Clustering

Filipe Cabral Pinto

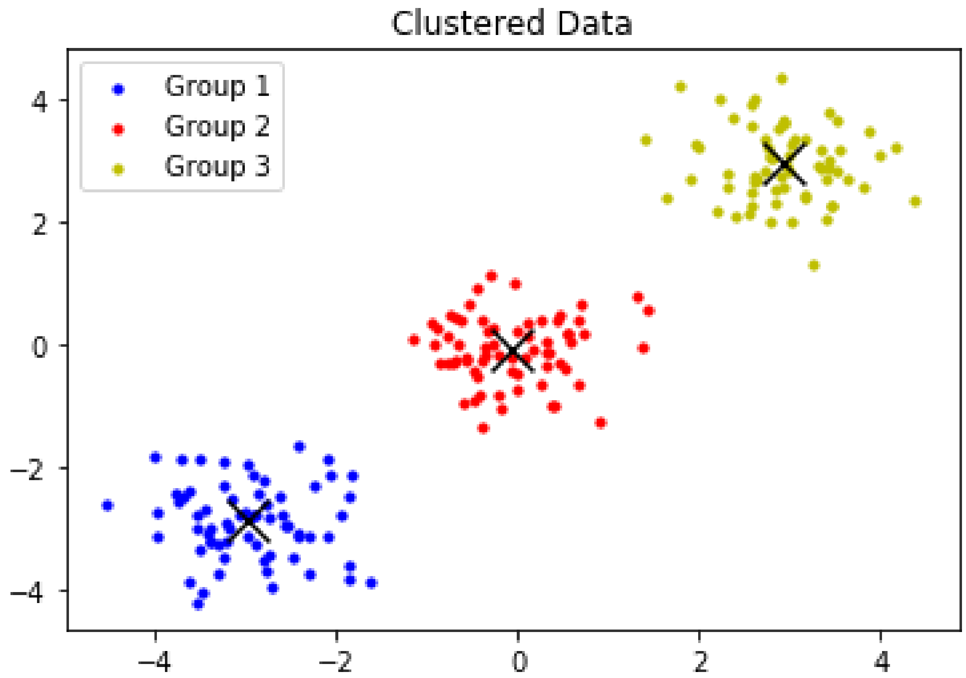
Key concept

- Clustering is an unsupervised machine learning topic intending to find data patterns having no target to predict
- Clustering aims at grouping similar data points in large datasets
  - A cluster should group similar observations
  - Observations with distinguishable characteristics should belong to different clusters
- Similarity can be defined in different ways

Clustering few examples

- Customer segmentation
- Network anomalies
- Fraud detection
- Document Analysis
- Streaming Services
- Sports science

Grouping data points



Clustering approaches

Centroid-based models

Data points are grouped based on the proximity to the centroids.

k-means is the most well-known centroid-based clustering algorithm.

Density-based models

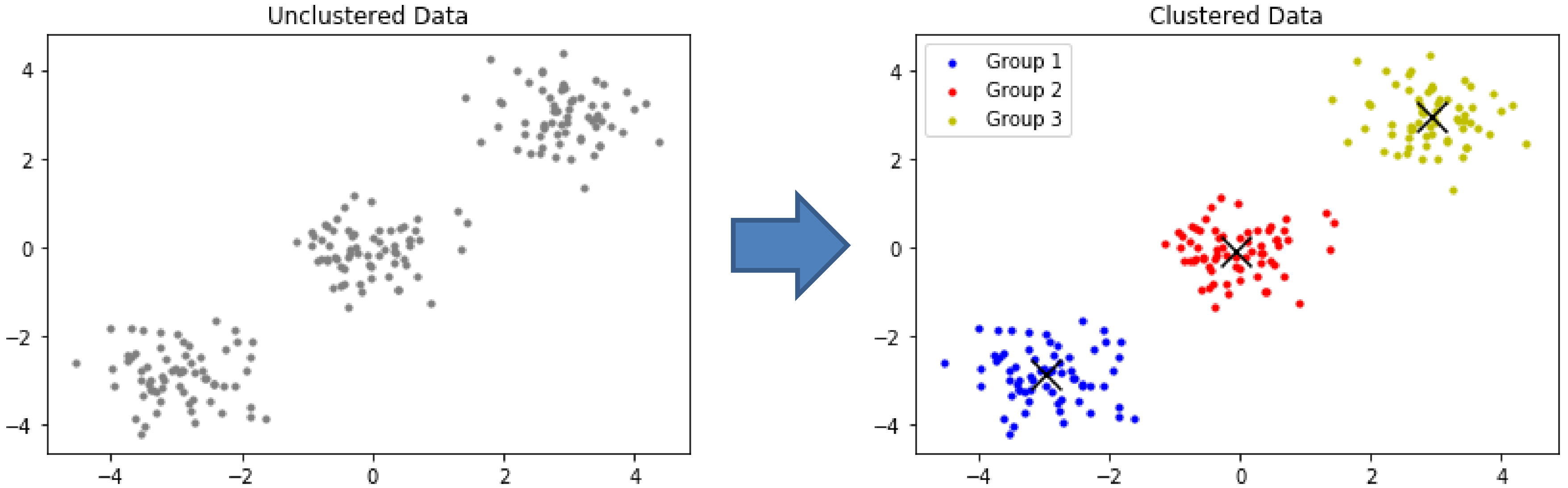
Data points concentrate over contiguous regions, and empty or sparse areas separate clusters.

DBSCAN is probably the most widely-used algorithm.

K-Means algorithm

- K-Means is an unsupervised algorithm aiming to group unlabelled data points into K distinct clusters.
  - A data point belongs to one and only one cluster.
- K-Means finds similar data points and group them together.
- K-Means uses Euclidean distances to measure the proximity.
  - It allocates each data point dynamically to the nearest cluster centred in one of the K centroids identified.

K-Means



Unclustered Data
4
2
0
−2
−4
−4
−2
0
2
4
Clustered Data
Group 1
Group 2
Group 3
4
2
0
−2
−4
−4
−2
0
2
4

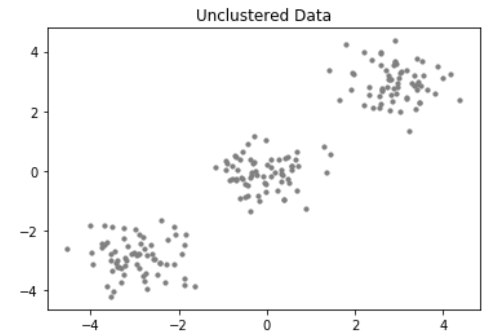
Inputs and outputs

Inputs

Dataset: $S_n = \{x^{(i)}, i = 1 \ldots n\}$

Number of clusters: K

Note: $\mathbf{x}^{(i)}$ = $\mathbf{i}^{\text{th}}$ data point of the dataset



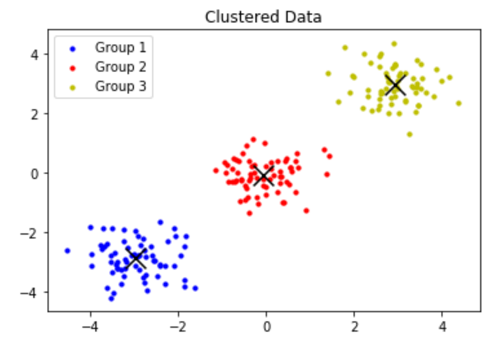
Outputs

Set of partitions: $C_{1,} C_{2,} \ldots, C_k$

$$\bigcup_{k=1}^{K} C_k = S_n$$

$$C_i \bigcap C_j = \emptyset, i \neq j$$

Note: $\mathbf{C_k}$ includes a set of data points



How to group?

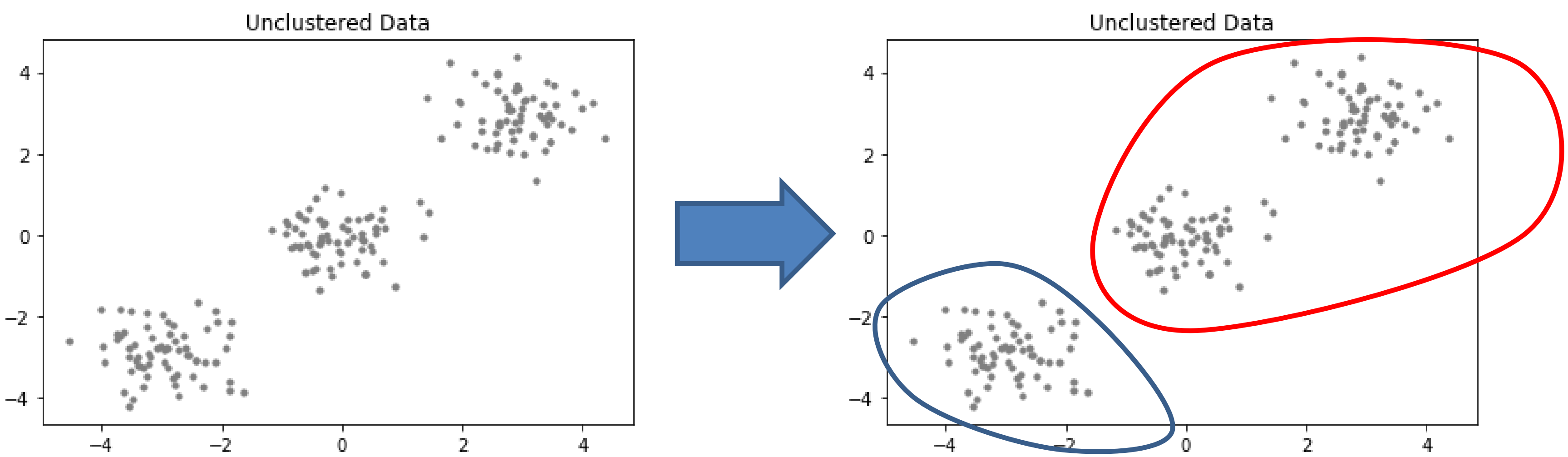
K = 2 => 2 Clusters

Total cost or loss function

Cost of all clusters

$$\text{Cost}(C_1, C_2, \dots, C_k) = \sum_{k=1}^{K} \text{Cost}(C_k)$$

C_k is the k^{th} cluster and includes a set of data points



Cluster cost

$$\mathrm{Cost}\left(C_k, c^{(k)}\right) = \sum_{x^{(i)} \in C_k} \mathrm{distance\_metric}\left(c^{(k)}, x^{(i)}\right)$$

$$\mathrm{Cost}\left(C_k, c^{(k)}\right) = \sum_{x^{(i)} \in C_k} \left\|\left(c^{(k)} - x^{(i)}\right)\right\|^2 = \sum_{x^{(i)} \in C_k} \left(c^{(k)} - x^{(i)}\right)^2$$

C_k is the k^{th} cluster and includes a set of data points

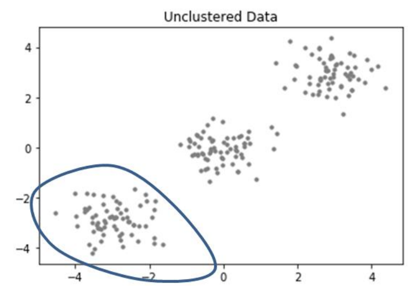
$c^{(k)}$ is the representative centroid of cluster C_k

$x^{(i)}$ is the i^{th} data point of the dataset

Note: $\|c - x\| = \sqrt{(c_1 - x_1)^2 + (c_2 - x_2)^2 + \cdots + (c_m - x_m)^2}$

m is the number of features

Cost of the k cluster



K-Means best partitions iterative steps

$$Cost(C_1, C_2, \dots, C_k, c^{(1)}, \dots, c^{(1)}) = \sum_{k=1}^{K} \sum_{x^{(i)} \in C_k} \left\| c^{(k)} - x^{(i)} \right\|^2$$

Objective: To minimize the **total cost**

Given the representative centroids - $\mathbf{c}^{(1)}, \mathbf{c}^{(2)} \dots, \mathbf{c}^{(K)}$ - allocate each **x** to the closest **c**

$$Cost(c^{(1)}, c^{(2)}, \dots, c^{(K)}) = \sum_{i=1}^{n} \min_{k=1\dots K} \left(\left\| \left(c^{(k)} - x^{(i)} \right) \right\|^2 \right)$$

Use Euclidean distance to select the closest **c**

Given the Clusters - $\mathbf{C_1}, \mathbf{C_2}, \dots, \mathbf{C_K}$ - select the best representatives centroids **c**

That's the tricky part!

$$Cost(C_1, C_2, \dots, C_k) = \min_{c^{(1)}, c^{(2)}, \dots, c^{(K)}} \sum_{k=1}^{K} \sum_{x^{(i)} \in C_k} \left\| c^{(k)} - x^{(i)} \right\|^2$$

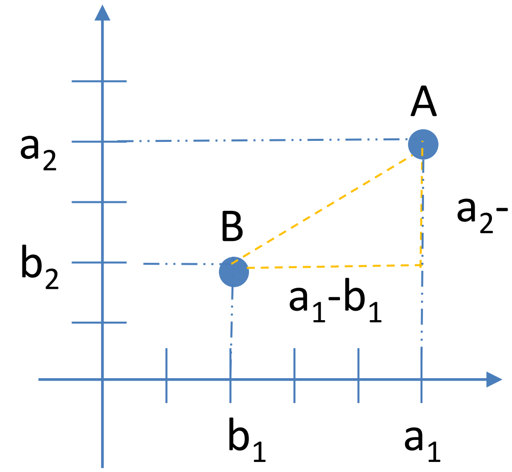
Use the mean to find the centroids

Euclidean distance

The most widely used distance function is the Euclidean distance

$$\text{distance}(A,B) = \sqrt{\sum_{i=1}^{m}(a_i - b_i)^2}$$

$A = (a_1, a_2, \ldots, a_m)$ and $B = (b_1, b_2, \ldots, b_m)$



$$\text{distance}(A,B) = \sqrt{\sum_{i=1}^{m}(a_i - b_i)^2} = \sqrt{(a_2 - b_2)^2 + (a_1 - b_1)^2}$$

Remember Pythagoras Theorem

The mean from K-Means

$$c^{(j)} = \frac{\sum_{x^{(i)} \in C_j} (x^{(i)})}{|C_j|}$$

K-Means uses the mean of each cluster to find the new representative centroids

$$\mathrm{Cost}(C_1, C_2, \dots, C_k) = \min_{c^{(1)}, c^{(2)}, \dots, c^{(K)}} \sum_{k=1}^{K} \sum_{x^{(i)} \in C_k} \left\| c^{(k)} - x^{(i)} \right\|^2$$

Partial derivative

$$\frac{\partial C}{\partial c^{(j)}} = \frac{\partial}{\partial c^{(j)}} \left(\sum_{k=1}^{K} \sum_{x^{(i)} \in C_k} \left\| c^{(k)} - x^{(i)} \right\|^2 \right) \Rightarrow \frac{\partial C}{\partial c^{(j)}} = \frac{\partial}{\partial c^{(j)}} \left(\sum_{k=1}^{K} \sum_{x^{(i)} \in C_k} \left(c^{(k)} - x^{(i)} \right)^2 \right)$$

$$\frac{\partial C}{\partial c^{(j)}} = \sum_{x^{(i)} \in C_j} \frac{\partial}{\partial c^{(j)}} \left(c^{(j)} - x^{(i)} \right)^2 \Rightarrow \frac{\partial C}{\partial c^{(j)}} = \sum_{x^{(i)} \in C_j} 2.\left(c^{(j)} - x^{(i)} \right)$$

The mean from K-Means

$$c^{(j)} = \frac{\sum_{x^{(i)} \in C_j} (x^{(i)})}{|C_j|}$$

K-Means uses the mean of each cluster to find the new representative centroids

$$\sum_{x^{(i)} \in C_j} 2.\left(c^{(j)} - x^{(i)}\right) = 0 \Rightarrow \sum_{x^{(i)} \in C_j} \left(c^{(j)} - x^{(i)}\right) = 0$$

… continuation

$$\sum_{x^{(i)} \in C_j} \left(c^{(j)}\right) - \sum_{x^{(i)} \in C_j} \left(x^{(i)}\right) = 0 \Rightarrow c^{(j)}. \sum_{x^{(i)} \in C_j} (1) - \sum_{x^{(i)} \in C_j} \left(x^{(i)}\right) = 0$$

$$c^{(j)}.\left|C_j\right| = \sum_{x^{(i)} \in C_j} \left(x^{(i)}\right) = 0 \Rightarrow \boxed{c^{(j)} = \frac{\sum_{x^{(i)} \in C_j} \left(x^{(i)}\right)}{\left|C_j\right|}}$$

The new representative centroid of a specific cluster is the mean of all data points in that cluster

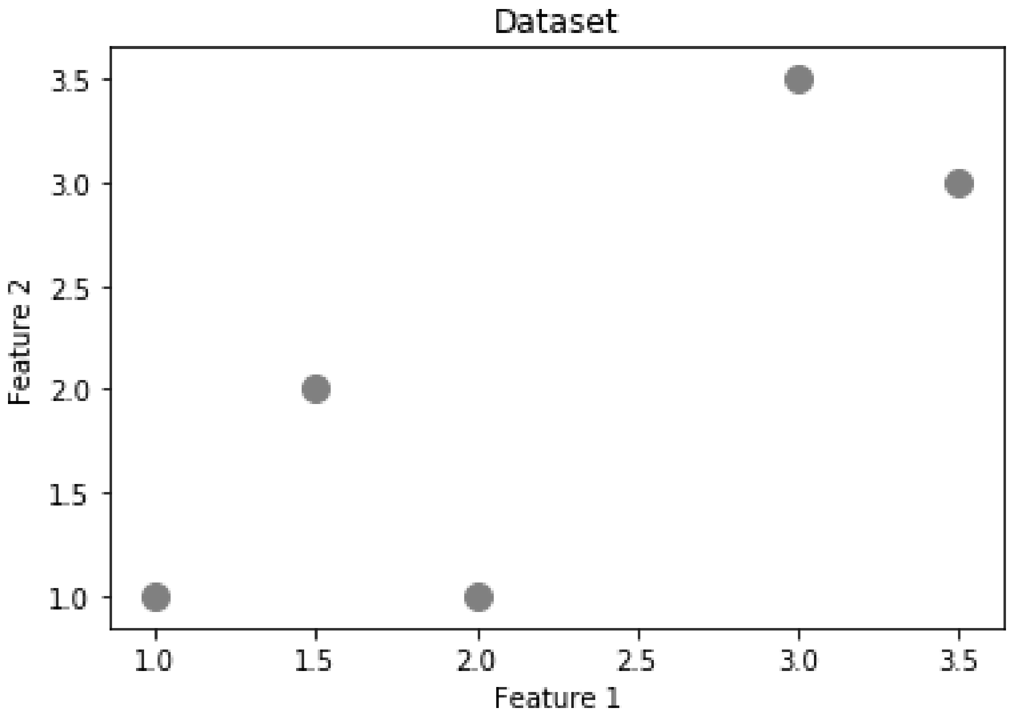
Note: $\left|C_j\right|$ is the cardinality of the set of C_j

K-Means steps

- Consider K centroids randomly
  - It is typical to select K random points from the dataset

- For each data point, check the distances to the centroids
  - Use Euclidean distance
- Associate each point with the closest centroid
  - Centroids are the cluster representatives
- Reassign the centroids positions
  - Set centroids to the mean of each cluster
- Repeat the process until there are no more changes in the clusters.

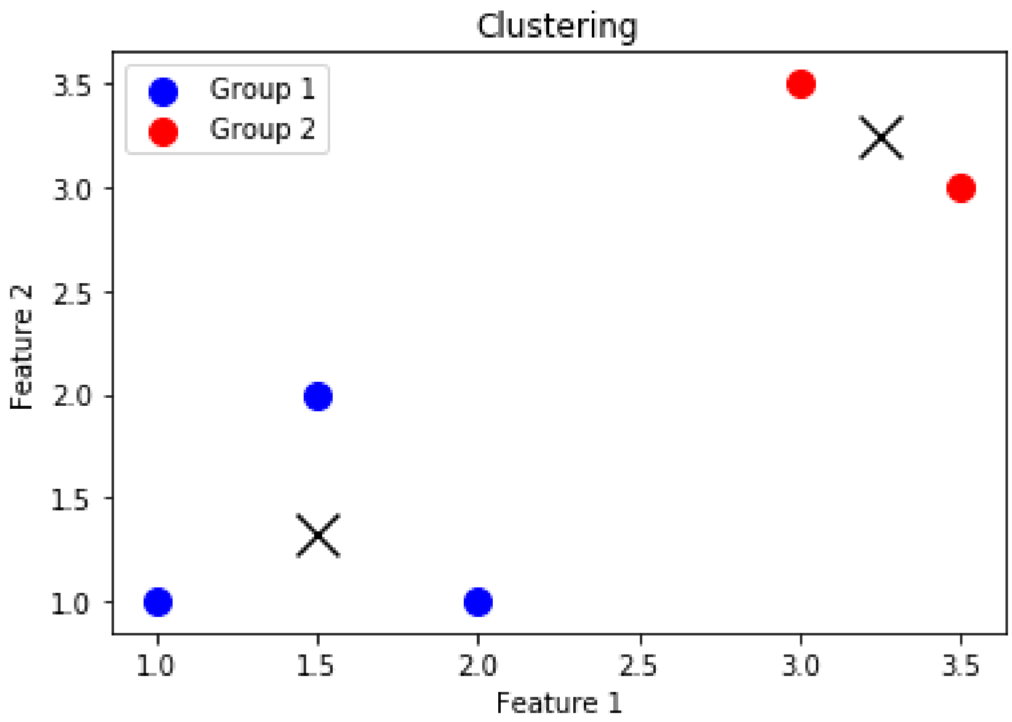
Dataset



	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

K = 2

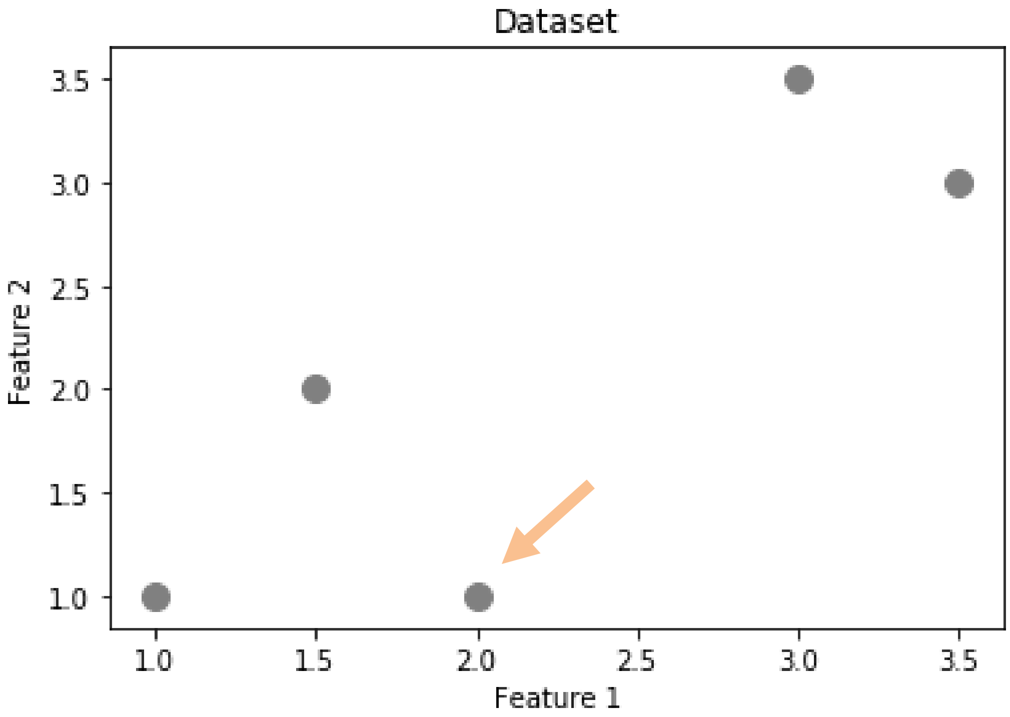
Target



	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

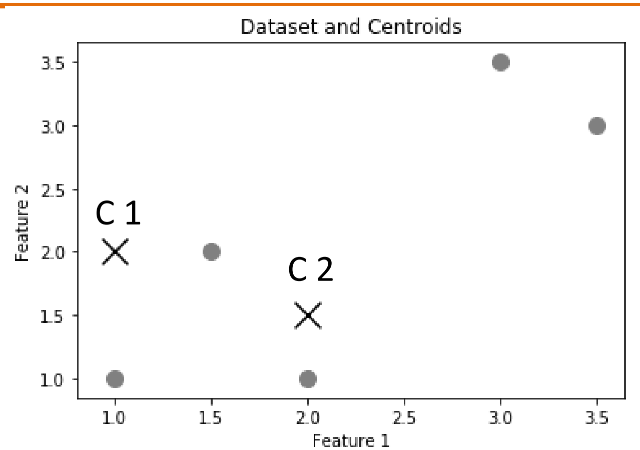
K = 2

Point 1



	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

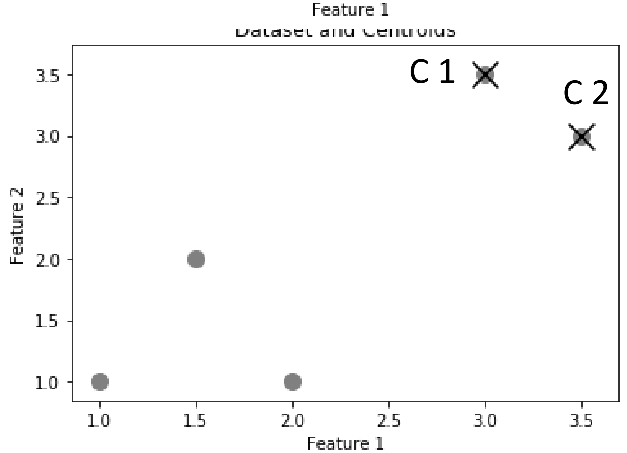
Centroids



There are different initialization methods with varying levels of complexity.

	Feature 1	Feature 2
C 1	1.0	2.0
C 2	2.0	1.5

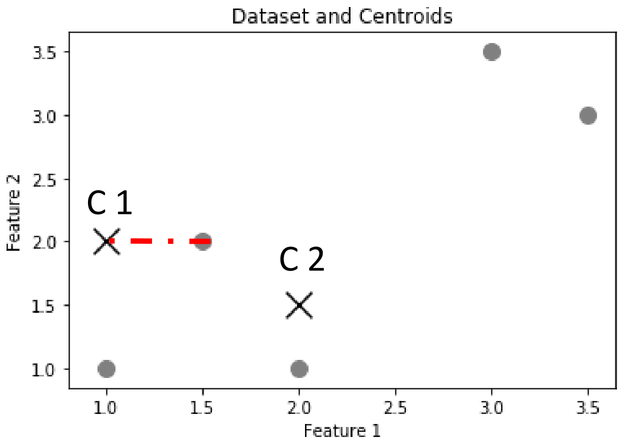
Random selection from the data space is an inefficient possibility

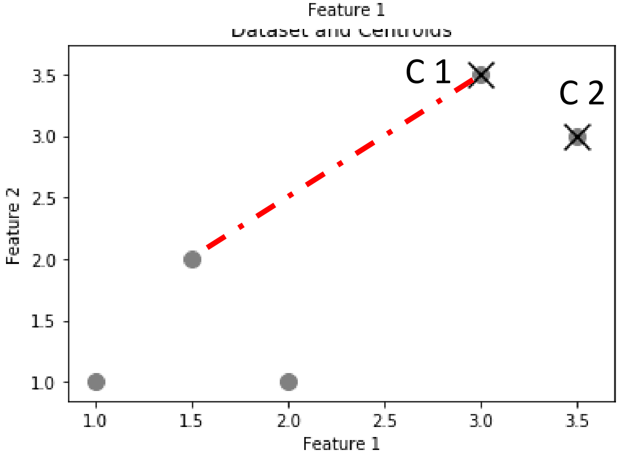


	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

Random selection from the dataset is the common one

Distance between points and centroids





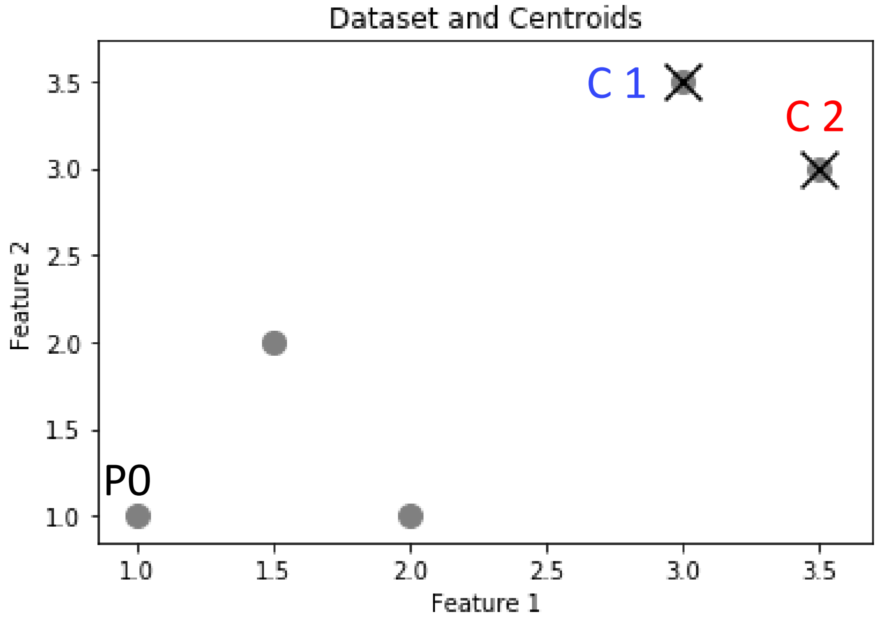
Different metrics can be applied

	Feature 1	Feature 2
C 1	1.0	2.0
C 2	2.0	1.5

	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

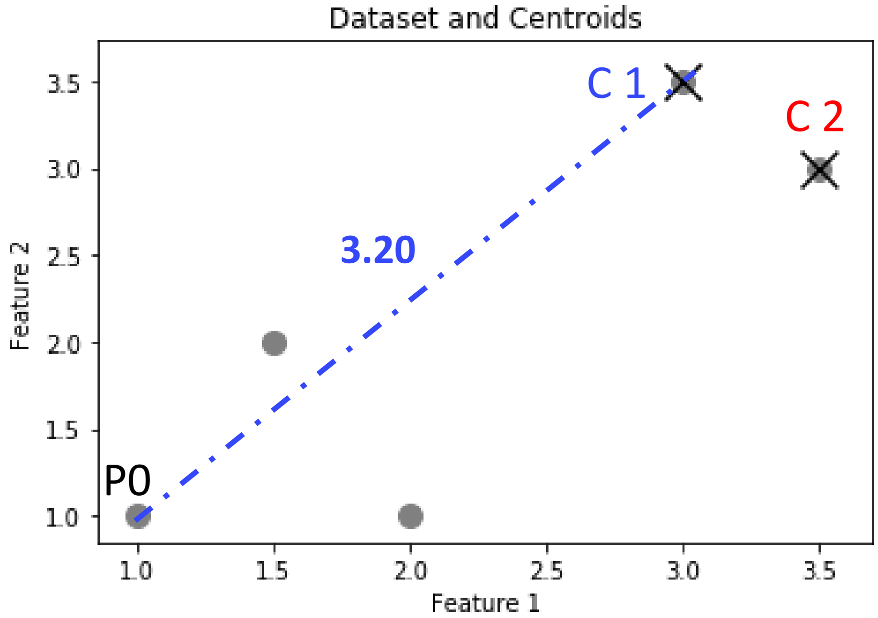
Dataset and centroids



	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

Point 0 to C1

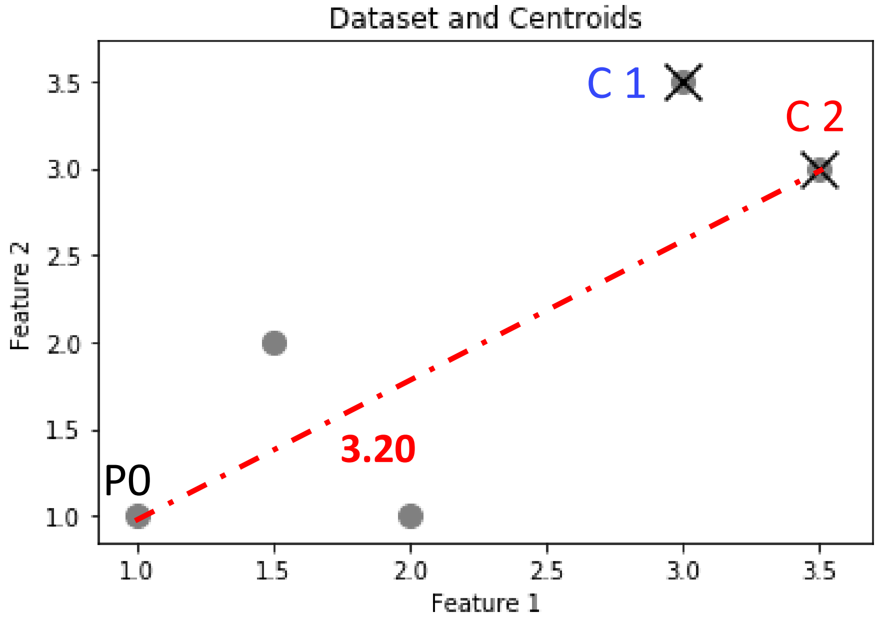


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$D\ (P_0, C_1) = \sqrt{(1-3)^2 + (1-3.5)^2} = 3.20$$

Point 0 to C2

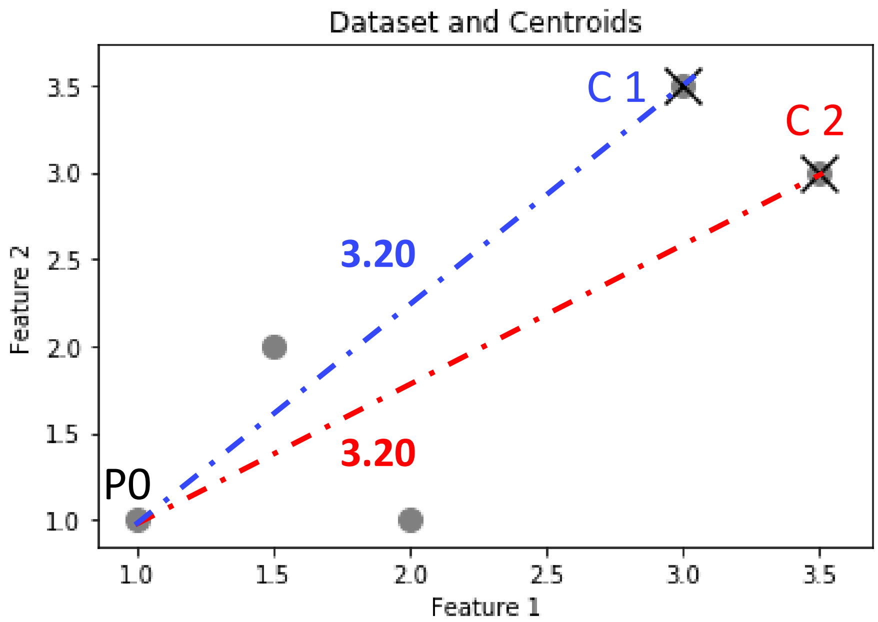


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$D\ (P_0, C_2) = \sqrt{(1-3.5)^2 + (1-3)^2} = 3.20$$

Grouping Point 0

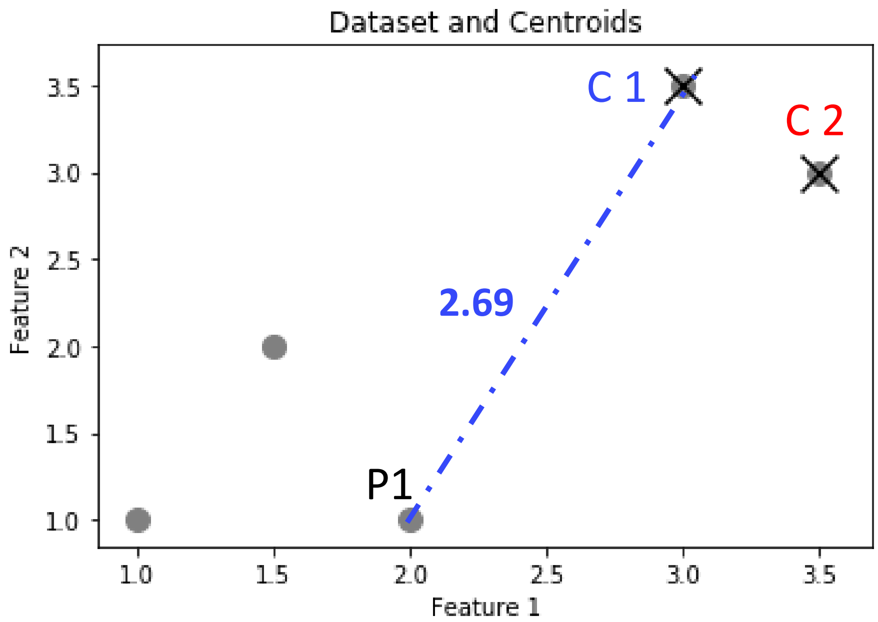


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

Point 0 is the same distance from C1 and C2, so assume it belongs to Cluster 1

Point 1 to C1

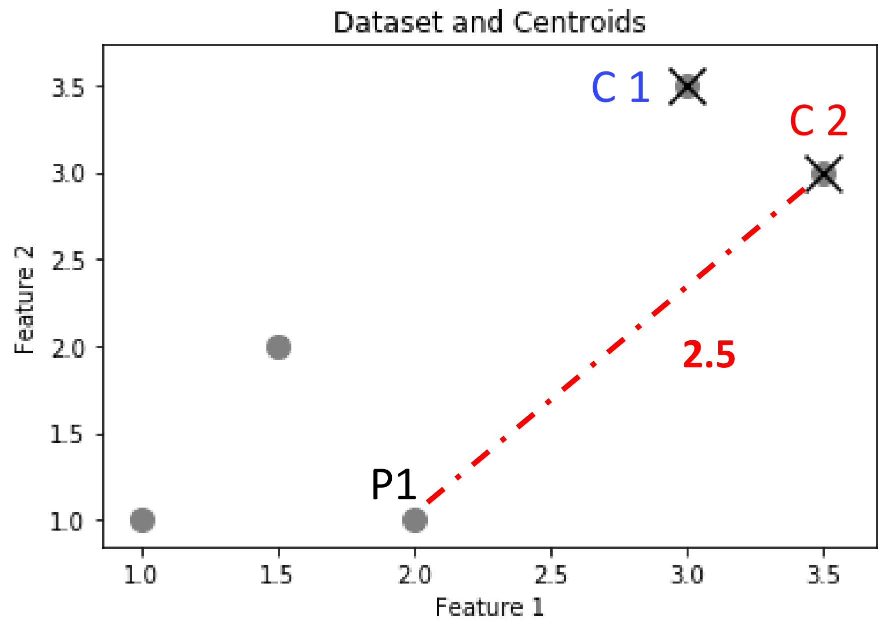


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$D\,(P_1, C_1) = \sqrt{(2-3)^2 + (1-3.5)^2} = 2.69$$

Point 1 to C2

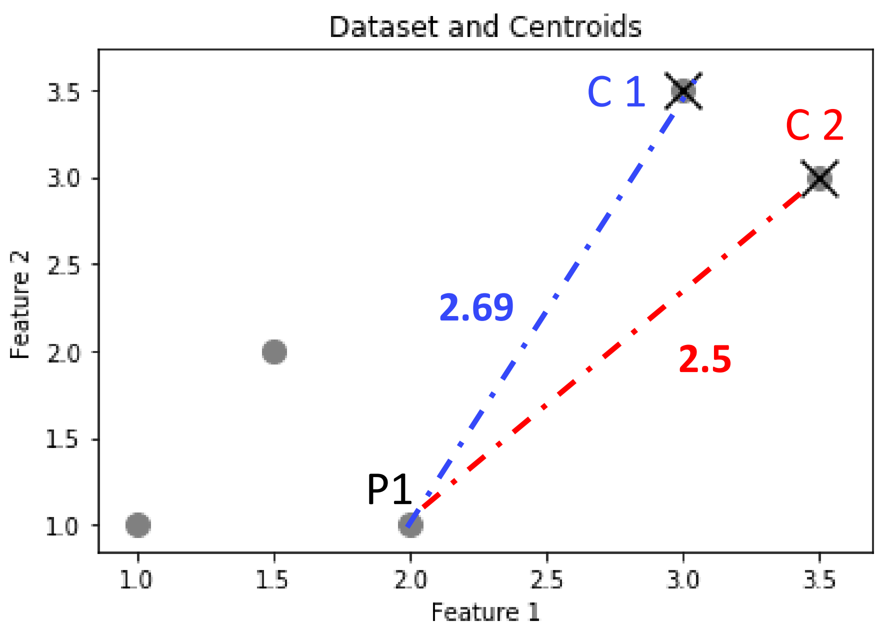


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$D\,(P_1, C_2) = \sqrt{(2-3.5)^2 + (1-3)^2} = 2.5$$

Grouping Point 1

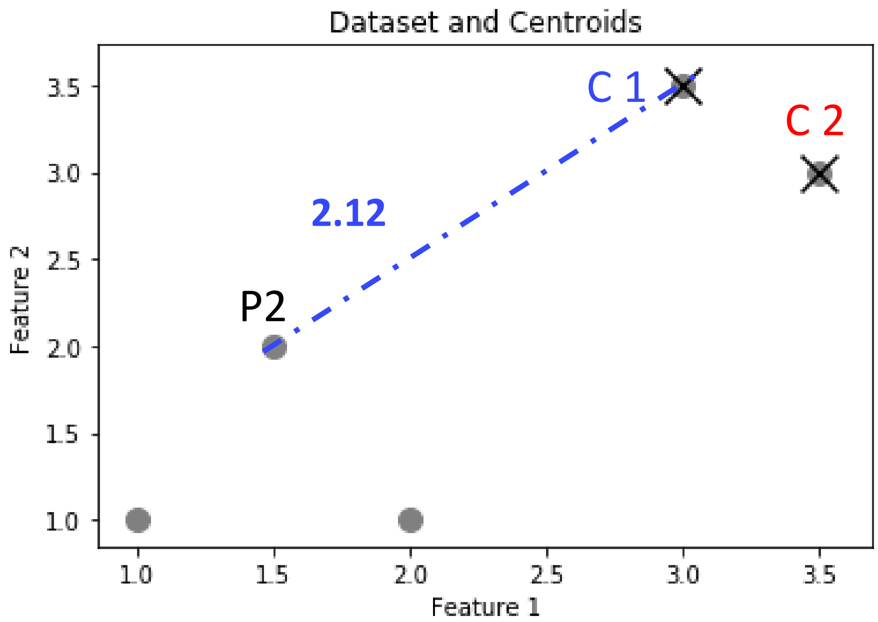


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

Point 1 is closer to C2 so it belongs to Cluster 2

Point 2 to C1

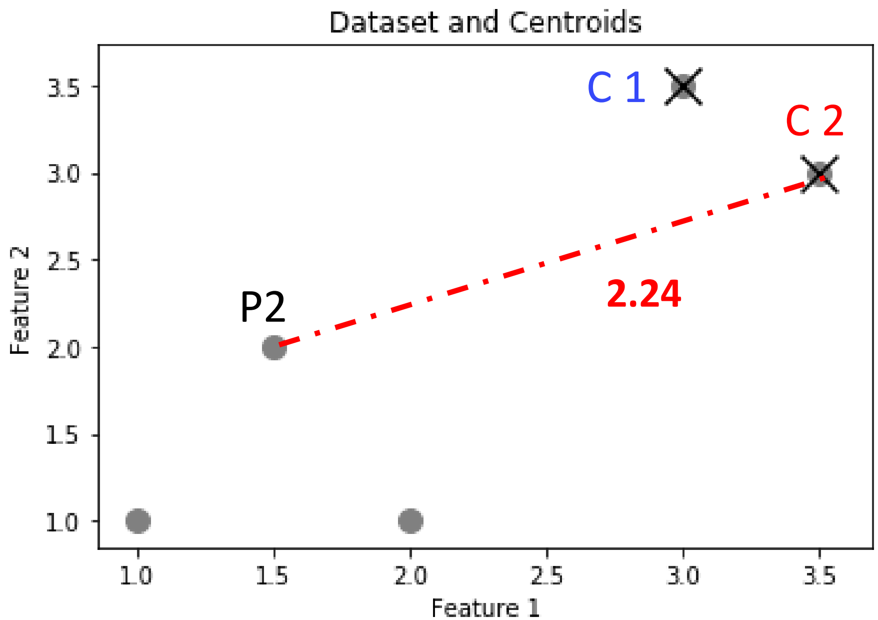


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$D\,(P_2, C_1) = \sqrt{(1.5-3)^2 + (2-3.5)^2} = 2.12$$

Point 2 to C2

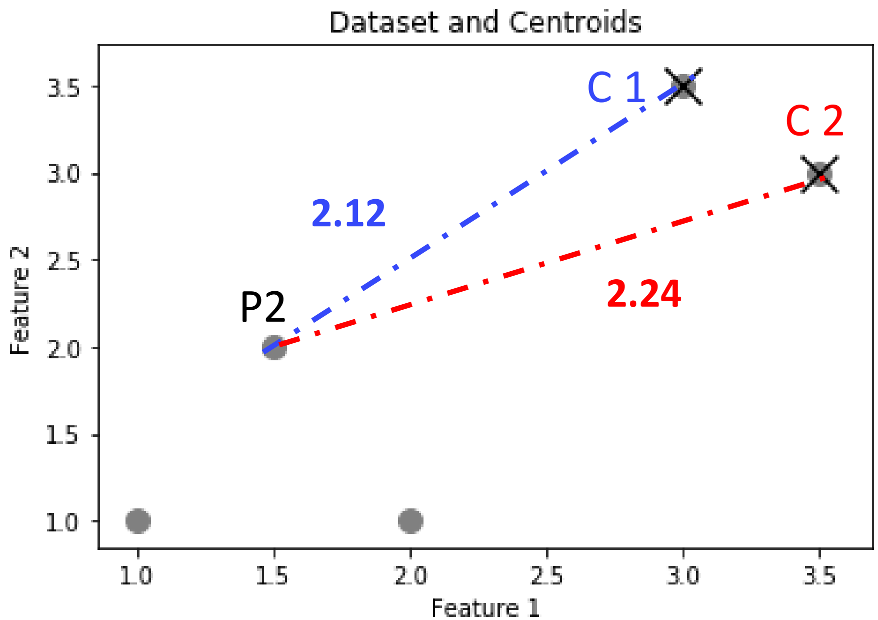


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$\mathrm{D}\,(\mathrm{P}_2, \mathrm{C}_2) = \sqrt{(1.5 - 3.5)^2 + (2 - 3)^2} = 2.24$$

Grouping Point 2

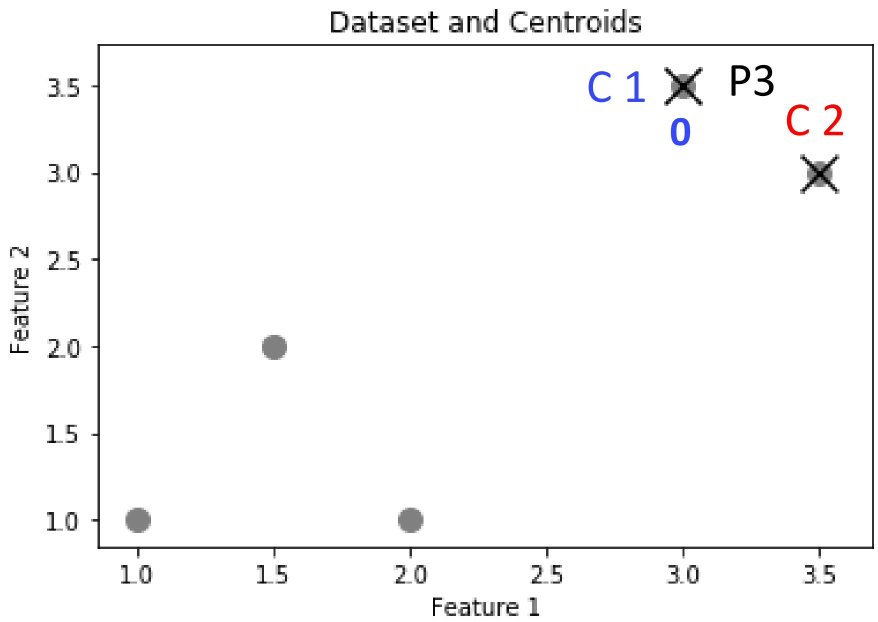


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

Point 2 is closer to C1 so it belongs to Cluster 1

Point 3 to C1

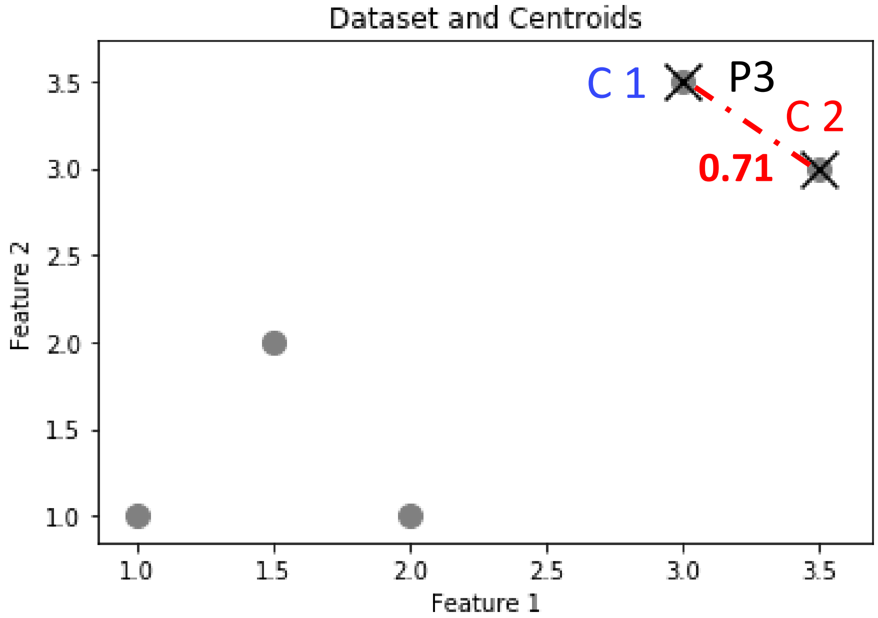


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$D\ (P_3, C_1) = \sqrt{(3-3)^2 + (3.5-3.5)^2} = 0$$

Point 3 to C2

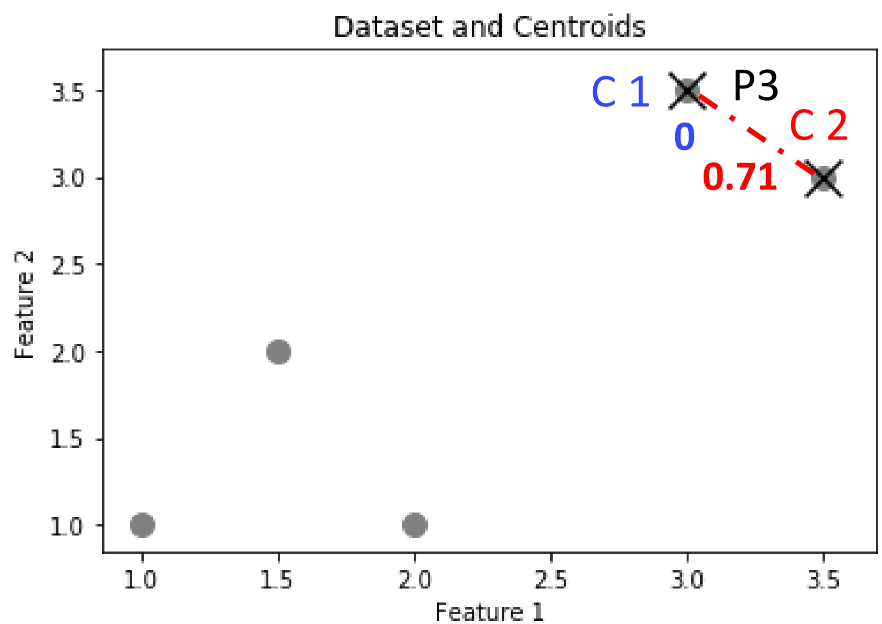


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$D\,(P_3, C_2) = \sqrt{(3-3.5)^2 + (3.5-3)^2} = 0.71$$

Grouping Point 3

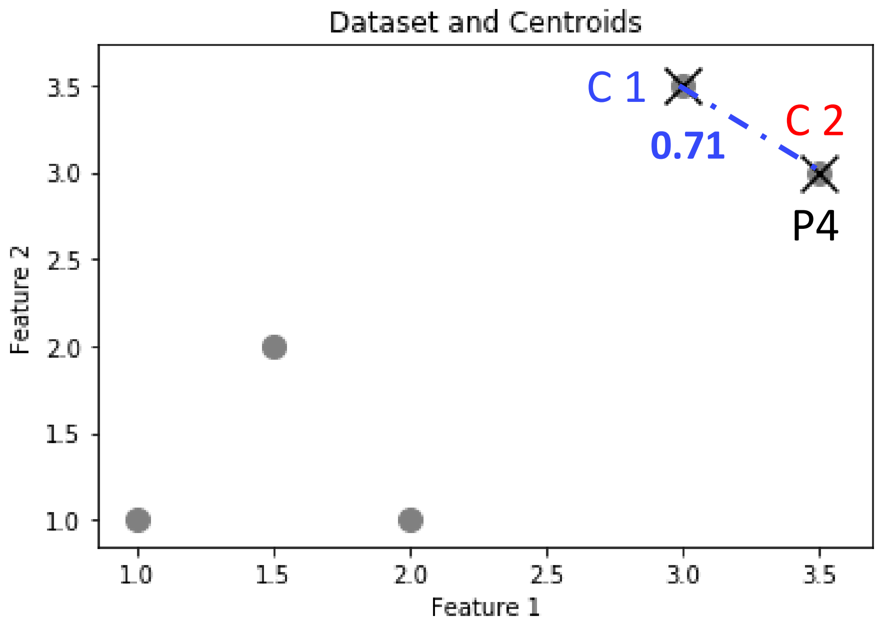


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

Point 3 is closer to C1 so it belongs to Cluster 1

Point 4 to C1

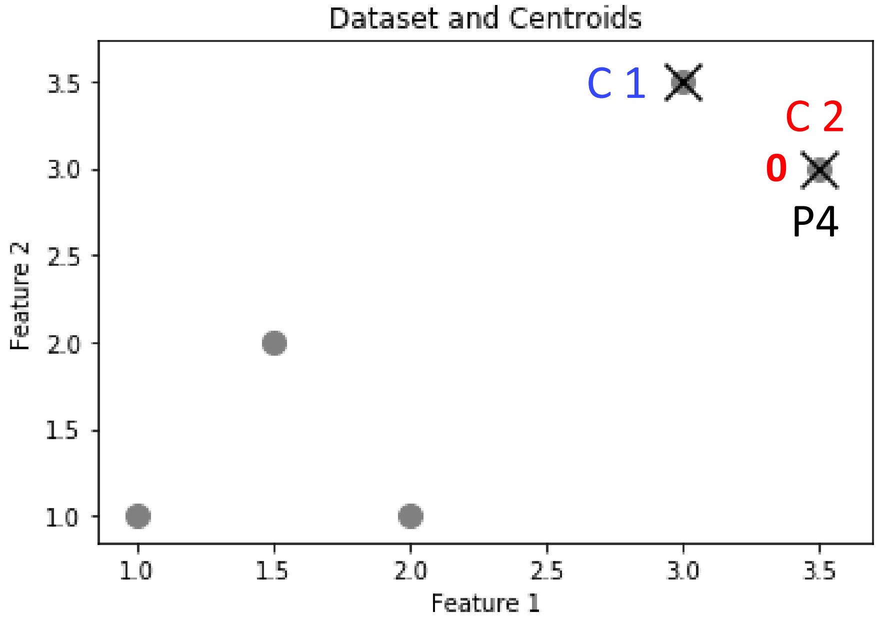


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$D\ (P_4, C_1) = \sqrt{(3.5-3)^2 + (3-3.5)^2} = 0.71$$

Point 4 to C2

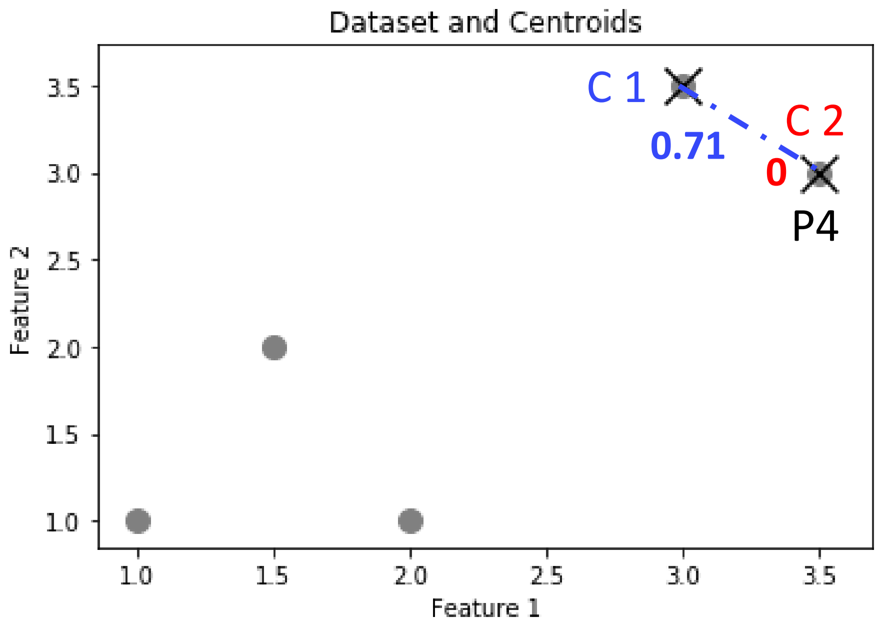


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

$$D\,(P_4, C_2) = \sqrt{(3.5-3.5)^2 + (3-3)^2} = 0$$

Grouping Point 4

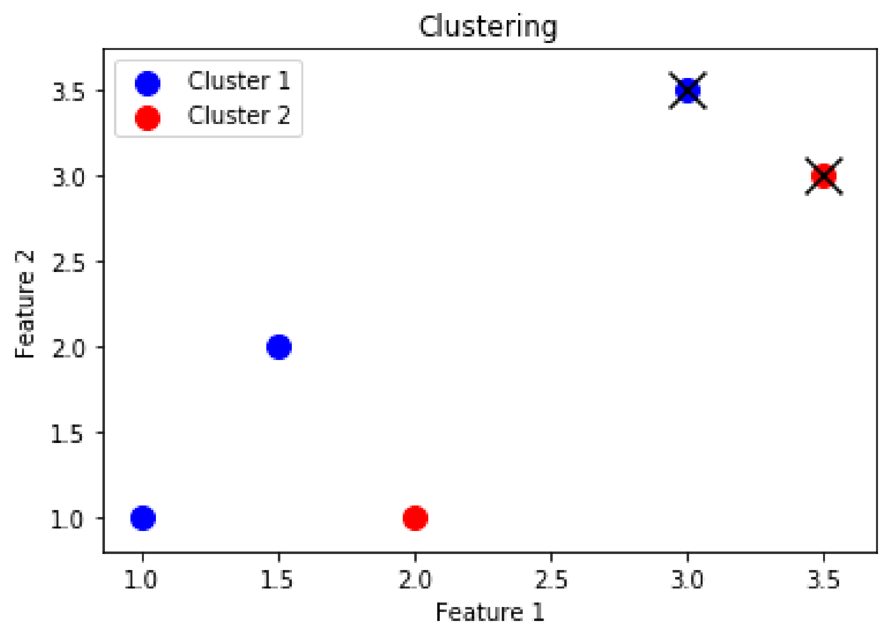


	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

Point 4 is closer to C2 so it belongs to Cluster 2

Data clustering after 1st iteration



	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

Remember: K = 2

Centroids reassignment

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

C 1

	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

C 2

	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

New centroids

	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

$$\text{New C }1\_\mathbf{F1} = \frac{1 + 1.5 + 3}{3} = 1.83$$

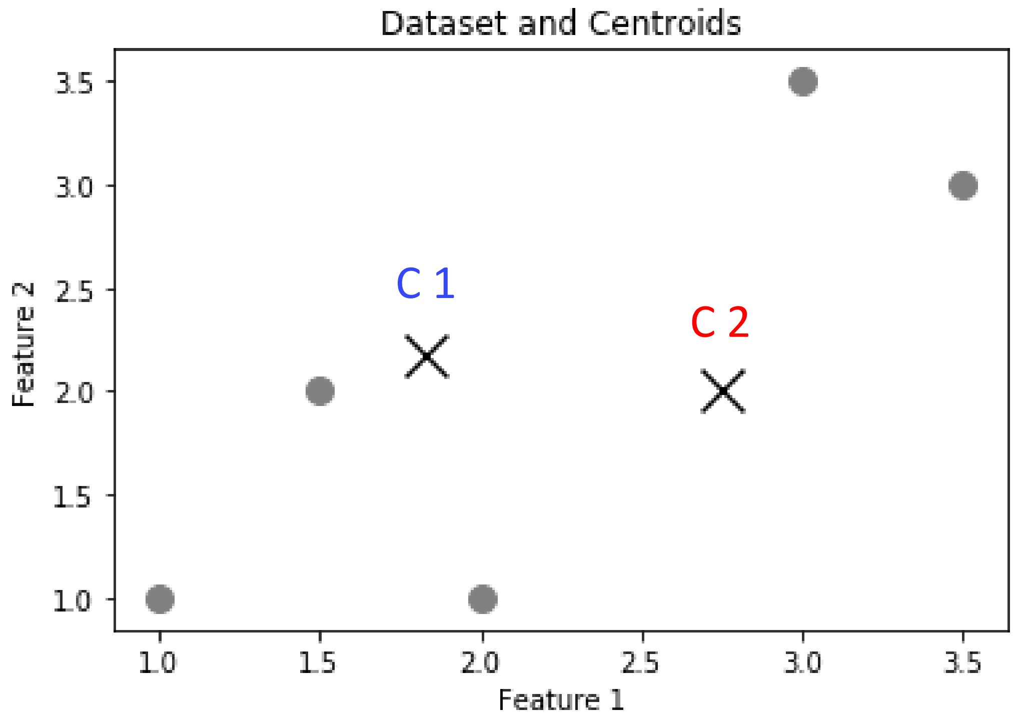
$$\text{New C }1\_\mathbf{F2} = \frac{1 + 2 + 3.5}{3} = 2.17$$

	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

$$\text{New C }2\_\mathbf{F1} = \frac{2 + 3.5}{2} = 2.75$$

$$\text{New C }2\_\mathbf{F2} = \frac{1 + 3}{2} = 2$$

Dataset and new centroids



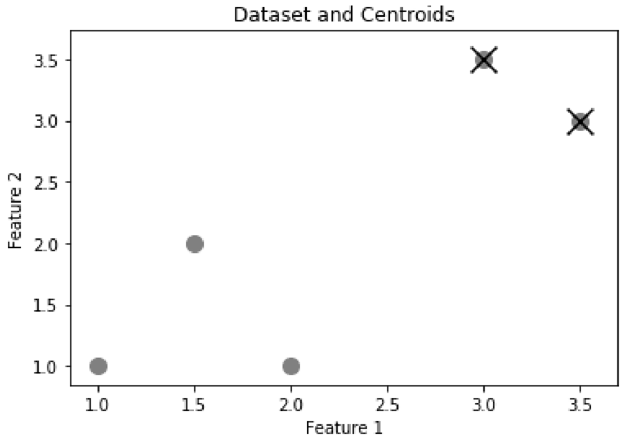
	Feature 1	Feature 2
0	1.0	1.0
1	2.0	1.0
2	1.5	2.0
3	3.0	3.5
4	3.5	3.0

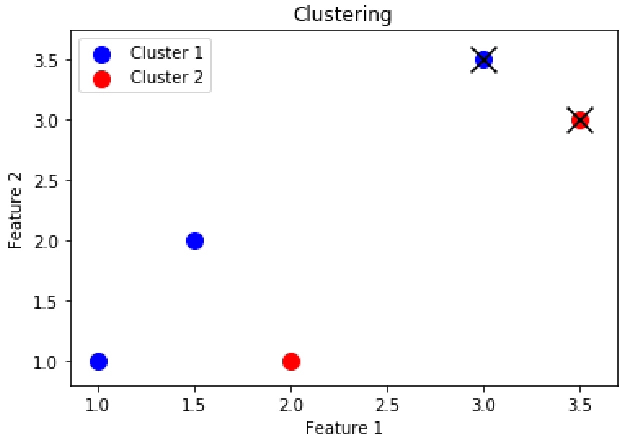
	Feature 1	Feature 2
C 1	1.83	2.17
C 2	2.75	2.00

Repeat the whole process with the new centroids!

The whole process: iteration 1

1st Iteration



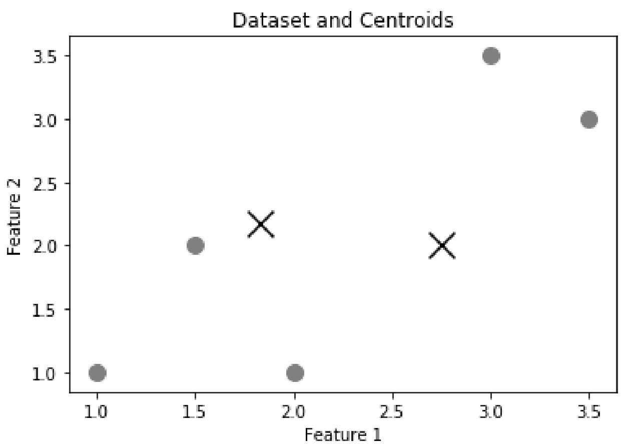


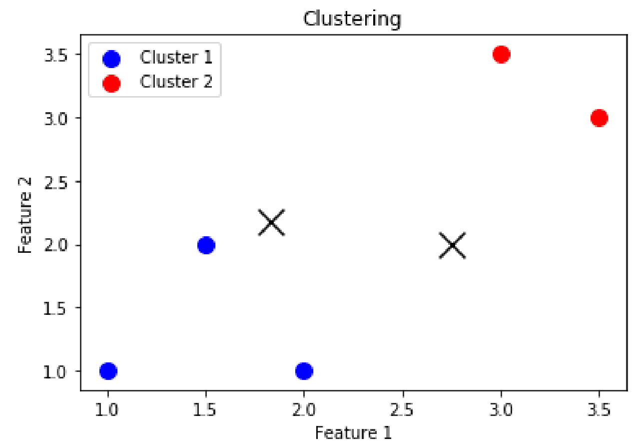
	Feature 1	Feature 2
C 1	3.0	3.5
C 2	3.5	3.0

Random centroids from the dataset

The whole process: iteration 2

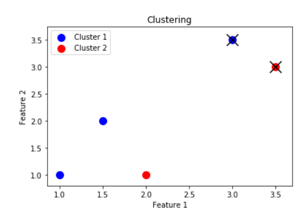
2^{nd} Iteration





	Feature 1	Feature 2
C 1	1.83	2.17
C 2	2.75	2.00

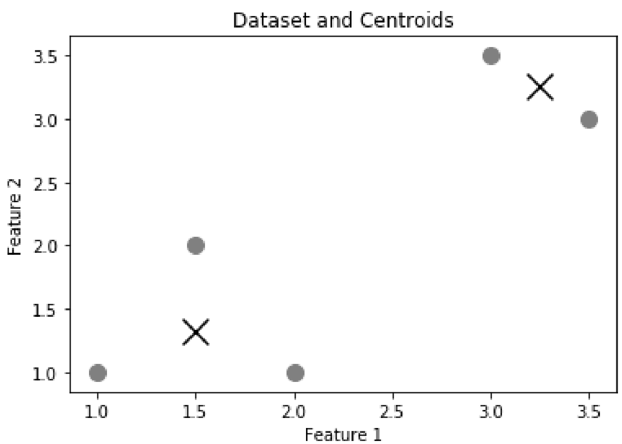
Centroids calculated in the first iteration

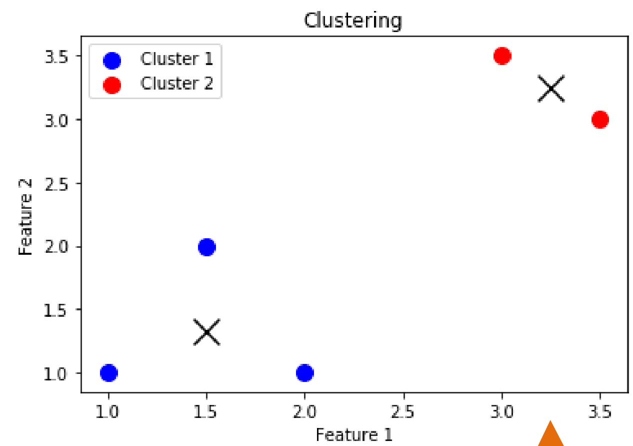


Check

The whole process: iteration 3

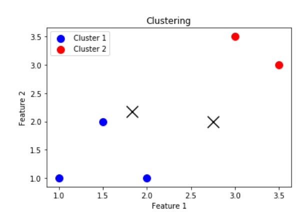
3rd Iteration





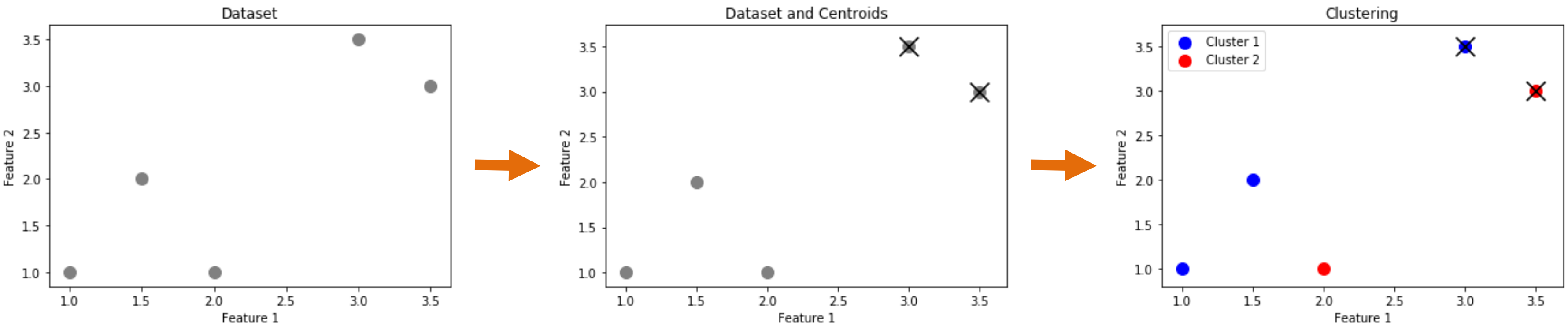
	Feature 1	Feature 2
C 1	1.50	1.33
C 2	3.25	3.25

Centroids calculated in the second iteration



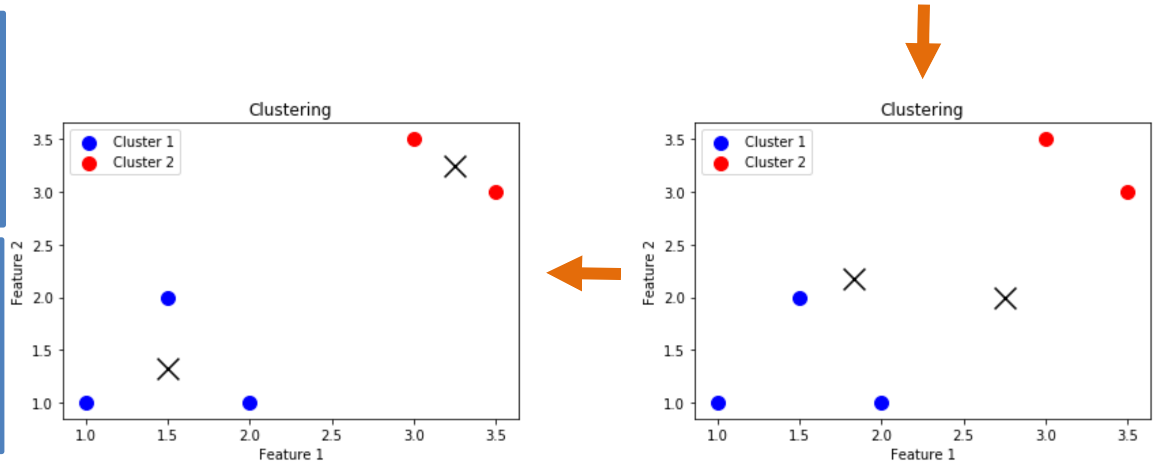
No reassignments! Centroids do not change anymore!

The whole process: stopping criteria



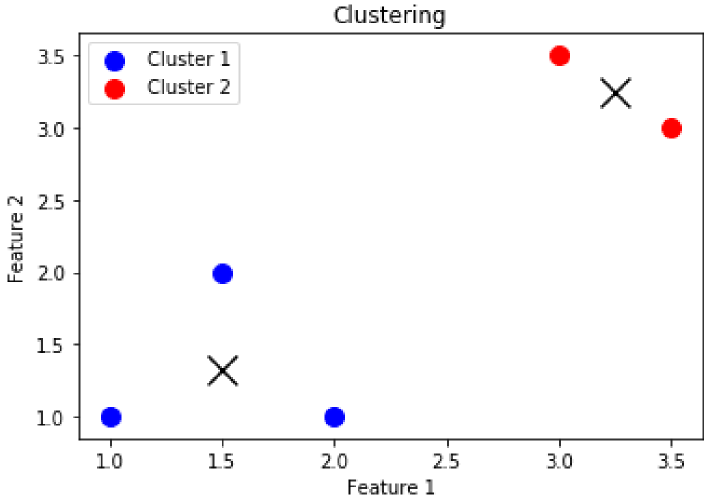
1 – There are no reassignments, so centroids do not change their values anymore

2 - The defined maximum number of iterations has been reached



Final total cost: compute it

$$Cost\left(C_1, C_2, \dots, C_k, c^{(1)}, \dots, c^{(1)}\right) = \sum_{k=1}^{K} \sum_{x^{(i)} \in C_k} \left\| c^{(k)} - x^{(i)} \right\|^2$$



The K-Means cost monotonically decreases

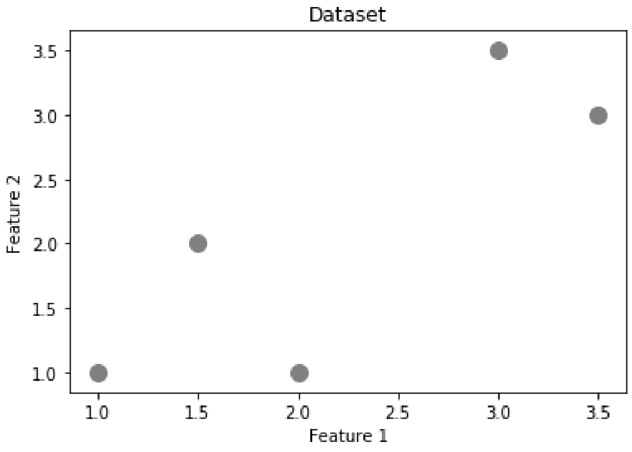
The K-Means algorithm converges to a local minimum

The cost depends on the representative centroids initialization

Sklearn runs by default 10 times using different centroids and selects the one with the lower final total cost, and defines 300 as the default maximum number of iterations

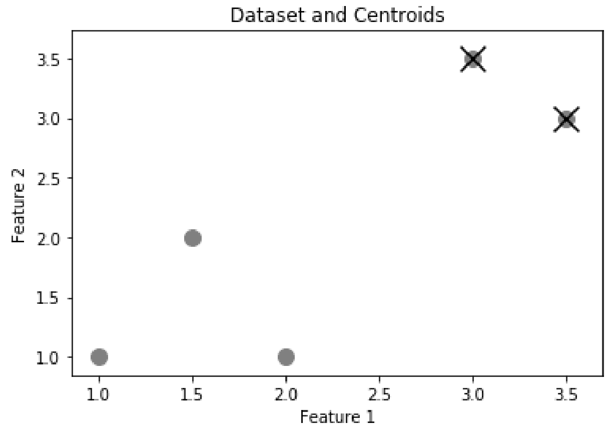
Plotting the dataset

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

descriptive = {'Feature 1':[1,2,1.5,3,3.5], 'Feature 2':[1,1,2,3.5,3]}
dataset_features = pd.DataFrame(descriptive)

plt.scatter(dataset_features['Feature 1'].values,
            dataset_features['Feature 2'].values,
            s=100, c='grey', label='Data')
plt.title('Dataset')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.show()
```



Plotting the dataset and centroids

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

descriptive = {'Feature 1':[1,2,1.5,3,3.5], 'Feature 2':[1,1,2,3.5,3]}
dataset_features = pd.DataFrame(descriptive)

centroids = np.array([[3,3.5], [3.5,3]])

indexes = ['C1', 'C2']
column_names = ['Feature 1', 'Feature 2']
dataset_centroids = pd.DataFrame(centroids,
                                 index = indexes, columns = column_names)
plt.scatter(dataset_features['Feature 1'].values,
            dataset_features['Feature 2'].values ,
            s=100, c='grey', label='Data')
plt.title('Dataset and Centroids')
plt.scatter(dataset_centroids['Feature 1'].values,
            dataset_centroids['Feature 2'].values,
            s=250, marker = 'x', c = 'black')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.show()
```

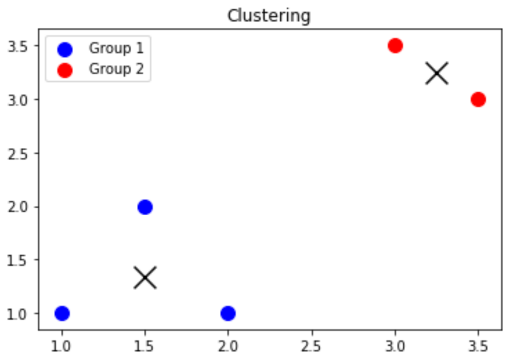


Computing and plotting the clusters

```
from sklearn.cluster import KMeans
import numpy as np

k_means = KMeans(n_clusters = 2,
                 random_state = 0, n_init = 1,
                 init = dataset_centroids.values).fit(dataset_features.values[:,:2])
dataset_features['Cluster'] = k_means.labels_

group_1 = dataset_features.loc[dataset_features['Cluster'] == 0]
group_2 = dataset_features.loc[dataset_features['Cluster'] == 1]

plt.scatter(group_1['Feature 1'].values, group_1['Feature 2'].values ,
            s=100, c='b', label='Group 1')
plt.scatter(group_2['Feature 1'].values, group_2['Feature 2'].values ,
            s=100, c='r', label='Group 2')
plt.scatter(k_means.cluster_centers_[:,0], k_means.cluster_centers_[:,1],
            s=250, marker = 'x', c='black')
plt.legend()
plt.title('Clustered Data')
plt.show()
```

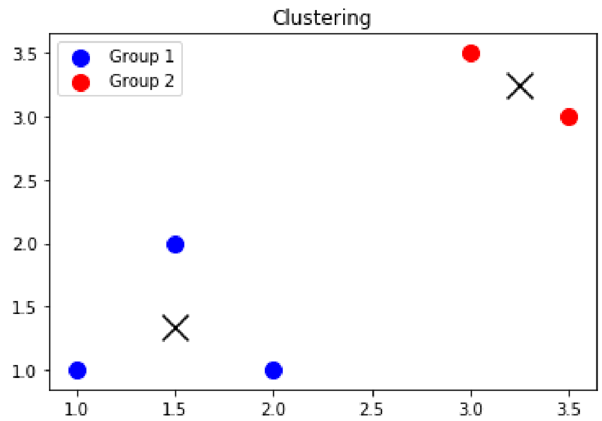
init sets the initial representative centroids

Sklearn provides the KMeans class



Final values

```
print('K-Means labels\n',k_means.labels_)
print('K-Means cluster centers\n',k_means.cluster_centers_)
print('Cost\n', k_means.inertia_)
print('Iterations\n', k_means.n_iter_)
```




```
K-Means labels
 [0 0 0 1 1]
K-Means cluster centers
 [[1.5        1.33333333]
 [3.25       3.25      ]]
Iterations
 3
Cost
 1.4166666666666667
```