Chapter 6

K Means Clustering

Clustering

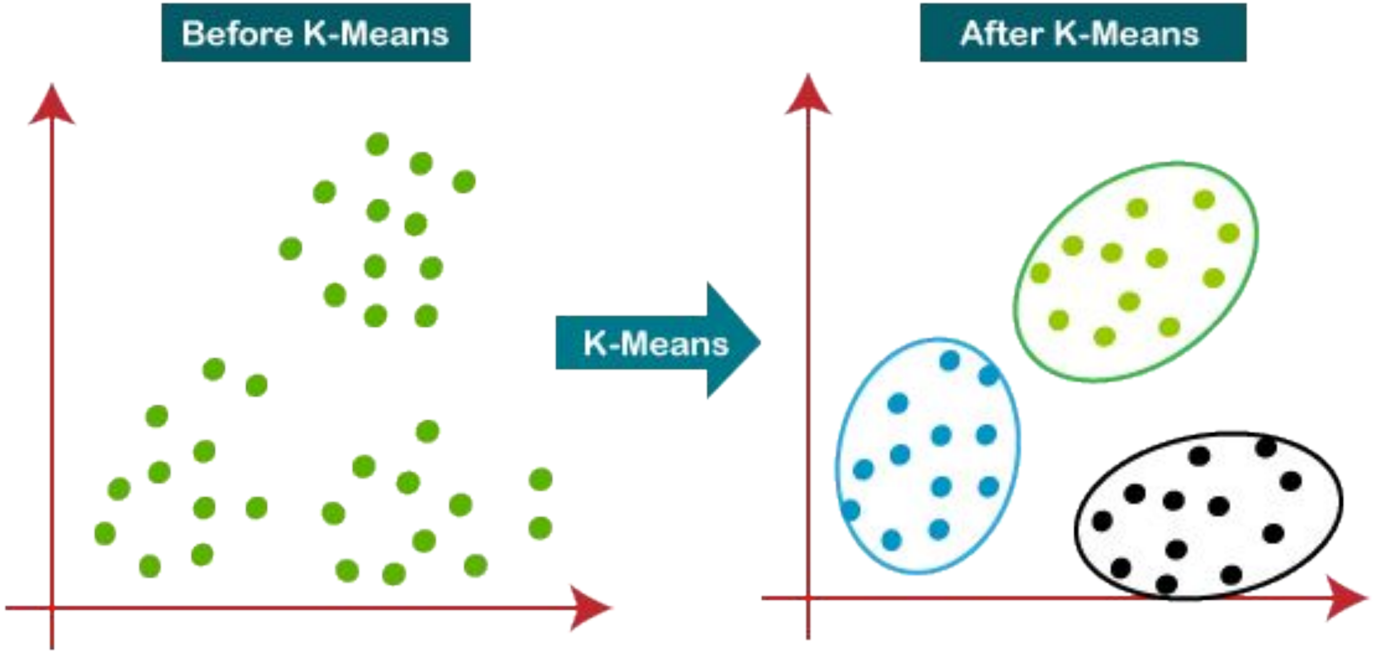
- <u>Unsupervised learning</u>
- Requires data, but ___________

Clustering

- <u>Unsupervised learning</u>
- Requires data, but NO labels

Clustering

- Basic idea: Group together *similar instances*



Clustering

- Image segmentation
  - Goal: Partition an image into perceptually similar regions

Other examples

- Group emails or search results
- Customer shopping patterns

K-Means Algorithm

- K −means is an <u>iterative algorithm</u>
- Consider the *dataset* of n −dimensional vectors

K-Means Algorithm

- K −means is an <u>iterative algorithm</u>
- Consider the *dataset* of n −dimensional vectors

$$\bar{\mathbf{x}}(1), \bar{\mathbf{x}}(2), \dots, \bar{\mathbf{x}}(M)$$

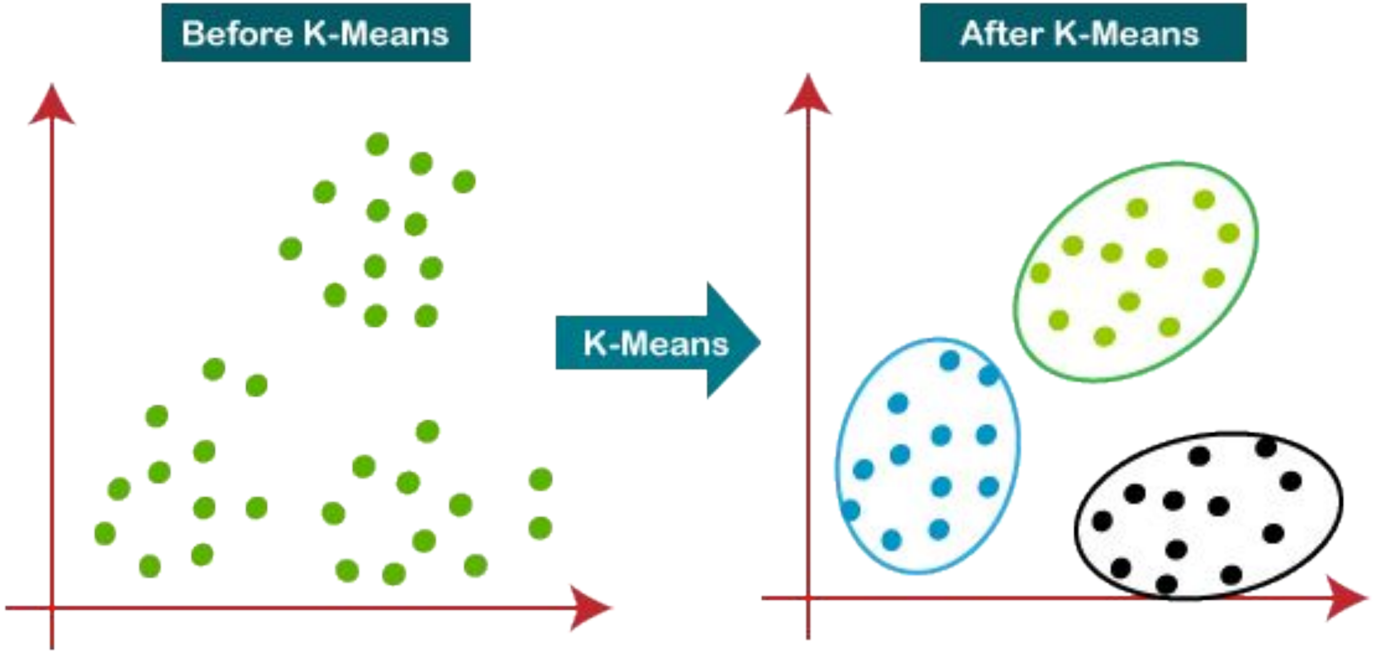
Clusters

- Organize the data into K *clusters*

Clusters

- Organize the data into K *clusters*

$$\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_K$$



Clusters

- The centroids for the clusters are

Clusters

- The centroids for the clusters are

$$\bar{\boldsymbol{\mu}}_1, \bar{\boldsymbol{\mu}}_2, \dots, \bar{\boldsymbol{\mu}}_K$$

Cluster assignment

- Let $\alpha_i(j)$ denote the *cluster assignment indicator*

Cluster assignment

Cluster assignment

- Let $\alpha_i(j)$ denote the *cluster assignment indicator*

$$\alpha_i(j) = \begin{cases} 1 & \bar{\mathbf{x}}(j) \in c_i \\ 0 & \bar{\mathbf{x}}(j) \notin c_i \end{cases}$$

K-Means procedure

- *Initialize centroids* randomly

- $\overline{\boldsymbol{\mu}}_i^{(l-1)}$ denotes centroids in iteration $l-1$

K-Means procedure

- *Initialize centroids* randomly

$$\bar{\boldsymbol{\mu}}_1^{(0)}, \bar{\boldsymbol{\mu}}_2^{(0)}, \dots, \bar{\boldsymbol{\mu}}_K^{(0)}$$

- $\bar{\boldsymbol{\mu}}_i^{(l-1)}$ denotes centroids in iteration $l-1$

Cluster determination

- Assign $\bar{\mathbf{x}}(j)$ to the cluster $\tilde{\imath}$ with *closest centroid* $\bar{\boldsymbol{\mu}}_{\tilde{\imath}}^{(l-1)}$

Cluster determination

- This is *minimized* when $\alpha_{\tilde{i}}^{(l)}(j) = 1$, where

Cluster determination

- This is *minimized* when $\alpha_{\tilde{i}}^{(l)}(j) = 1$, where

$$\tilde{\imath} = \arg\min \left\| \bar{\mathbf{x}}(j) - \bar{\boldsymbol{\mu}}_i^{(l-1)} \right\|^2$$

Centroid determination

- The centroid of cluster i can be determined as below
- The average of all points assigned to cluster i in iteration l

Centroid determination

- This can be mathematically expressed as

Centroid determination

- This can be mathematically expressed as

$$\bar{\boldsymbol{\mu}}_i^{(l)} = \frac{\sum_{j=1}^{M} \alpha_i^{(l)}(j)\bar{\mathbf{x}}(j)}{\sum_{j=1}^{M} \alpha_i^{(l)}(j)}$$

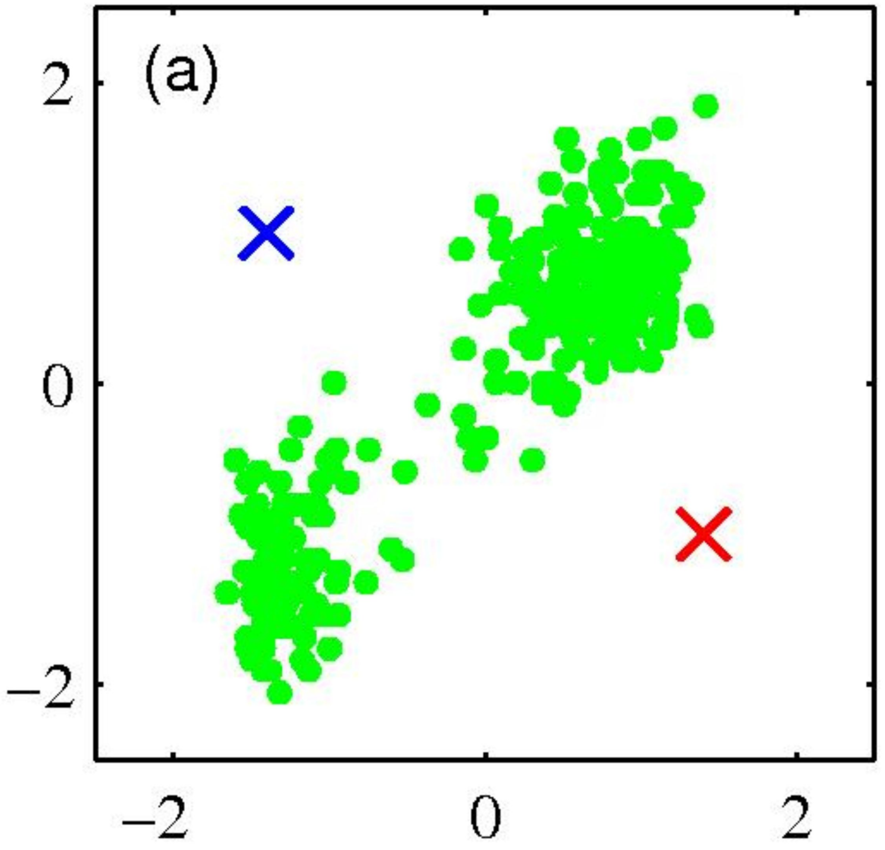
$$= \frac{\sum_{j:\bar{\mathbf{x}}(j)\in C_i} \bar{\mathbf{x}}(j)}{\sum_{j:\bar{\mathbf{x}}(j)\in C_i} 1}$$

Convergence

- The cluster and centroid computation steps are repeated until convergence
- i.e., when cluster assignments do NOT change

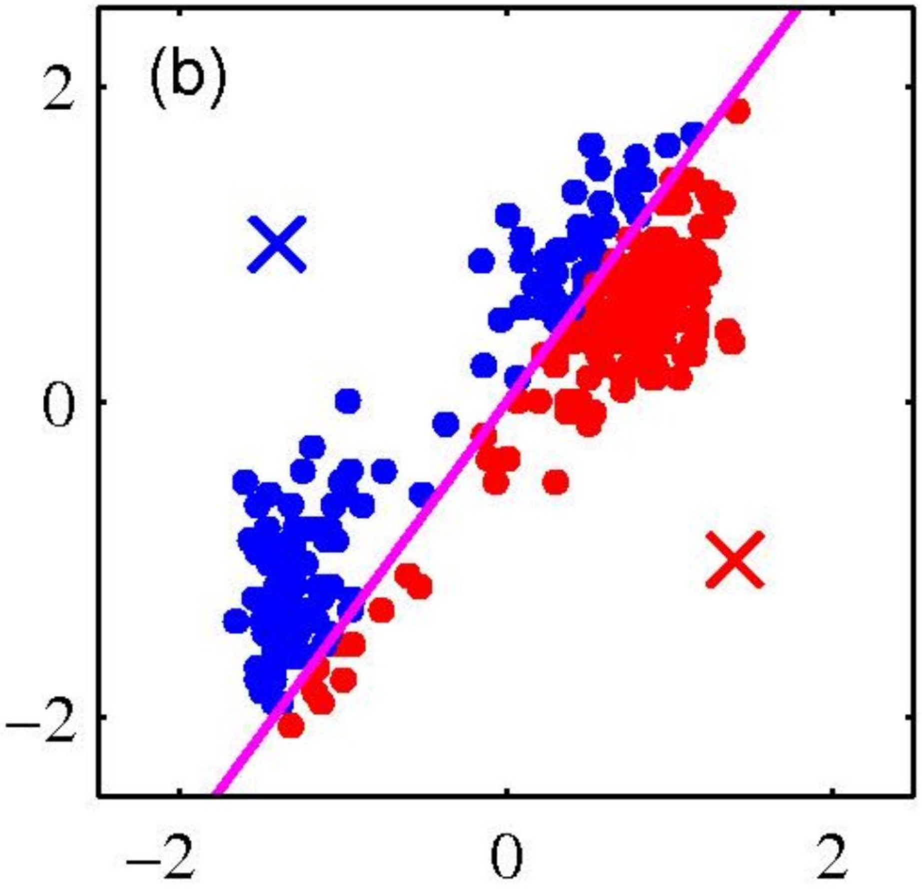
K-Means Example

- Pick K *random points* as cluster *centroids*
- Shown here for $K = 2$



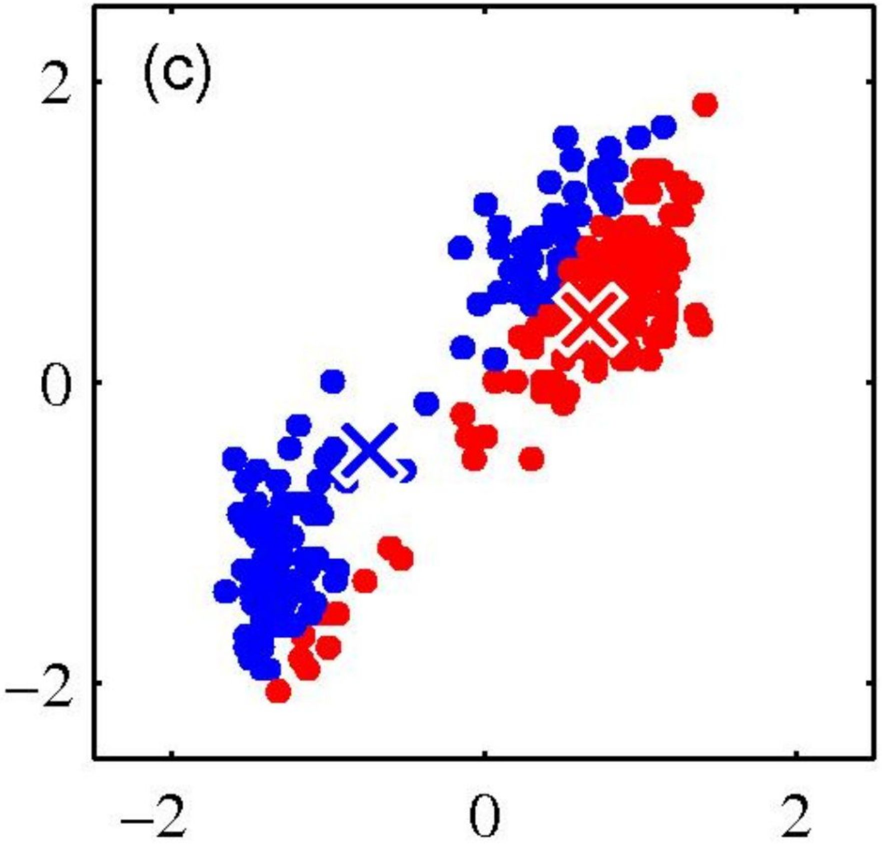
Iteration 1

- Assign data points to *closest centroid*



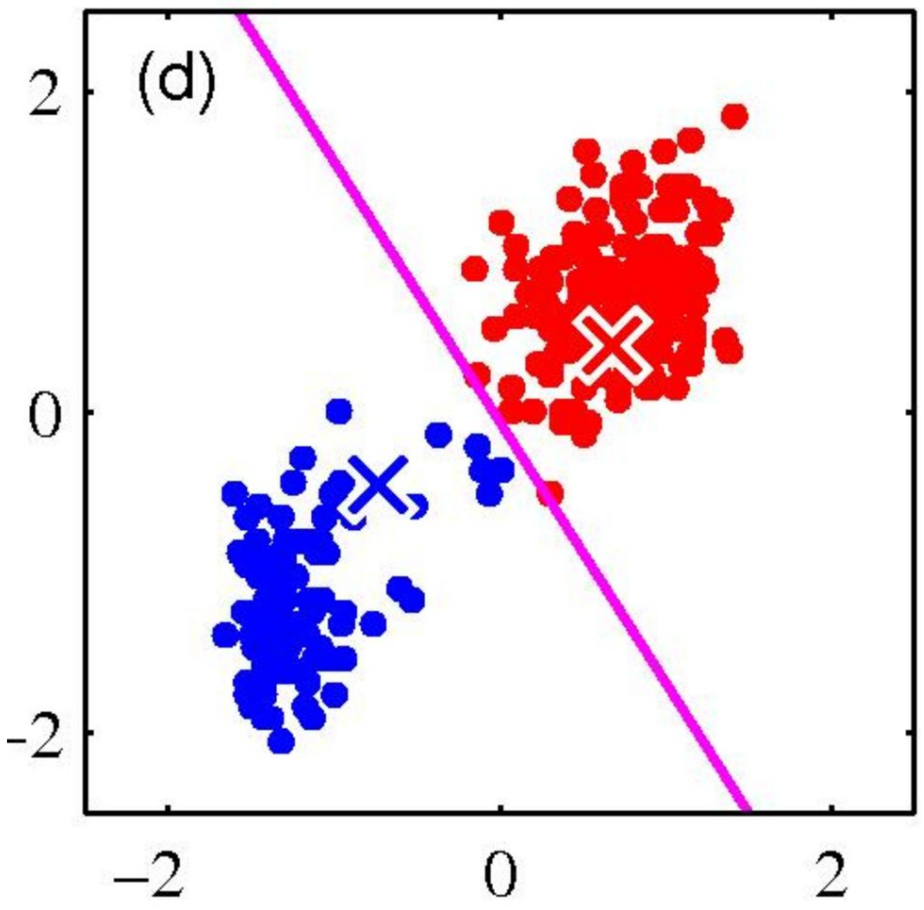
Iteration 1

- Change each *centroid* to the average of the assigned points



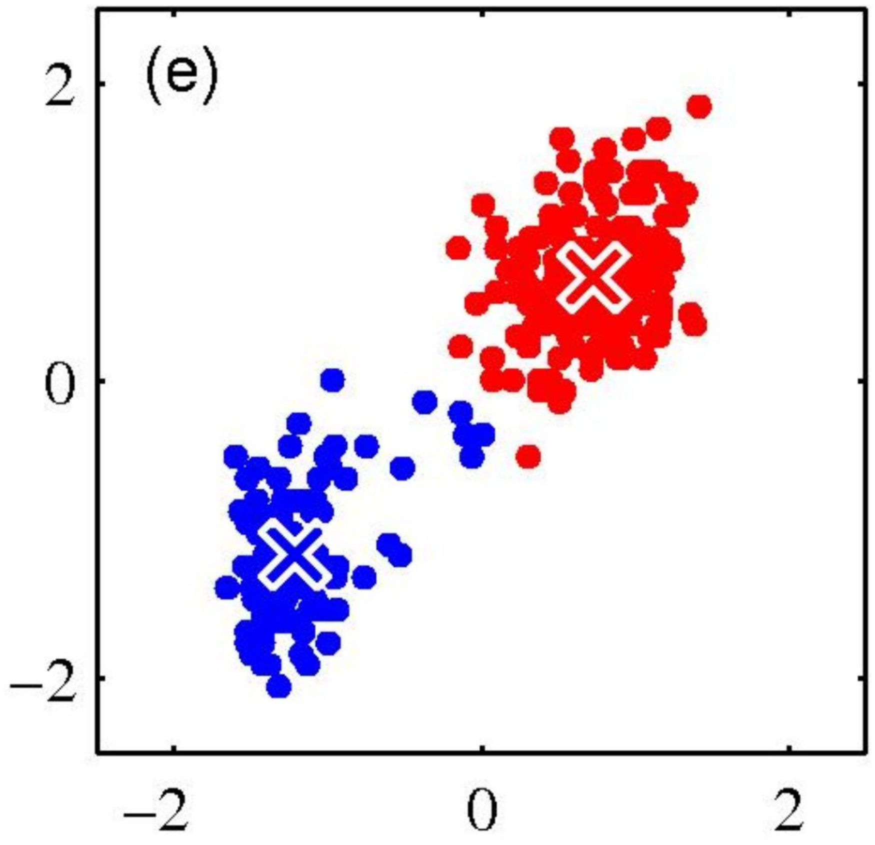
Iteration 2

- Assign data points to *closest centroid*



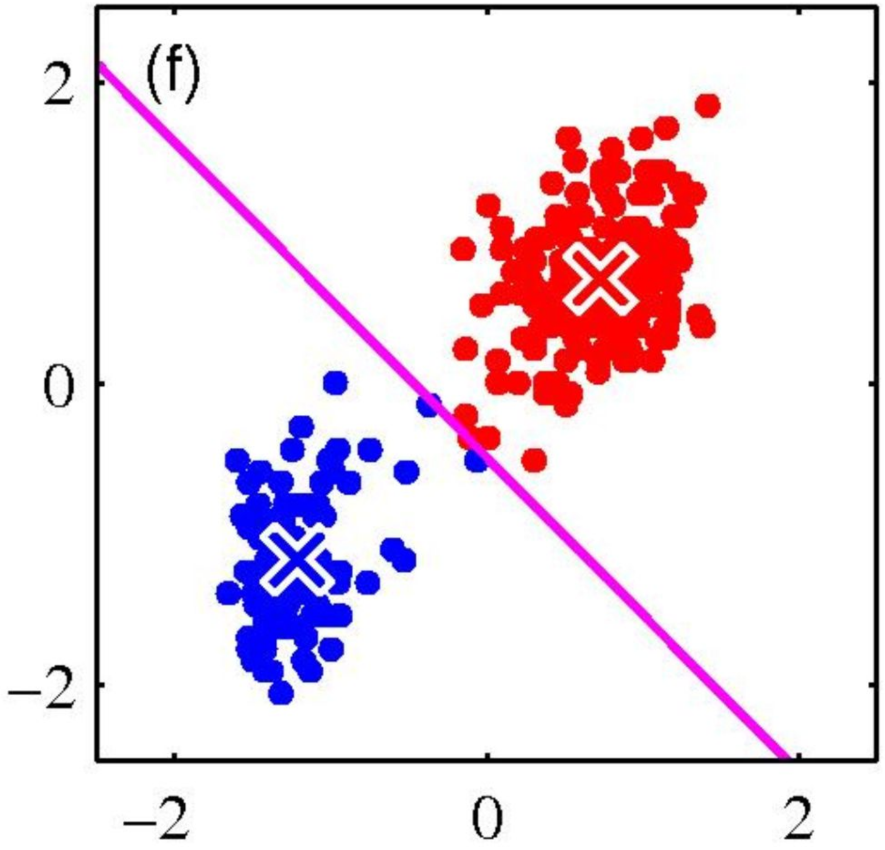
Iteration 2

- Change each *centroid* to the average of the assigned points



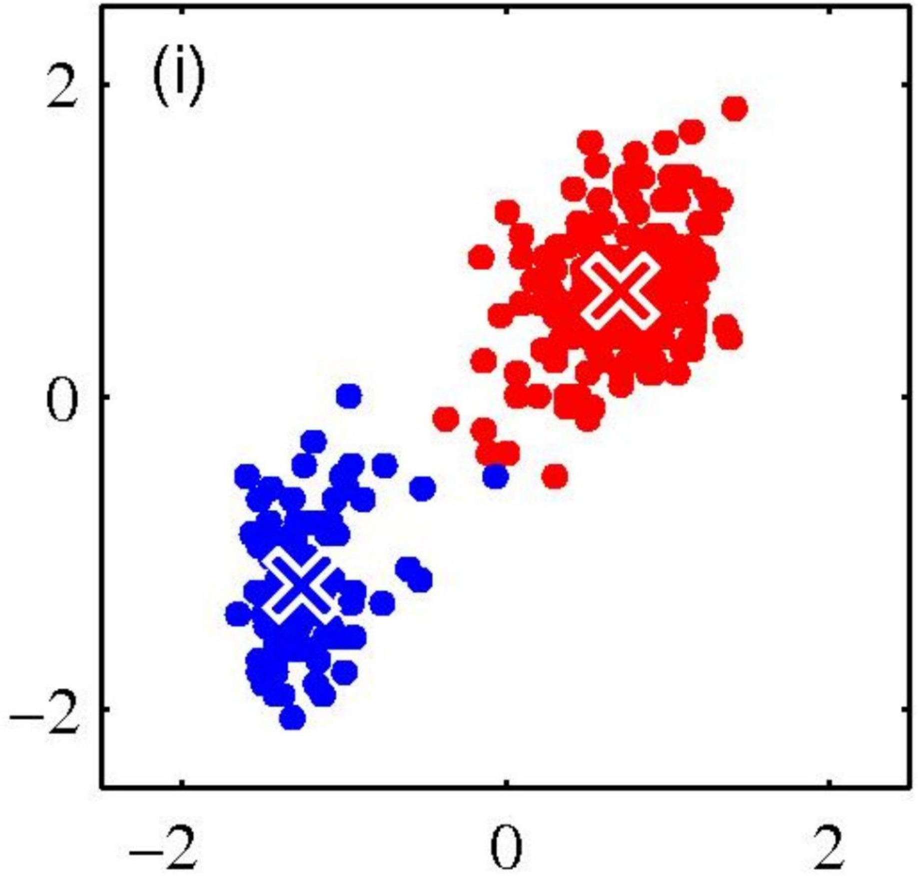
Iteration 3

- Assign data points to *closest centroid*



Iteration 3

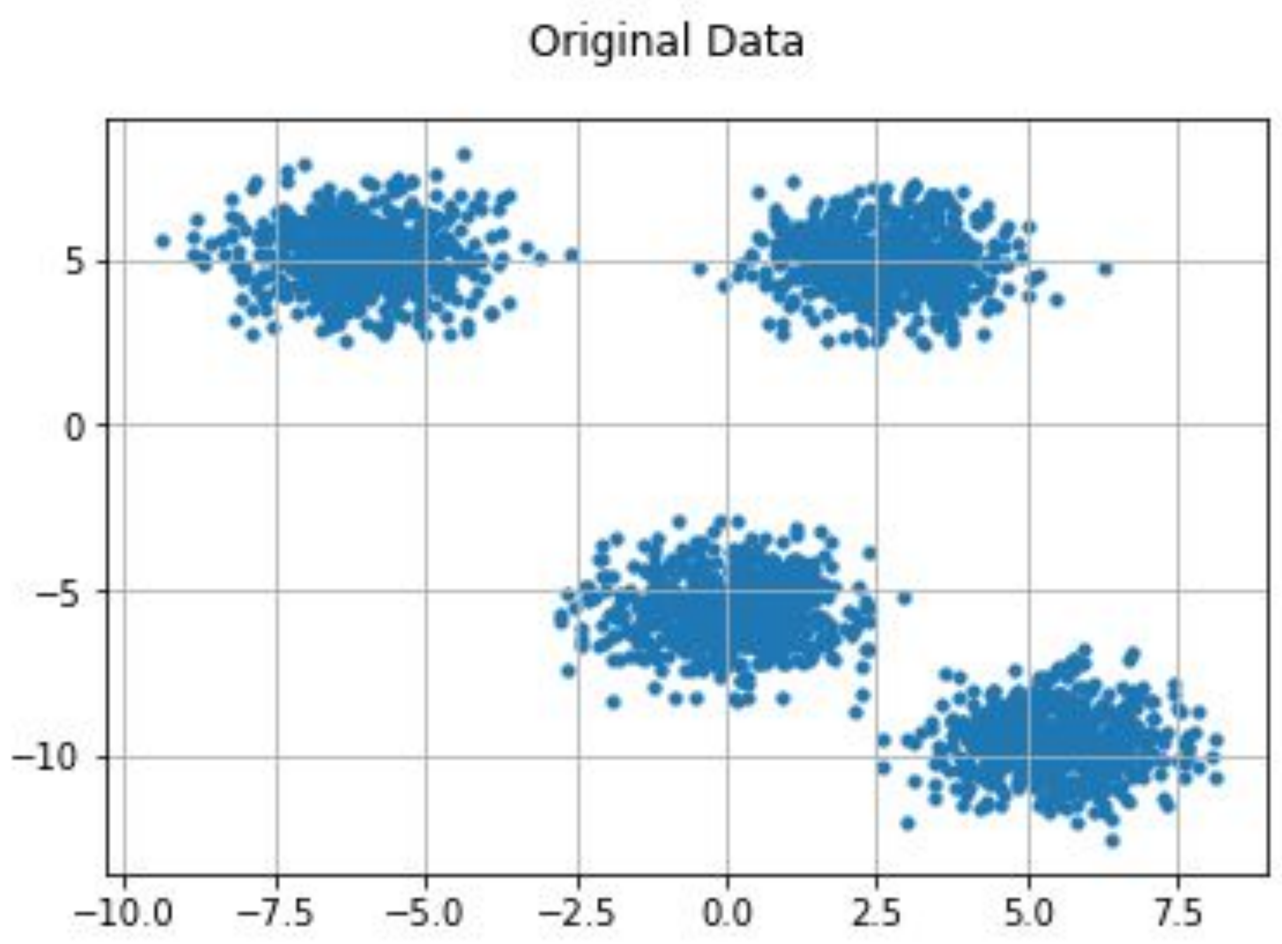
- Change each *centroid* to the average of the assigned points
- *Convergence achieved*!



```
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs
```

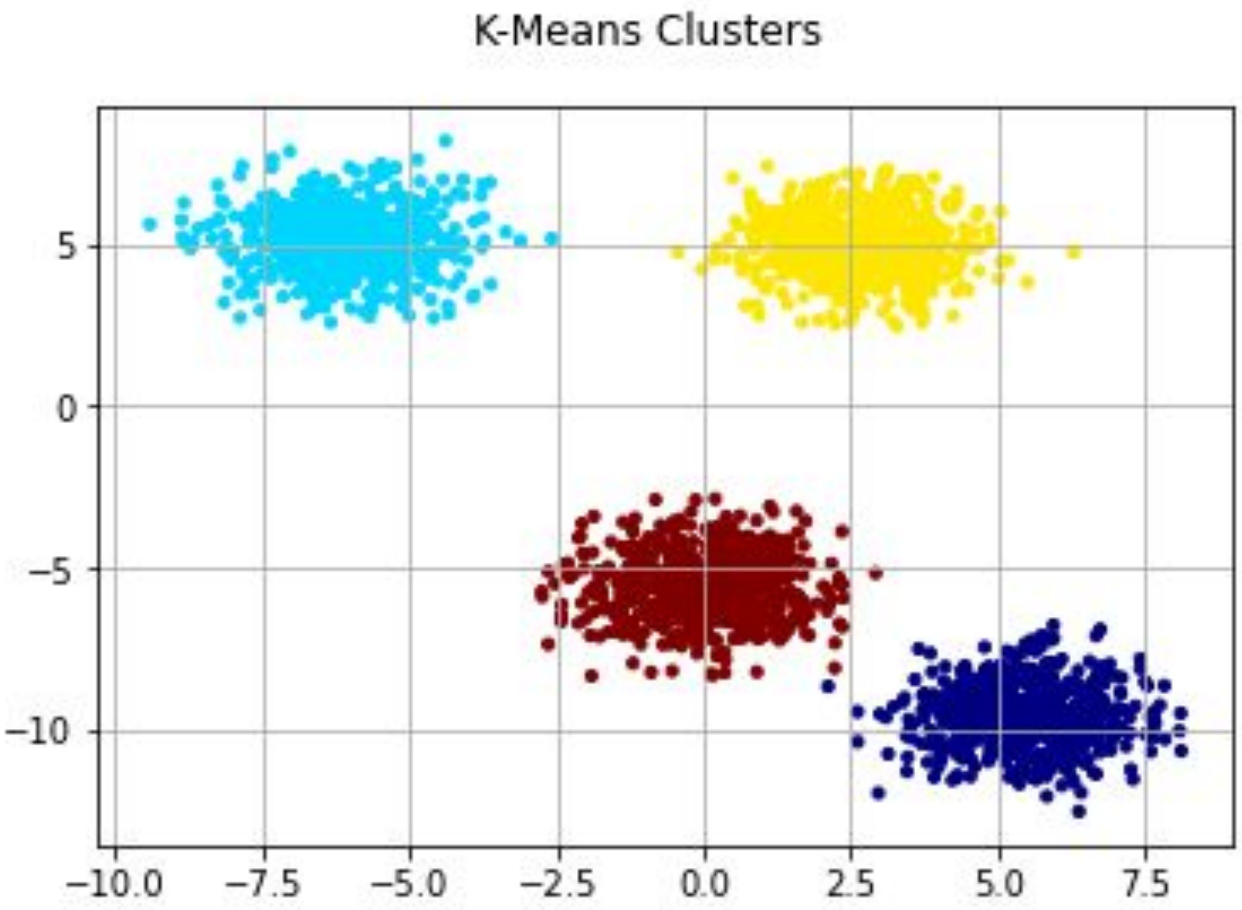
```
X, y = make_blobs(n_samples=2500,centers=4, n_features=2,random_state = 10)
```

```python
plt.figure()
plt.scatter(X[:, 0], X[:, 1], c=None, cmap='jet',s=10)
plt.suptitle('Original Data')
plt.grid(1,which='both')
plt.axis('tight')
plt.show()
```


Original Data
5
0
−5
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5

```
y_pred = KMeans(n_clusters=4, random_state=0).fit_predict(X)
```

```
plt.figure()
plt.scatter(X[:, 0], X[:, 1], c=y_pred, cmap='jet',s=10)
plt.suptitle('K-Means Clusters')
plt.grid(1,which='both')
plt.axis('tight')
plt.show()
```


K-Means Clusters
5
0
−5
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5

Number of clusters

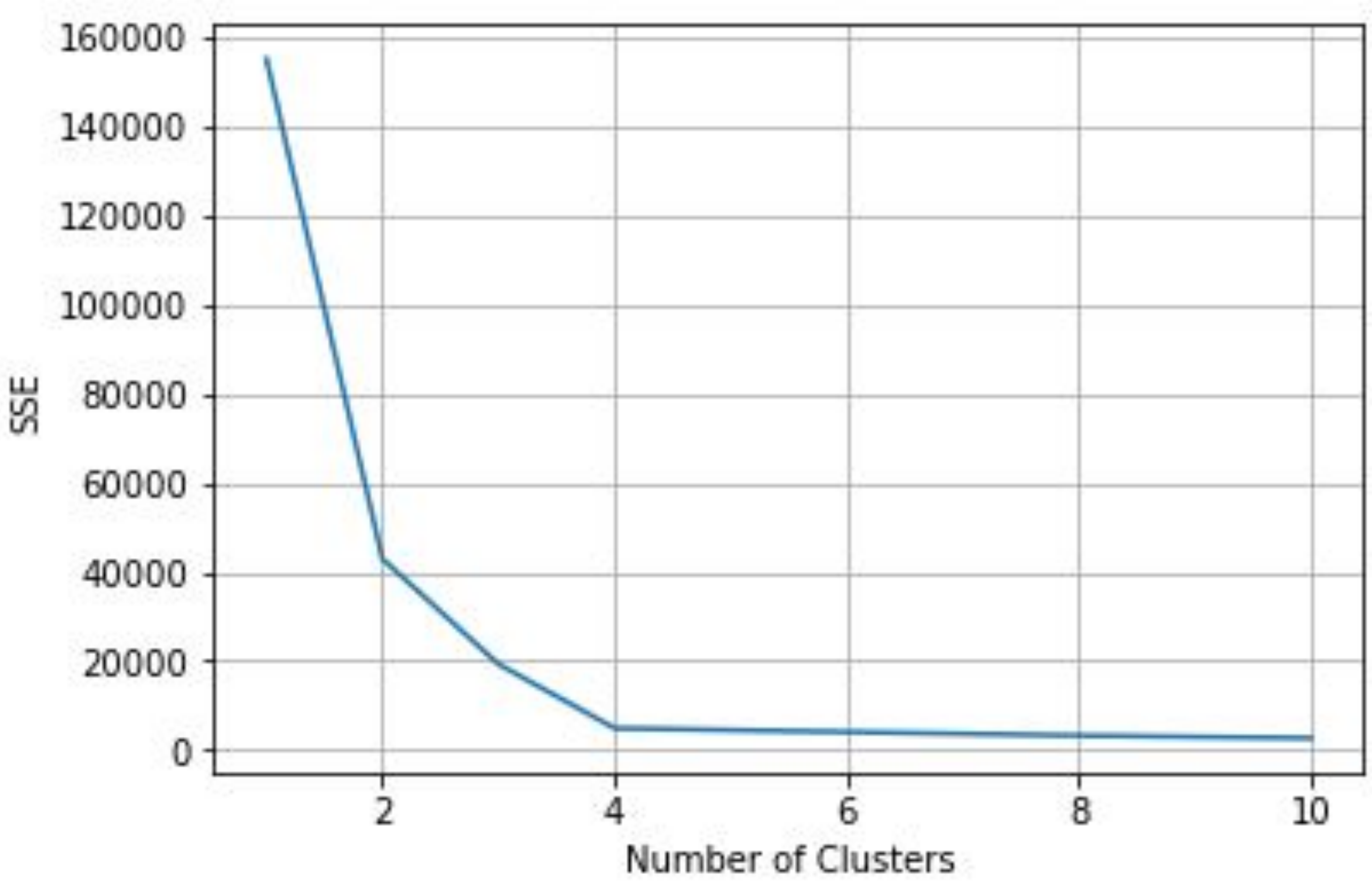
- How to choose number of clusters?

- Use the Sum Square Error (SSE) as a metric
- Also termed inertia

```python
#Sum squared error
SSE = []

for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, random_state=0)
    kmeans.fit(X)
    SSE.append(kmeans.inertia_)
```

```python
plt.plot(range(1, 11), SSE)
plt.xlabel("Number of Clusters")
plt.ylabel("SSE")
plt.grid(1,which='both')
plt.axis('tight')
plt.show()
```



160000
140000
120000
100000
80000
60000
40000
20000
0
SSE
2
4
6
8
10
Number of Clusters

Instructors may use this white area (14.5 cm / 25.4 cm) for the text. Three options provided below for the font size.

Font: Avenir (Book), Size: 32, Colour: Dark Grey

Font: Avenir (Book), Size: 28, Colour: Dark Grey

Font: Avenir (Book), Size: 24, Colour: Dark Grey

Do not use the space below.