}_k$ and mixing coefficients w_k, and evaluate the initial value of the log likelihood

$$p(\mathcal{D}\,|\,\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\mathbf{x}_n\,|\,\boldsymbol{\theta}) \quad \text{where } \boldsymbol{\theta} = [w_1 \ldots w_k \ldots w_K, \boldsymbol{\mu}_1 \ldots \boldsymbol{\mu}_k \ldots \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1 \ldots \boldsymbol{\Sigma}_k \ldots \boldsymbol{\Sigma}_K]^\mathsf{T}$$

$$\mathcal{L}(\boldsymbol{\theta}) = \ln p(\mathcal{D}\,|\,\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\mathbf{x}_n\,|\,\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln\left(\sum_{k=1}^{K} w_k \mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right)$$

2. **E-step**: Evaluate the responsibilities $\gamma_k(\mathbf{x})$ using the current parameter values – Assign the data points to each cluster

26

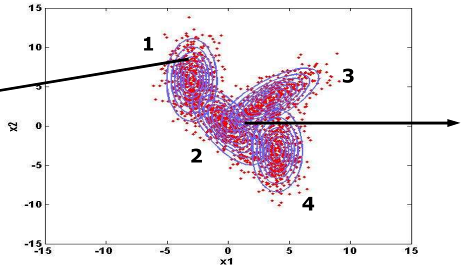
EM Method – Responsibility Term

- A quantity that plays an important role is the responsibility term, $\gamma_k(\mathbf{x})$
- It is given by

$$\gamma_k(\mathbf{x}) = \frac{w_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k=1}^{K} w_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

- w_k : mixture coefficient or prior probability of cluster k,
- $\gamma_k(\mathbf{x})$ gives the posterior probability of the cluster k for the observation $\mathbf{x}$

$\gamma_1(\mathbf{x}_n) = 0.99$
$\gamma_2(\mathbf{x}_n) = 0.01$
$\gamma_3(\mathbf{x}_n) = 0.00$
$\gamma_4(\mathbf{x}_n) = 0.00$



$\gamma_1(\mathbf{x}_m) = 0.08$
$\gamma_2(\mathbf{x}_m) = 0.42$
$\gamma_3(\mathbf{x}_m) = 0.34$
$\gamma_4(\mathbf{x}_m) = 0.16$

Expectation-Maximization (EM) for GMMs

- Given a Gaussian mixture model, the goal is to maximize the likelihood function with respect to the parameters

1. Initialize the mean vectors $\boldsymbol{\mu}_k$, covariance matrices $\boldsymbol{\Sigma}_k$ and mixing coefficients w_k, and evaluate the initial value of the log likelihood
2. **E-step**: Evaluate the responsibilities $\gamma_k(\mathbf{x})$ using the current parameter values
3. **M-step**: Re-estimate the parameters $\boldsymbol{\mu}_k^{new}$, $\boldsymbol{\Sigma}_k^{new}$ and w_k^{new} using the current responsibilities

$$\boldsymbol{\mu}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k(\mathbf{x}_n)\, \mathbf{x}_n$$

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k(\mathbf{x}_n)\, (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^{\mathsf{T}}$$

$$w_k^{new} = \frac{N_k}{N}$$

$$N_k = \sum_{n=1}^{N} \gamma_k(\mathbf{x}_n)$$

- N_k: Effective number of points assigned to the cluster k

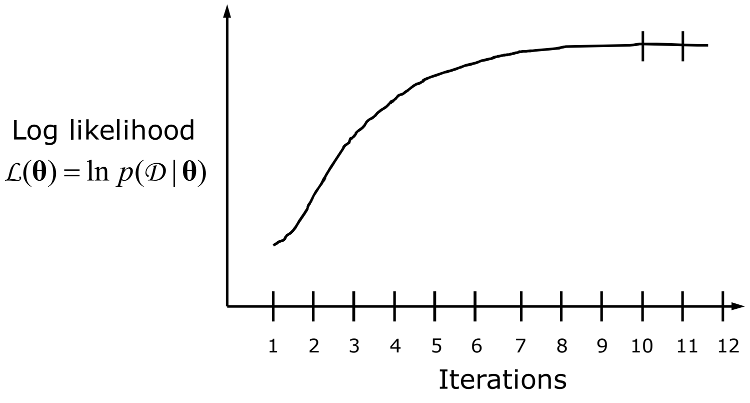
Expectation-Maximization (EM) for GMMs

- Given a Gaussian mixture model, the goal is to maximize the likelihood function with respect to the parameters

1. Initialize the mean vectors $\boldsymbol{\mu}_k$, covariance matrices $\boldsymbol{\Sigma}_k$ and mixing coefficients w_k, and evaluate the initial value of the log likelihood
2. **E-step**: Evaluate the responsibilities $\gamma_k(\mathbf{x})$ using the current parameter values
3. **M-step**: Re-estimate the parameters $\boldsymbol{\mu}_k^{new}$, $\boldsymbol{\Sigma}_k^{new}$ and w_k^{new} using the current responsibilities
4. Evaluate the log likelihood and check for convergence of the log likelihood
   - If the convergence criterion is not satisfied return to step 2

Expectation-Maximization (EM) for GMMs

- Convergence criterion: Difference between log likelihoods of successive iterations fall below a threshold (E.g. 10^{-3})



$$\mathcal{L}(\boldsymbol{\theta}) = \ln p(\mathcal{D} \mid \boldsymbol{\theta}) = \sum_{n=1}^{N} \ln\left(\sum_{k=1}^{K} w_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

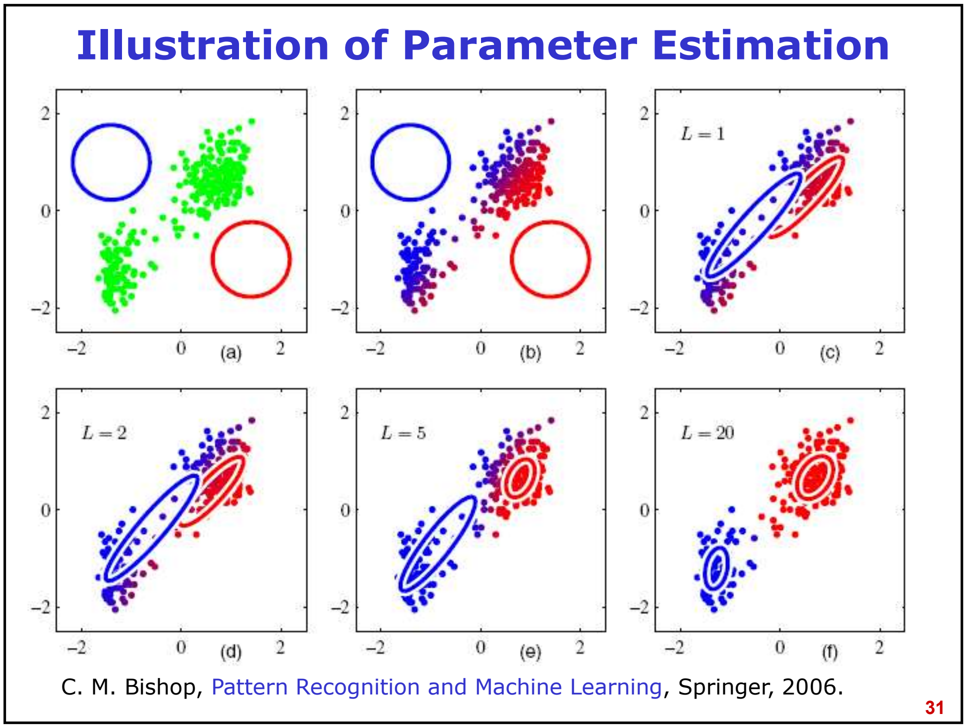
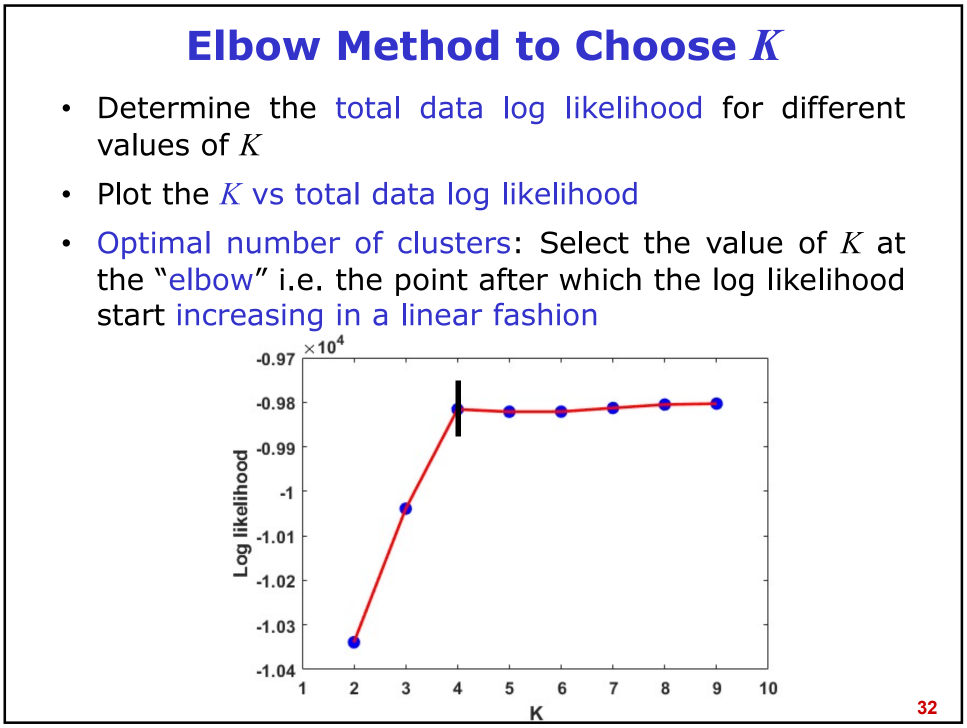


Illustration of Parameter Estimation

C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

31

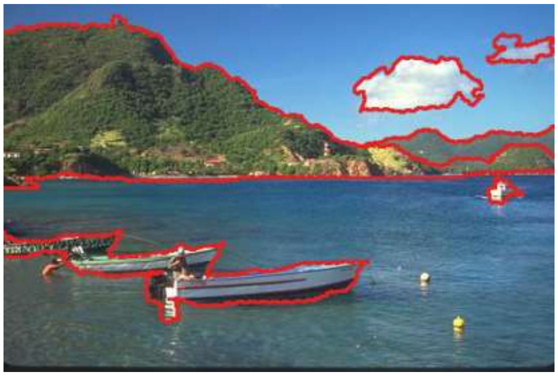


Elbow Method to Choose K

- Determine the total data log likelihood for different values of K
- Plot the K vs total data log likelihood
- Optimal number of clusters: Select the value of K at the "elbow" i.e. the point after which the log likelihood start increasing in a linear fashion

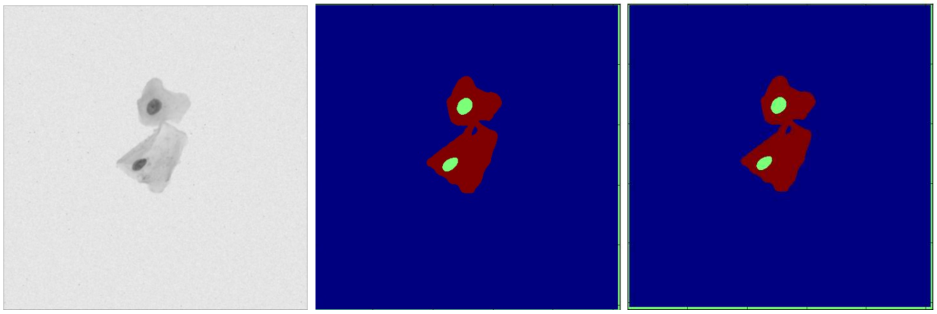
32

Application of Clustering: Segmentation using Clustering



Application of Clustering: Cell and Nucleus Segmentation

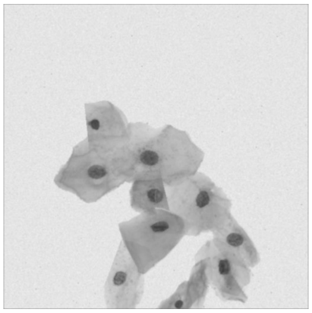
- An image is divided into patches of size 7 x 7
- From each patch, mean and variance of pixel values is considered as features
- 100 images with different numbers of cells are considered for training

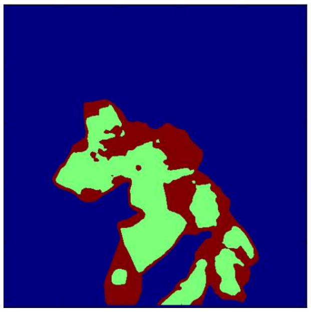


K-Means GMM

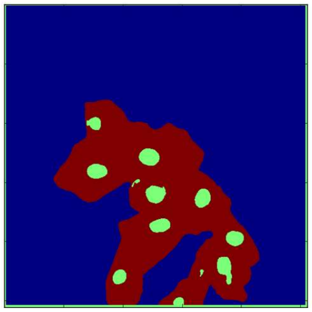
Application of Clustering: Cell and Nucleus Segmentation

- An image is divided into patches of size 7 x 7
- From each patch, mean and variance of pixel values is considered as features
- 100 images with different numbers of cells are considered for training



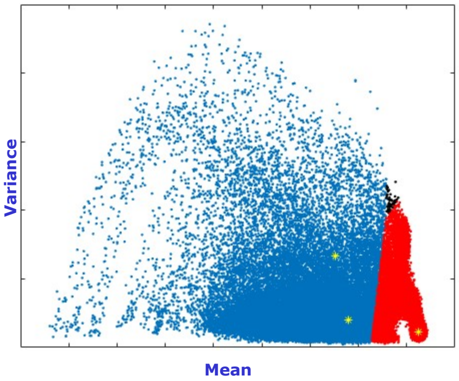


K-Means

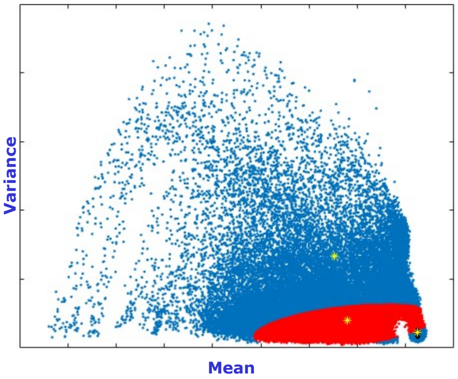


GMM

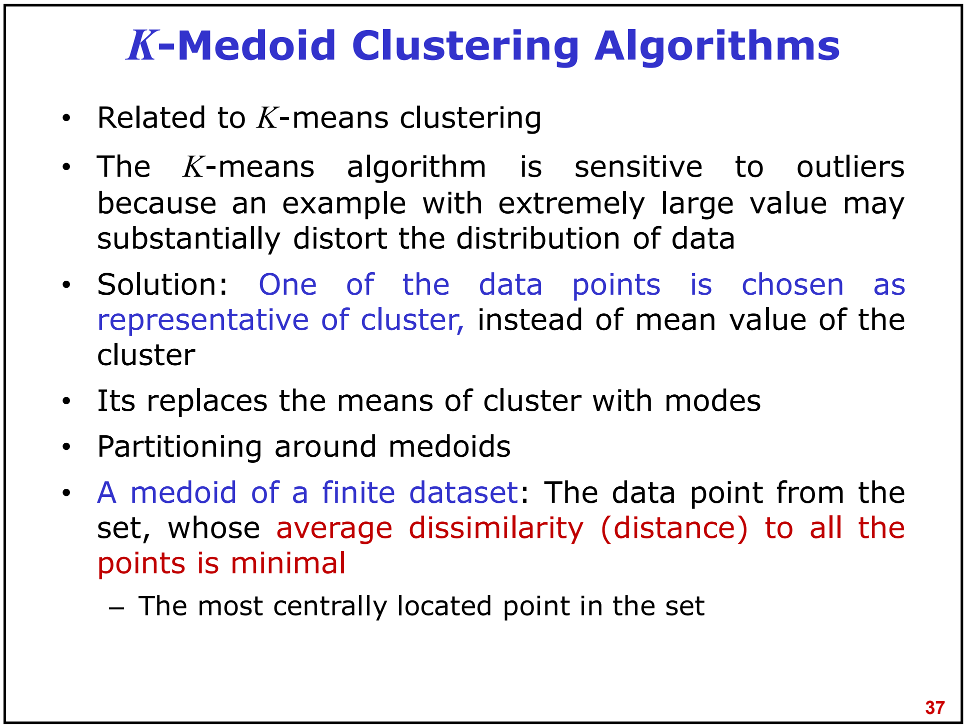
Hard Clustering vs Soft Clustering



K-Means

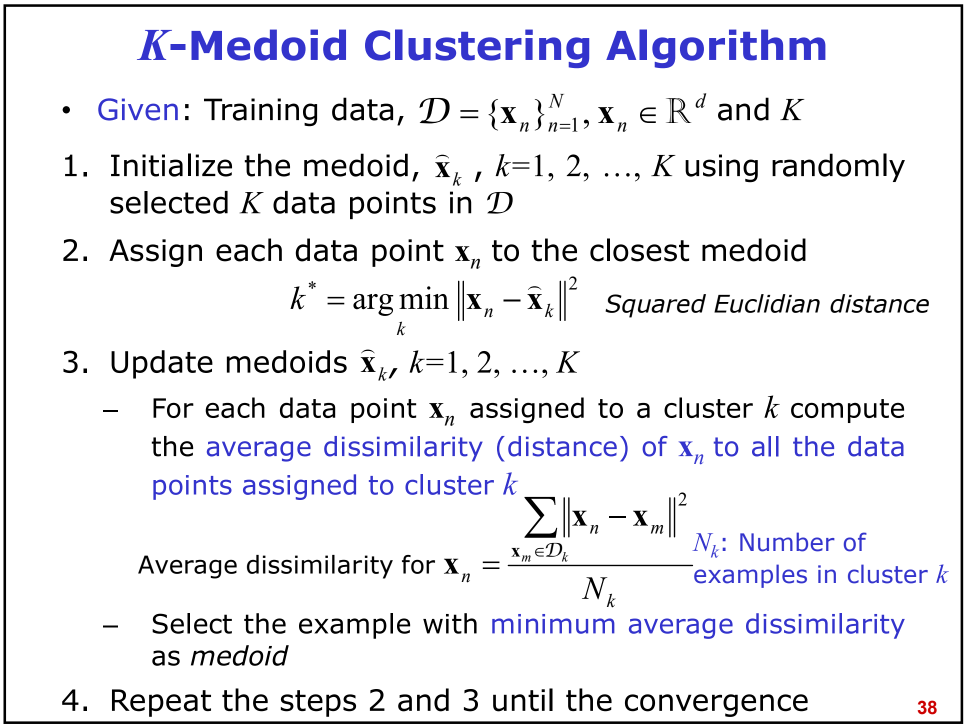


GMM



K-Medoid Clustering Algorithms

- Related to K-means clustering
- The K-means algorithm is sensitive to outliers because an example with extremely large value may substantially distort the distribution of data
- Solution: One of the data points is chosen as representative of cluster, instead of mean value of the cluster
- Its replaces the means of cluster with modes
- Partitioning around medoids
- A medoid of a finite dataset: The data point from the set, whose average dissimilarity (distance) to all the points is minimal
  - The most centrally located point in the set



K-Medoid Clustering Algorithm

- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$ and K

1. Initialize the medoid, $\hat{\mathbf{x}}_k$, k=1, 2, …, K using randomly selected K data points in $\mathcal{D}$
2. Assign each data point $\mathbf{x}_n$ to the closest medoid

$$k^* = \arg\min_k \|\mathbf{x}_n - \hat{\mathbf{x}}_k\|^2$$ *Squared Euclidian distance*

3. Update medoids $\hat{\mathbf{x}}_k$, k=1, 2, …, K
   - For each data point $\mathbf{x}_n$ assigned to a cluster k compute the average dissimilarity (distance) of $\mathbf{x}_n$ to all the data points assigned to cluster k

     Average dissimilarity for $\mathbf{x}_n = \dfrac{\sum_{\mathbf{x}_m \in \mathcal{D}_k} \|\mathbf{x}_n - \mathbf{x}_m\|^2}{N_k}$ N_k: Number of examples in cluster k
   - Select the example with minimum average dissimilarity as *medoid*
4. Repeat the steps 2 and 3 until the convergence

K-Medoid Clustering Algorithm

- Convergence criteria:
  - No change in the cluster assignment **OR**
  - The difference between the distortion measure (absolute-error) (J) in the successive iteration falls below the threshold
    - Distortion measure (J) : Sum of the squares of the distance of each example to its corresponding reference point (medoid)

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \left\| \mathbf{x}_n - \hat{\mathbf{x}}_k \right\|^2$$

z_{nk} is 1 if $\mathbf{x}_n$ belongs to cluster k, otherwise 0

- Optimal number of clusters (k) can obtained using elbow method

39

Evaluation of Clustering: Purity Score

- Lets us assume that class index for each example is given
- Purity score: Purity is a measure of the extent to which clusters contain a single class
  - For each cluster, count the number of data points from the most common class
  - Take the sum over all clusters and divide by the total number of data points
- Let M be the number of classes, $C_1, C_2, \ldots, C_m, \ldots, C_M$
- Let K be the number of clusters, $k = 1, 2, \ldots, K$
- Let N be the number of data points

40

Evaluation of Clustering: Purity Score

- For each cluster k,
  - Count the number of data points from each class
  - Consider the number of data points of most common class

$$\max_{m} \left| N_k \cap C_m \right|$$

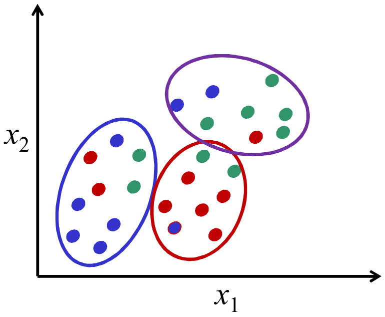
$\left| N_k \cap C_m \right|$ is the number of data points in k^{th} cluster belonging to class m

- Take the sum over all clusters, k
- Divide by the total number of data points (N)

$$\text{Purity Score} = \frac{1}{N} \sum_{k=1}^{K} \max_{m} \left| N_k \cap C_m \right|$$

Illustration of Computing Purity Score

- Number of data points, N = 25
- number of classes, M = 3
- number of clusters, K = 3



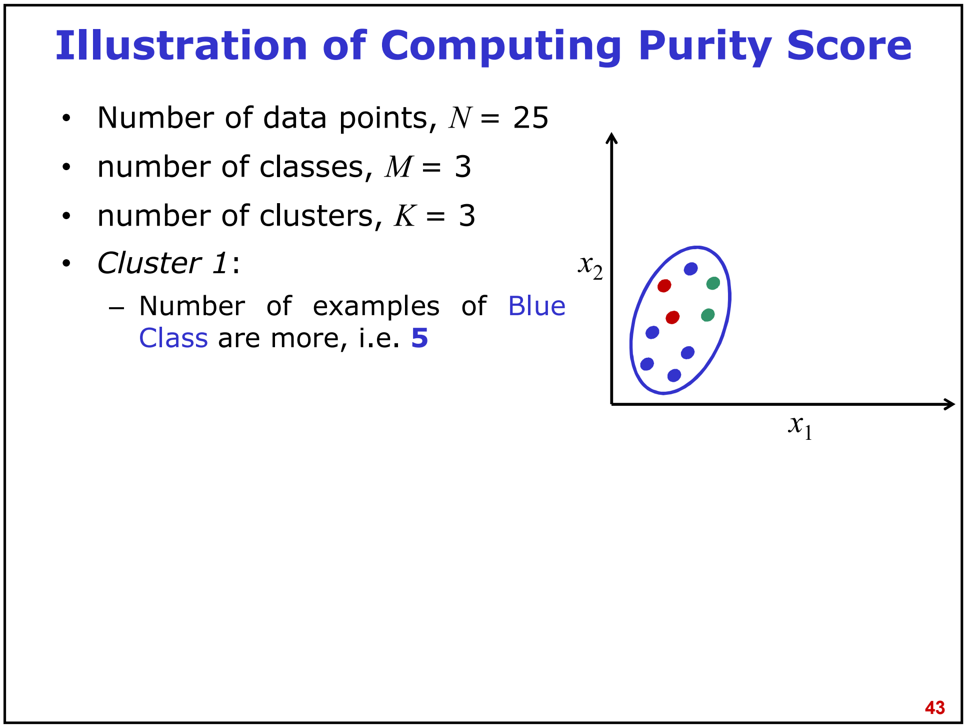


Illustration of Computing Purity Score

- Number of data points, $N = 25$
- number of classes, $M = 3$
- number of clusters, $K = 3$
- *Cluster 1*:
  - Number of examples of Blue Class are more, i.e. **5**

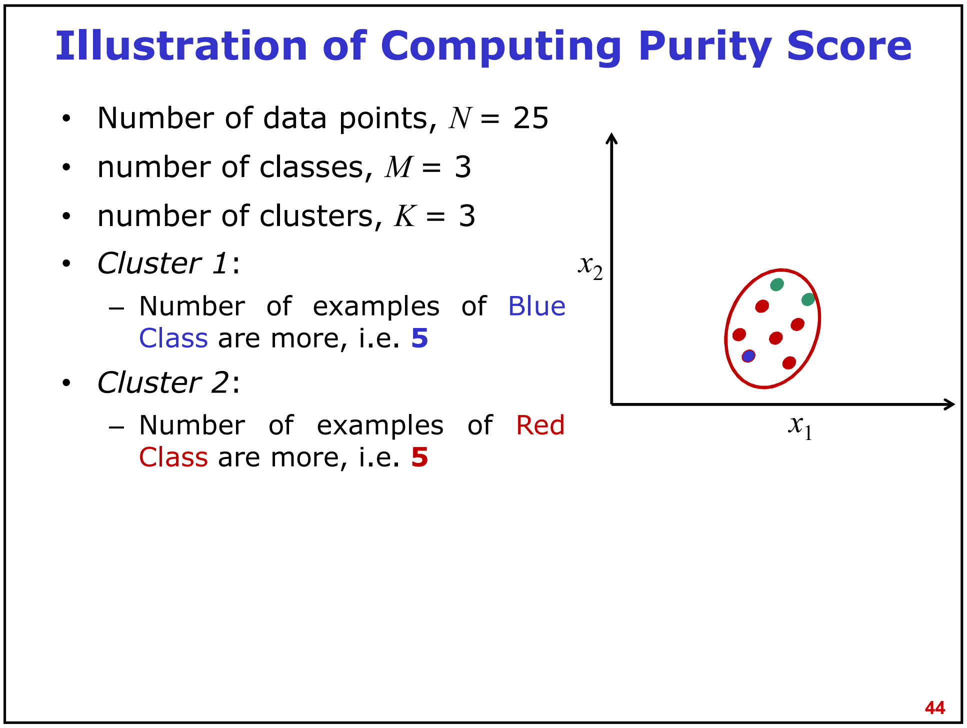


Illustration of Computing Purity Score

- Number of data points, $N = 25$
- number of classes, $M = 3$
- number of clusters, $K = 3$
- *Cluster 1*:
  - Number of examples of Blue Class are more, i.e. **5**
- *Cluster 2*:
  - Number of examples of Red Class are more, i.e. **5**

Illustration of Computing Purity Score

- Number of data points, $N = 25$
- number of classes, $M = 3$
- number of clusters, $K = 3$
- *Cluster 1*:
  - Number of examples of Blue Class are more, i.e. **5**
- *Cluster 2*:
  - Number of examples of Red Class are more, i.e. **5**
- *Cluster 3*:
  - Number of examples of Green Class are more, i.e. **5**

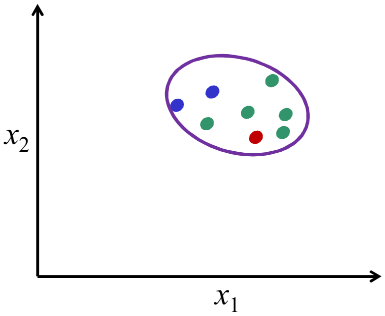
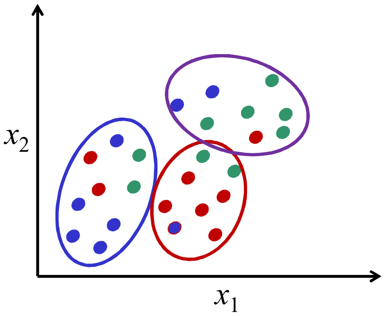


Illustration of Computing Purity Score

- Number of data points, $N = 25$
- number of classes, $M = 3$
- number of clusters, $K = 3$
- *Cluster 1*:
  - Number of examples of Blue Class are more, i.e. **5**
- *Cluster 2*:
  - Number of examples of Red Class are more, i.e. **5**
- *Cluster 3*:
  - Number of examples of Green Class are more, i.e. **5**

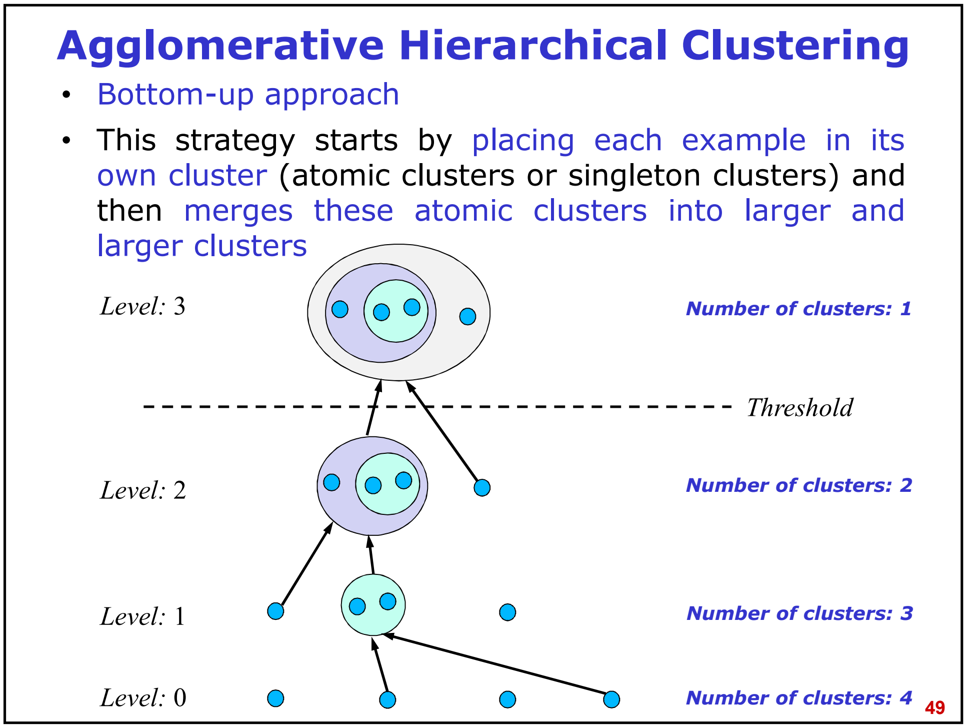


- **Purity score: (5+5+5)/25 = 0.60**

Hierarchical Clustering

Hierarchical Clustering Algorithms

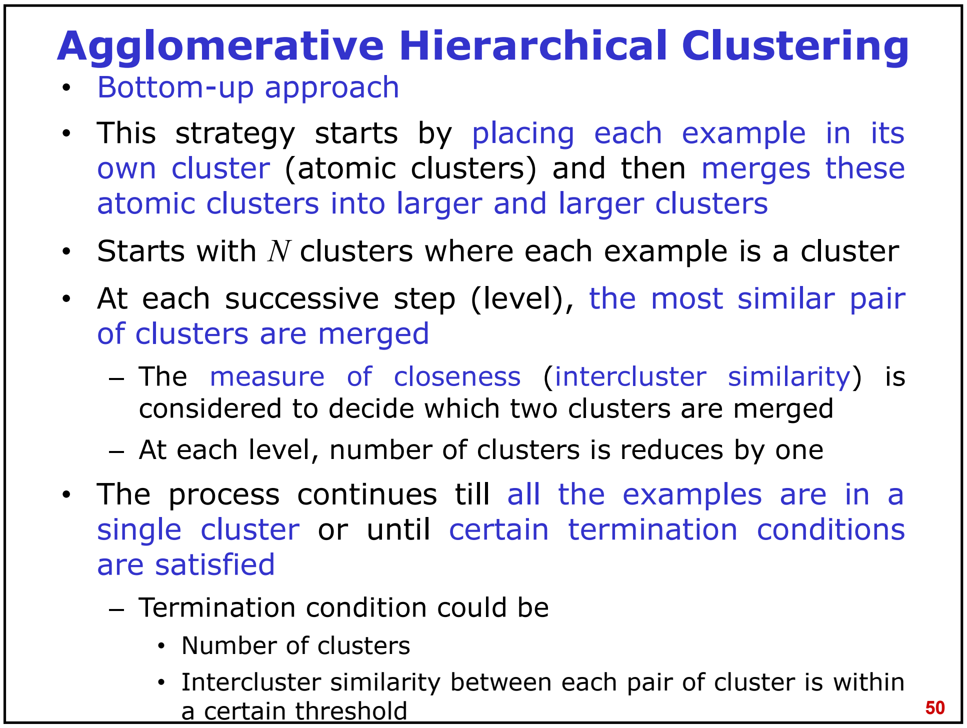
- These methods create a hierarchical decomposition of the collection of examples
- Produce nested sequence of data partitions
- These sequence can be depicted using a tree structure
- Hierarchical clustering method works by grouping data points into a tree of clusters
- Hierarchical algorithms are either agglomerative or divisive
  - This classification of hierarchical clustering is depending on whether the hierarchical decomposition is formed in a
    - Bottom-up (merging) OR
    - Top-down (splitting) fashion
- Need not have to specify the number of clusters



Agglomerative Hierarchical Clustering

- Bottom-up approach
- This strategy starts by placing each example in its own cluster (atomic clusters or singleton clusters) and then merges these atomic clusters into larger and larger clusters

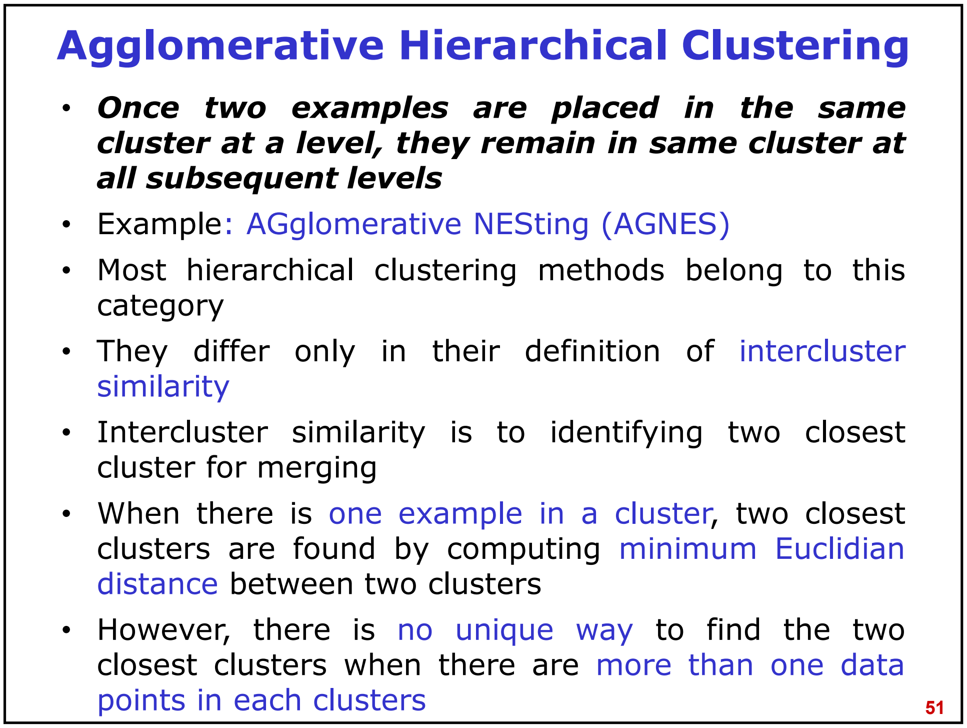
49



Agglomerative Hierarchical Clustering

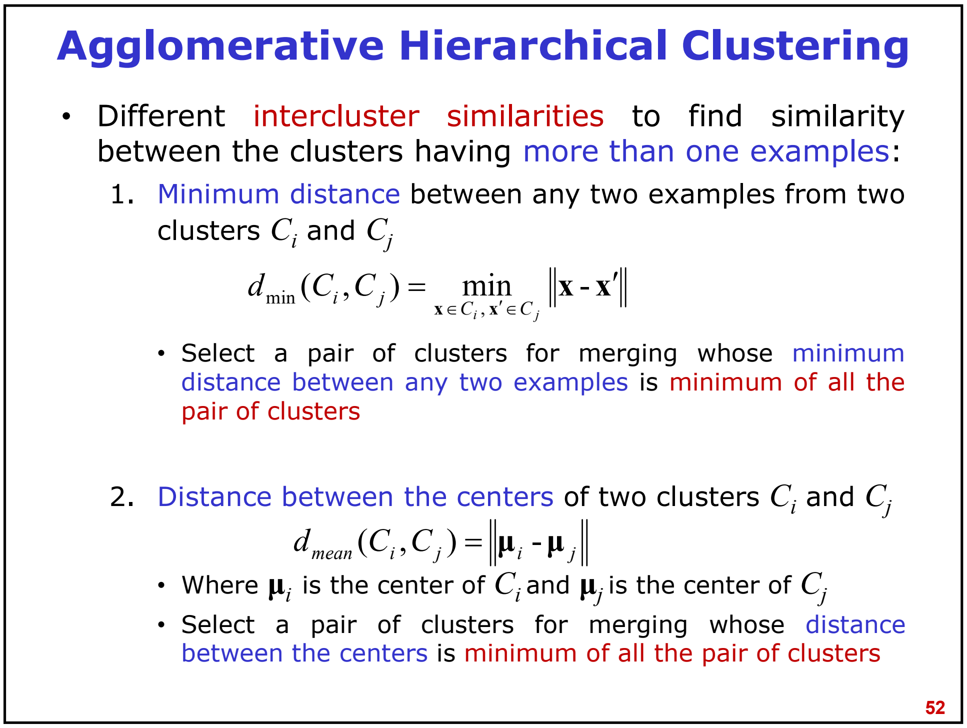
- Bottom-up approach
- This strategy starts by placing each example in its own cluster (atomic clusters) and then merges these atomic clusters into larger and larger clusters
- Starts with N clusters where each example is a cluster
- At each successive step (level), the most similar pair of clusters are merged
  - The measure of closeness (intercluster similarity) is considered to decide which two clusters are merged
  - At each level, number of clusters is reduces by one
- The process continues till all the examples are in a single cluster or until certain termination conditions are satisfied
  - Termination condition could be
    - Number of clusters
    - Intercluster similarity between each pair of cluster is within a certain threshold

50



Agglomerative Hierarchical Clustering

- ***Once two examples are placed in the same cluster at a level, they remain in same cluster at all subsequent levels***
- Example: AGglomerative NESting (AGNES)
- Most hierarchical clustering methods belong to this category
- They differ only in their definition of intercluster similarity
- Intercluster similarity is to identifying two closest cluster for merging
- When there is one example in a cluster, two closest clusters are found by computing minimum Euclidian distance between two clusters
- However, there is no unique way to find the two closest clusters when there are more than one data points in each clusters



Agglomerative Hierarchical Clustering

- Different intercluster similarities to find similarity between the clusters having more than one examples:
  1. Minimum distance between any two examples from two clusters C_i and C_j

     $$d_{\min}(C_i, C_j) = \min_{\mathbf{x} \in C_i, \mathbf{x}' \in C_j} \|\mathbf{x} - \mathbf{x}'\|$$

     - Select a pair of clusters for merging whose minimum distance between any two examples is minimum of all the pair of clusters

  2. Distance between the centers of two clusters C_i and C_j

     $$d_{mean}(C_i, C_j) = \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|$$

     - Where $\boldsymbol{\mu}_i$ is the center of C_i and $\boldsymbol{\mu}_j$ is the center of C_j
     - Select a pair of clusters for merging whose distance between the centers is minimum of all the pair of clusters

Agglomerative Hierarchical Clustering

- Different intercluster similarities to find similarity between the clusters having more than one examples:
  3. Average distance of all the points in one cluster (C_i) to all the points in another cluster (C_j)

     $$d_{avg}(C_i, C_j) = \frac{1}{N_i N_j} \sum_{\mathbf{x} \in C_i} \sum_{\mathbf{x}' \in C_j} \left\| \mathbf{x} - \mathbf{x}' \right\|$$

     - Where N_i and N_j are the number of examples in clusters C_i and C_j respectively
     - Select a pair of clusters for merging whose average distance is minimum than that of all the pair of clusters

Agglomerative Hierarchical Clustering

- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$
- Target: Partition the data
- Step1: N clusters where each example is a cluster
- Step2: Compute intercluster similarity between each pair of clusters
- Step3: Choose a pair of clusters that are most similar (minimum intercluster distance) and merge them
- Step4: Repeat Step2 and Step3 until all the examples are in a single cluster or until certain termination conditions are satisfied

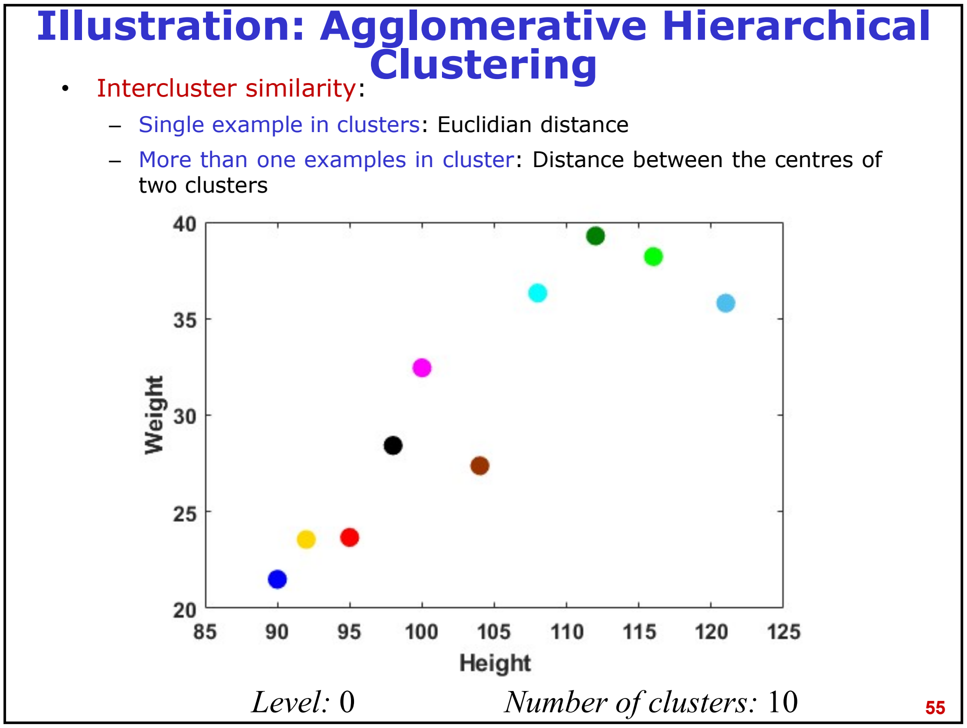


Illustration: Agglomerative Hierarchical Clustering

- Intercluster similarity:
  - Single example in clusters: Euclidian distance
  - More than one examples in cluster: Distance between the centres of two clusters

Level: 0 *Number of clusters:* 10

55

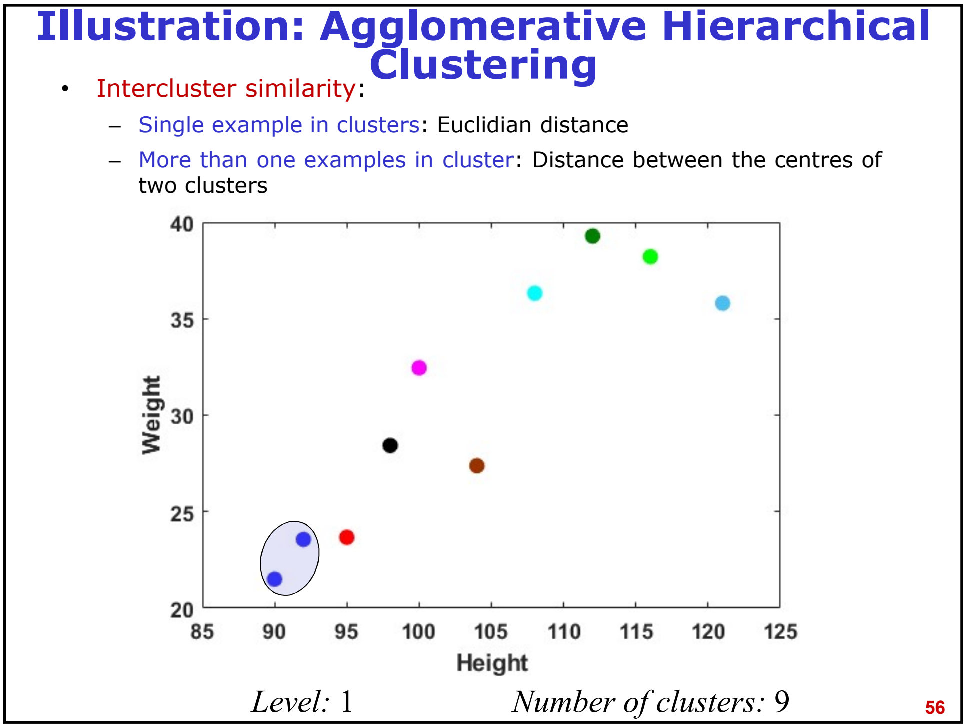


Illustration: Agglomerative Hierarchical Clustering

- Intercluster similarity:
  - Single example in clusters: Euclidian distance
  - More than one examples in cluster: Distance between the centres of two clusters

Level: 1 *Number of clusters:* 9

56

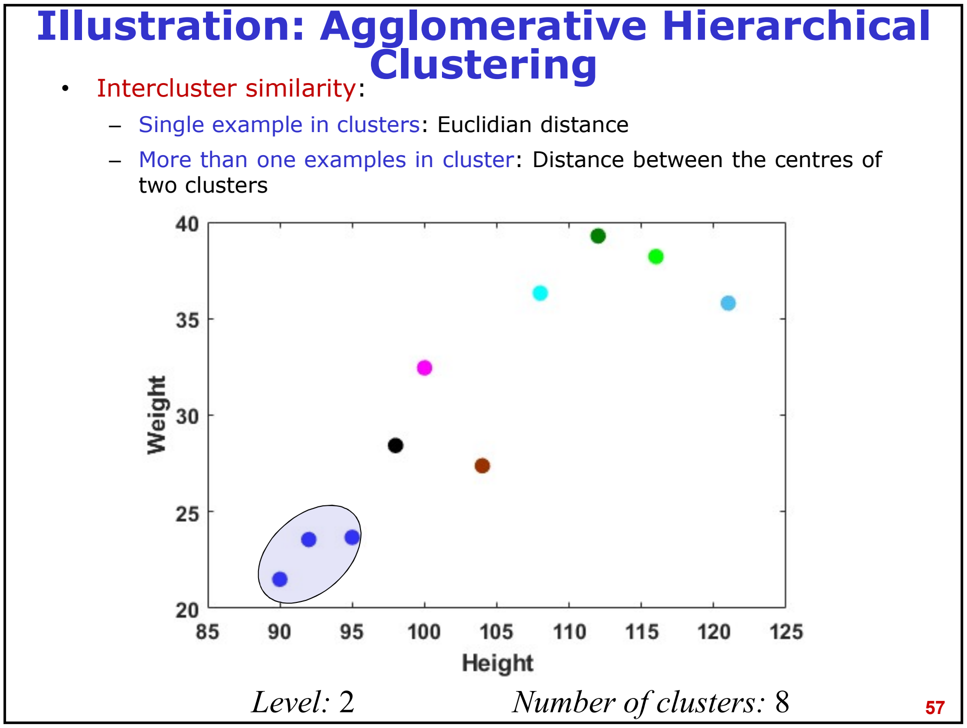


Illustration: Agglomerative Hierarchical Clustering

- Intercluster similarity:
  - Single example in clusters: Euclidian distance
  - More than one examples in cluster: Distance between the centres of two clusters

Level: 2 *Number of clusters:* 8

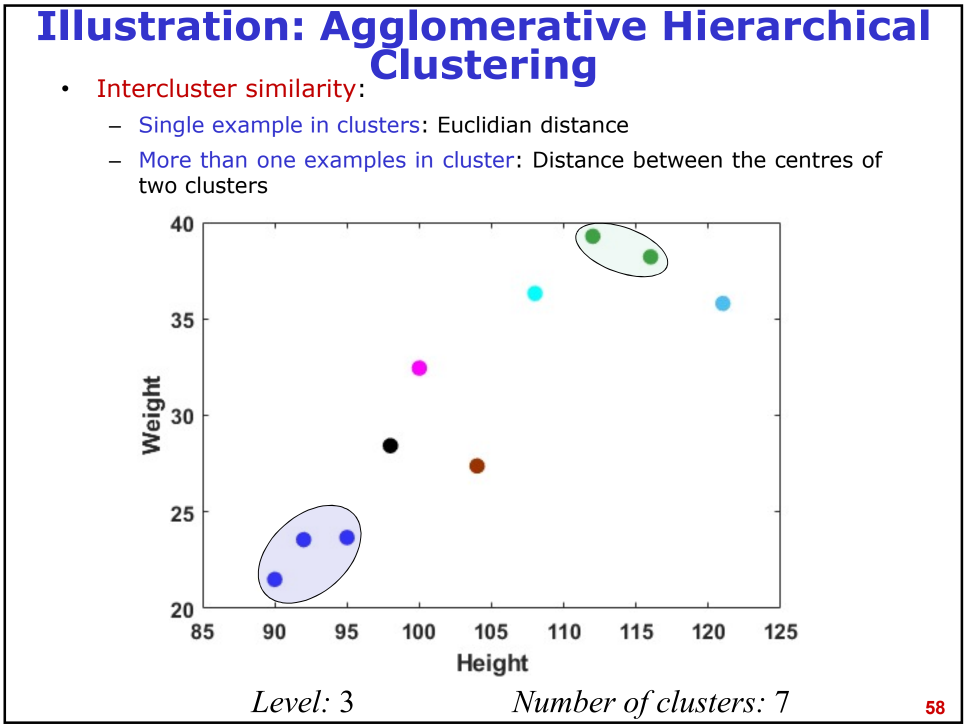


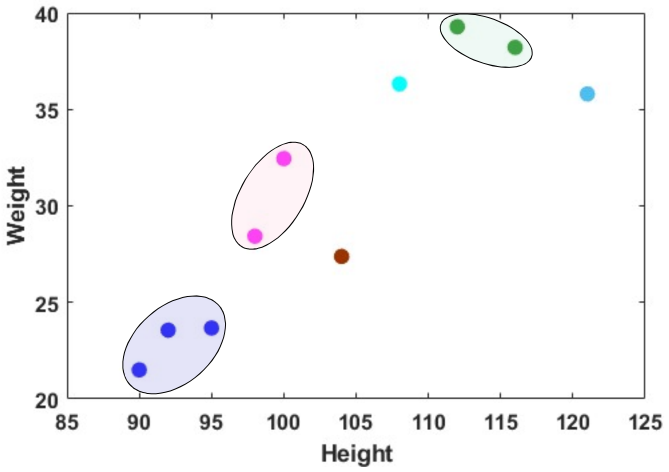
Illustration: Agglomerative Hierarchical Clustering

- Intercluster similarity:
  - Single example in clusters: Euclidian distance
  - More than one examples in cluster: Distance between the centres of two clusters

Level: 3 *Number of clusters:* 7

Illustration: Agglomerative Hierarchical Clustering

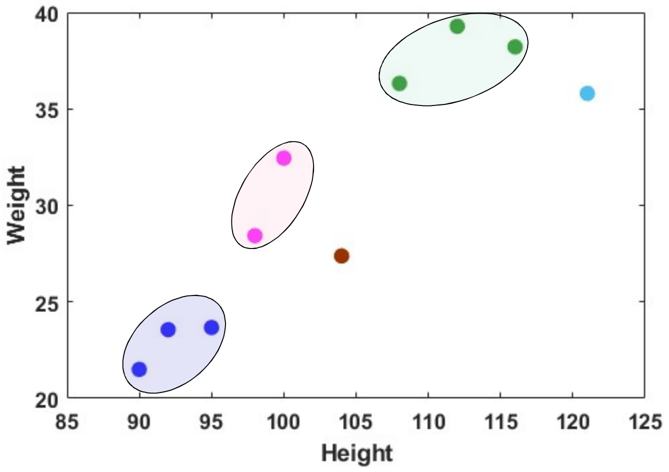
- Intercluster similarity:
  - Single example in clusters: Euclidian distance
  - More than one examples in cluster: Distance between the centres of two clusters



Level: 4 *Number of clusters:* 6

Illustration: Agglomerative Hierarchical Clustering

- Intercluster similarity:
  - Single example in clusters: Euclidian distance
  - More than one examples in cluster: Distance between the centres of two clusters



Level: 5 *Number of clusters:* 5

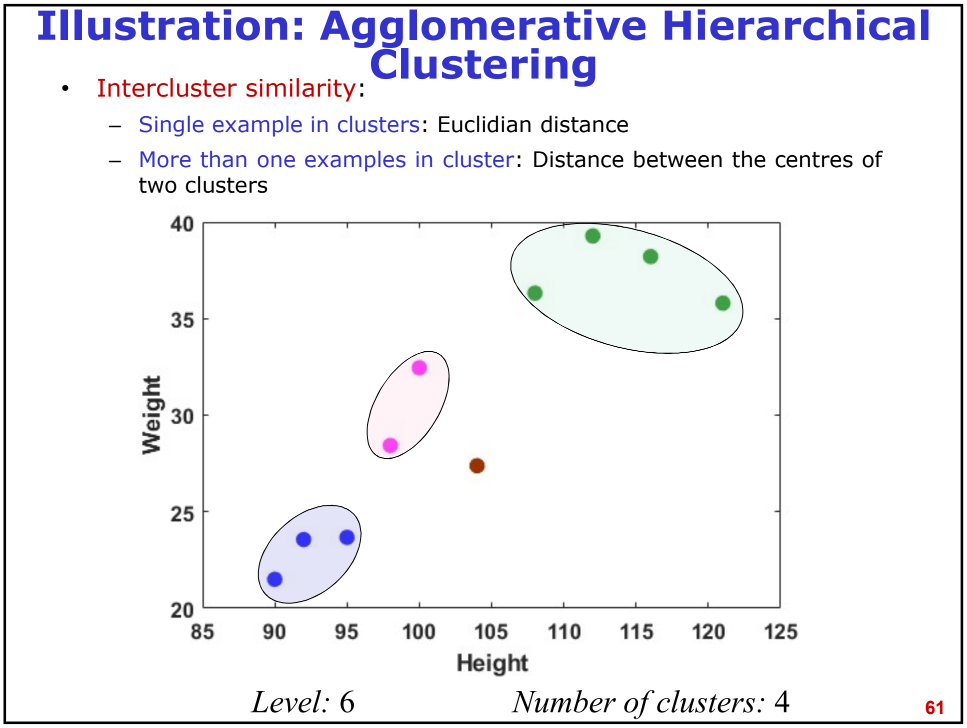


Illustration: Agglomerative Hierarchical Clustering

- Intercluster similarity:
  - Single example in clusters: Euclidian distance
  - More than one examples in cluster: Distance between the centres of two clusters

Level: 6 *Number of clusters:* 4

61

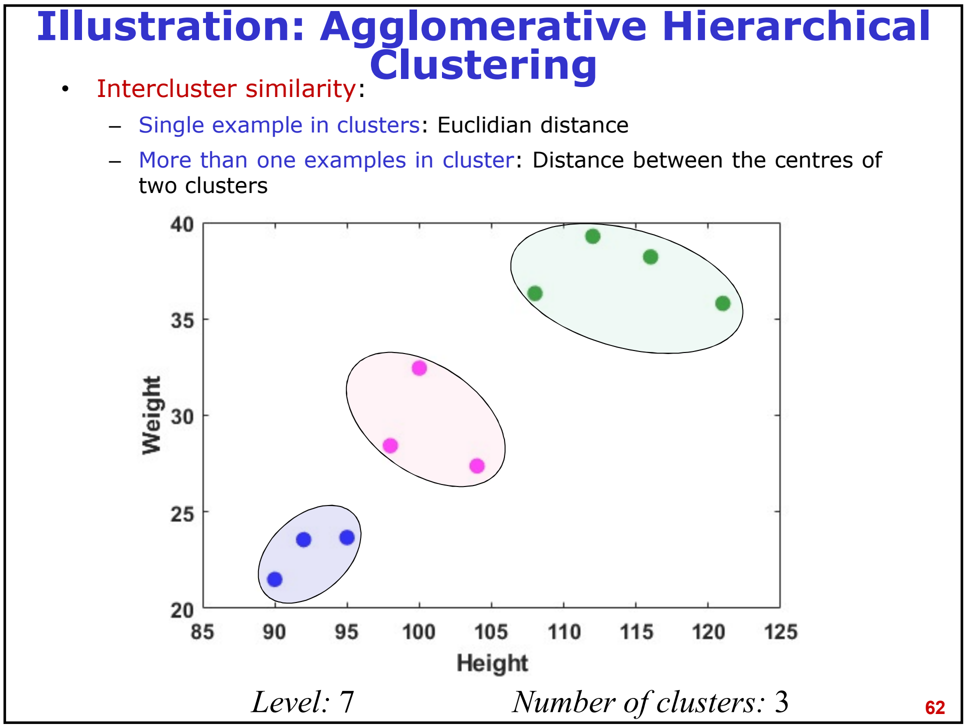


Illustration: Agglomerative Hierarchical Clustering

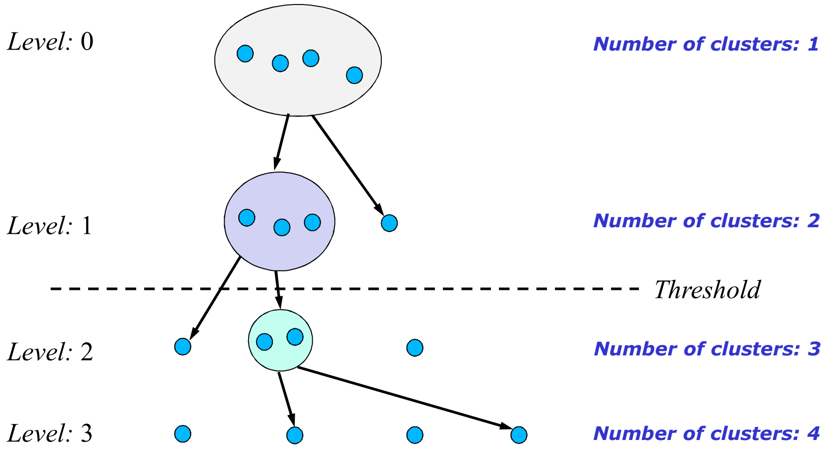
- Intercluster similarity:
  - Single example in clusters: Euclidian distance
  - More than one examples in cluster: Distance between the centres of two clusters

Level: 7 *Number of clusters:* 3

62

Divisive Hierarchical Clustering

- Top-down approach
- Starts with single cluster having all the examples
- It subdivides the cluster into smaller and smaller clusters in the successive step



Level: 0 — ***Number of clusters: 1***

Level: 1 — ***Number of clusters: 2***

Threshold

Level: 2 — ***Number of clusters: 3***

Level: 3 — ***Number of clusters: 4***

Divisive Hierarchical Clustering

- Top-down approach
- Starts with single cluster having all the examples
- It subdivides the cluster into smaller and smaller clusters in the successive step
- At each successive step, a compactness measure is used to choose which cluster to split
  - Compactness measure: Average value of distance between the data points of a cluster
  - Compactness measure (CM_i) of a cluster C_i:

$$CM_i = \frac{1}{N_i^2}\sum_{\mathbf{x}\in C_i}\sum_{\mathbf{x}'\in C_i}\|\mathbf{x} - \mathbf{x}'\|$$

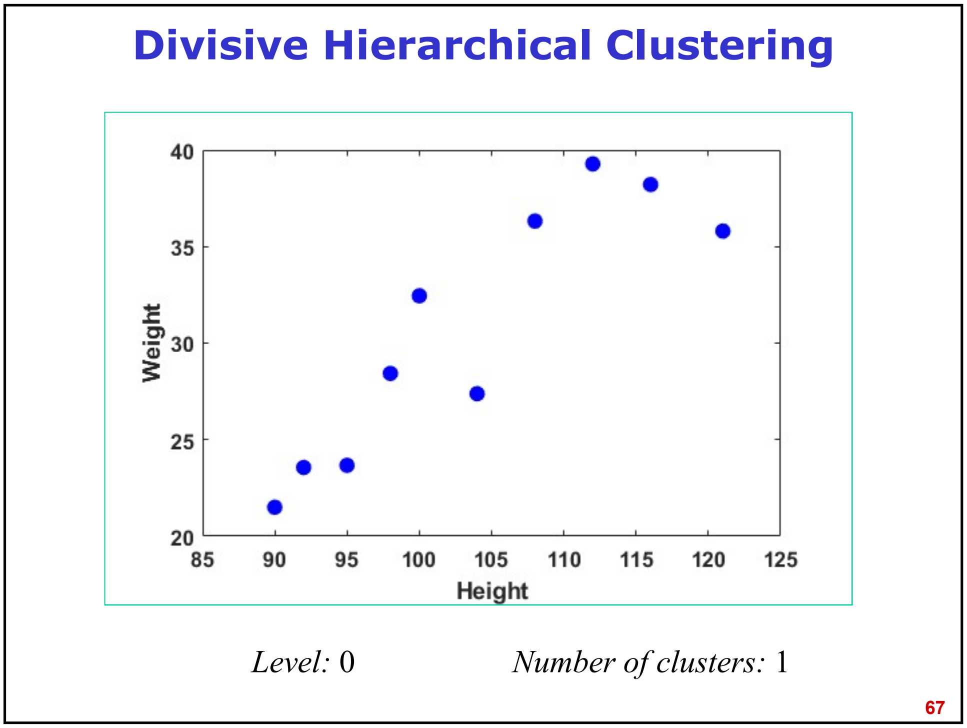
    - Where N_i is the number of examples in cluster C_i
  - Choose the cluster with larger value of compactness measure to split

Divisive Hierarchical Clustering

- To split a cluster, find a pair of examples having maximum Euclidian distance and split around these two examples (keeping them as centroids)
- At each level, number of clusters is increases by one
- The process continues until each example forms a cluster (atomic or singleton cluster) or until it satisfies certain termination condition
  - Termination condition could be
    - Number of clusters
    - Compactness measure of each cluster is within a certain threshold
- ***Once two examples are placed in two different clusters at a level, they remain in different clusters at all subsequent levels***
- Example: DIvisive ANAlysis (DIANA)

Divisive Hierarchical Clustering

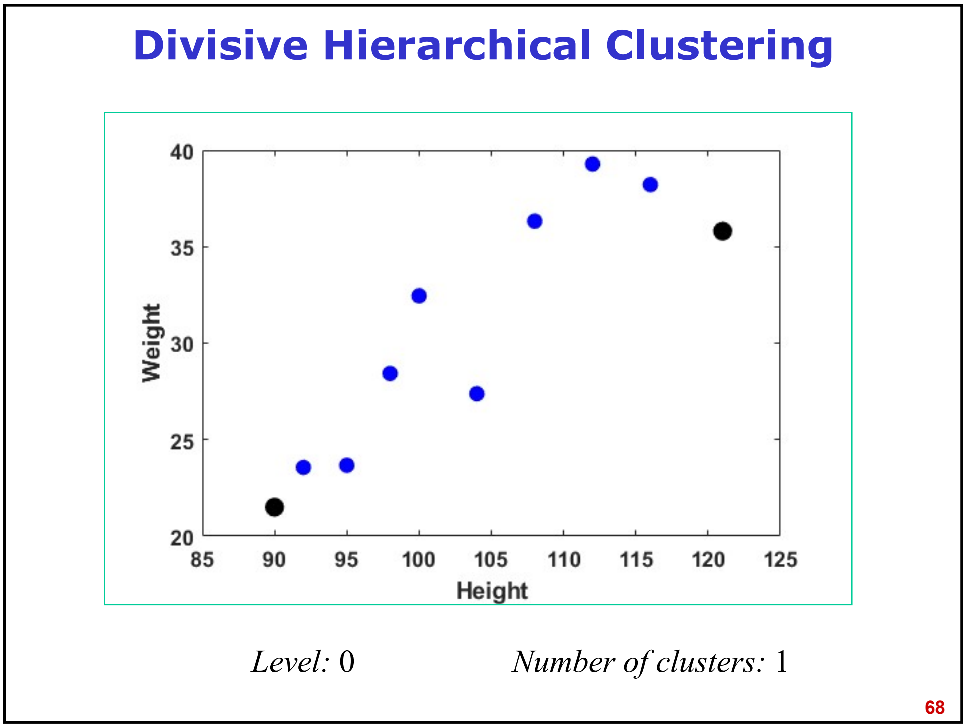
- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$
- Target: Partition the data
- Step1: Single cluster having all the examples
- Step2: Find a pair of examples with in a cluster having maximum Euclidian distance
  - These examples act as centroid
- Step3: Split into two clusters by assigning each data point to one of these two examples using Euclidian distance
- Step4: Compute compactness measure for each cluster
- Step5: Choose the cluster with larger value of compactness measure to split
- Step6: Repeat Step2 to Step5 until each example forms a cluster (atomic or singleton cluster) or until it satisfies certain termination condition



Divisive Hierarchical Clustering

Level: 0 *Number of clusters:* 1

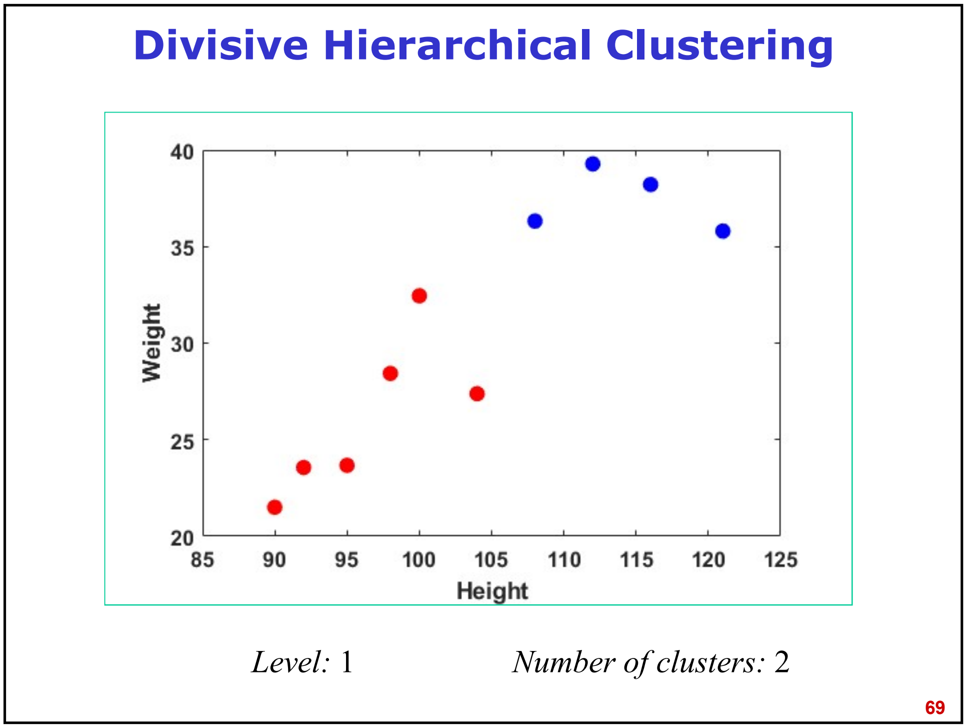
67



Divisive Hierarchical Clustering

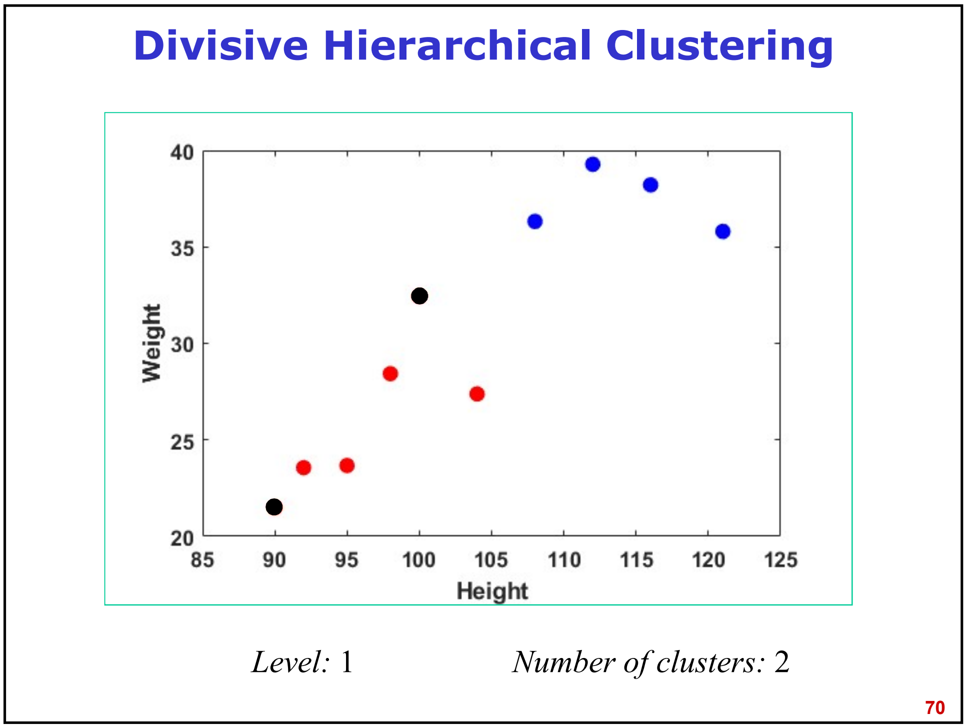
Level: 0 *Number of clusters:* 1

68



Divisive Hierarchical Clustering

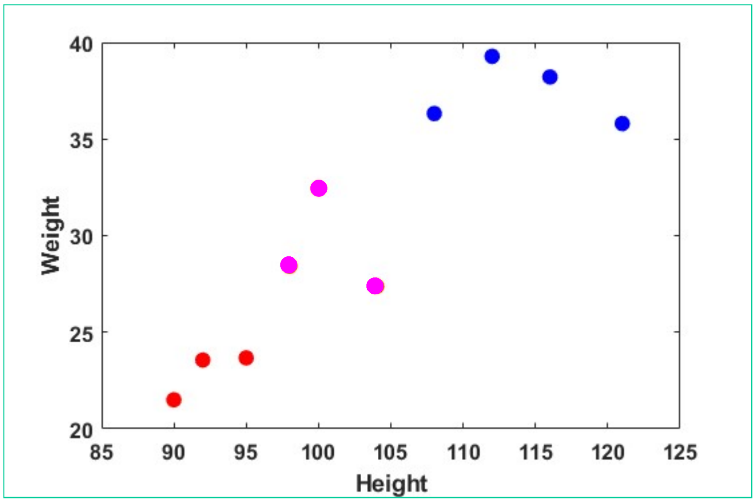
Level: 1 *Number of clusters:* 2



Divisive Hierarchical Clustering

Level: 1 *Number of clusters:* 2

Divisive Hierarchical Clustering

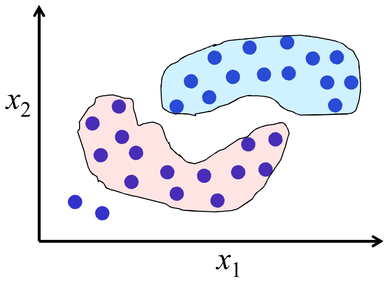


Level: 2 *Number of clusters:* 3

Density-Based Clustering

Density-Based Clustering

- These methods cluster collection of examples based on the notion of density
- These methods regard clusters as dense regions of examples in the data space that are separated by regions of low density (i.e. noise)
- They discover clusters with arbitrary shape

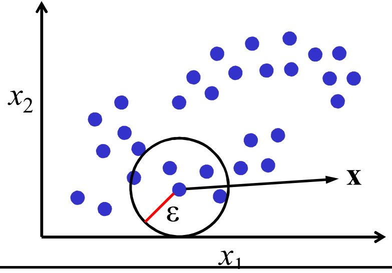


Density-Based Clustering

- These methods cluster collection of examples based on the notion of density
- These methods regard clusters as dense regions of examples in the data space that are separated by regions of low density (i.e. noise)
- They discover clusters with arbitrary shape
- They automatically identifies the number of clusters
- General idea: To continue growing the given cluster as long as density (number of examples) in the neighbourhood exceeds some threshold
- Example: Density-based Spatial Clustering of Applications with Noise (DBSCAN)
  - It grows the clusters according to a density-based connectivity analysis

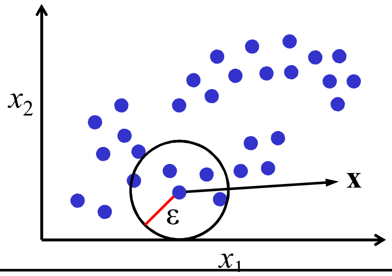
Density-based Spatial Clustering of Applications with Noise (DBSCAN)

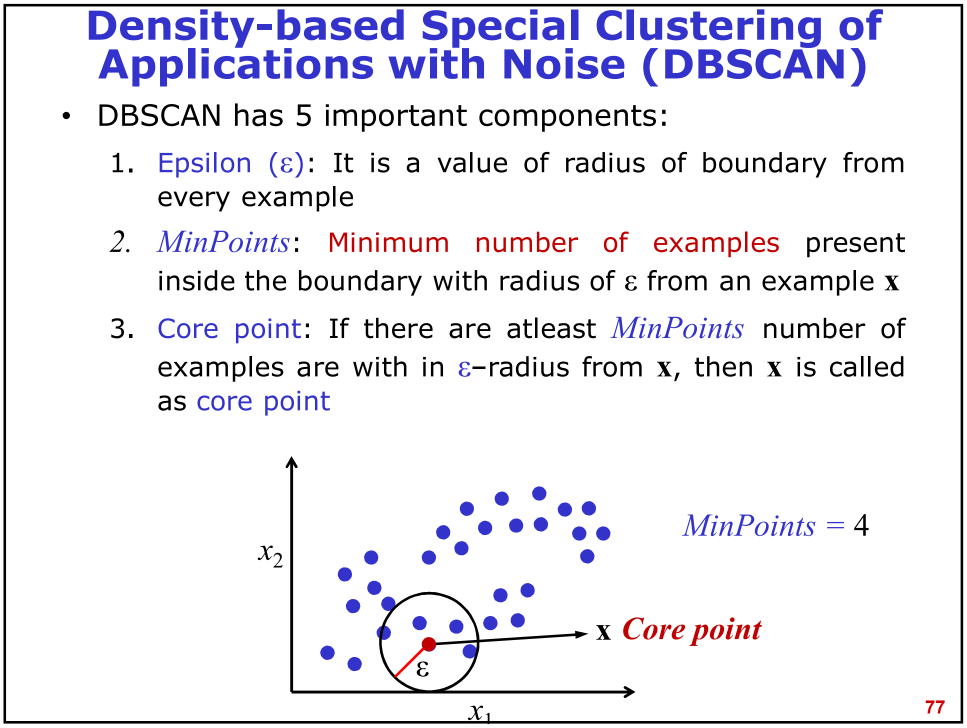
- DBSCAN is a density-based clustering included with noise
- It grows the regions with sufficiently high density (neighbors) into clusters with arbitrary shape
- It defines a cluster as a maximal set of density-connected points
- DBSCAN has 5 important components:
  1. Epsilon (ε): It is a value of radius of boundary from every example



Density-based Special Clustering of Applications with Noise (DBSCAN)

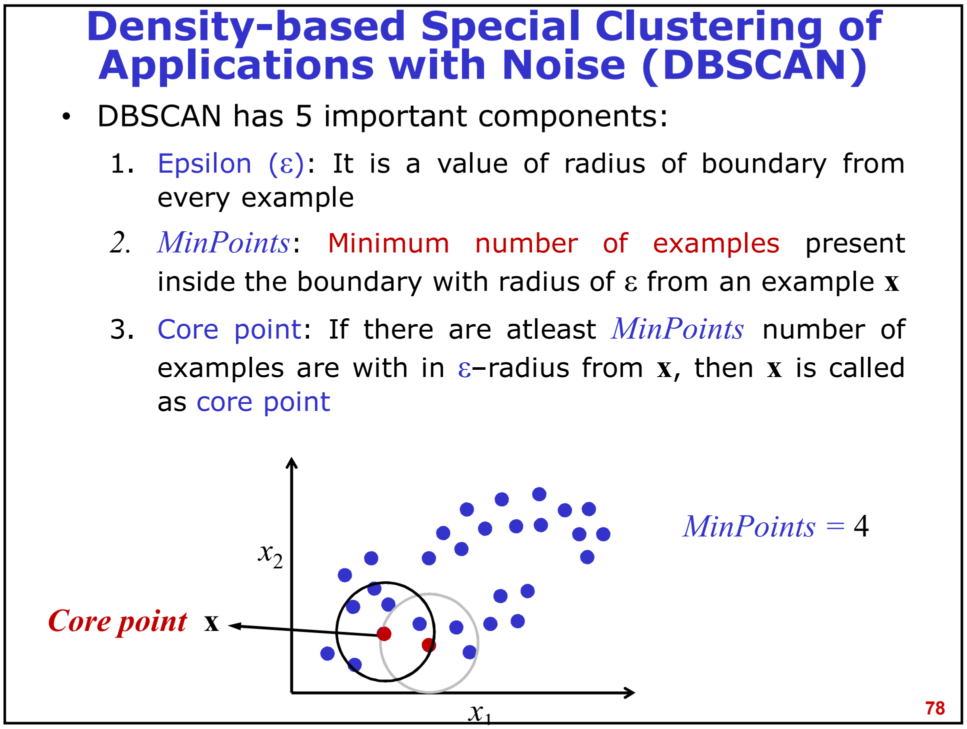
- DBSCAN has 5 important components:
  1. Epsilon (ε): It is a value of radius of boundary from every example
  2. *MinPoints*: Minimum number of examples present inside the boundary with radius of ε from an example $\mathbf{x}$
     - These examples with in a boundary are neighbors to $\mathbf{x}$ and called as ε-neighborhood of an example, $\mathbf{x}$





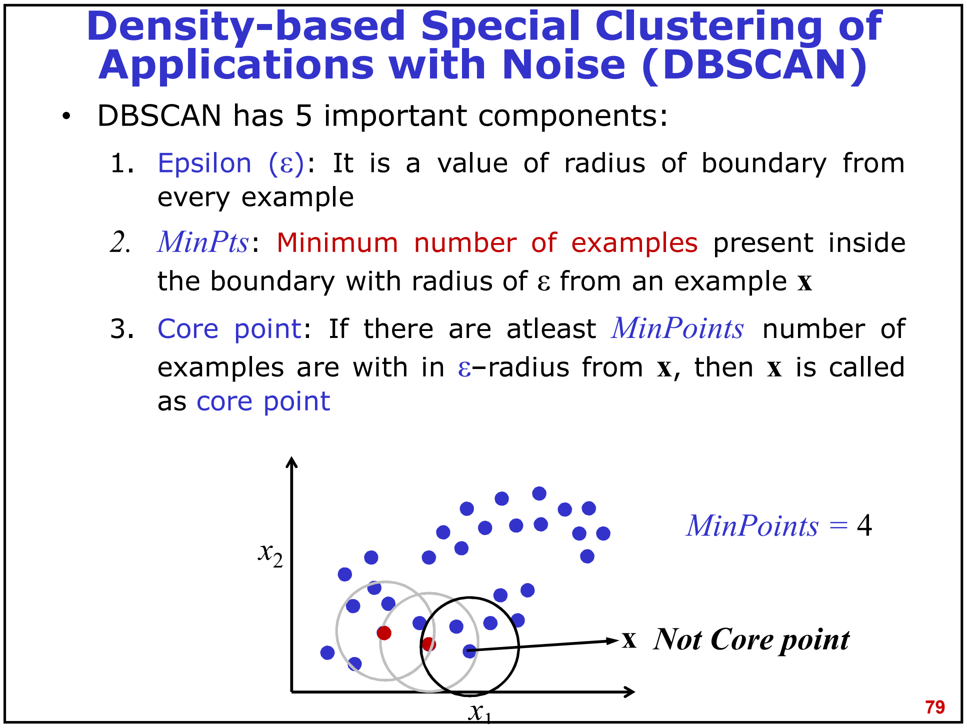
Density-based Special Clustering of Applications with Noise (DBSCAN)

- DBSCAN has 5 important components:
  1. Epsilon (ε): It is a value of radius of boundary from every example
  2. *MinPoints*: Minimum number of examples present inside the boundary with radius of ε from an example **x**
  3. Core point: If there are atleast *MinPoints* number of examples are with in ε–radius from **x**, then **x** is called as core point



Density-based Special Clustering of Applications with Noise (DBSCAN)

- DBSCAN has 5 important components:
  1. Epsilon (ε): It is a value of radius of boundary from every example
  2. *MinPoints*: Minimum number of examples present inside the boundary with radius of ε from an example **x**
  3. Core point: If there are atleast *MinPoints* number of examples are with in ε–radius from **x**, then **x** is called as core point

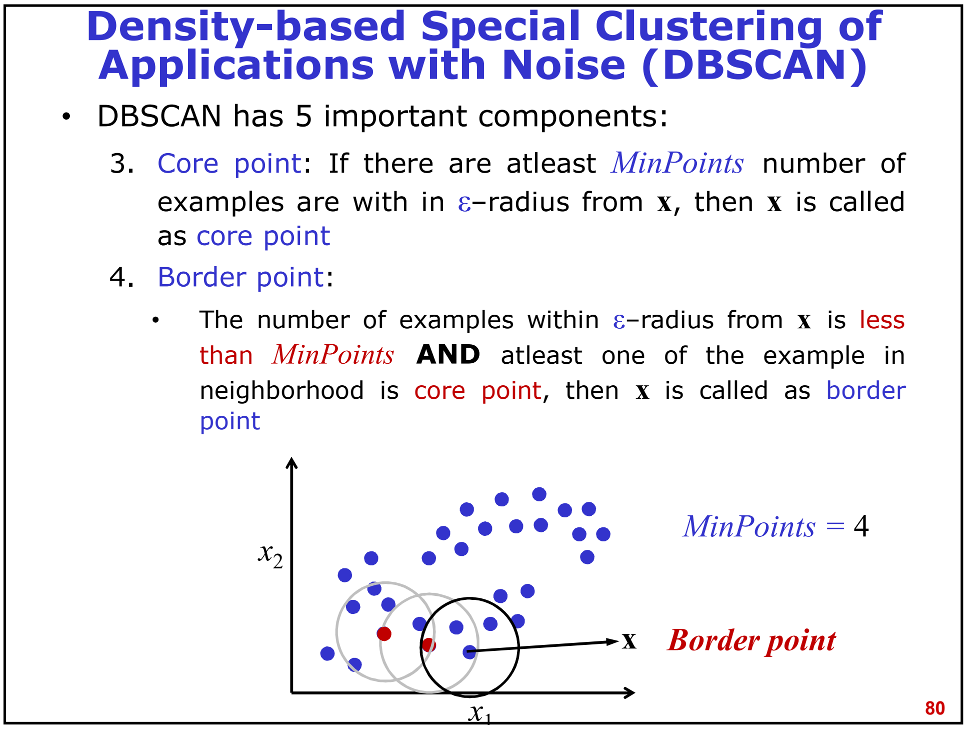


Density-based Special Clustering of Applications with Noise (DBSCAN)

- DBSCAN has 5 important components:
  1. Epsilon (ε): It is a value of radius of boundary from every example
  2. *MinPts*: Minimum number of examples present inside the boundary with radius of ε from an example $\mathbf{x}$
  3. Core point: If there are atleast *MinPoints* number of examples are with in ε–radius from $\mathbf{x}$, then $\mathbf{x}$ is called as core point

$MinPoints = 4$

$\mathbf{x}$ ***Not Core point***

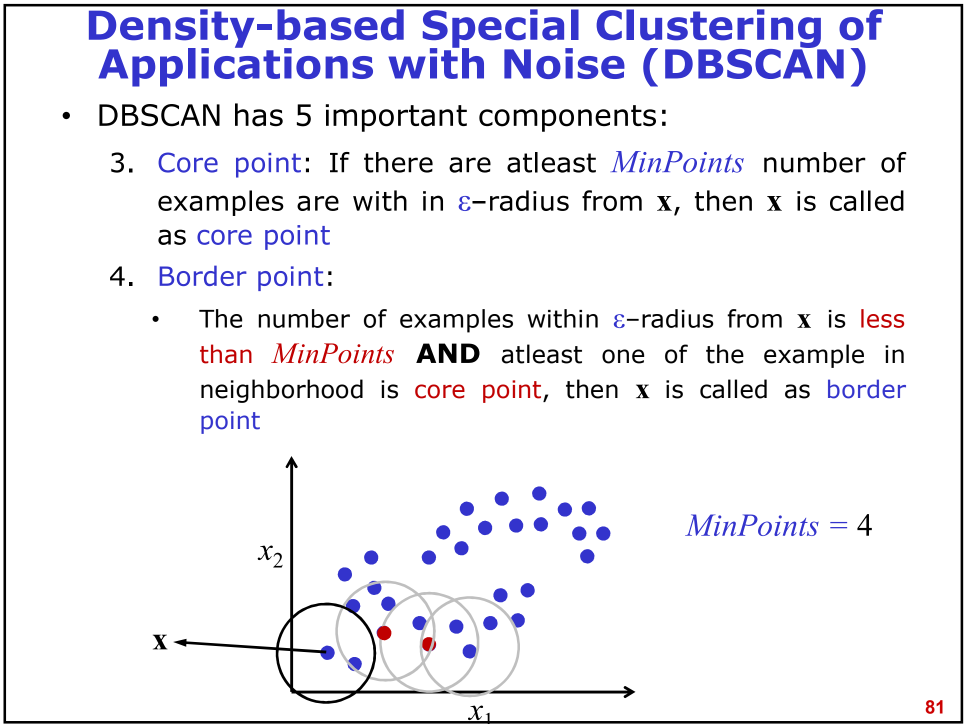


Density-based Special Clustering of Applications with Noise (DBSCAN)

- DBSCAN has 5 important components:
  3. Core point: If there are atleast *MinPoints* number of examples are with in ε–radius from $\mathbf{x}$, then $\mathbf{x}$ is called as core point
  4. Border point:
     - The number of examples within ε–radius from $\mathbf{x}$ is less than *MinPoints* **AND** atleast one of the example in neighborhood is core point, then $\mathbf{x}$ is called as border point

$MinPoints = 4$

$\mathbf{x}$ ***Border point***

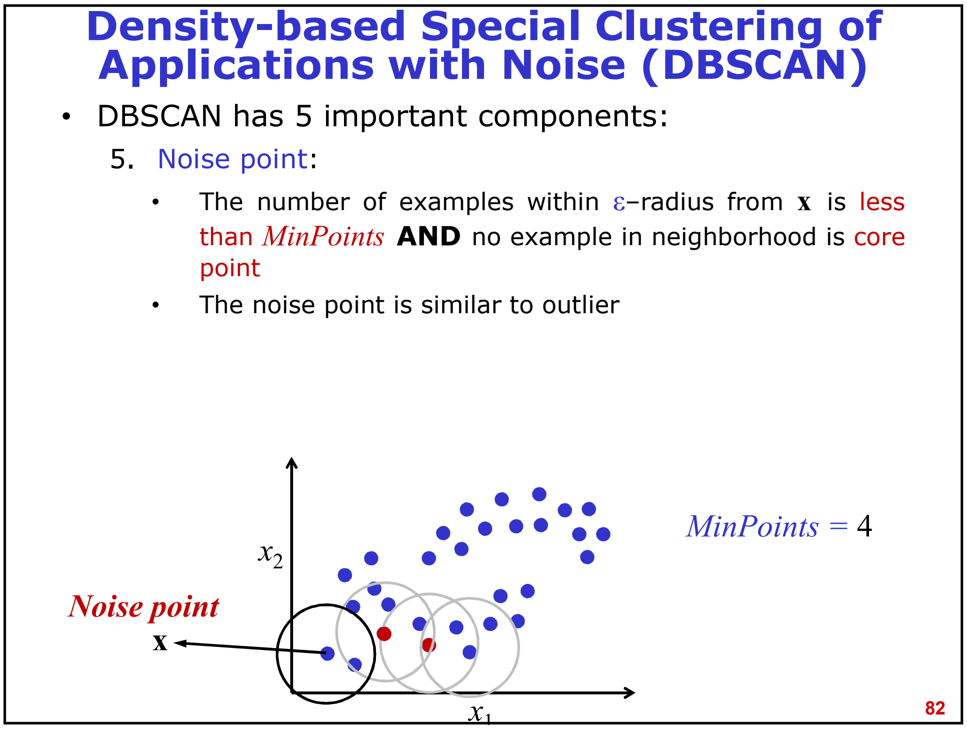


Density-based Special Clustering of Applications with Noise (DBSCAN)

- DBSCAN has 5 important components:

3. Core point: If there are atleast *MinPoints* number of examples are with in ε–radius from **x**, then **x** is called as core point
4. Border point:
   - The number of examples within ε–radius from **x** is less than *MinPoints* **AND** atleast one of the example in neighborhood is core point, then **x** is called as border point

$MinPoints = 4$



Density-based Special Clustering of Applications with Noise (DBSCAN)

- DBSCAN has 5 important components:

5. Noise point:
   - The number of examples within ε–radius from **x** is less than *MinPoints* **AND** no example in neighborhood is core point
   - The noise point is similar to outlier

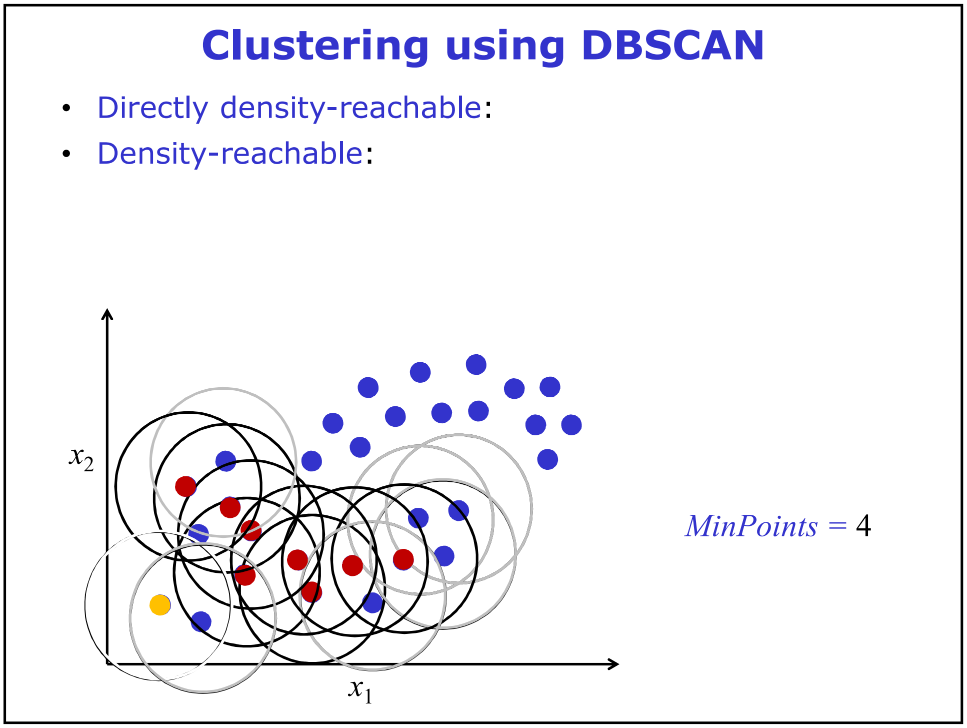
$MinPoints = 4$

Clustering using DBSCAN

- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$
- First step is to identify core points, border points and noise points
  - Only core points and border points are considered inside the cluster
  - Noise points are not taken into the cluster
  - Thus DBSCAN is robust to outliers
- Next step is to find the connected components of core points
- Connected component of core points: Connecting the core points that are reachable from any point
  - All the connected (reachable) core points form a cluster

Clustering using DBSCAN

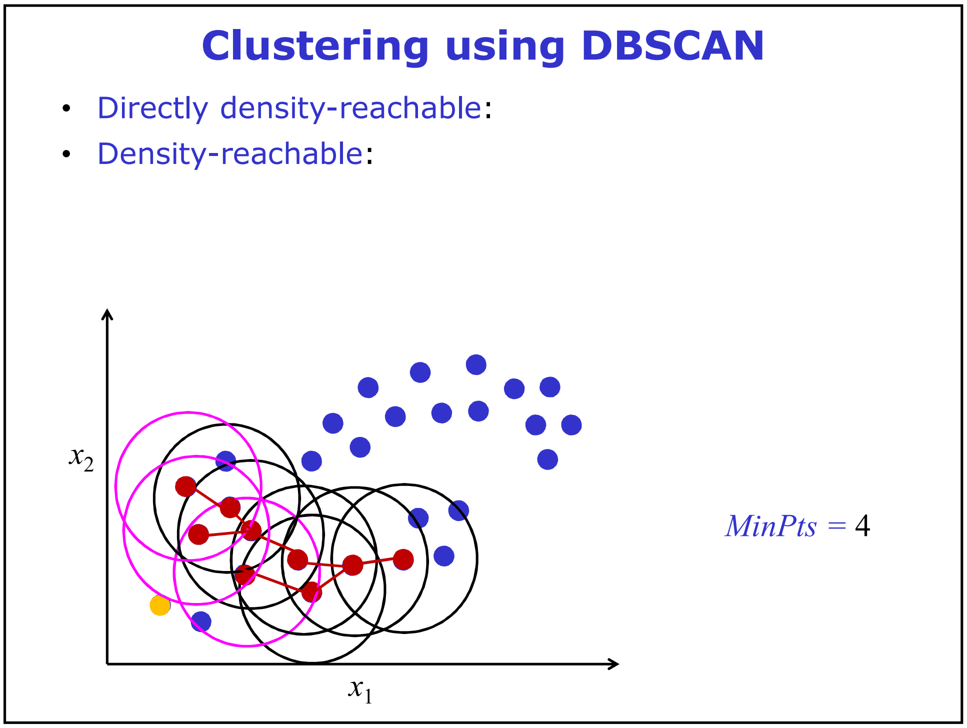
- The connected component of core points is obtained by understanding following *two* definitions.
- Directly density-reachable: A core point $\mathbf{x}_i$ is directly density-reachable to a core point $\mathbf{x}_j$, if the core point $\mathbf{x}_j$ is within ε–distance from core point $\mathbf{x}_i$
- Density-reachable: A core point $\mathbf{x}_i$ is indirectly reachable to another core point $\mathbf{x}_j$ through other core points, $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_K$ such that
  - $\mathbf{x}_i$ is directly density-reachable to $\mathbf{x}_1$
  - $\mathbf{x}_1$ is directly density-reachable to $\mathbf{x}_2$
  - ….
  - $\mathbf{x}_k$ is directly density-reachable to $\mathbf{x}_{k+1}$
  - ….
  - $\mathbf{x}_K$ is directly density-reachable to $\mathbf{x}_j$



Clustering using DBSCAN

- Directly density-reachable:
- Density-reachable:

MinPoints = 4



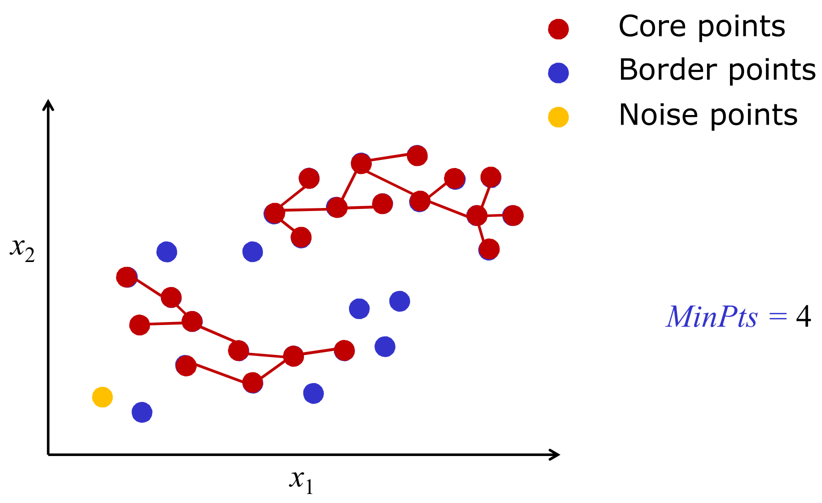
Clustering using DBSCAN

- Directly density-reachable:
- Density-reachable:

MinPts = 4

Clustering using DBSCAN

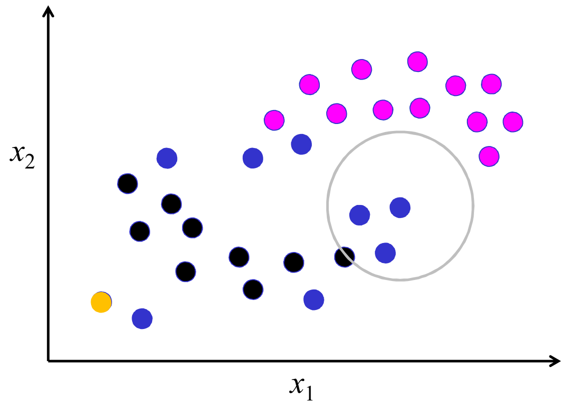
- All the core points with connected component forms a cluster



$MinPts = 4$

Clustering using DBSCAN

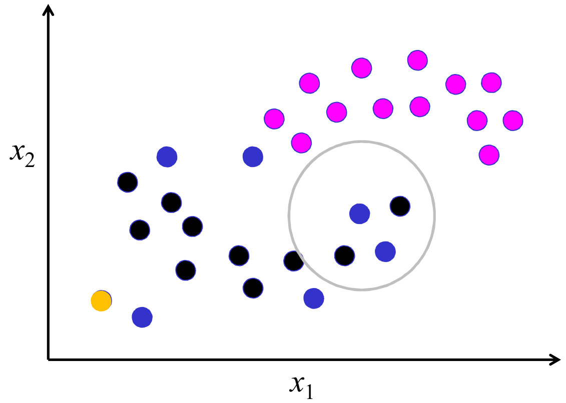
- All the core points with connected component forms a cluster
- Assign the border points to nearby cluster which is at ε–radius from that border point



$MinPts = 4$

Clustering using DBSCAN

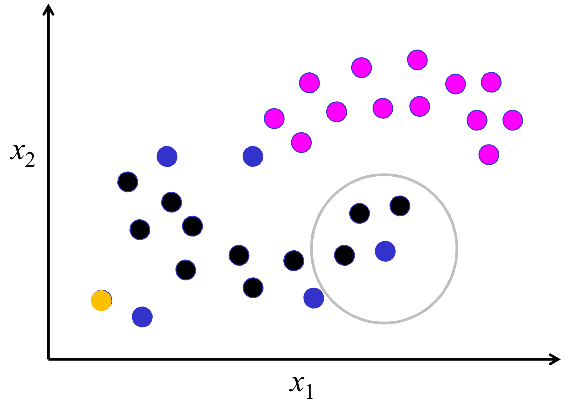
- All the core points with connected component forms a cluster
- Assign the border points to nearby cluster which is at ε–radius from that border point



$MinPts = 4$

Clustering using DBSCAN

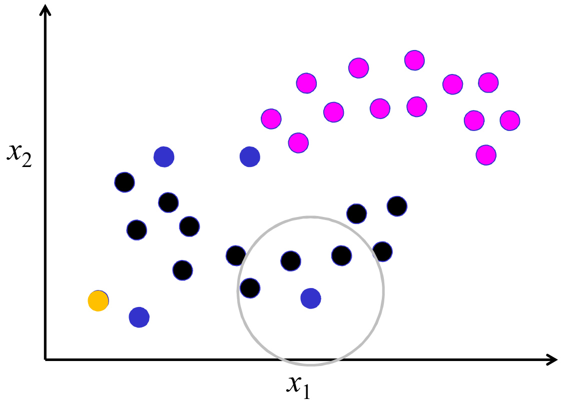
- All the core points with connected component forms a cluster
- Assign the border points to nearby cluster which is at ε–radius from that border point



$MinPts = 4$

Clustering using DBSCAN

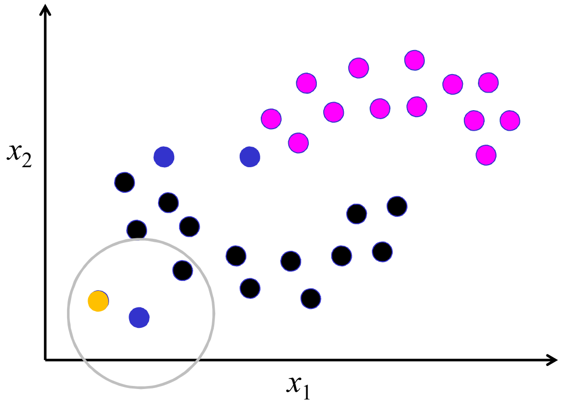
- All the core points with connected component forms a cluster
- Assign the border points to nearby cluster which is at ε–radius from that border point



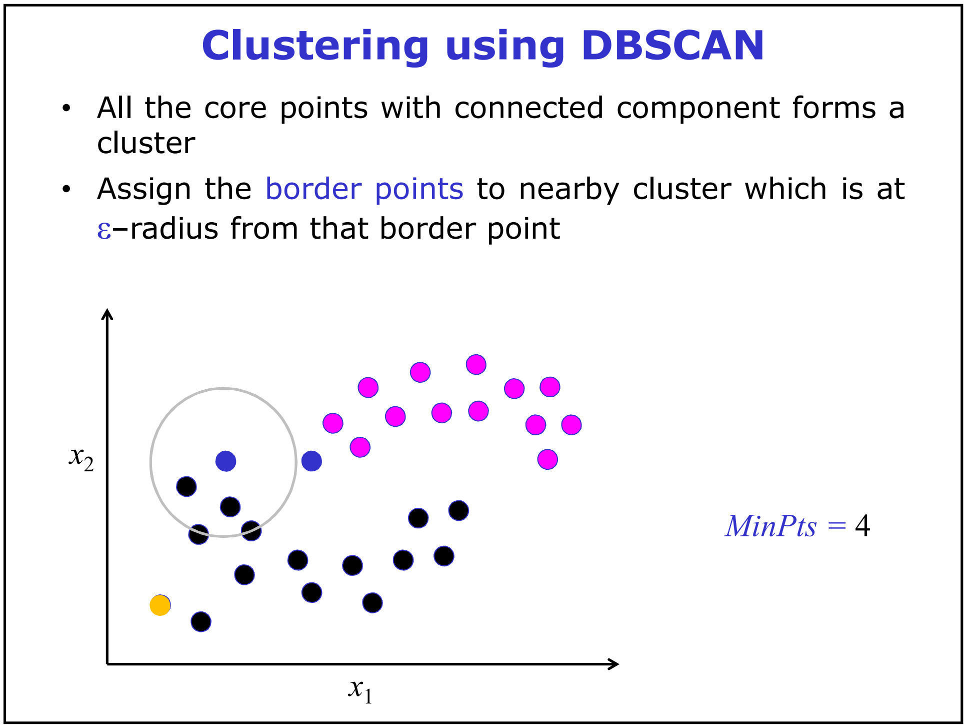
MinPts = 4

Clustering using DBSCAN

- All the core points with connected component forms a cluster
- Assign the border points to nearby cluster which is at ε–radius from that border point



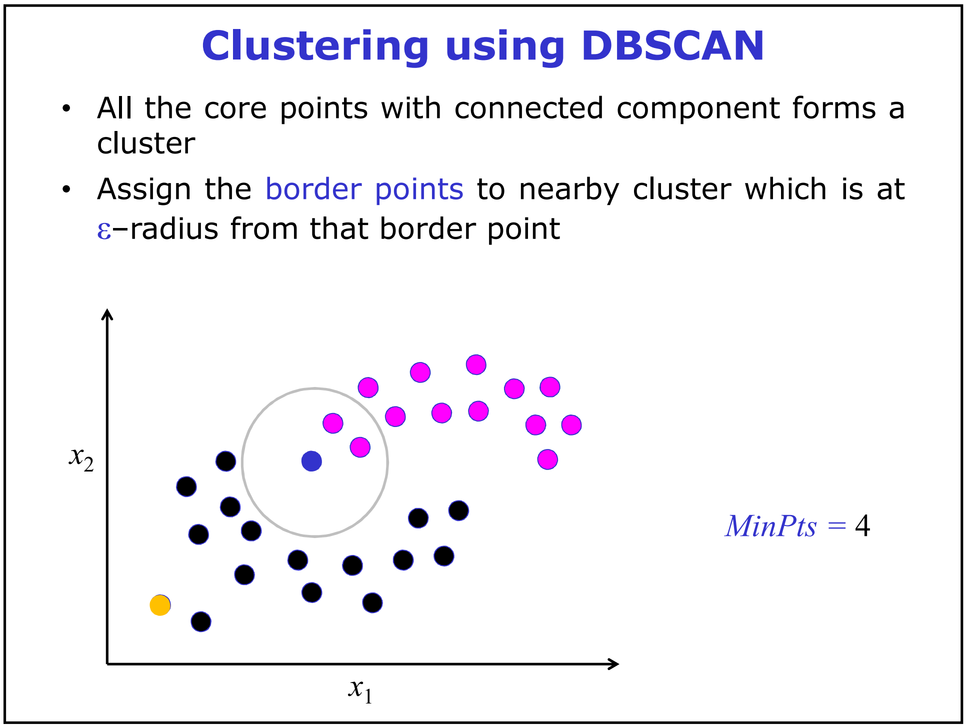
MinPts = 4



Clustering using DBSCAN

- All the core points with connected component forms a cluster
- Assign the border points to nearby cluster which is at ε–radius from that border point

$MinPts = 4$



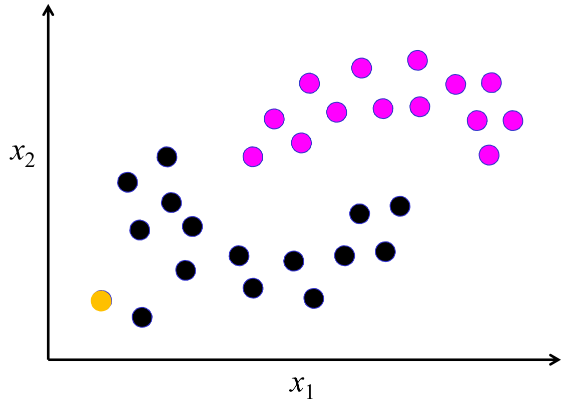
Clustering using DBSCAN

- All the core points with connected component forms a cluster
- Assign the border points to nearby cluster which is at ε–radius from that border point

$MinPts = 4$

Clustering using DBSCAN

- All the core points with connected component forms a cluster
- Assign the border points to nearby cluster which is at ε–radius from that border point



$MinPts = 4$

Clustering using DBSCAN

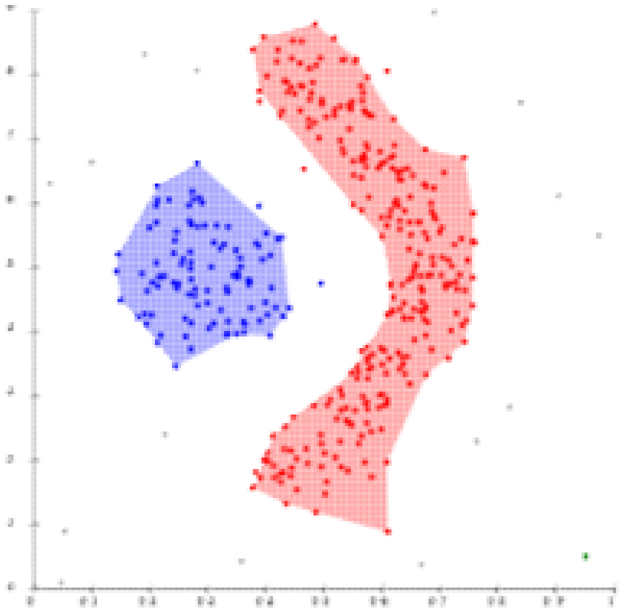
- **Training process**:
- Given: Training data, $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^{N}, \mathbf{x}_n \in \mathbb{R}^d$
- Identify the core points, border points and noise points
- Find the connected components of core points
- Each connected component forms a cluster
- Assign each of the border points to a nearby cluster which is at ε–radius from that border point
- Noise points are not assigned to any clusters

- Training process, stores the core points as model

Clustering using DBSCAN

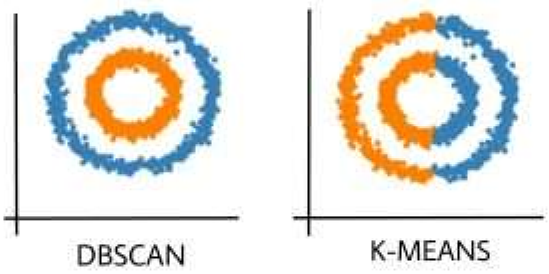
- **Test process**:
- For a test example, identify it as core point or border point or noise point
- If it is a core point, assign it to a cluster to which it is directly density-reachable or density-reachable
- If it is a border point, assign it to a nearby cluster which is at ε–radius from that border point
- If it is a noise point, do not assign to any cluster

Clustering using DBSCAN



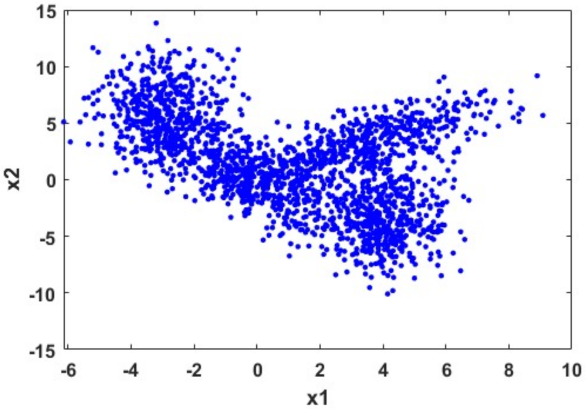
Advantages of DBSCAN

- DBSCAN does not require to specify the number of clusters in the data a priori
- DBSCAN can find arbitrarily shaped clusters
- DBSCAN has a notion of noise, and is robust to outliers
- The parameters ε and *MinPoints* are experimentally set by the users

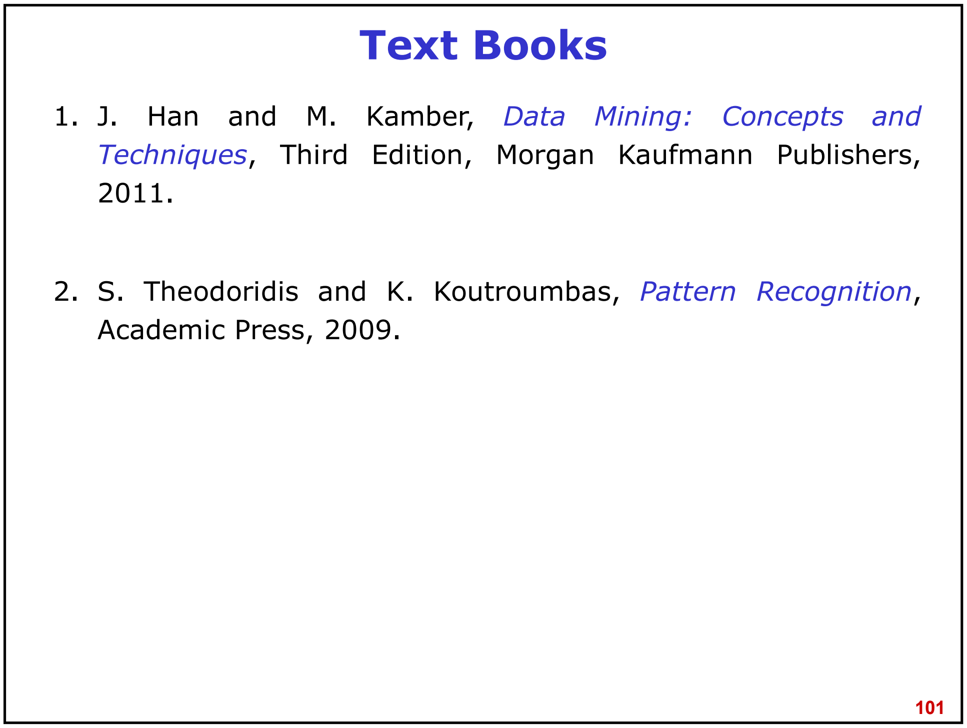


Limitation of DBSCAN

- DBSCAN is not suitable when the data is completely dense and there is no low dense area to separate



- The parameters ε and *MinPoints* should be chosen carefully



Text Books

1. J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Third Edition, Morgan Kaufmann Publishers, 2011.

2. S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, 2009.

101