1495 UNIVERSITY OF ABERDEEN | CELEBRATING 525 YEARS 1495 – 2020

ABERDEEN 2040

Clustering

Data Mining & Visualisation

Lecture 17

2025

Today…

- Unsupervised Learning
- Clustering
- K-Means Clustering
- K-Means ++

Recap: Unsupervised Learning

Unsupervised Learning

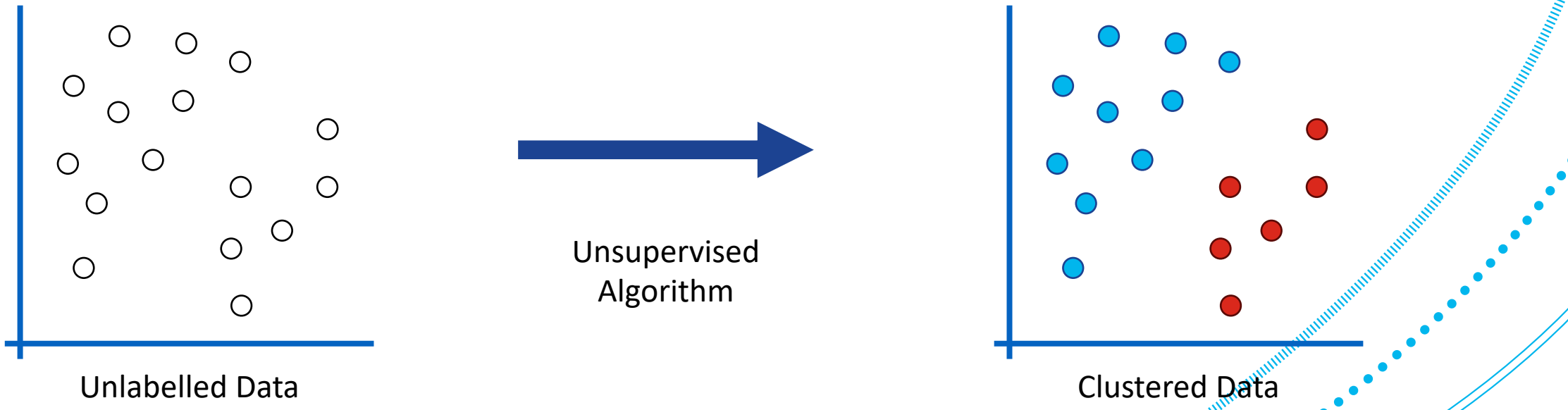
Unsupervised learning is a type of machine learning where unlabelled training data is used to train models.

Our models learn the relationships between these variables, attributing some labels or values on its own.

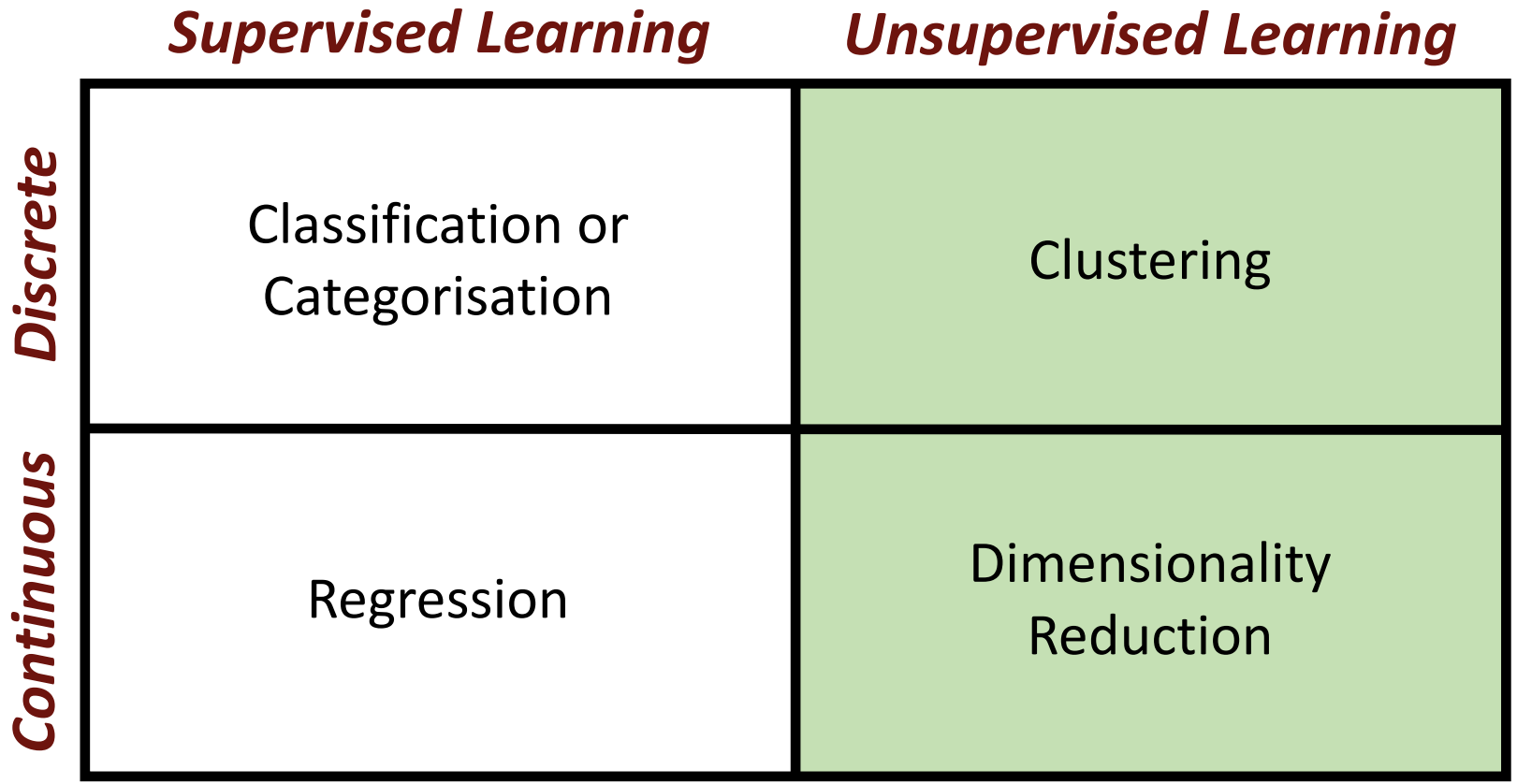
Unlike supervised learning, there is no direct measure of success.

Unsupervised Learning

Example: Based on an unlabelled dataset, we could identify clusters, or groups, of similar datapoints within that dataset.



Statistical Learning Problems



Clustering

What is Clustering

Let’s say we have our dataset representing the sizes and price of houses, and we want to group together similar datapoints.

Price ($)	Size (Sq. ft.)
100,000	1,000
250,000	2,500
400,000	4,000
…	…

Unlabelled Data

No such label exists in the dataset.

So let’s use an unsupervised learning algorithm to identify one.

What is Clustering

Let’s say we have our dataset representing the sizes and price of houses, and we want to group together similar datapoints.

Price ($)	Size (Sq. ft.)	Label
100,000	1,000	Small
250,000	2,500	Large
400,000	4,000	Large
…	…	

Clustered Data

No such label exists in the dataset.

So let’s use an unsupervised learning algorithm to identify one.

Clustering

More formally, we might say:

Problem: Identify (a small number of) groups of similar objects within a given (large) dataset.

Goals: Find “natural” clusters of points / Identify representative points for these clusters / Identify unusual datapoints (outliers), etc.

Clustering

Clustering algorithms group a set of datapoints in such a way that datapoints are more similar within-group than they are to other data.

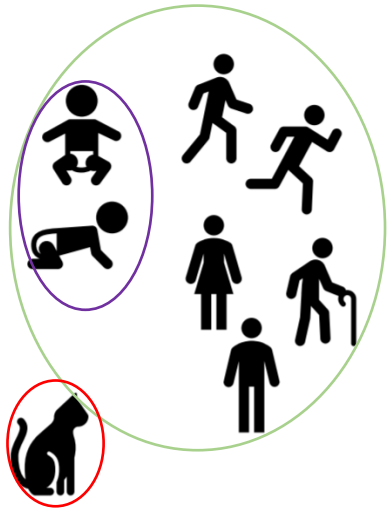
The **input** of these algorithms is a dataset (with no DV).

The **output** of these algorithms is a set of labels (groups) for each datapoint, where members of a group, or cluster, are more ‘similar’ to each other than they are to members of another cluster.

Aims of Clustering

There are a few reasons that we might want to cluster data:

- Discover the underlying structure of the data
- Identify sub-populations in the data:
  - How many?
  - What size?
  - Common properties?
  - Outliers?



Types of Clustering

In this course, we will cover a few different clustering approaches. These will broadly fall into two categories:

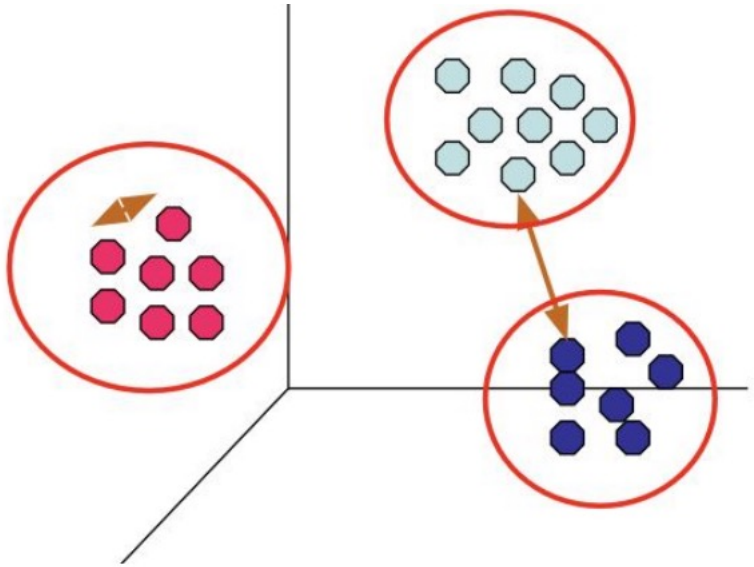
Partitional Clustering: A division of data objects into non-overlapping subsets (clusters) such that each datapoint is in exactly one subset.

Hierarchical Clustering: A set of nested clusters organized as a hierarchical tree, where each cluster can be a subset of another cluster.

Defining Cluster Similarity

Defining Cluster Similarity

Members of a cluster are more 'similar' to each other than they are to members of another cluster.



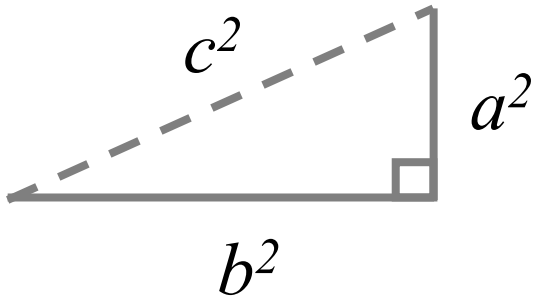
Similarity is often defined through (Euclidean) distance.

Defining Cluster Similarity

In all clustering examples (and in the exam), we will use Euclidean distance to determine similarity.

This means that the only equation you will need for these examples is based on the Pythagorean theorem:

$$a^2 + b^2 = c^2$$



Defining Cluster Similarity

Our distance formula is:

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Where i and j represent the two datapoints we are comparing…

…and x_1 and x_2 represent the numerical attributes in our dataset.

	x_1	x_2
i	1	4
j	2	5
…	7	3
	…	…

Defining Cluster Similarity

Our distance formula is:

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

$$dist(i,j)^2 = (2 - 1)^2 + (6 - 4)^2$$

$$= 1^2 + 2^2 = 1 + 4$$

$$= 5$$

	x_1	x_2
i	2	6
j	1	4
...	7	3
	...	...

Defining Cluster Similarity

Our distance formula is:

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

$$dist(i,j)^2 = 5$$

	x_1	x_2
i	2	6
j	1	4
…	7	3
	…	…

Note: We are only interested in comparing these distances, so to make our lives easier,

we will not find $dist(i,j)$; i.e. $\sqrt{5}$

Defining Cluster Similarity

	x_1	x_2
i	2	6
j	1	4
k	7	3
	…	…

Our distance formula is:

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

$$dist(i,j)^2 = (2-1)^2 + (6-4)^2 = 1^2 + 2^2 = 1 + 4 = 5$$

$$dist(i,k)^2 = (2-7)^2 + (6-3)^2 = -5^2 + 3^2 = 25 + 9 = 34$$

In other words, point i is closer to j than it is to k.

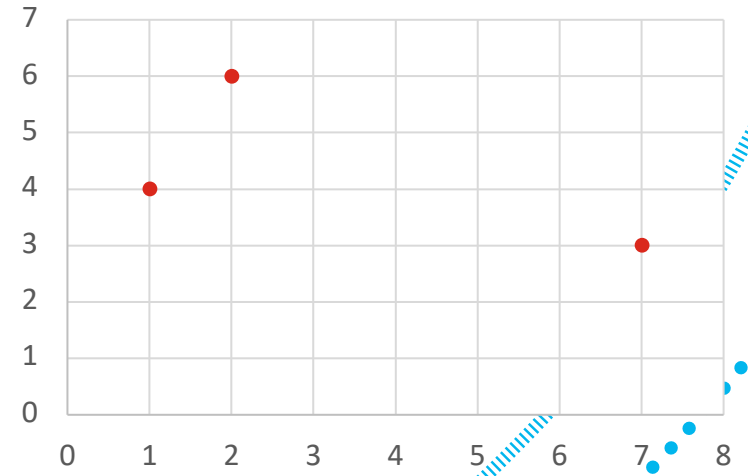
Defining Cluster Similarity

	x_1	x_2
i	2	6
j	1	4
k	7	3
	…	…

When two points are closer to each other, that *usually* means that they're more likely to be in a cluster together.

Whether or not this is actually the case will depend on the clustering algorithm that we use.

But typically, we're looking for closeness between points.



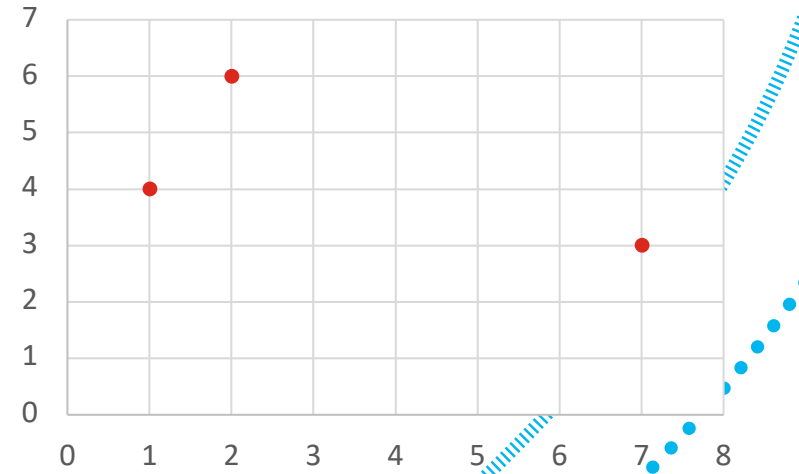
Evaluating Our Clusters

	x_1	x_2
i	2	6
j	1	4
k	7	3
	…	…

Once we have our clusters, we can then use the Sum of Squared Errors (SSE) to evaluate them.

The overall SSE is simply the sum of the squared distances for each datapoint to its assigned cluster's centroid:

$$SSE = \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$



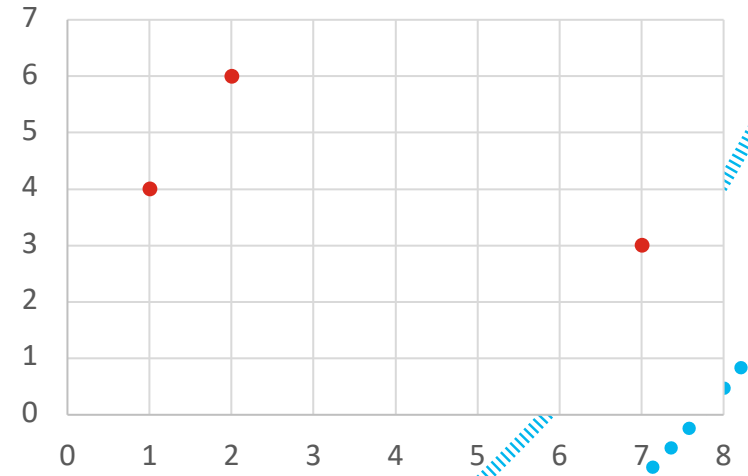
Evaluating Our Clusters

$$SSE = \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

where:

- K = number of clusters
- C_k = a set of points in cluster k
- x_i = a datapoint in cluster k
- μ_k = the centroid (mean) of cluster k
- $\|x_i - \mu_k\|^2$ = the squared Euclidean distance

	x_1	x_2
i	2	6
j	1	4
k	7	3
	…	…



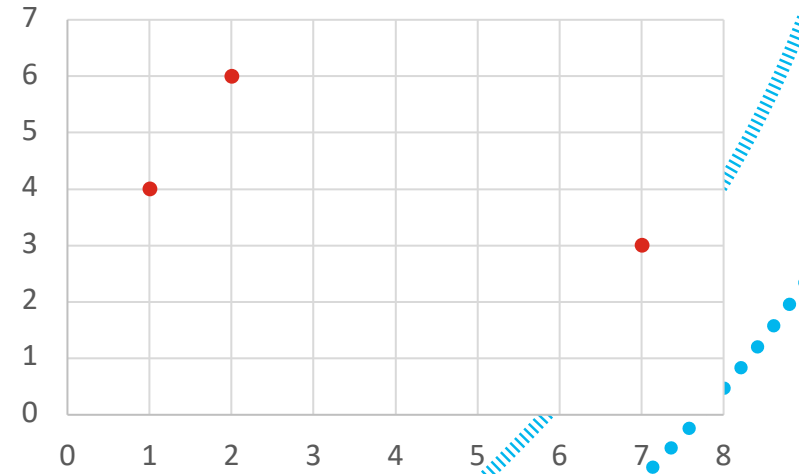
Evaluating Our Clusters

	x_1	x_2
i	2	6
j	1	4
k	7	3
	…	…

We can also work out the SSE value for each cluster:

For a given cluster, C_k, the SSE would simply be the sum of the squared distances for each datapoint to that cluster's centroid:

$$\mathrm{SSE}_{C_k} = \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$



K-Means Clustering

K-Means Clustering

K-Means clustering is a *partitional approach:*

- Datapoints are clustered into non-overlapping subsets
- Each datapoint is grouped into exactly one cluster

One of the defining aspects of K-Means clustering is that we start by defining the number of clusters, *K*, that we want.

K-Means Clustering

K datapoints are randomly picked as the ‘centroids’ (i.e. the centre point) of our clusters.

Each datapoint is then assigned to be a member of the cluster that has the closest centroid to it.

Once all datapoints are assigned to a cluster, we recompute the centroids of our clusters, and then repeat the assignment process, until the centroids converge (no longer change).

K-Means Clustering Algorithm

The K-Means algorithm is very simple:

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** the centroids don't change.

K-Means Clustering – Overview

Let’s take a look at an overview of what this process looks like, step-by-step.

We will use the ‘Old Faithful’ dataset, which contains records about the *duration of*, and the *time between*, eruptions of the ‘Old Faithful’ geyser.

K-Means Clustering – Overview

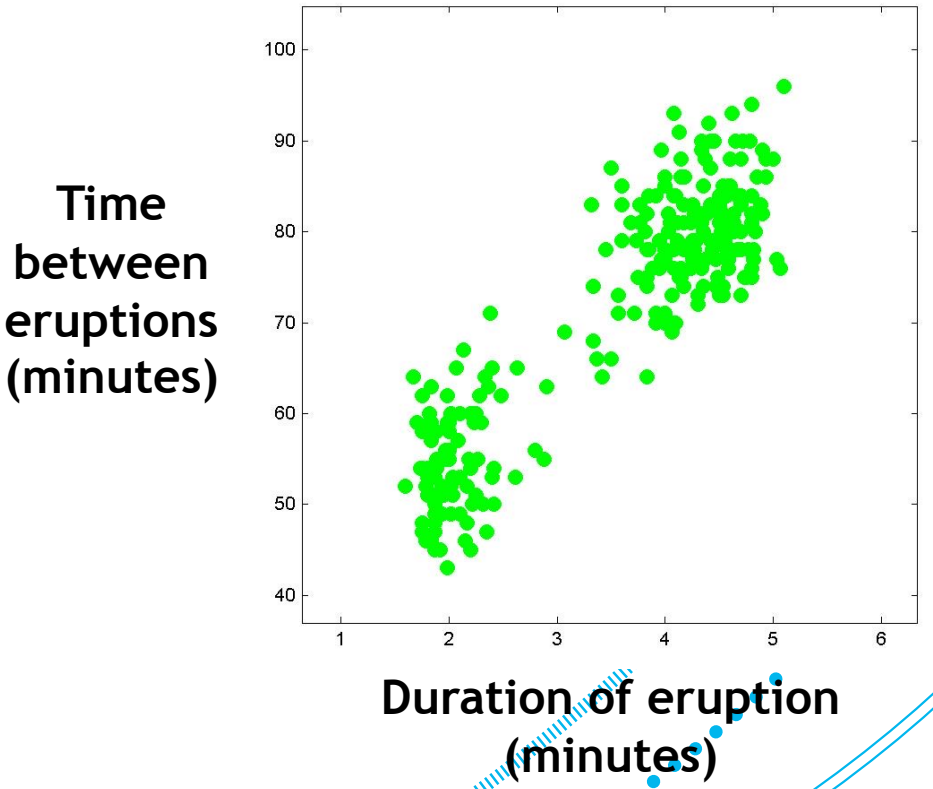
Goal: Given the old faithful dataset, partition the data into 2 clusters.

K-Means Clustering:

Initialize K centroids (cluster means) based on K random points, then iterate between:

step 1: Given these centroids, assign each datapoint to the nearest cluster.

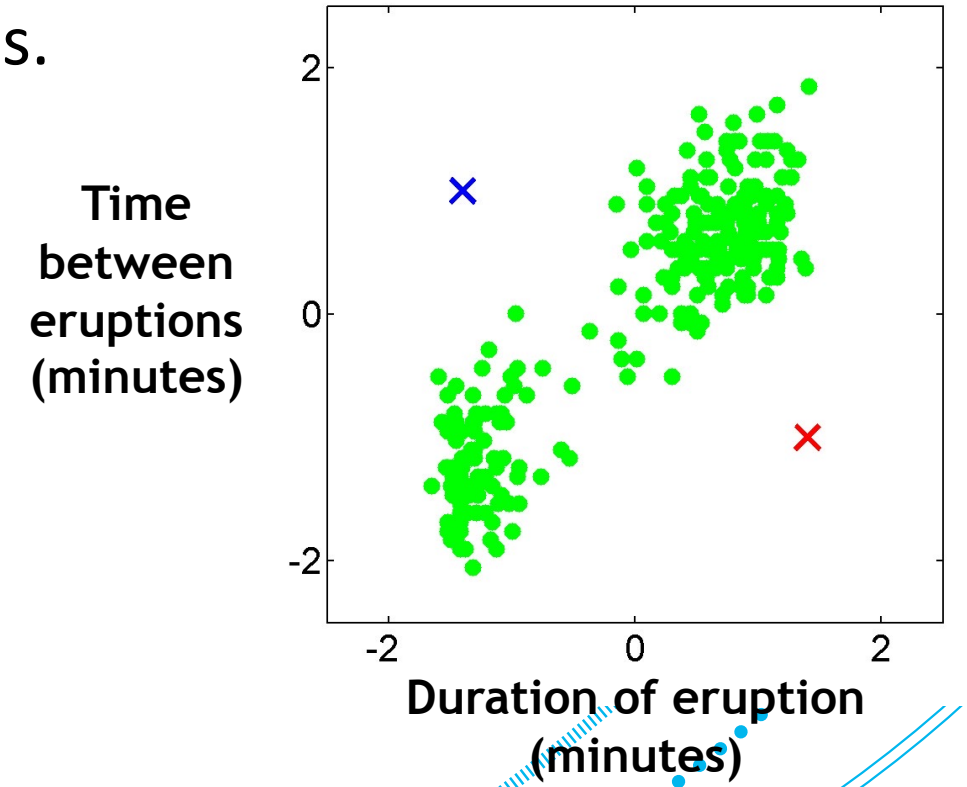
step 2: Given these assignments, update each centroid to be the mean of the cluster.



K-Means Clustering – Overview

For K-Means clustering, our initial cluster centroids would represent random datapoints.

However, for illustrative purposes, let's just say we start off with these two centroids.

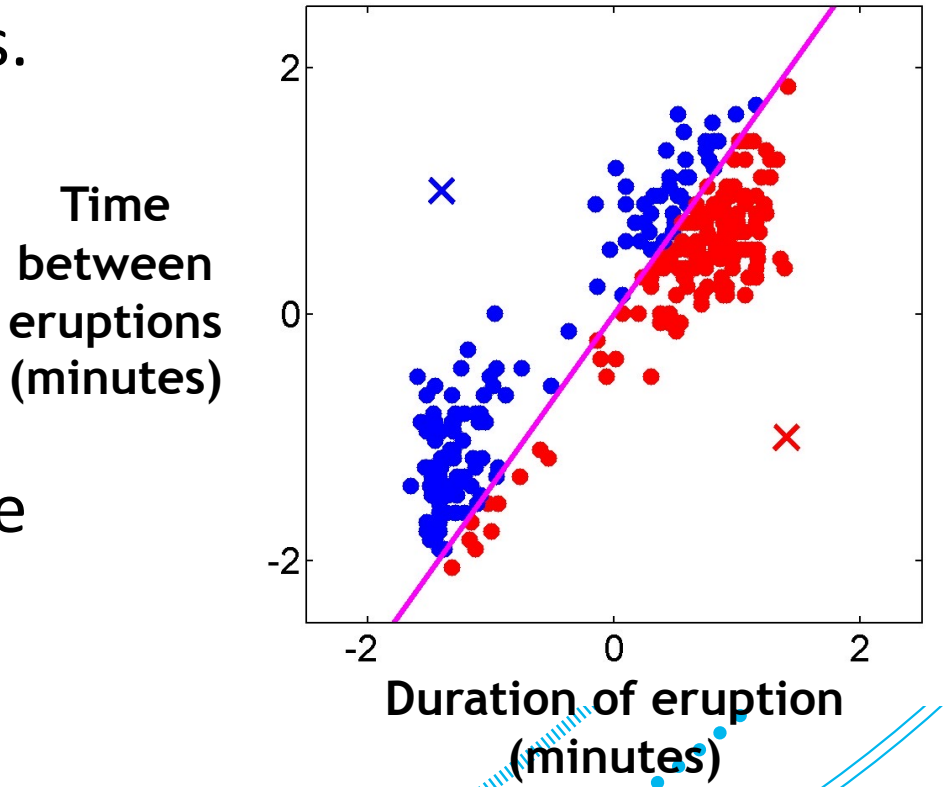


K-Means Clustering – Overview

For K-Means clustering, our initial cluster centroids would represent random datapoints.

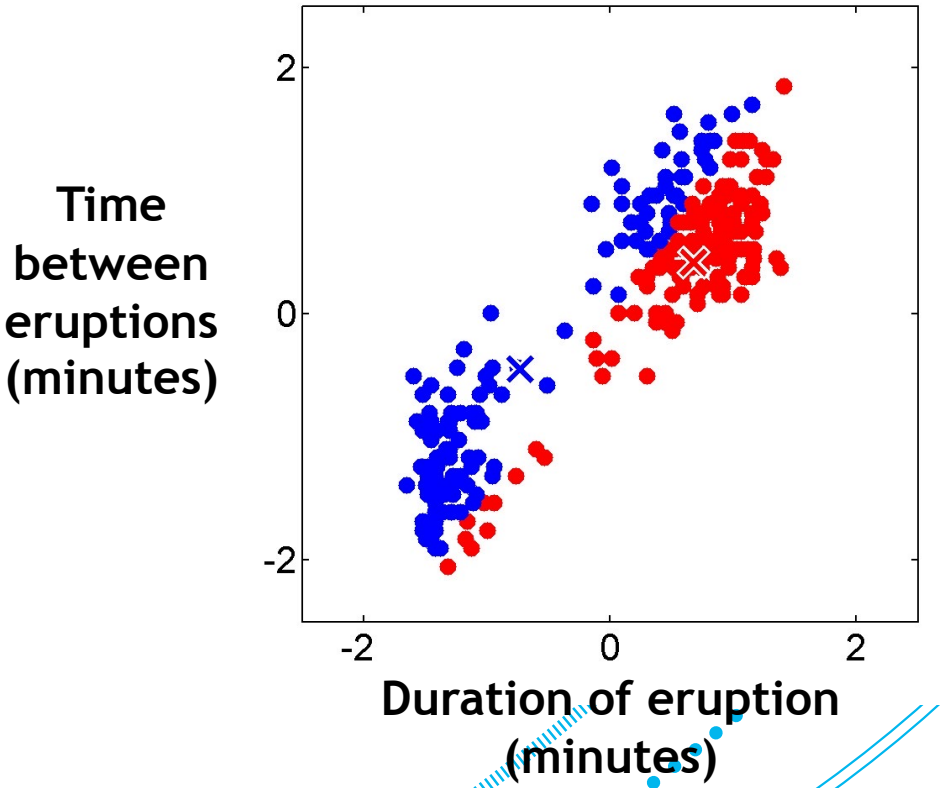
However, for illustrative purposes, let's just say we start off with these two centroids.

We then assign each datapoint to one of these two clusters, based on *closeness*.



K-Means Clustering – Overview

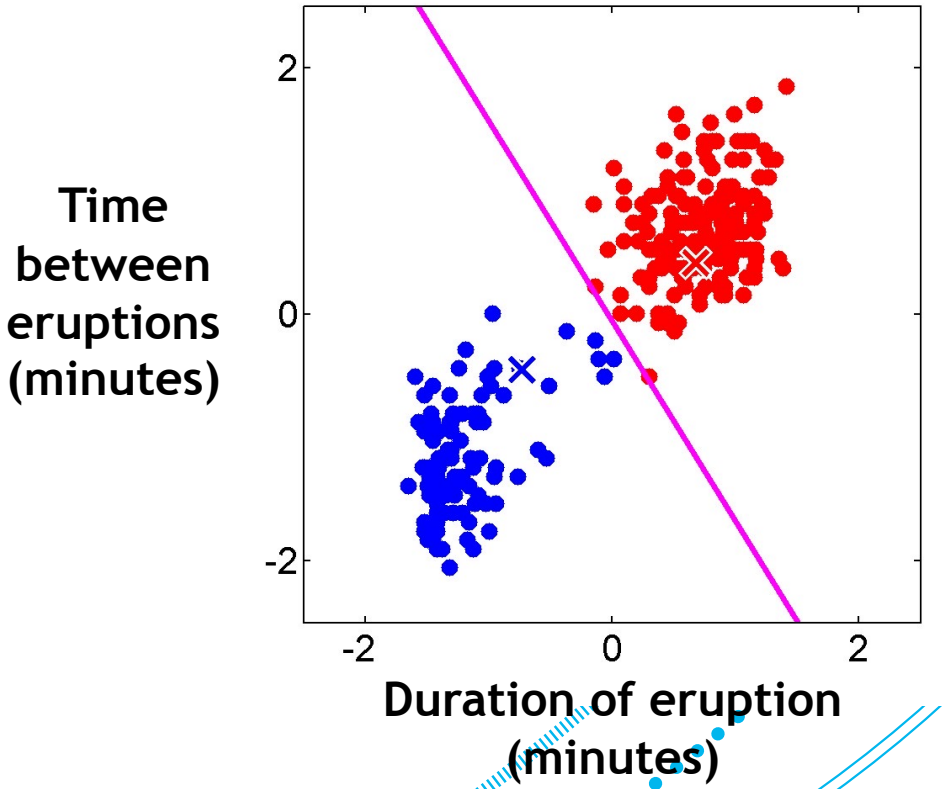
Given these assignments, our next step is to recompute and update the centroid (i.e. mean position) of each of the clusters…



K-Means Clustering – Overview

Given these assignments, our next step is to recompute and update the centroid (i.e. mean position) of each of the clusters…

…before again assigning each datapoint to the cluster with the closest centroid.

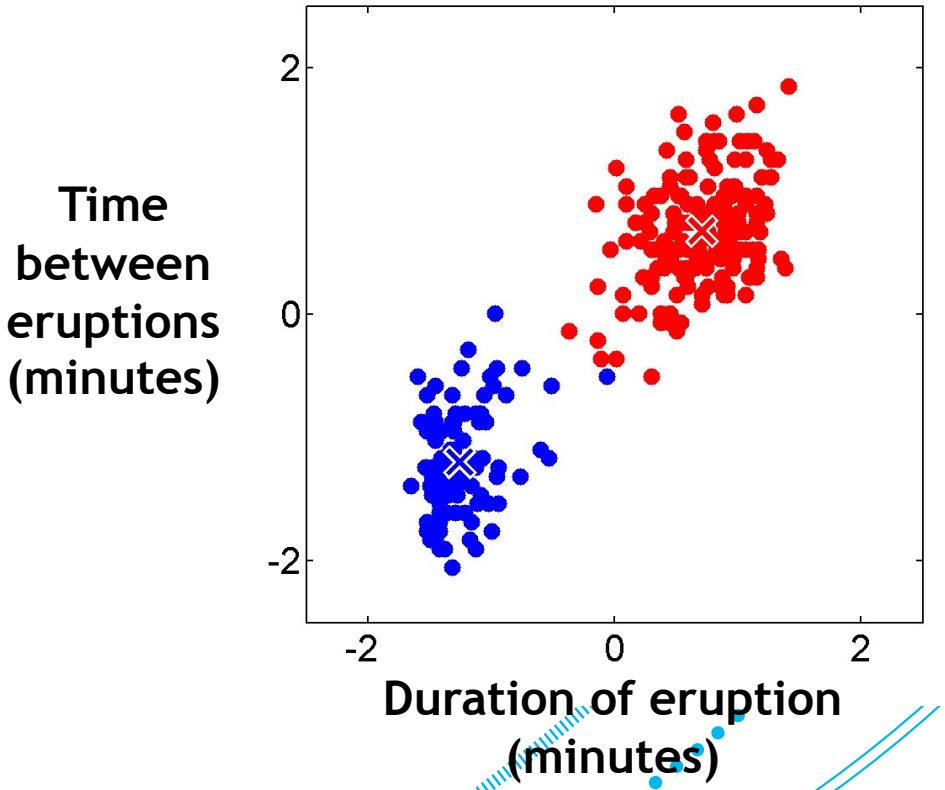


K-Means Clustering – Overview

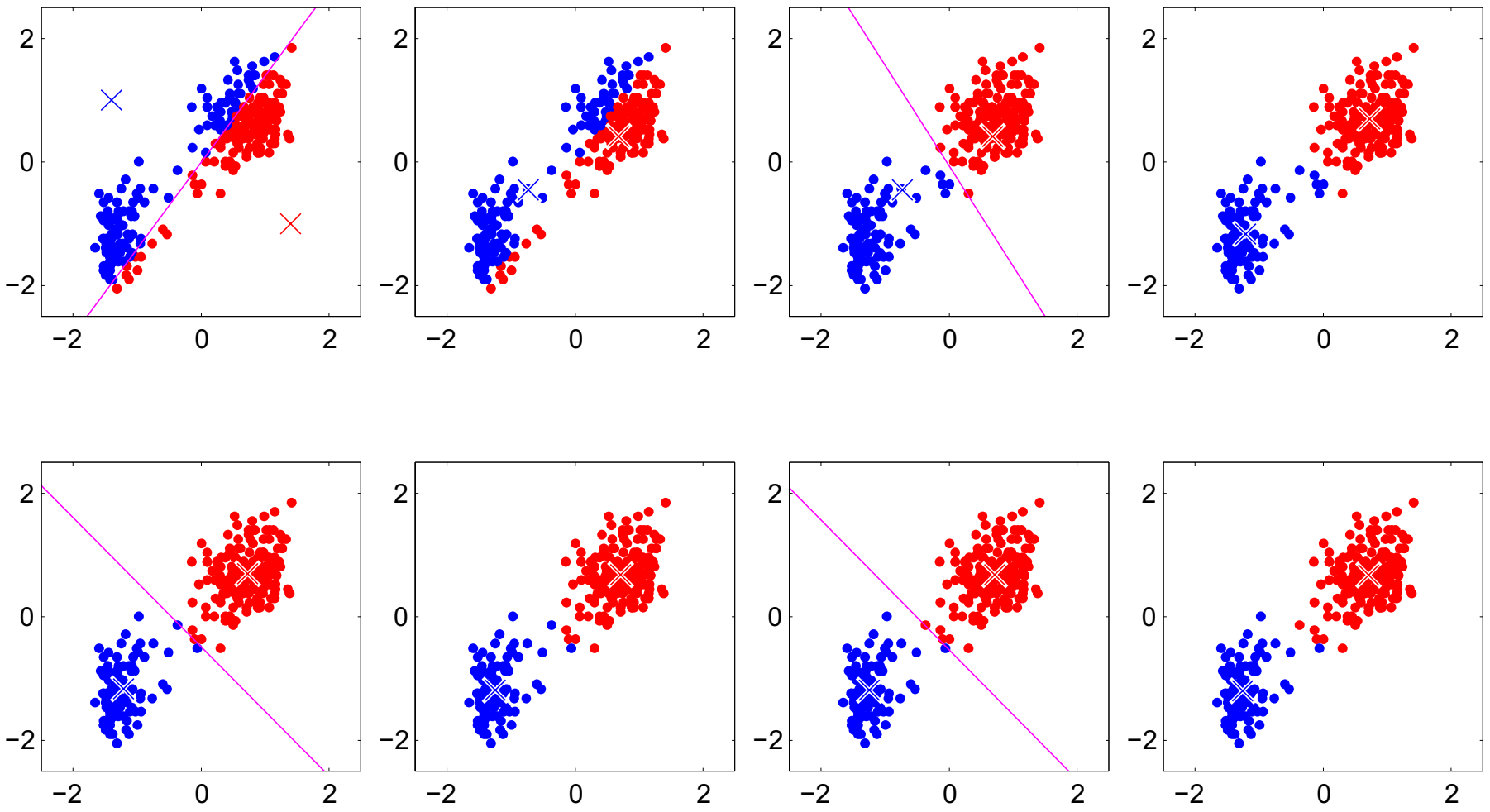
Given these assignments, our next step is to recompute and update the centroid (i.e. mean position) of each of the clusters…

…before again assigning each datapoint to the cluster with the closest centroid.

We repeat this process until the cluster assignments no longer change.



K-Means Clustering – Overview



K-Means: Step-by-Step

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Let's say we have the following six data objects (a—f) in two-dimensional Euclidean space.

Using K-Means clustering, let's see how we would cluster these objects into ***three*** clusters.

Let's say that our initial centroids are *b*, *d*, and *f*.

Cluster Centroids:

C1: 2, 2

C2: 4, 2

C3: 7, 2

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

We start by working out the distance between each point and each cluster centroid.

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Let's start with point a (2, 1):

Cluster Centroids:

C1: 2, 2

C2: 4, 2

C3: 7, 2

dist(a, c1)^2 = (2 - 2)^2 + (1 – 2)^2 = (0)^2 + (-1)^2 = 1

dist(a, c2)^2 = (2 - 4)^2 + (1 – 2)^2 = (-2)^2 + (-1)^2 = 5

dist(a, c3)^2 = (2 - 7)^2 + (1 – 2)^2 = (-5)^2 + (-1)^2 = 26

In this case, a is closest to C1, so let's allocate a to C1.

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

We start by working out the distance between each point and each cluster centroid.

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

We know that point b is our initial cluster centroid for C1.

So let's just allocate b to C1 without calculating!

Cluster Centroids:

C1: 2, 2 {a}

C2: 4, 2

C3: 7, 2

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

We start by working out the distance between each point and each cluster centroid.

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Let's move on to point c (4, 1):

$dist(c, c1)^2 = (4 - 2)^2 + (1 - 2)^2 = (2)^2 + (-1)^2 = 5$

$dist(c, c2)^2 = (4 - 4)^2 + (1 - 2)^2 = (0)^2 + (-1)^2 = 1$

$dist(c, c3)^2 = (4 - 7)_2 + (1 - 2)^2 = (-3)^2 + (-1)^2 = 10$

Cluster Centroids:

C1: 2, 2 {a, b}

C2: 4, 2

C3: 7, 2

In this case, c is closest to C2, so let's allocate c to C2.

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

We start by working out the distance between each point and each cluster centroid.

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

We know that point d is our initial cluster centroid for C2.

So let's just allocate d to C2 without calculating!

Cluster Centroids:

C1: 2, 2 {a, b}

C2: 4, 2 {c}

C3: 7, 2

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

We start by working out the distance between each point and each cluster centroid.

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Let's move on to point e (7, 1):

$dist(e, c1)^2 = (7 - 2)^2 + (1 - 2)^2 = (5)^2 + (-1)^2 = 26$

$dist(e, c2)^2 = (7 - 4)^2 + (1 - 2)^2 = (-3)^2 + (-1)^2 = 10$

$dist(e, c3)^2 = (7 - 7)^2 + (1 - 2)^2 = (0)^2 + (-1)^2 = 1$

Cluster Centroids:

C1: 2, 2 {a, b}

C2: 4, 2 {c, d}

C3: 7, 2

In this case, e is closest to C3, so let's allocate e to C3.

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

We start by working out the distance between each point and each cluster centroid.

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

We know that point f is our initial cluster centroid for C3.

So let's just allocate f to C3 without calculating!

Cluster Centroids:

C1: 2, 2 {a, b}

C2: 4, 2 {c, d}

C3: 7, 2 {e}

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

So after 1 iteration of assignments, our current clusters are:

C1 = {a, b}

C2 = {c, d}

C3 = {e, f}

Cluster Centroids:

C1: 2, 2 {a, b}

C2: 4, 2 {c, d}

C3: 7, 2 {e, f}

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

We then need to recompute our centroids:

C1 centroid = ((2 + 2) / 2, (1 + 2) / 2) = (2, 1.5)

C2 centroid = ((4 + 4) / 2, (1 + 2) / 2) = (4, 1.5)

C3 centroid = ((7 + 7) / 2, (1 + 2) / 2) = (7, 1.5)

Cluster Centroids:

C1: 2, 2 {a, b}

C2: 4, 2 {c, d}

C3: 7, 2 {e, f}

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Now, we need to repeat the process of allocating our points to clusters, based on their new centroids

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Let's start with point a (2, 1):

Cluster Centroids:

C1: 2, 1.5

C2: 4, 1.5

C3: 7, 1.5

dist(a, c1)^2 = (2 - 2)^2 + (1 – 1.5)^2 = (0)^2 + (-0.5)^2 = 0.25

dist(a, c2)^2 = (2 - 4)^2 + (1 – 1.5)^2 = (-2)^2 + (-0.5)^2 = 4.25

dist(a, c3)^2 = (2 - 7)^2 + (1 – 1.5)^2 = (-5)^2 + (-0.5)^2 = 25.25

In this case, a is still closest to C1, so let's re-allocate a to C1.

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Now, we need to repeat the process of allocating our points to clusters, based on their new centroids

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Let's move on to point b (2, 2):

Cluster Centroids:

C1: 2, 1.5 {a}

C2: 4, 1.5

C3: 7, 1.5

$dist(b, c1)^2 = (2 - 2)^2 + (2 - 1.5)^2 = (0)^2 + (0.5)^2 = 0.25$

$dist(b, c2)^2 = (2 - 4)^2 + (2 - 1.5)^2 = (-2)^2 + (0.5)^2 = 4.25$

$dist(b, c3)^2 = (2 - 7)^2 + (2 - 1.5)^2 = (-5)^2 + (0.5)^2 = 25.25$

In this case, b is still closest to C1, so let's re-allocate b to C1.

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Now, we need to repeat the process of allocating our points to clusters, based on their new centroids

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Let's move on to point c (4, 1):

Cluster Centroids:

C1: 2, 1.5 {a, b}

C2: 4, 1.5

C3: 7, 1.5

dist(c, c1)^2 = (4 - 2)^2 + (1 – 1.5)^2 = (2)^2 + (-0.5)^2 = 4.25

dist(c, c2)^2 = (4 - 4)^2 + (1 – 1.5)^2 = (0)^2 + (-0.5)^2 = 0.25

dist(c, c3)^2 = (4 - 7)^2 + (1 – 1.5)^2 = (-3)^2 + (-0.5)^2 = 9.25

In this case, c is still closest to C2, so let's re-allocate c to C2.

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Now, we need to repeat the process of allocating our points to clusters, based on their new centroids

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Let's move on to point d (4, 2):

Cluster Centroids:

C1: 2, 1.5 {a, b}

C2: 4, 1.5 {c}

C3: 7, 1.5

dist(c, c1)^2 = (4 - 2)^2 + (2 – 1.5)^2 = (2)^2 + (0.5)^2 = 4.25

dist(c, c2)^2 = (4 - 4)^2 + (2 – 1.5)^2 = (0)^2 + (0.5)^2 = 0.25

dist(c, c3)^2 = (4 - 7)^2 + (2 – 1.5)^2 = (-3)^2 + (0.5)^2 = 9.25

In this case, d is still closest to C2, so let's re-allocate d to C2.

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Now, we need to repeat the process of allocating our points to clusters, based on their new centroids

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Let's move on to point e (7, 1):

Cluster Centroids:

C1: 2, 1.5 {a, b}

C2: 4, 1.5 {c, d}

C3: 7, 1.5

$\text{dist(e, c1)}^2 = (7 - 2)^2 + (1 - 1.5)^2 = (5)^2 + (-0.5)^2 = 25.25$

$\text{dist(e, c2)}^2 = (7 - 4)^2 + (1 - 1.5)^2 = (3)^2 + (-0.5)^2 = 9.25$

$\text{dist(e, c3)}^2 = (7 - 7)^2 + (1 - 1.5)^2 = (0)^2 + (-0.5)^2 = 0.25$

In this case, e is still closest to C3, so let's re-allocate e to C3.

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Now, we need to repeat the process of allocating our points to clusters, based on their new centroids

$$dist(i,j)^2 = (i_{x_1} - j_{x_1})^2 + (i_{x_2} - j_{x_2})^2$$

Let's move on to point f (7, 2):

Cluster Centroids:

C1: 2, 1.5 {a, b}

C2: 4, 1.5 {c, d}

C3: 7, 1.5 {e}

dist(f, c1)^2 = (7 - 2)^2 + (2 – 1.5)^2 = (5)^2 + (0.5)^2 = 25.25

dist(f, c2)^2 = (7 - 4)^2 + (2 – 1.5)^2 = (3)^2 + (0.5)^2 = 9.25

dist(f, c3)^2 = (7 - 7)^2 + (2 – 1.5)^2 = (0)^2 + (0.5)^2 = 0.25

In this case, f is still closest to C3, so let's re-allocate f to C3.

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

So after 2 iterations of assignments, our current clusters are:

C1 = {a, b}

C2 = {c, d}

C3 = {e, f}

Cluster Centroids:

C1: 2, 1.5 {a, b}

C2: 4, 1.5 {c, d}

C3: 7, 1.5 {e, f}

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Note that, at this point, our cluster allocations have not changed since the last iteration.

Therefore, re-calculating the centroids will result in the same coordinates.

At this point, we have reached convergence, and our clusters will not change anymore.

Cluster Centroids:

C1: 2, 1.5 {a, b}

C2: 4, 1.5 {c, d}

C3: 7, 1.5 {e, f}

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Last step we need to do is to calculate the SSE.

Our clusters: C1 = {a, b}; C2 = {c, d}; C3 = {e, f}

And we've already calculated dist^2 for each of these values. This is our squared error!

Cluster Centroids:

C1: 2, 1.5 {a, b}

C2: 4, 1.5 {c, d}

C3: 7, 1.5 {e, f}

To calculate SSE for each cluster:

$$\mathrm{SSE}_{C_k} = \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

K-Means: Step-by-Step

Point	x_1	x_2
a	2	1
b	2	2
c	4	1
d	4	2
e	7	1
f	7	2

Last step we need to do is to calculate the SSE.

$$\text{SSE}_{C_k} = \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

SSE_{c1} = dist(a, c1)^2 + dist(b, c1)^2 = .25 + .25 = .5

SSE_{c2} = dist(c, c2)^2 + dist(d, c2)^2 = .25 + .25 = .5

SSE_{c3} = dist(e, c3)^2 + dist(f, c3)^2 = .25 + .25 = .5

Cluster Centroids:

C1: 2, 1.5 {a, b}

C2: 4, 1.5 {c, d}

C3: 7, 1.5 {e, f}

Our overall SSE = SSE_{c1} + SSE_{c2} + SSE_{c3} = 1.5

K-Means – Summary

K-Means – Pros

Pros:

- It is simple, fast to compute, and incredibly well-studied
- It converges to local minimum of within-cluster squared error

K-Means – Problem Cases

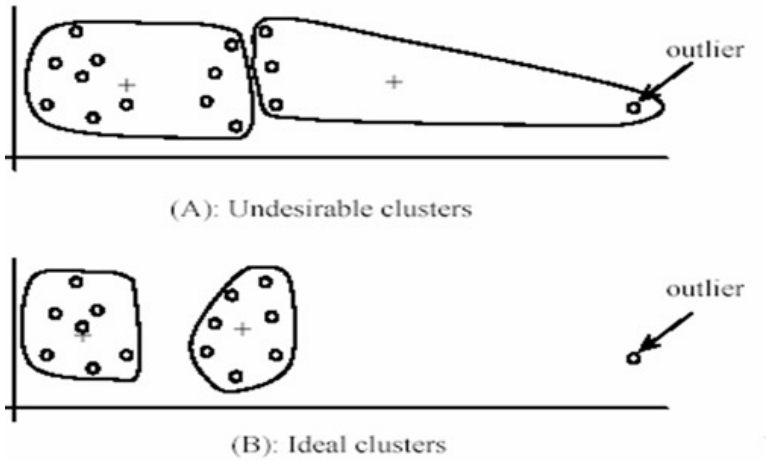
Problem cases:

- Setting K requires some prior knowledge, thought (or luck) to get right.
- Only allows for hard assignments of data points to clusters – a small shift of a data point can flip it to a different cluster
- It is sensitive to the initial centroids that are selected

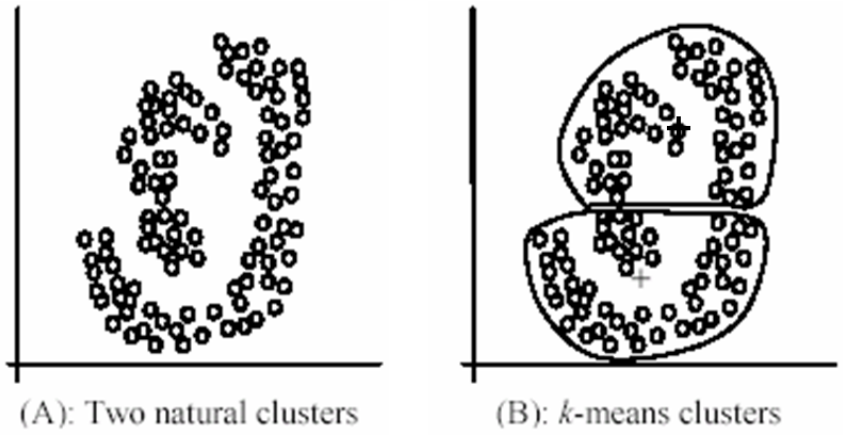
K-Means – Problem Cases

Problem cases:

It is sensitive to outliers:



It only detects spherical clusters:



K-Means ++

K-Means ++ – Smarter Initialization

K-means++ is an algorithm for choosing the initial values (or seeds) for k-Means clustering algorithm.

By doing so, we can eliminate one downside of K-Means clustering: That it is dependent on the initial cluster centroids.

K-Means ++

The exact algorithm is as follows:

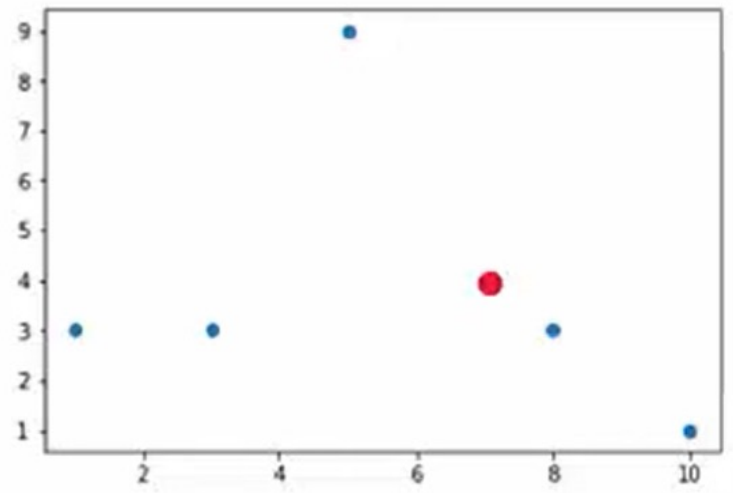
1. Choose the first centroid uniformly at random from the data points.
2. For each remaining data point, calculate the distance squared (usually the Euclidean distance) to the nearest existing centroid.
3. Choose the next centroid from the data points with probability proportional to the distance squared. This means that data points farther away from existing centroids are more likely to be selected as new centroids.
4. Repeat steps 2 and 3 until all K centroids have been chosen.

K-Means ++

Suppose we have a dataset, to which we want to apply 3 clusters.

In the beginning, we randomly select (7,4) to be a cluster centre:

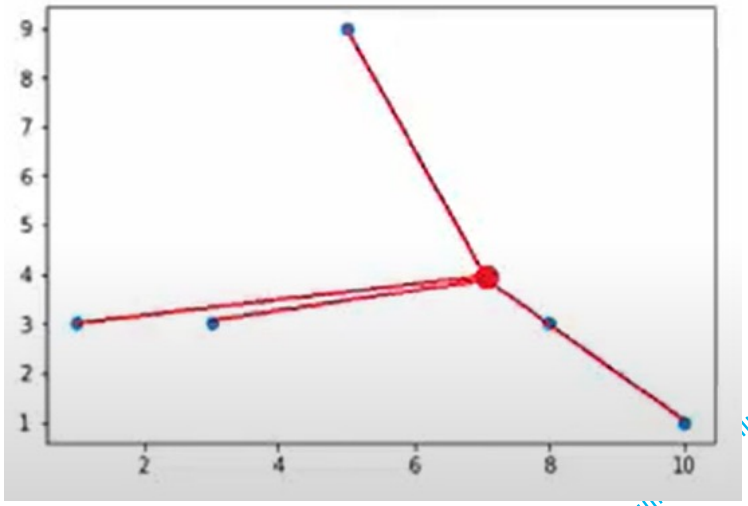
x	
(7,4)	-
(8,3)	
(5,9)	
(3,3)	
(1,3)	
(10,1)	



K-Means ++

We calculate all the distances between the other point and the selected centre:

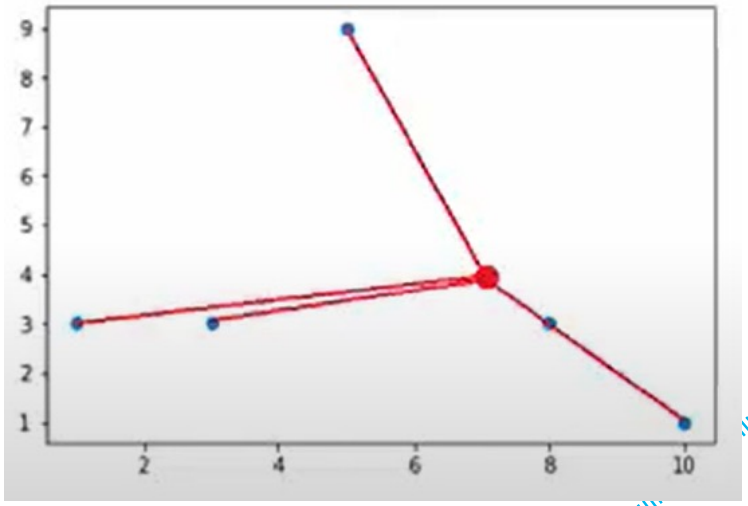
x	$min\ (d(x, z_j)^2)$
(7,4)	-
(8,3)	2
(5,9)	29
(3,3)	17
(1,3)	37
(10,1)	18



K-Means ++

And work out the probabilities for each other point to be chosen as the next cluster centroid:

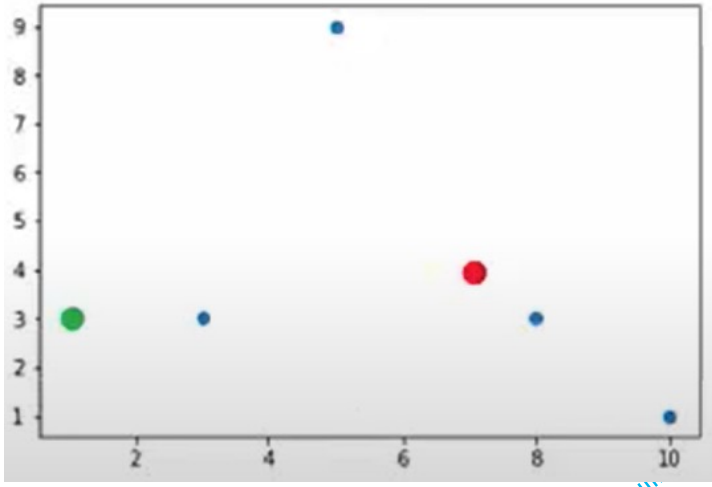
x	p
(7,4)	-
(8,3)	2/103
(5,9)	29/103
(3,3)	17/103
(1,3)	37/103
(10,1)	18/103



K-Means ++

Let’s add (1, 3) to the list of cluster centres:

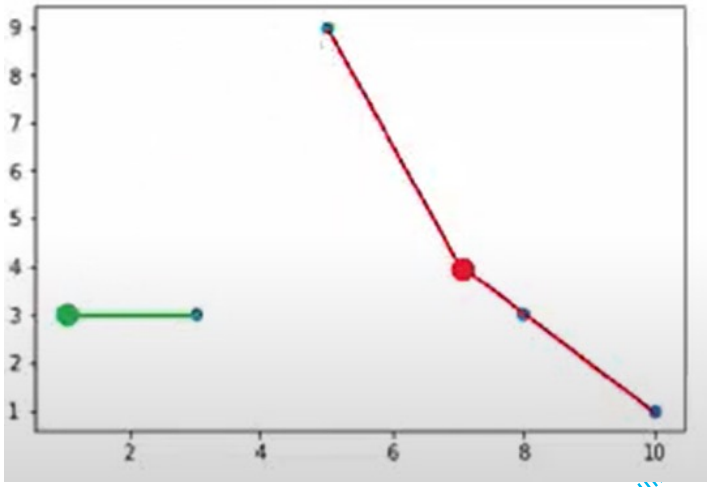
x	
(7,4)	-
(8,3)	
(5,9)	
(3,3)	
(1,3)	-
(10,1)	



K-Means ++

For each remaining point, we compute the distance to the nearest centre:

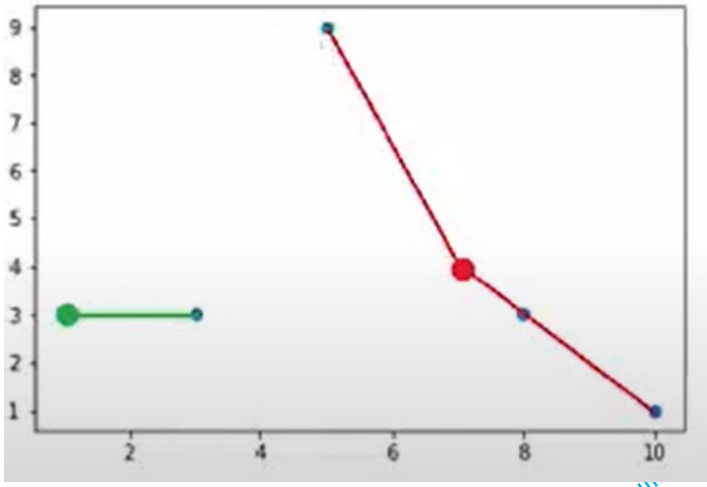
x	$min\ (d(x, z_j)^2)$
(7,4)	-
(8,3)	2
(5,9)	29
(3,3)	4
(1,3)	-
(10,1)	18



K-Means ++

Then work out the probabilities to be chosen for the next cluster centroid:

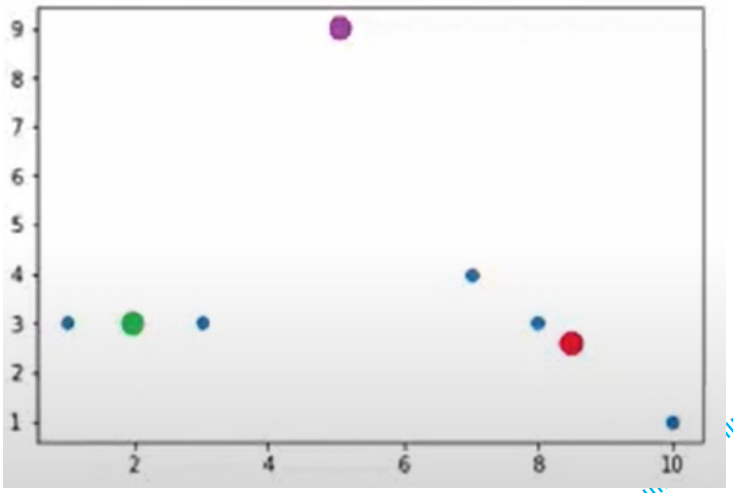
x	p
(7,4)	-
(8,3)	2/53
(5,9)	29/53
(3,3)	4/53
(1,3)	-
(10,1)	18/53



K-Means ++

Let’s add (5, 9) to the list of cluster centres:

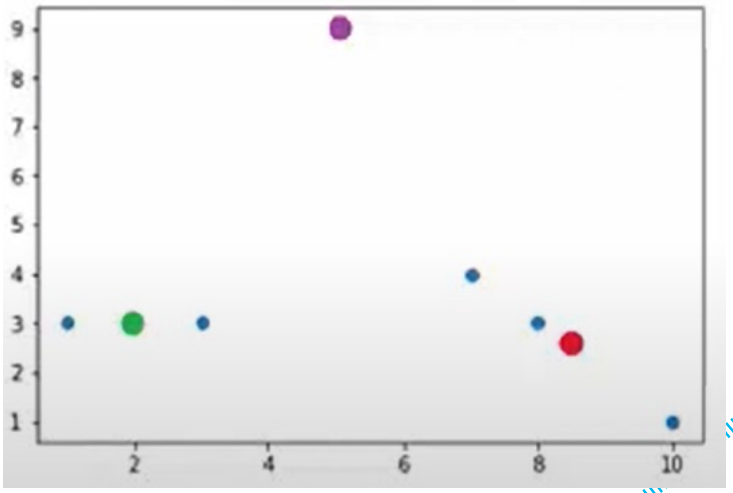
x	
(7,4)	-
(8,3)	
(5,9)	-
(3,3)	
(1,3)	-
(10,1)	



K-Means ++

Now, we've finished the process of initialization, and can run K-Means with the centroids: (7, 4), (5, 9) and (1, 3).

x	
(7,4)	-
(8,3)	
(5,9)	-
(3,3)	
(1,3)	-
(10,1)	



K-Means ++

This process has a setup cost, but convergence tends to be faster and better than standard K-Means clustering.

x	
(7,4)	-
(8,3)	
(5,9)	-
(3,3)	
(1,3)	-
(10,1)