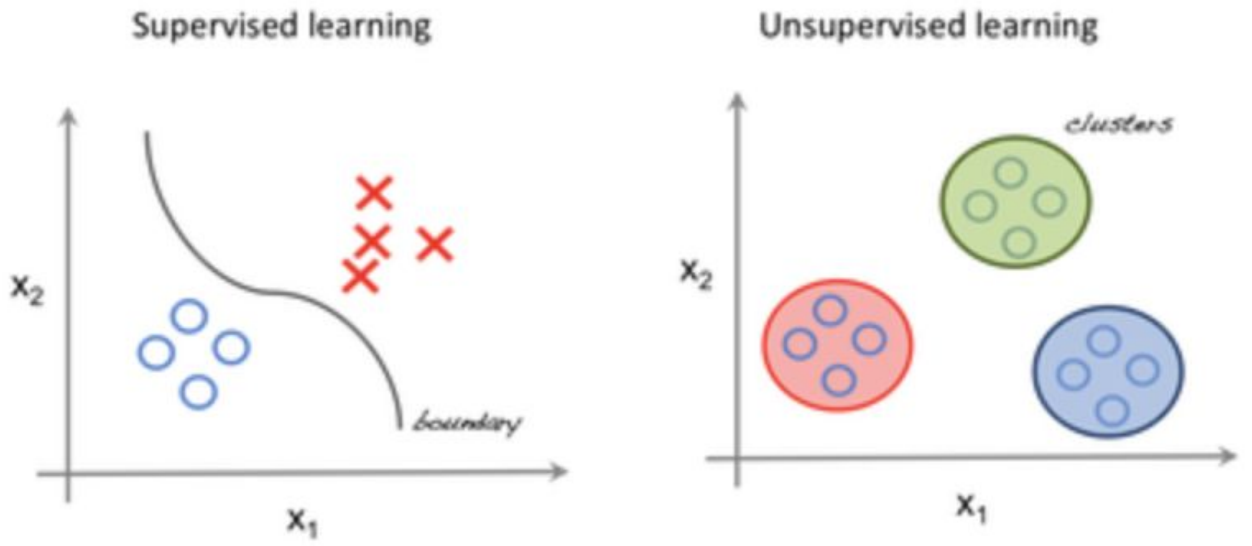
EC 414 Week 7 Discussion
Clustering – K-Means & DP-Means

What are Clustering Algorithms?

- Clustering algorithms are a form of unsupervised learning.
- Unsupervised learning:
  - no labels
  - algorithm attempts to find patterns in the data directly from the examples

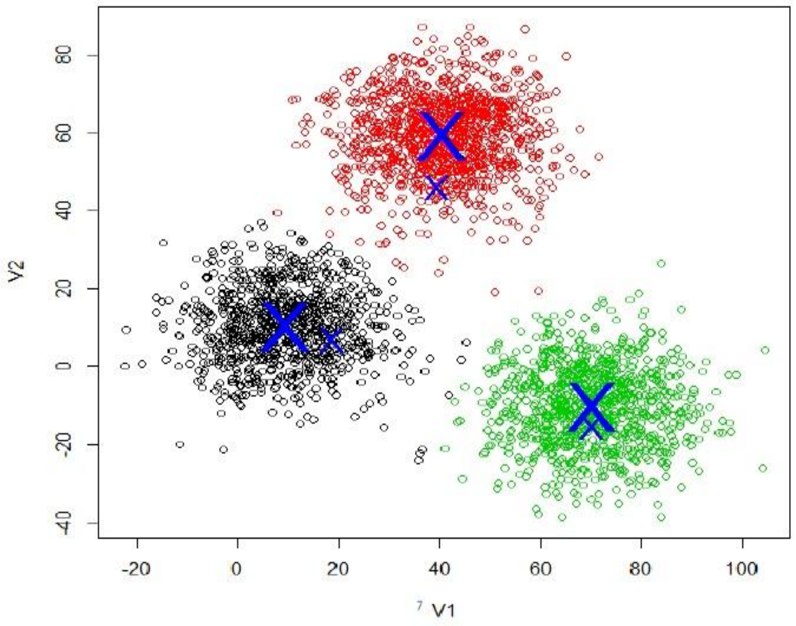


Some Applications of Clustering

- *Search Engines*
  - Grouping together similar objects to improve search experiences
- *Marketing*
  - Finding groups of customers with similar behavior given a large database of customer data containing their properties and past buying records
- *etc...*

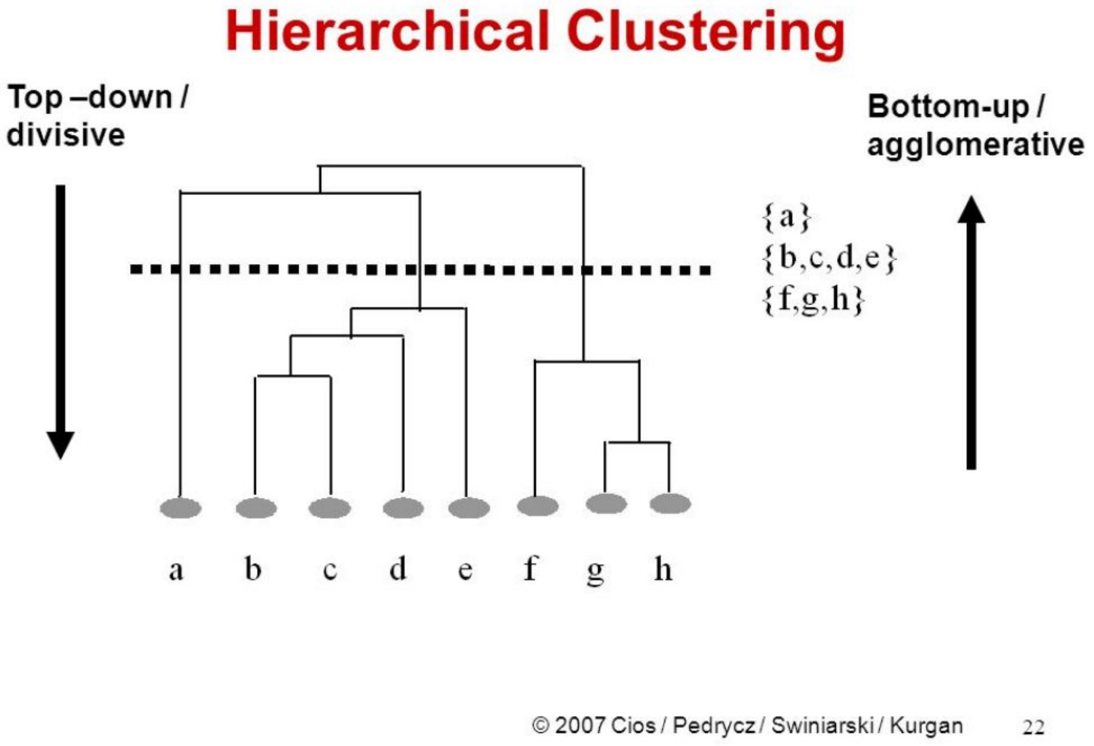
Types of Clustering Algorithms

- ***Hard Clustering***
  - Each point belongs either belongs completely to a cluster or it does not.
- Soft Clustering
  - Points can be “shared” by clusters. For example a point will have a probabilistic likelihood of belonging to several various clusters
- Partitional Clustering
  - Carving the space into parts
- ***Hierarchical Clustering***
  - Clusters themselves are further clusterable ...



Hierarchical Clustering: Agglomerative vs Divisive

- *Agglomerative: bottom-up*
  - each data point starts as its own cluster
  - we then merge clusters every iteration
- *Divisive: top-down*
  - starting with one big cluster, we recursively divide the data into smaller clusters until we reach each individual point
- "Tree" of clusters == **dendrogram**
- The method of joining and dividing clusters is sometimes done by a similarity matrix



$K - Means$

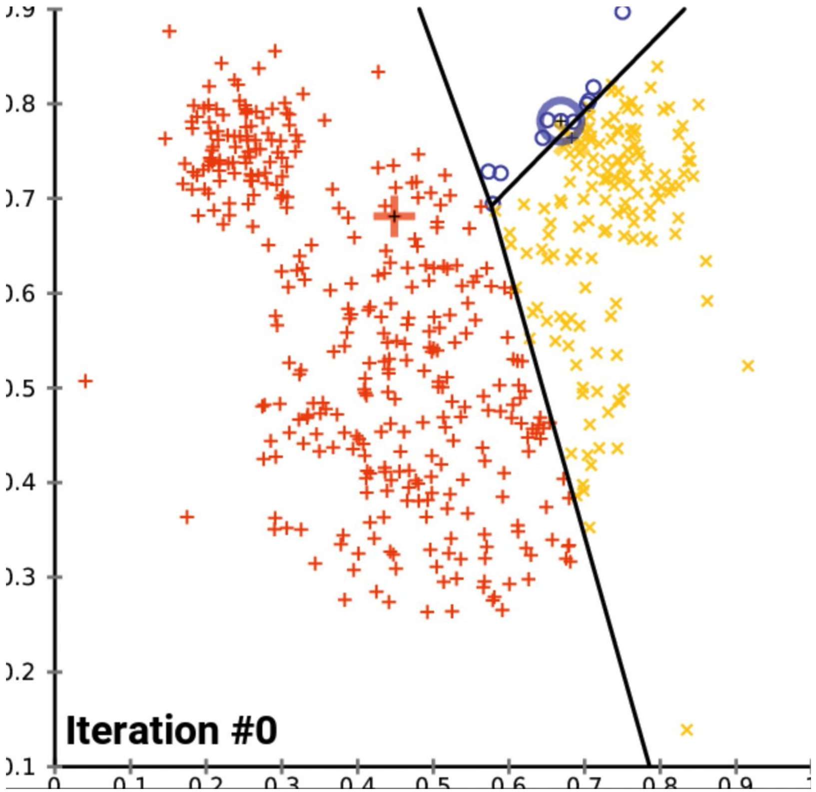
- Unsupervised learning that creates *k* clusters of data by minimizing a squared error function:

$$J = \sum_{i=1}^{k} \sum_{x \in S_i} ||x - \mu_i||^2$$

- Requires initial knowledge of number of clusters.

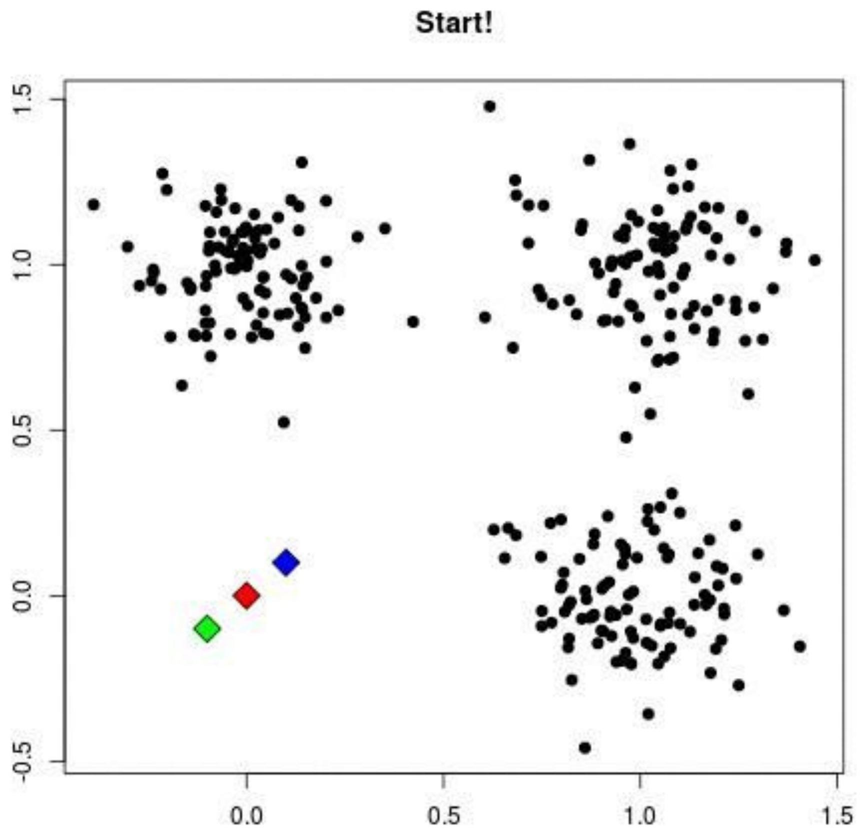
Algorithm Steps:

1) Initialize *k* means (centroids)
2) Assign all points to their closest centroids.
3) Recalculate positions of *k* centroids.
4) Repeat (2) and (3) until means become stationary.



K-means Clustering: Algorithm

1. Randomly assign K centroids to the space, which define the initial clusters.

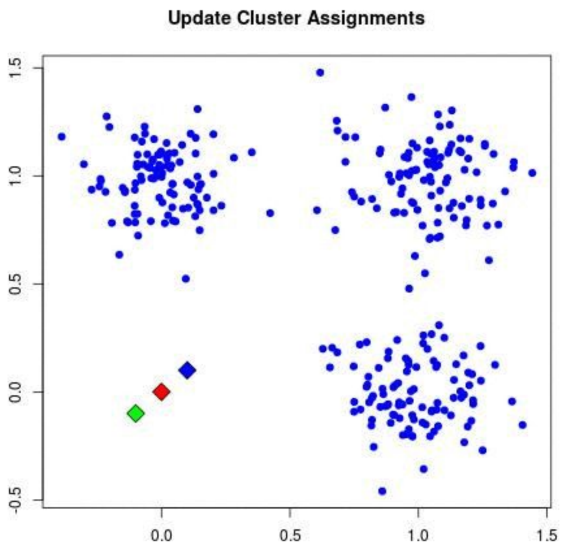


K-means Clustering

2. **Assignment step**: assign each point of training data to the closest centroid,

$$\underset{c_i \in C}{\operatorname{argmin}}\, dist(c_i, x)^2$$

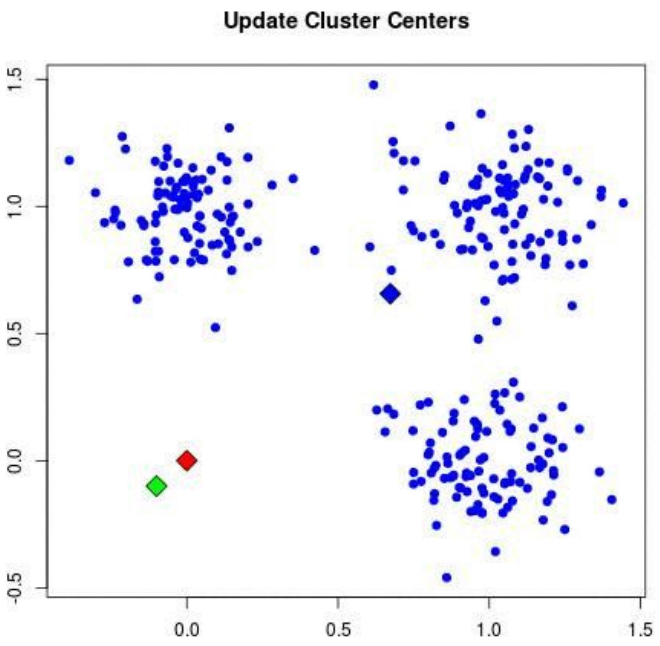
where $\boldsymbol{c_i}$ is the current centroid, and $\boldsymbol{C}$ is the set of all centroids. Usually Euclidean (L2) distance is used as the distance function.



K-means Clustering

3. **Update step:** Find the new centroids by taking the average of all the points assigned to that cluster,

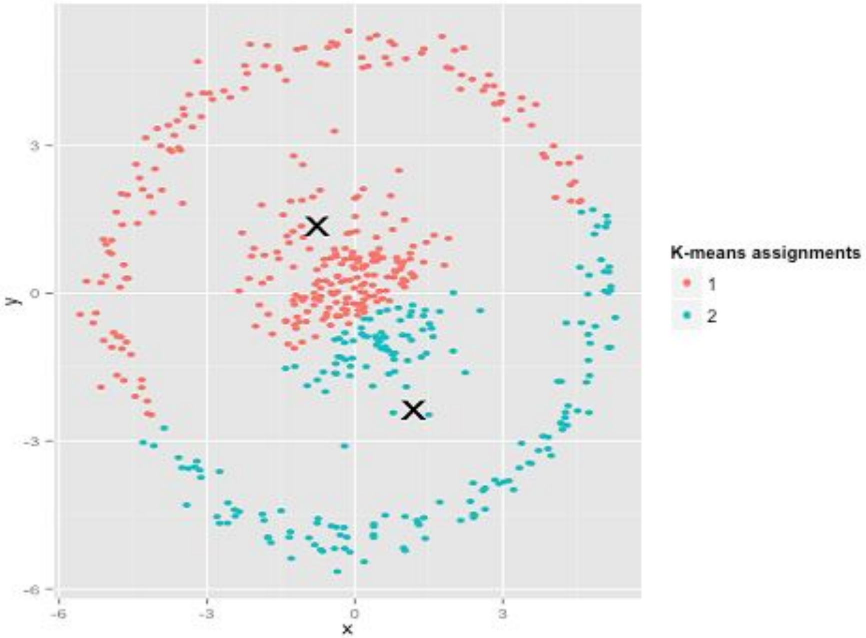
$$c_i = \frac{1}{|S_i|} \sum_{x_i \in S_i} x_i$$



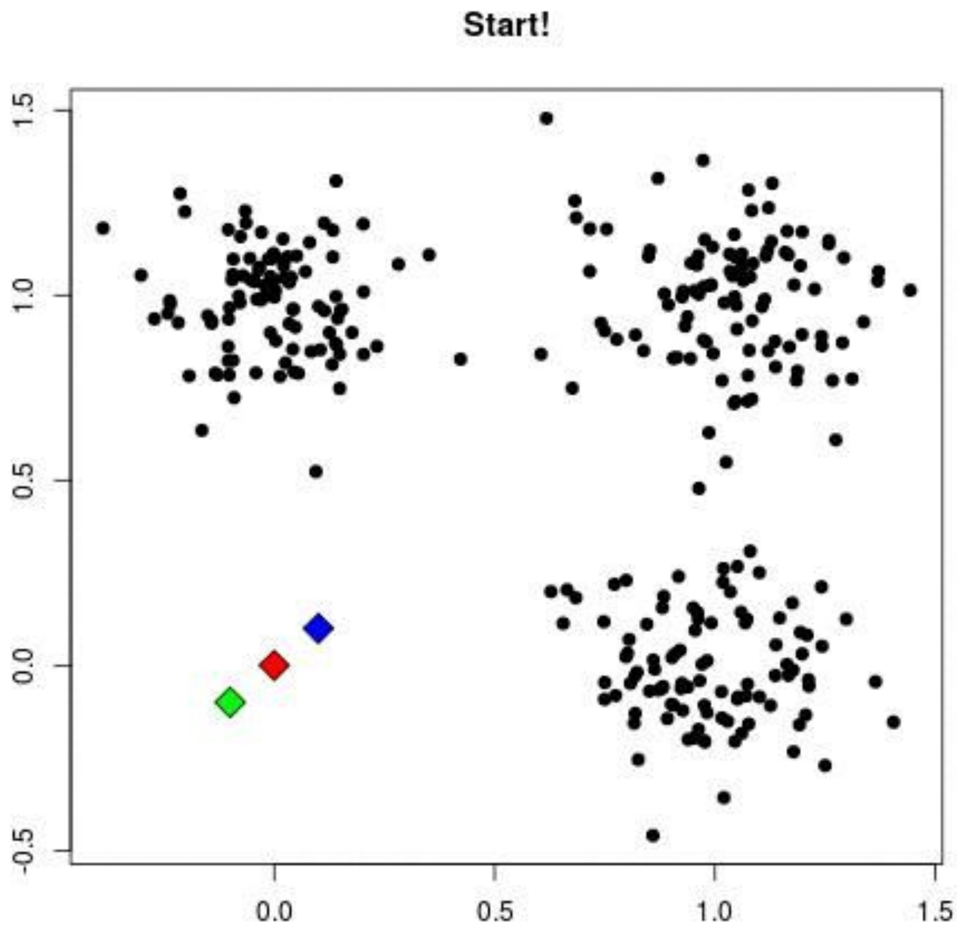
4. Repeat assignment & update until clusters no longer move

Characteristics of K-means

- Fixed number of clusters
- Often produces clusters of uniform size
- **Sensitive**: Order of data traversal affects final results, inconsistent due to random initialization
- Fast, easy to implement



K-means Clustering Animation



interactive visualization: https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

K-means++ : The Scikit-learn Implementation

Take one center c_1, chosen uniformly at random from $\mathcal{X}$.

Take a new center c_i, choosing $x \in \mathcal{X}$ with probability $\frac{D(x)^2}{\sum_{x \in \mathcal{X}} D(x)^2}$.

D(x) is the distance of a data point to the closest center already chosen.

Repeat Step 1b. until we have taken k centers altogether.

Proceed as with the standard `k-means` algorithm.

DP-means Clustering

1. Initialize only 1 cluster (k=1, $\ell_c = \{x_1 \ldots x_n\}$ initially), where the value of the centroid is the global mean,

$$\boldsymbol{\mu}_c = \frac{1}{|\ell_c|} \sum_{\boldsymbol{x} \in \ell_c} \boldsymbol{x}.$$

2. **Assign** each point x_i to the closest centroid. Usually Euclidean (L2) distance is used as the distance function.
3. If the distance from $x_i > \lambda$ (cluster penalty parameter), add a new cluster (k=k+1), and set $\mu_c = x_i$. Otherwise, place the point into the nearest cluster.
4. **Update:** Find the new centroid by taking the average of all points assigned to each cluster ℓ_j:

$$\boldsymbol{\mu}_j = \frac{1}{|\ell_j|} \sum_{\boldsymbol{x} \in \ell_j} \boldsymbol{x}.$$

5. Repeat steps 2-4 until clusters become stable (they no longer reassign with each iteration).

DP Means - Algorithm

Input: $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$: input data, λ : cluster penalty parameter
Output: Clustering $\ell_1, ..., \ell_k$ and number of clusters k

1. Init. $k = 1, \ell_1 = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ and $\boldsymbol{\mu}_1$ the global mean.
2. Init. cluster indicators $z_i = 1$ for all $i = 1, ..., n$.
3. Repeat until convergence

- For each point $\boldsymbol{x}_i$
  - Compute $d_{ic} = \|\boldsymbol{x}_i - \boldsymbol{\mu}_c\|^2$ for $c = 1, ..., k$
  - If $\min_c d_{ic} > \lambda$, set $k = k + 1$, $z_i = k$, and $\boldsymbol{\mu}_k = \boldsymbol{x}_i$.
  - Otherwise, set $z_i = \text{argmin}_c d_{ic}$.
- Generate clusters $\ell_1, ..., \ell_k$ based on $z_1, ..., z_k$: $\ell_j = \{\boldsymbol{x}_i \mid z_i = j\}$.
- For each cluster ℓ_j, compute $\boldsymbol{\mu}_j = \frac{1}{|\ell_j|} \sum_{\boldsymbol{x} \in \ell_j} \boldsymbol{x}$.

Characteristics of DP-means

- Fast and scalable nonparametric extension of k-means (clusters can grow as a function of data)
- Number of clusters is not fixed (do not need prior knowledge of k)
- Can minimize the number of clusters due to the cluster penalty

K-means vs. DP-means objectives

number of clusters

number of cases

centroid for cluster j

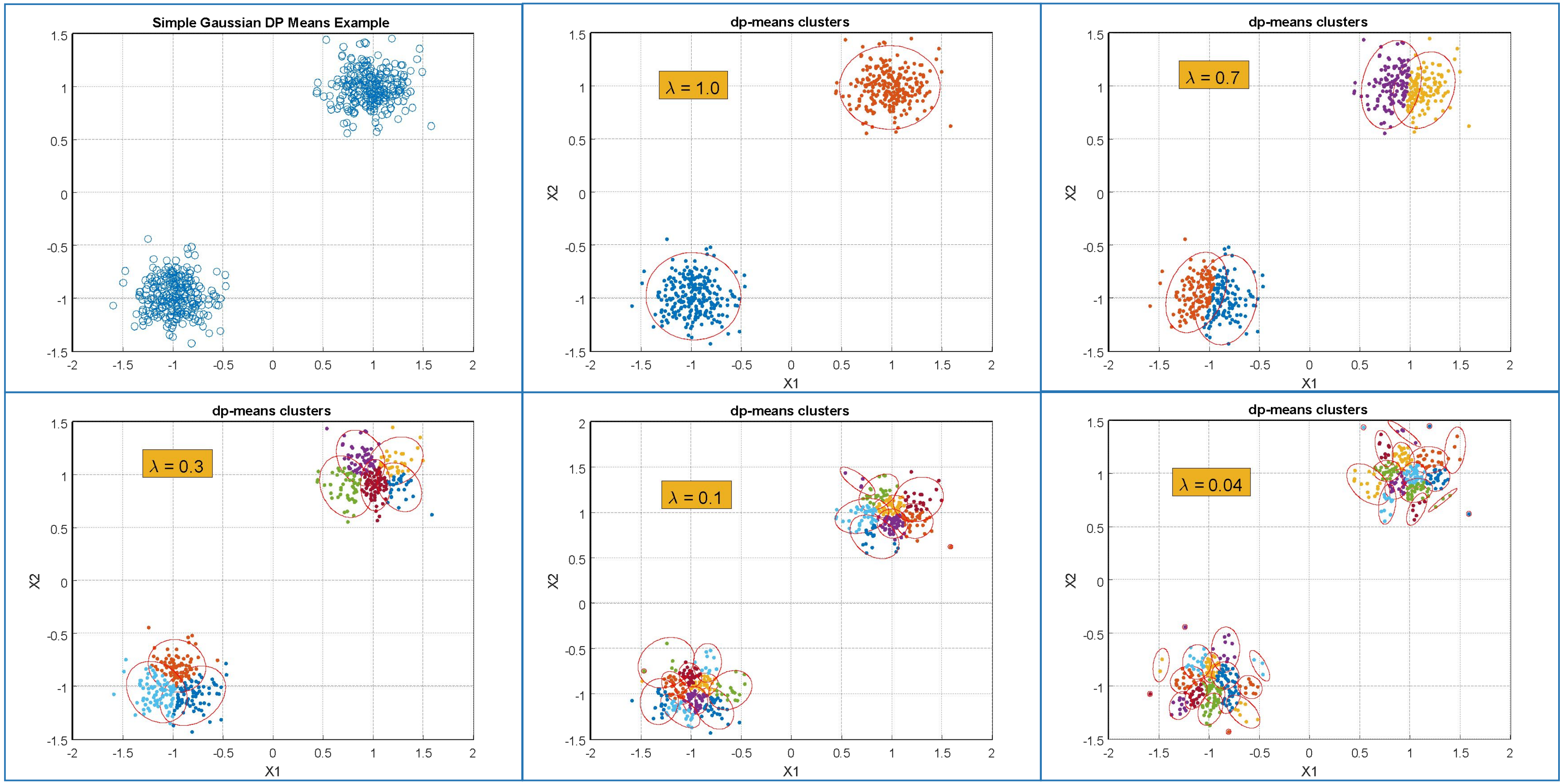
case i

objective function ← $J = \sum_{j=1}^{k} \sum_{i=1}^{n} \underbrace{\left\| x_i^{(j)} - c_j \right\|^2}_{\text{Distance function}}$

$$\min_{\{\ell_c\}_{c=1}^{k}} \quad \sum_{c=1}^{k} \sum_{\boldsymbol{x} \in \ell_c} \|\boldsymbol{x} - \boldsymbol{\mu}_c\|^2 + \boxed{\lambda k}$$

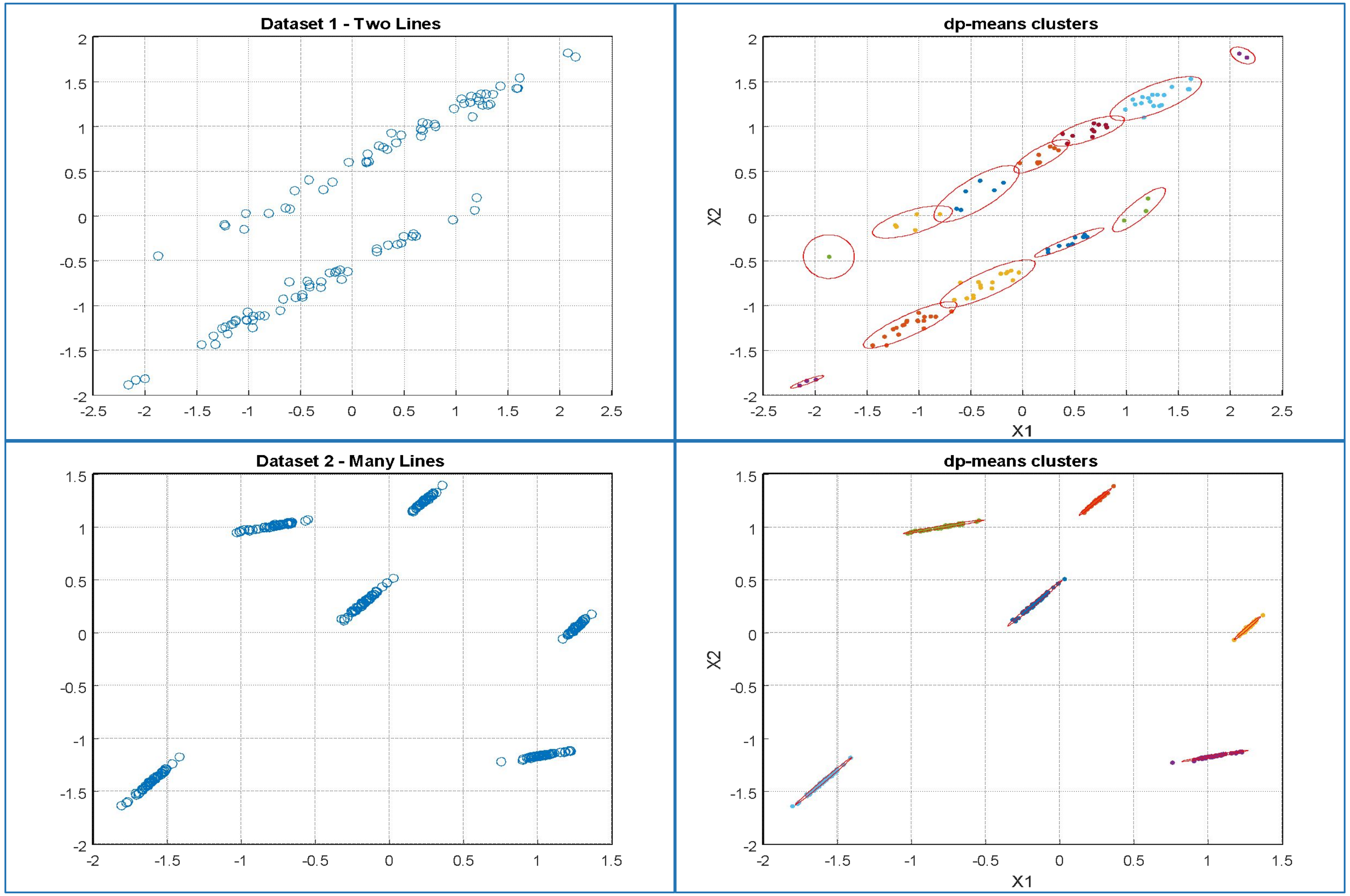
$$\text{where} \qquad \boldsymbol{\mu}_c = \frac{1}{|\ell_c|} \sum_{\boldsymbol{x} \in \ell_c} \boldsymbol{x}.$$

DP Means – Effect of λ on Simple Example:



- Smaller λ ➔ Finer clusters

DP Means – Examples 1-2



DP Means – Examples 3-4