4^{th}-week-2

k-Means

Wang Jianfang(王建芳 in Chinese)

Agenda

- Example
- The Basic Idea
- Code

Algorithm Procedure

① Randomly select K points from complete samples as the initial center.(That's what k means in K-means)

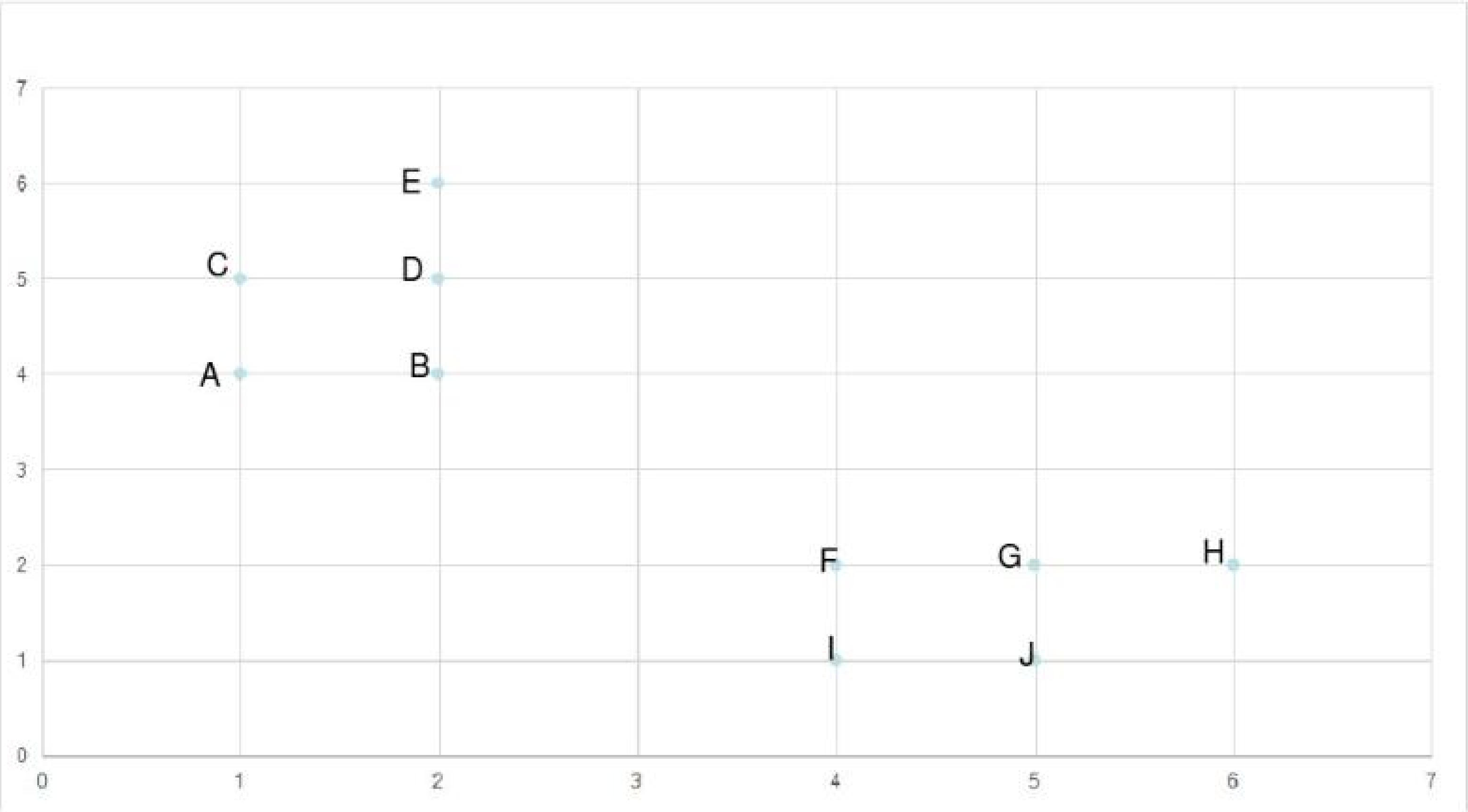
② Each point in the dataset is assigned to the closed cluster,based upon the Euclidean distance between each point and each cluster center.

$$S=\sqrt{(X_1-X_2)^2+(Y_1-Y_2)^2}$$

③ Each cluster's center is recomputed as the average of the points in that cluster.

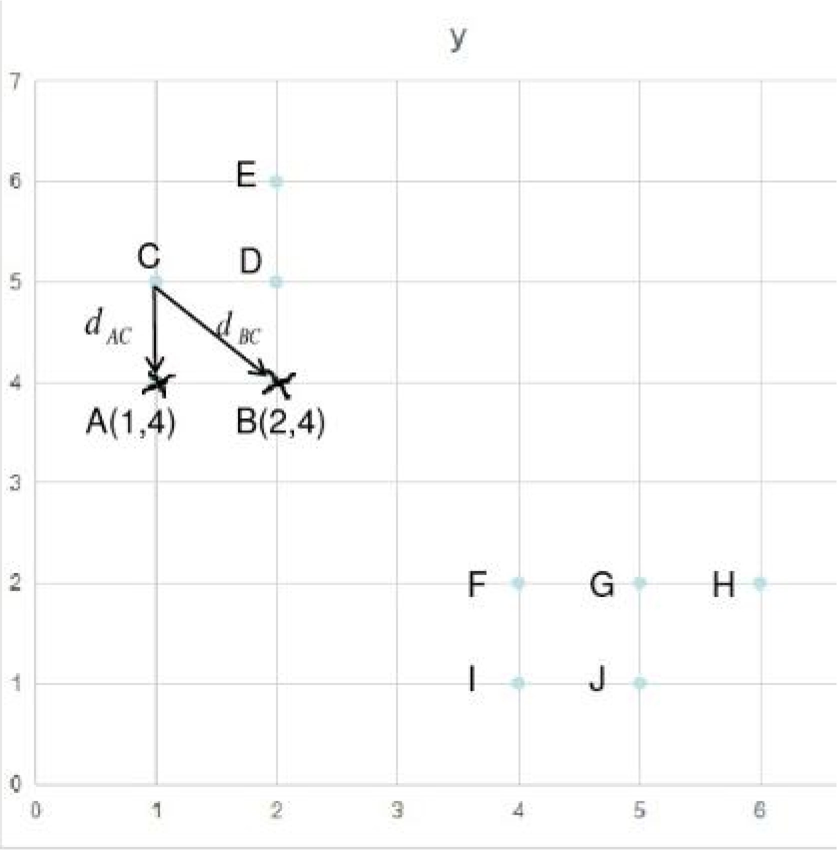
④ Iterate step 2 or more until the new center of cluster equals to the original center of cluster or less than a specified threshold,then clustering finished.

Example



How to cluster A,B...H,J into two clusters?

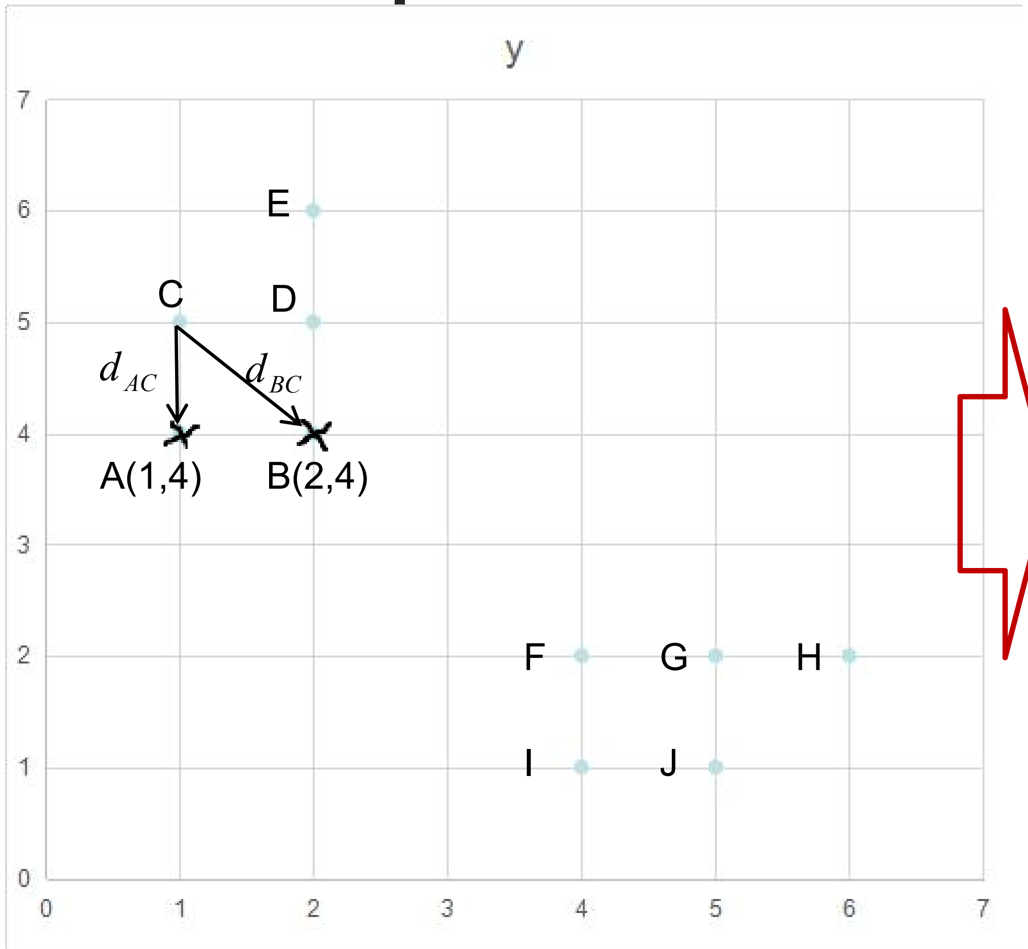
Example



Randomly choose A,B as the centre and K=2.

Step 1 and 2.

Example



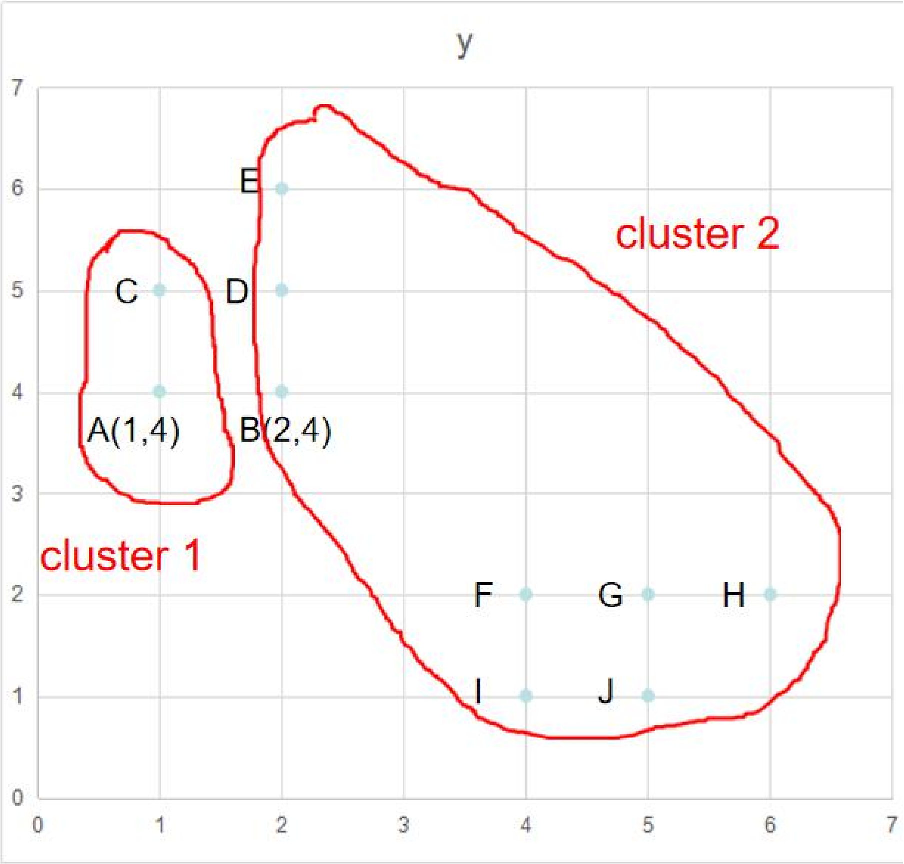
$$S=\sqrt{(X_1-X_2)^2+(Y_1-Y_2)^2}$$

A	$d_{AA}=$	$d_{BA}=$
B	$d_{AB}=$	$d_{BB}=$
C	$d_{AC}=$	$d_{BC}=$
D	$d_{AD}=$	$d_{BD}=$
E	$d_{AE}=$	$d_{BE}=$
F	$d_{AF}=$	$d_{BF}=$
G	$d_{AG}=$	$d_{BG}=$
H	$d_{AH}=$	$d_{BH}=$
I	$d_{AI}=$	$d_{BI}=$
J	$d_{AJ}=$	$d_{BJ}=$

Randomly choose A,B as the centre and K=2.

Step 1 and 2.

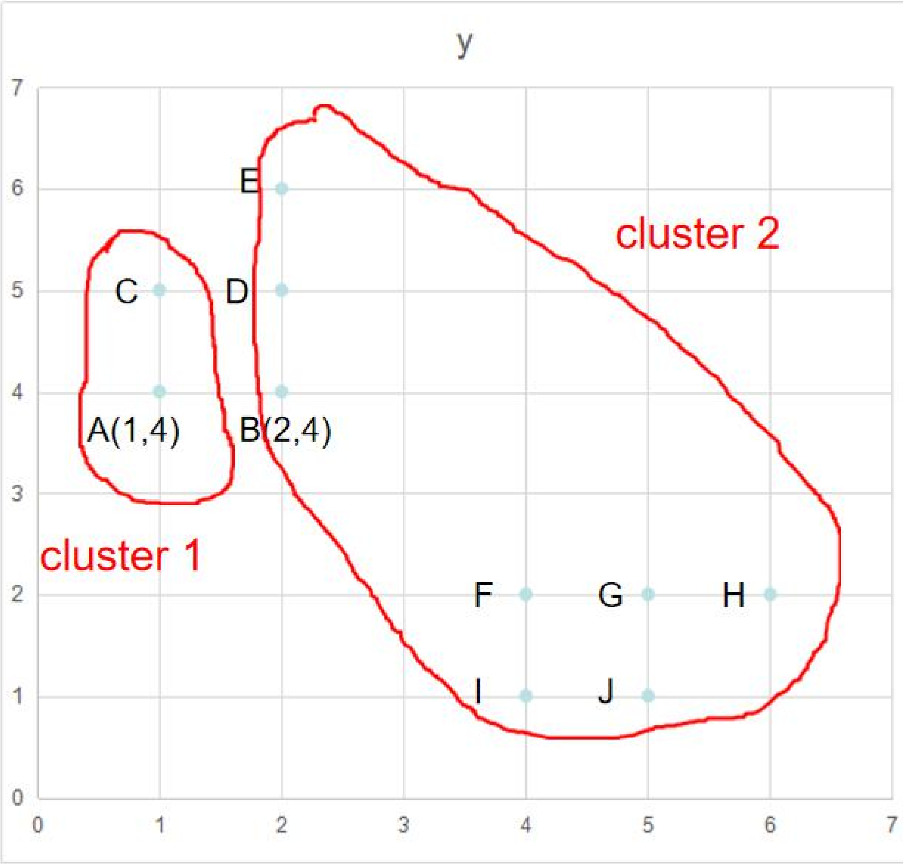
Example



Randomly choose A,B as the centre and K=2.

Step 3.

Example



Randomly choose A,B as the centre and K=2.

Step 3.

$$center = \left(\frac{\sum x_i}{i}, \frac{\sum y_j}{j}\right)$$

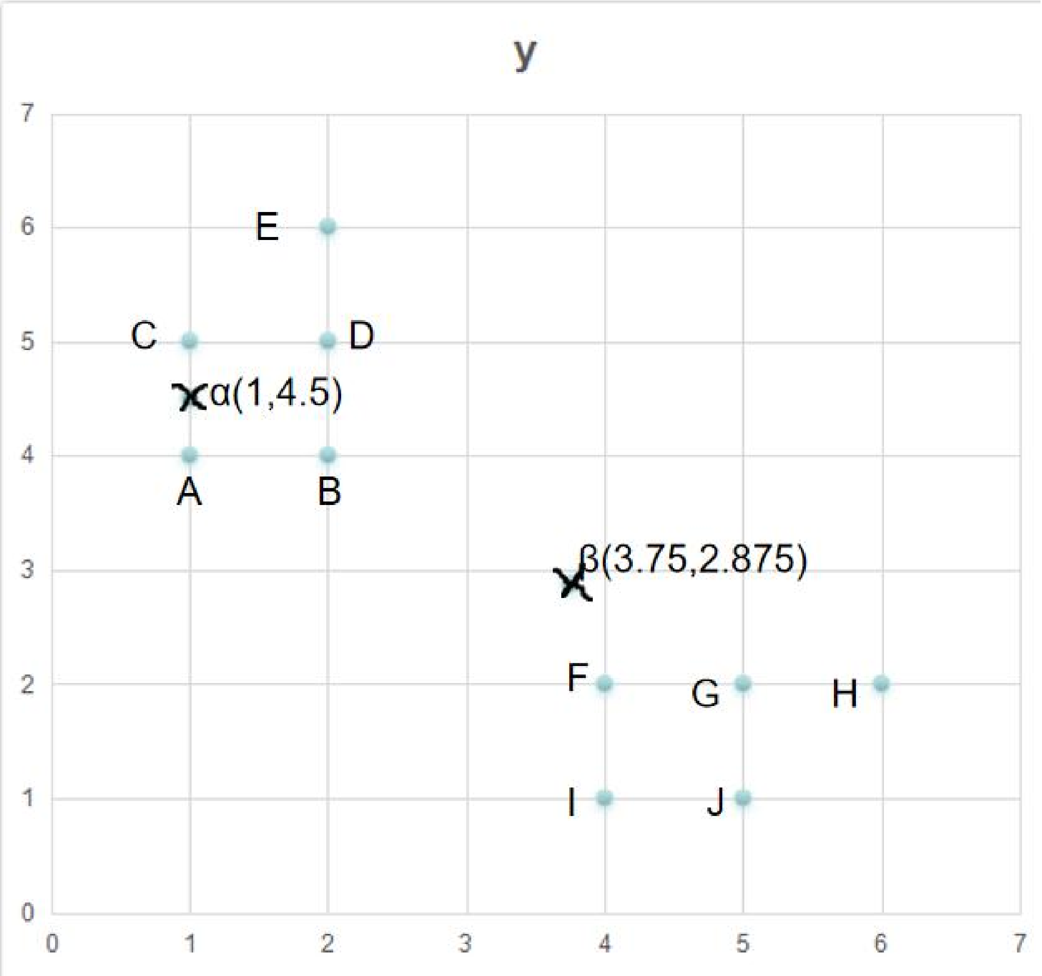
new center

$\alpha_{A,C} =$

$\beta_{B,D,E,F,G,H,I,J} =$

The new centers of the two clusters are (,) and (,)

Example



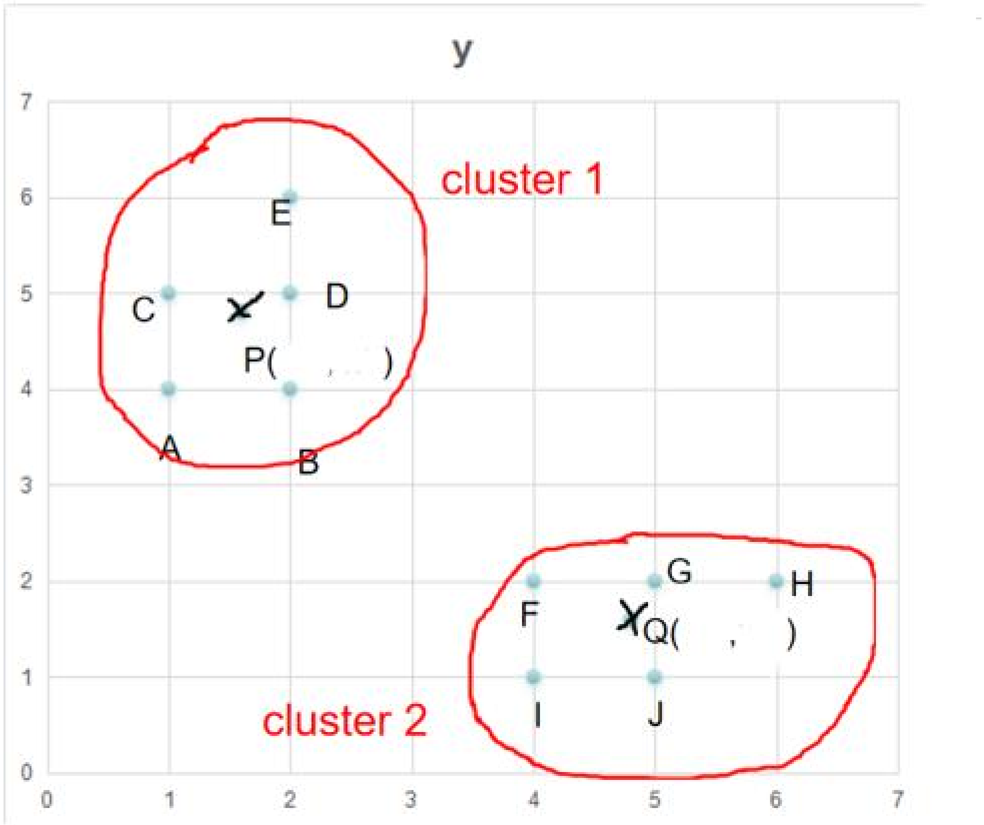
α , β as the centre and K=2.

Step 2 again.

A	$d_{\alpha A} =$		$d_{\beta A} =$
B	$d_{\alpha B} =$		$d_{\beta B} =$
C	$d_{\alpha C} =$		$d_{\alpha C} =$
D	$d_{\alpha D} =$		$d_{\beta D} =$
E	$d_{\alpha E} =$		$d_{\beta E} =$
F	$d_{\alpha F} =$		$d_{\beta F} =$
G	$d_{\alpha G} =$		$d_{\beta G} =$
H	$d_{\alpha H} =$		$d_{\beta H} =$
I	$d_{\alpha I} =$		$d_{\beta I} =$
J	$d_{\alpha J} =$		$d_{\beta J} =$

So,we classify A,B,C,D,E as a cluster and F,G,H,I,J as another cluster.

Example



P , Q as the centre and K=2.

Step 3 again.

$$center = \left(\frac{\sum x_i}{i}, \frac{\sum y_j}{j}\right)$$

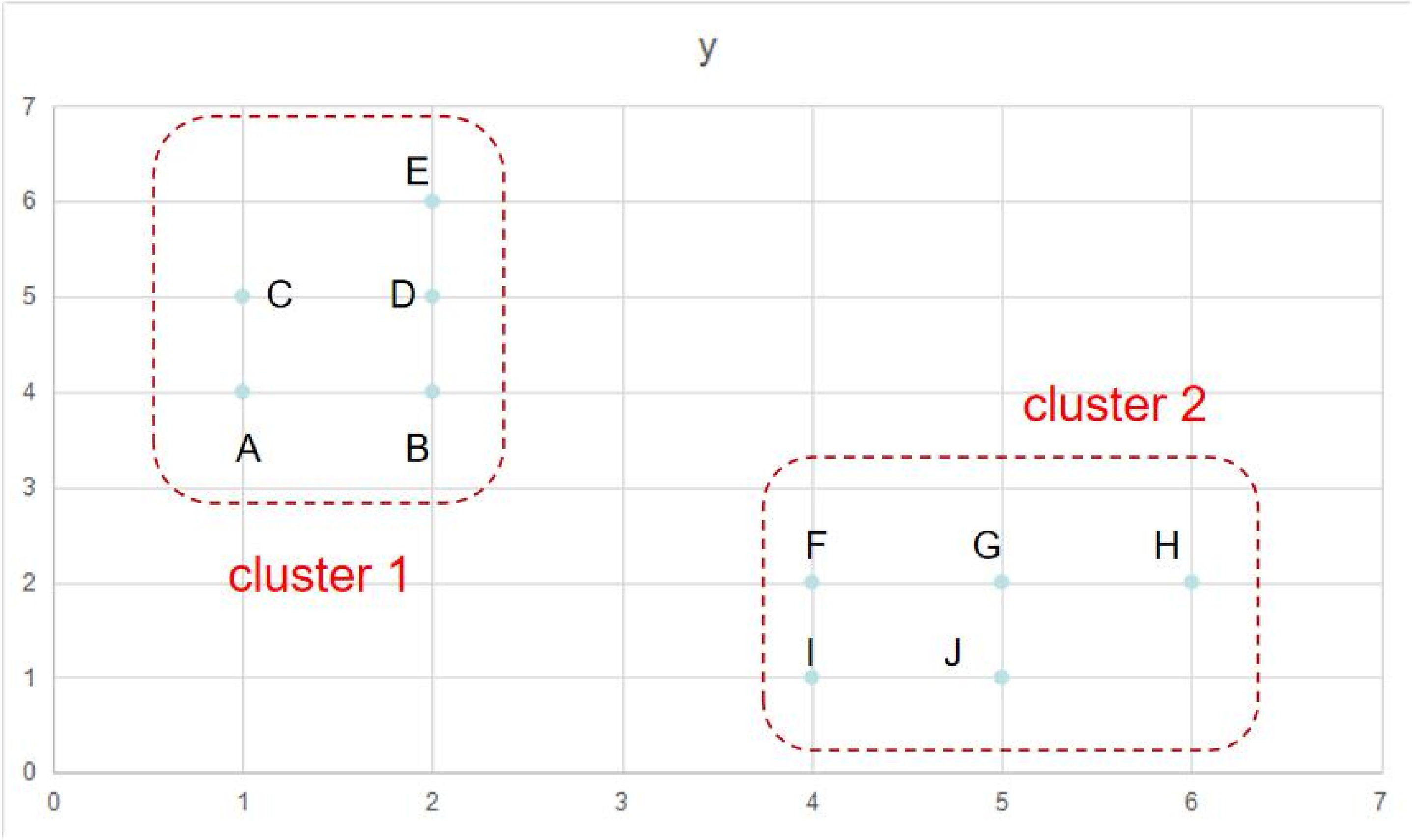
$M_{A,B,C,D,E} = (\quad , \quad)$

new center

$N_{F,G,H,I,J} = (\quad , \quad)$

The new centers of the two clusters are equal to the original P (,) and Q(,)

Example



Clustering finished !

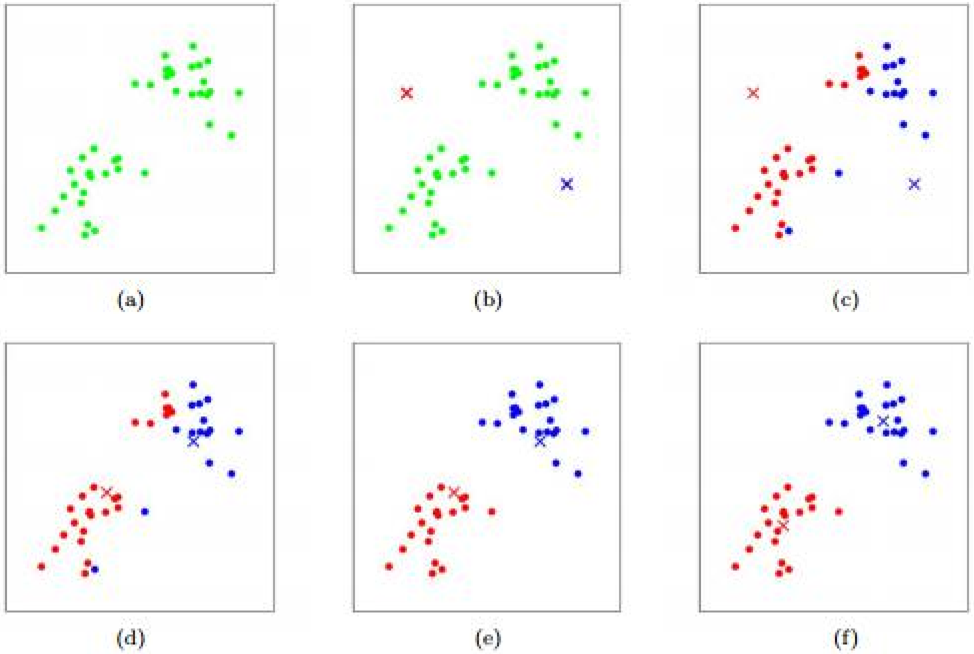
The Basic Idea

- Say you are given a data set where each observed example has a set of features, but has no labels. Labels are an essential ingredient to a supervised algorithm like Support Vector Machines, which learns a hypothesis function to predict labels given features. So we can't run supervised learning. What can we do?
- One of the most straightforward tasks we can perform on a data set without labels is to find groups of data in our dataset which are similar to one another -- what we call clusters.
- K-Means is one of the most popular "clustering" algorithms. K-means stores k centroids that it uses to define clusters. A point is considered to be in a particular cluster if it is closer to that cluster's centroid than any other centroid.

The Basic Idea

- K-Means finds the best centroids by alternating between (1) assigning data points to clusters based on the current centroids (2) chosing centroids (points which are the center of a cluster) based on the current assignment of data points to clusters.

Example



- Figure 1: K-means algorithm. Training examples are shown as dots, and cluster centroids are shown as crosses. (a) Original dataset. (b) Random initial cluster centroids. (c-f) Illustration of running two iterations of k-means. In each iteration, we assign each training example to the closest cluster centroid (shown by "painting" the training examples the same color as the cluster centroid to which is assigned); then we move each cluster centroid to the mean of the points assigned to it. Images courtesy of Michael Jordan.

The Algorithm

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.
2. Repeat until convergence: {

For every i, set

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

For each j, set

$$\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$$

}

- Important note: You might be tempted to calculate the distance between two points manually, by looping over values. This will work, but it will lead to a slow k-means! And a slow k-means will mean that you have to wait longer to test and debug your solution.
- Let's define three vectors:
- x = np.array([1, 2, 3, 4, 5]
- y = np.array([8, 8, 8, 8, 8])
- z = np.ones((5, 9))

To calculate the distance between x and y we can use:

```
np.sqrt(sum((x - y) ** 2))
```

To calculate the distance between all the length 5 vectors in z and x we can use:

```
np.sqrt(((z-x)**2).sum(axis=0))
```

Code

Homework

- update the distance formulas.