Categorization with *k*-Means Clustering

CISC 3225
Spring 2024
DSFS 20, PDSH 47

Machine learning crash course

Two basic approaches:

1. **Supervised learning**: We have data labeled with the correct answer
2. **Unsupervised learning**: The data is unlabeled

k-means clustering is an example of unsupervised learning.

Irises unsupervised

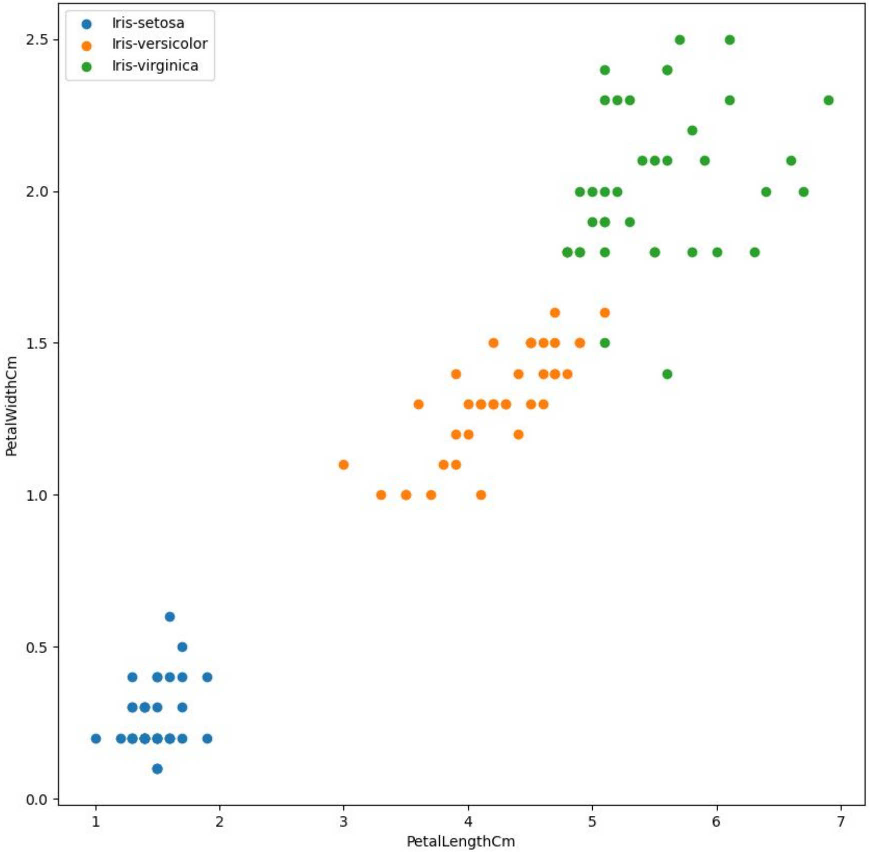
This is a **supervised** problem

Observations Label

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	6.9	3.1	5.1	2.3	Iris-virginica
1	6.2	2.2	4.5	1.5	Iris-versicolor
2	6.9	3.1	5.4	2.1	Iris-virginica
3	5.4	3.9	1.3	0.4	Iris-setosa
4	5.1	3.5	1.4	0.2	Iris-setosa

...

Our data



Irises unsupervised

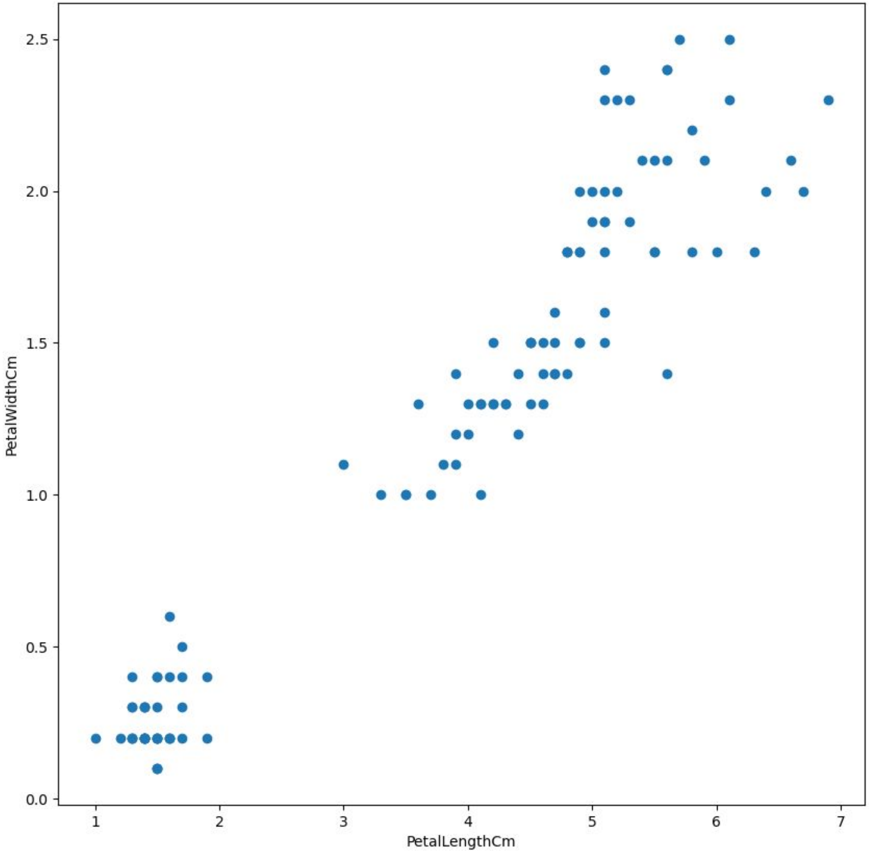
This is an **unsupervised** problem: there are no species labels!

Observations

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
0	6.9	3.1	5.1	2.3
1	6.2	2.2	4.5	1.5
2	6.9	3.1	5.4	2.1
3	5.4	3.9	1.3	0.4
4	5.1	3.5	1.4	0.2

…

Our data

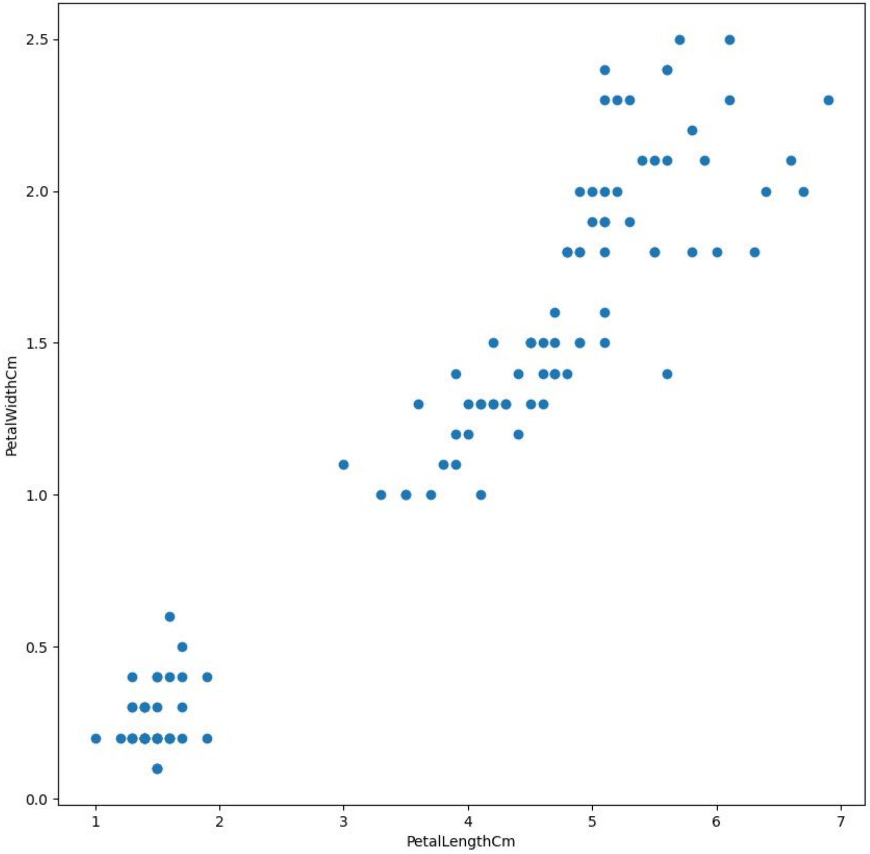


k-means clustering

k-means clustering is an **unsupervised** technique that allows us to identify **clusters** of similar points.

Clusters are defined by a *centroid*: the mean of all points in the cluster

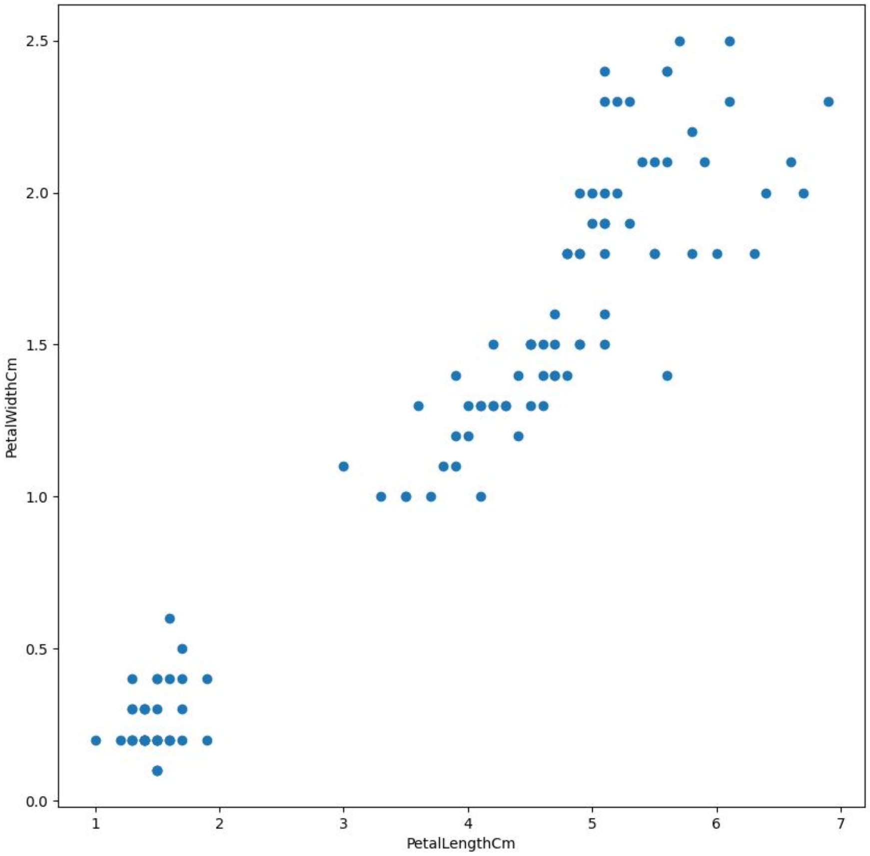
We have *k* centroids corresponding to *k* clusters.



k-means clustering

Uses for *k*-means clustering:

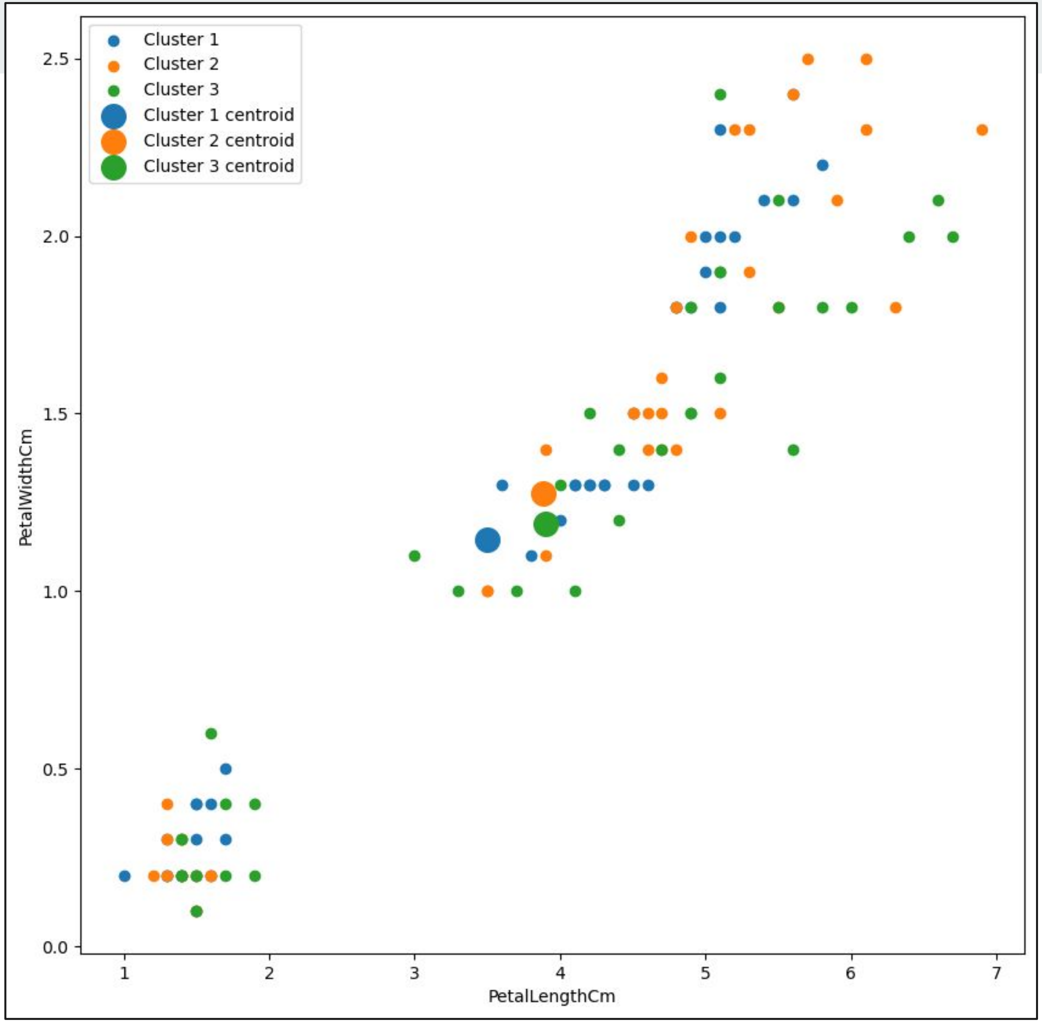
- Customer segmentation for advertising
- Cybersecurity and IT: Use clusters to identify outliers in your data
- Document clustering: Find groups of related articles for recommendation



Clustering algorithm

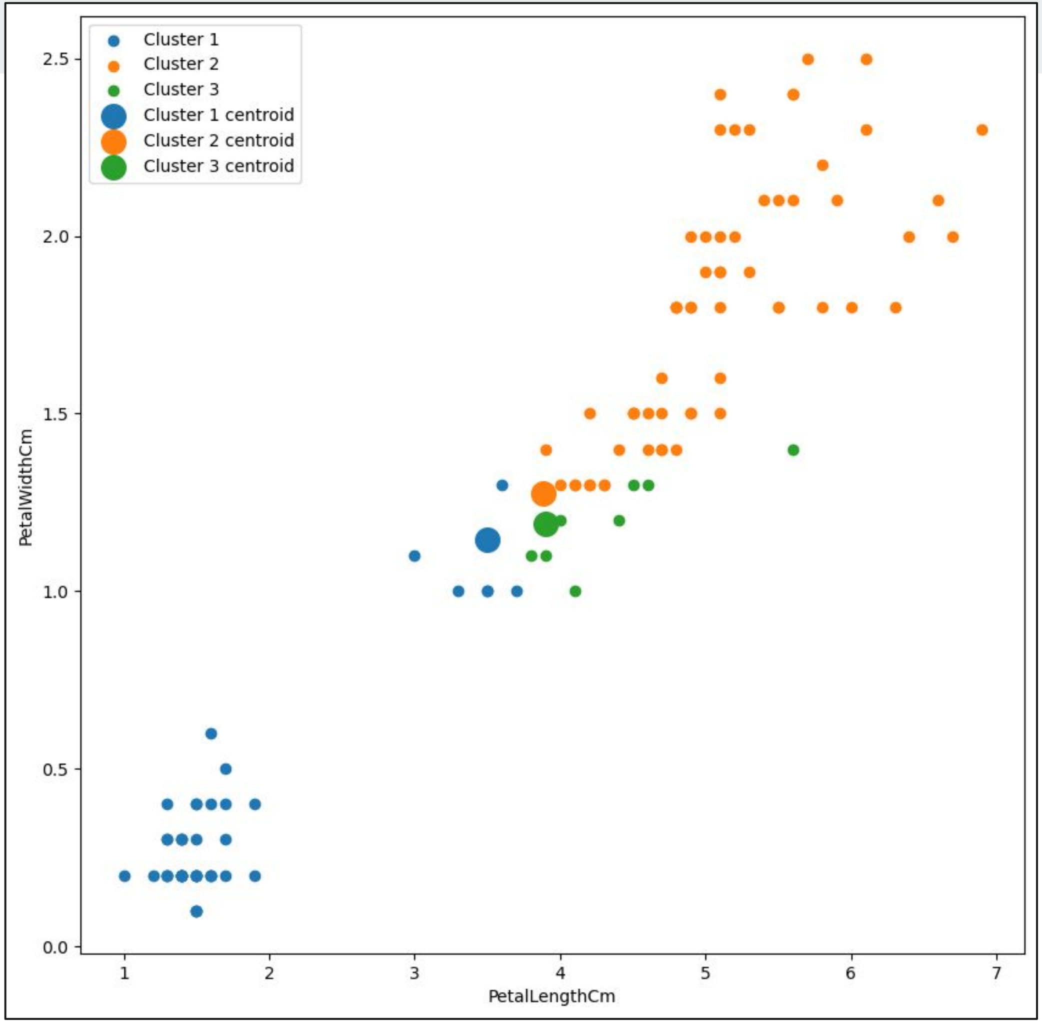
Assign clusters randomly (k=3)

Compute centroids as the mean of all points in the cluster



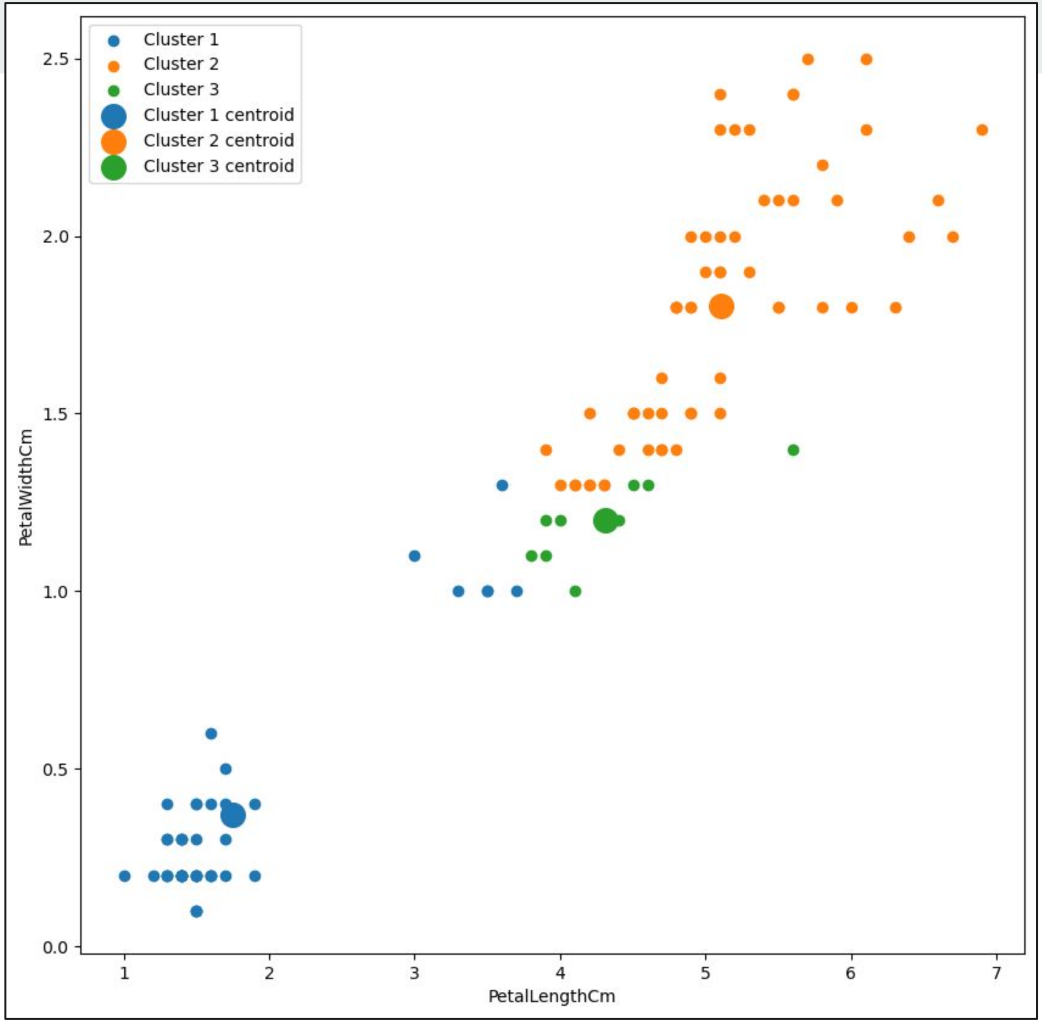
Clustering algorithm

Reassign points based on nearest centroid



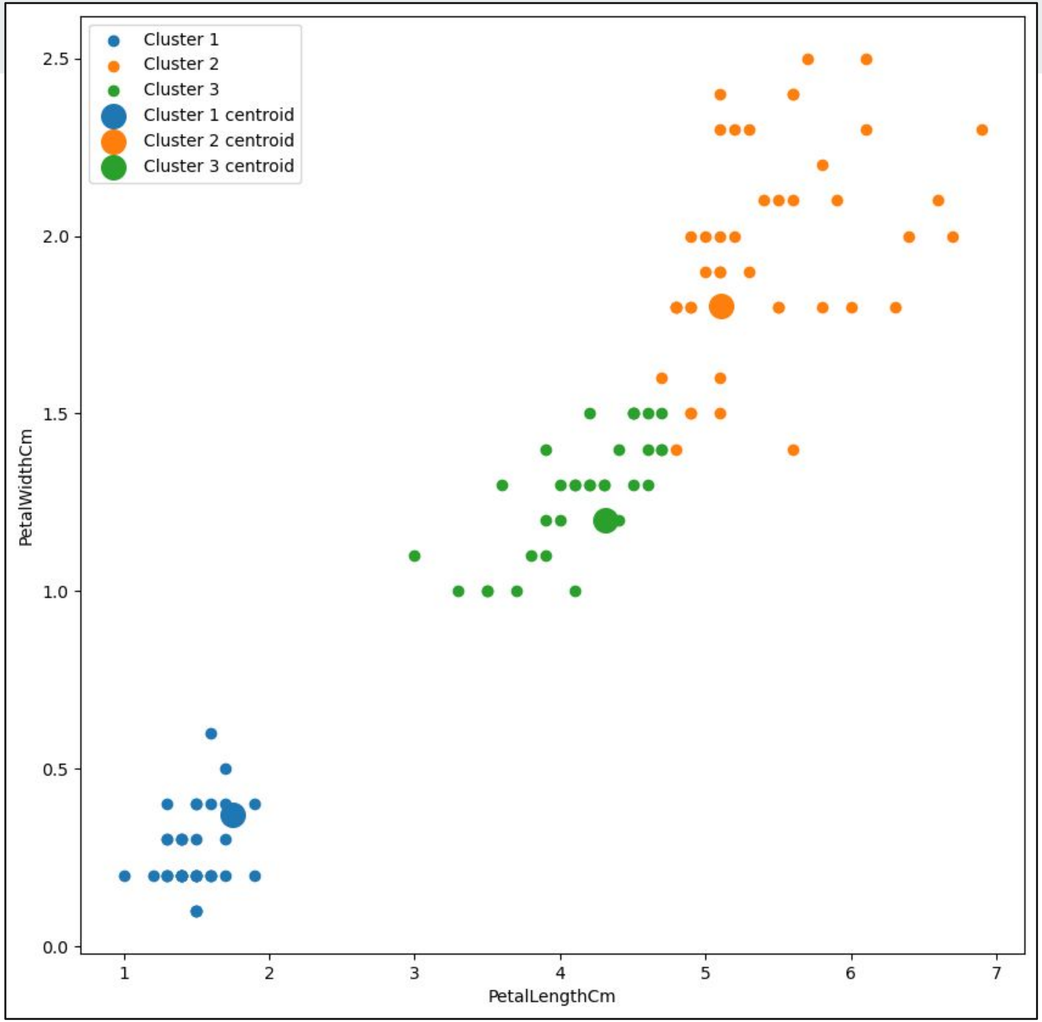
Clustering algorithm

Recompute centroids as the mean of all points in the cluster



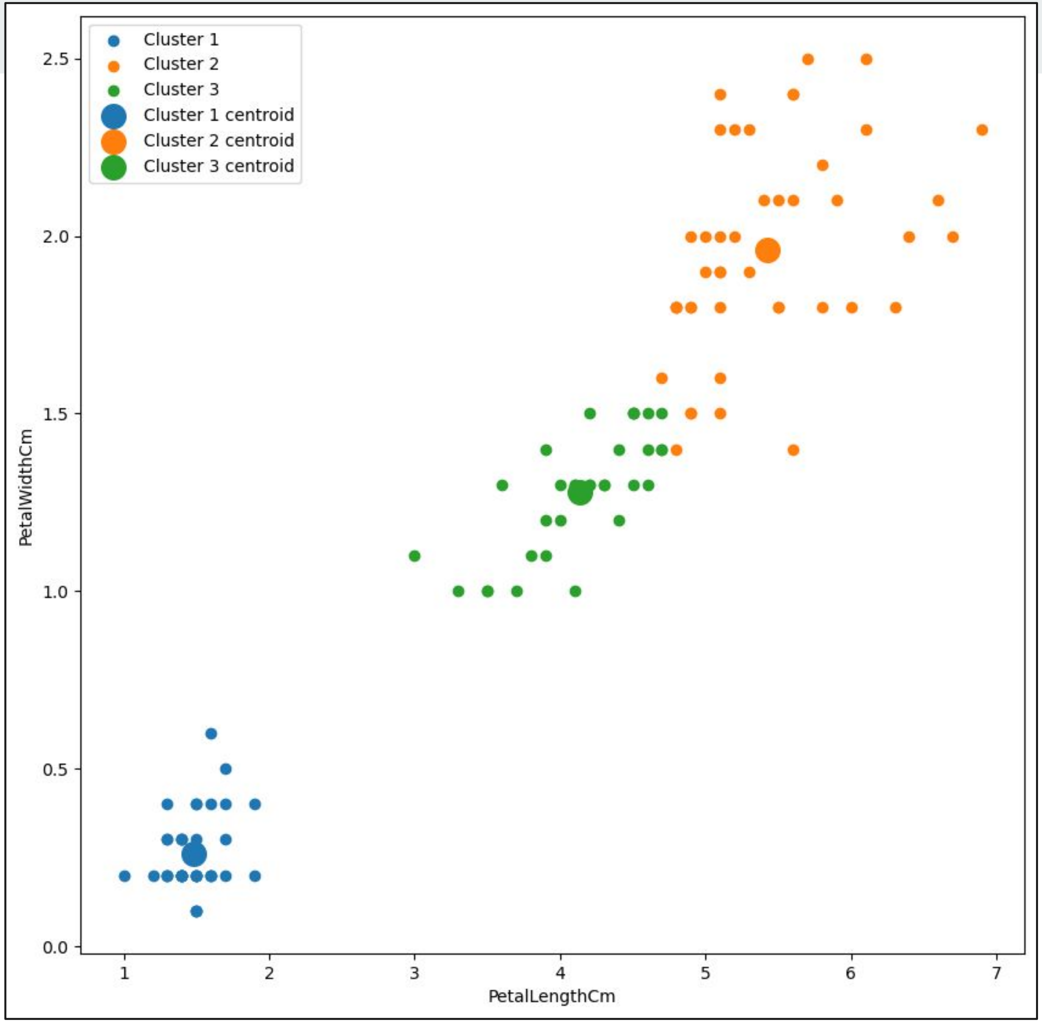
Clustering algorithm

Reassign points based on nearest centroid



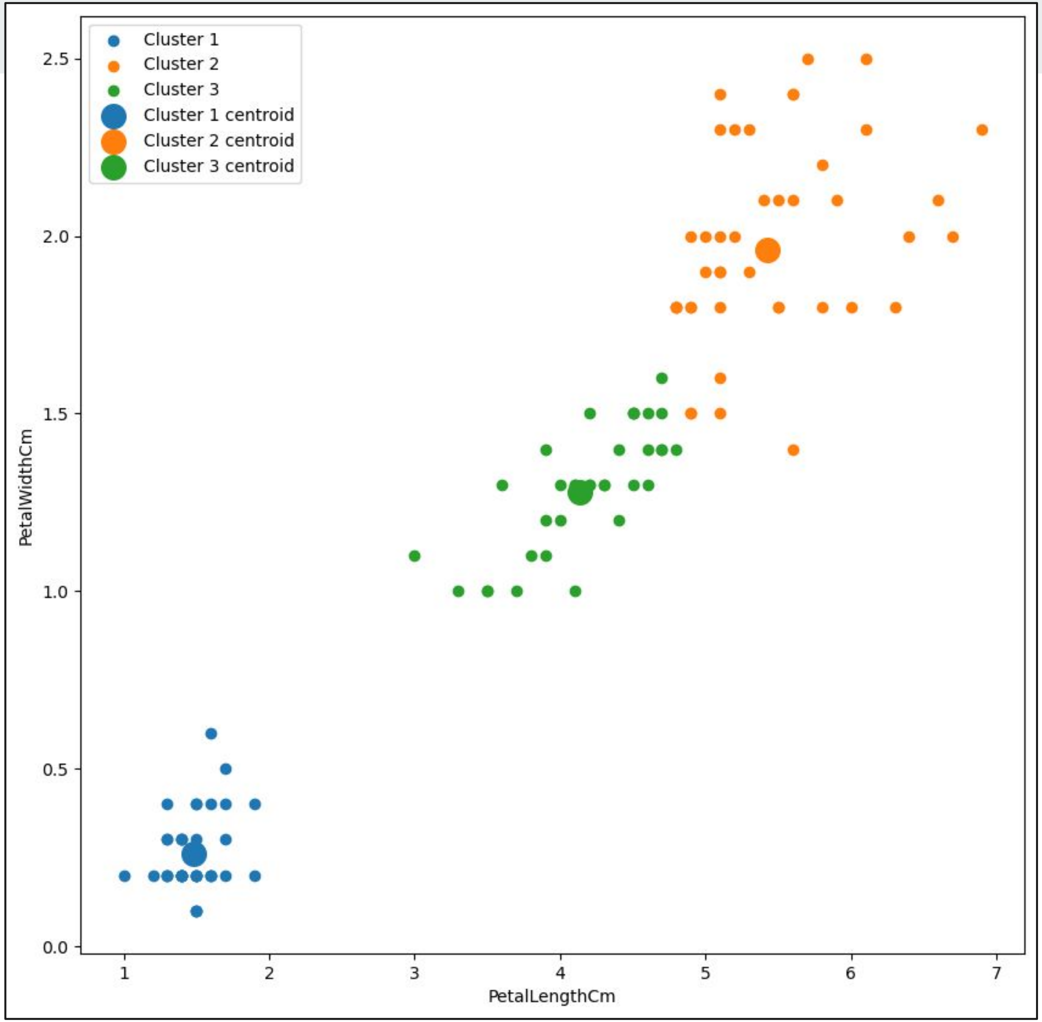
Clustering algorithm

Recompute centroids as the mean of all points in the cluster



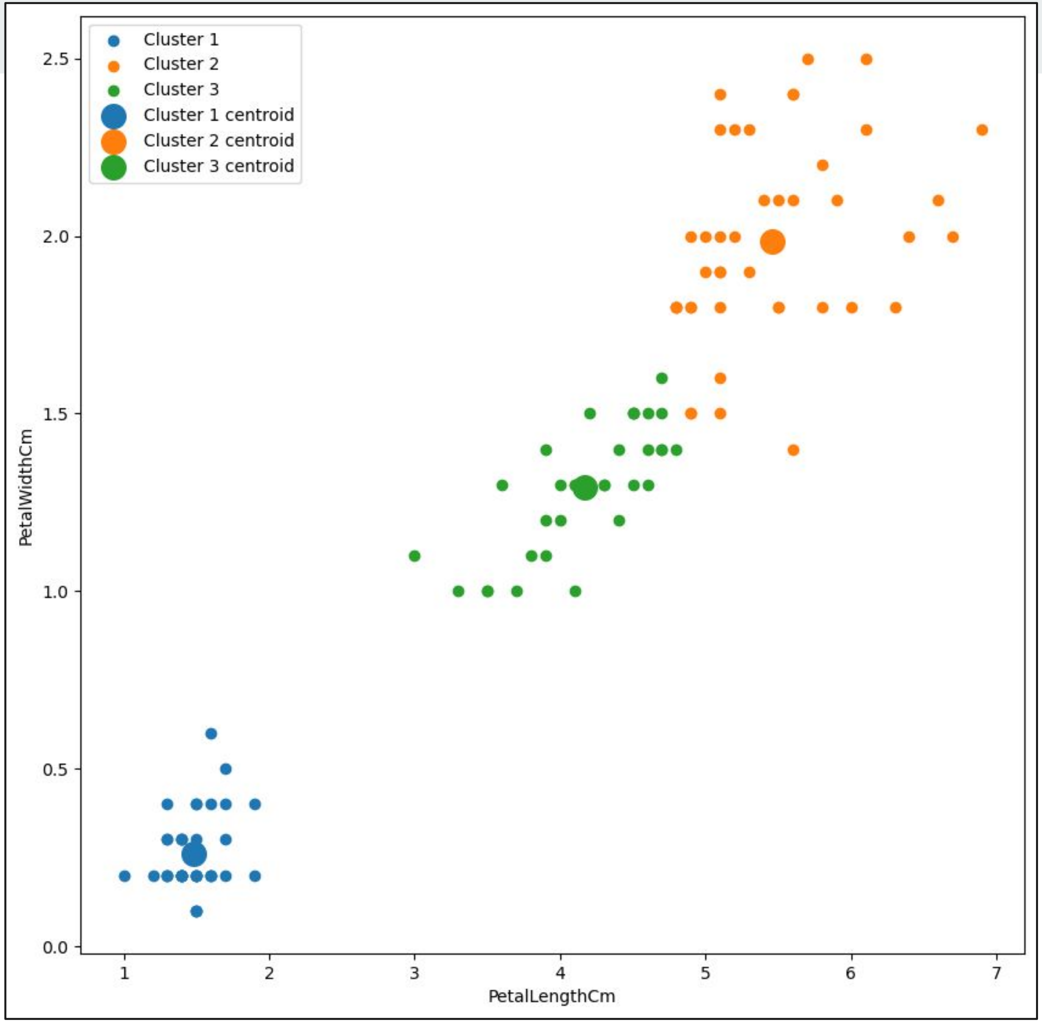
Clustering algorithm

Reassign points based on nearest centroid



Clustering algorithm

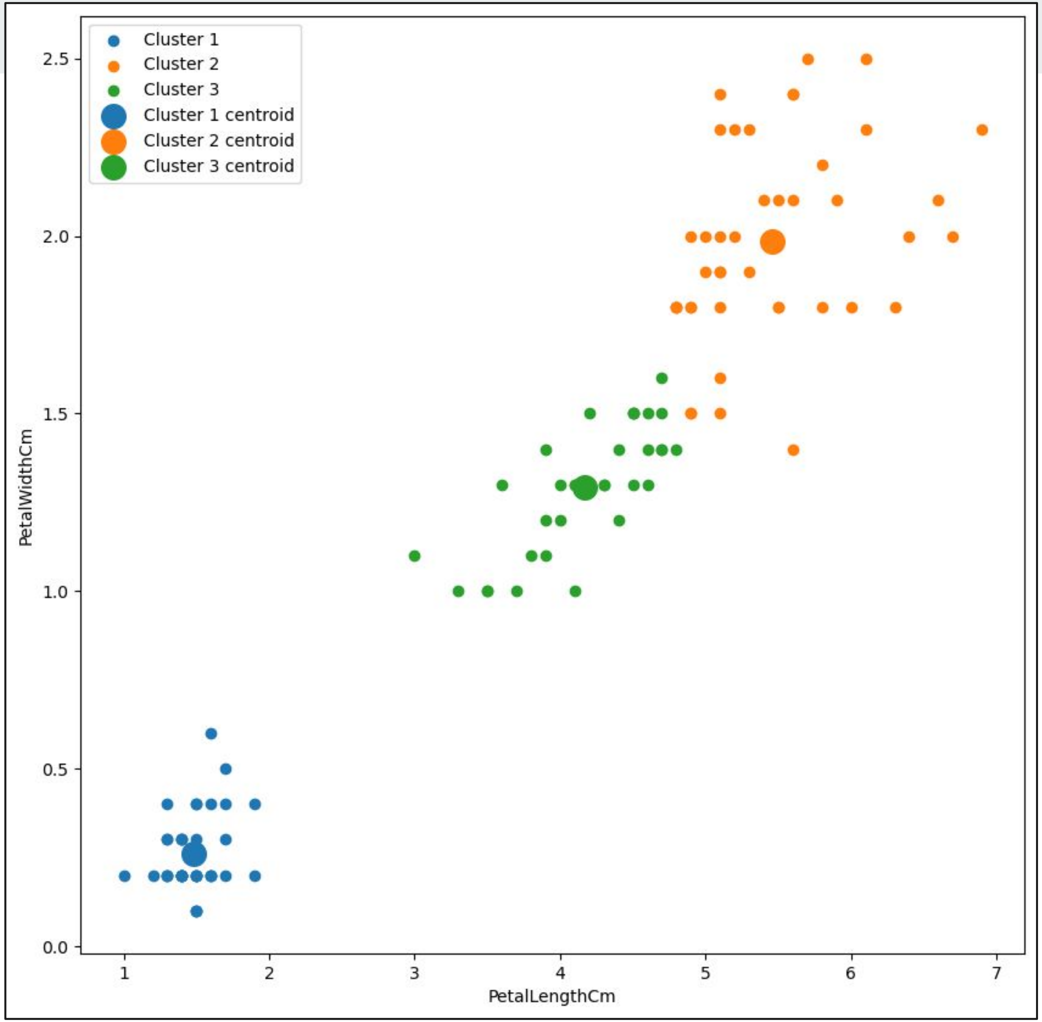
Recompute centroids as the mean of all points in the cluster



Clustering algorithm

Reassign points based on nearest centroid

Nothing changed: we're done



k-Means Clustering Algorithm

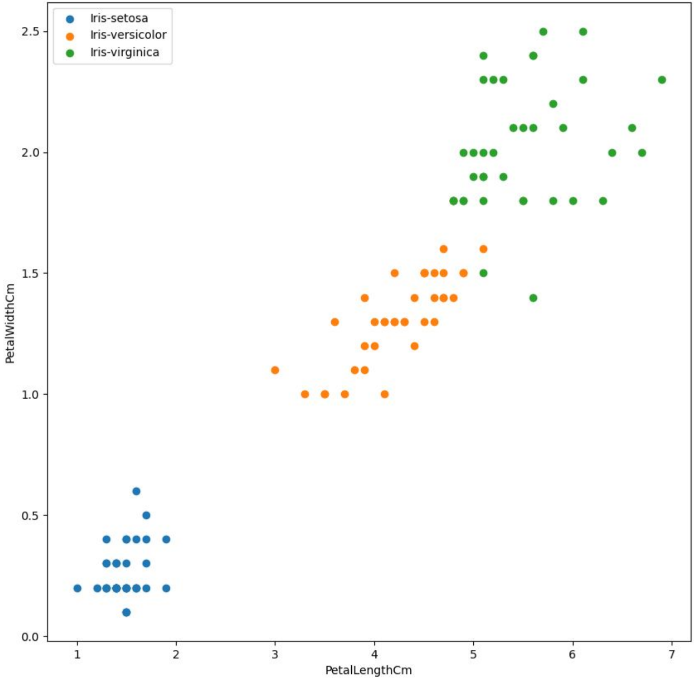
$$d(\mathbf{p},\ \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

1. Assign clusters randomly to all instances in the dataset
2. Repeat until cluster assignments do not change:
   a. Recompute centroids as the mean of each point in the cluster
   b. Reassign points based on the nearest centroid (Euclidean distance)

Implementation

In Colab

k-Means Clustering Caveats

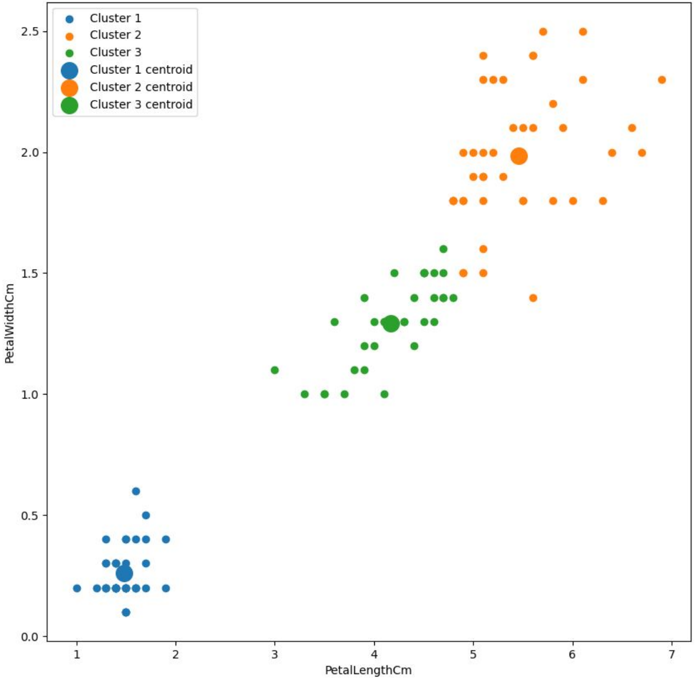


k-means clustering **does not discover class labels!**

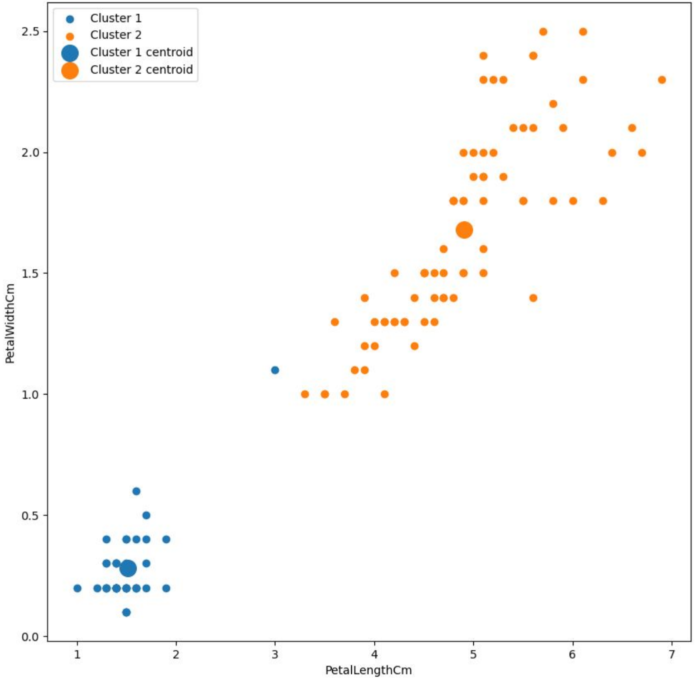
Instead, it groups similar points together.

If class labels are available, **clusters may not be homogeneous!**

Iris dataset: **We cannot confuse clusters with species!**



k-Means Clustering Caveats

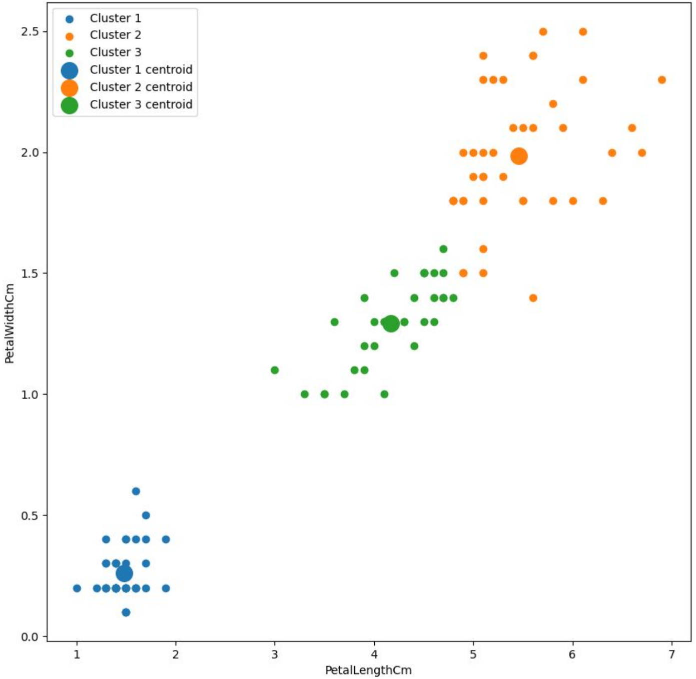


The clusters discovered by *k*-means clustering *depend on the value of k*.

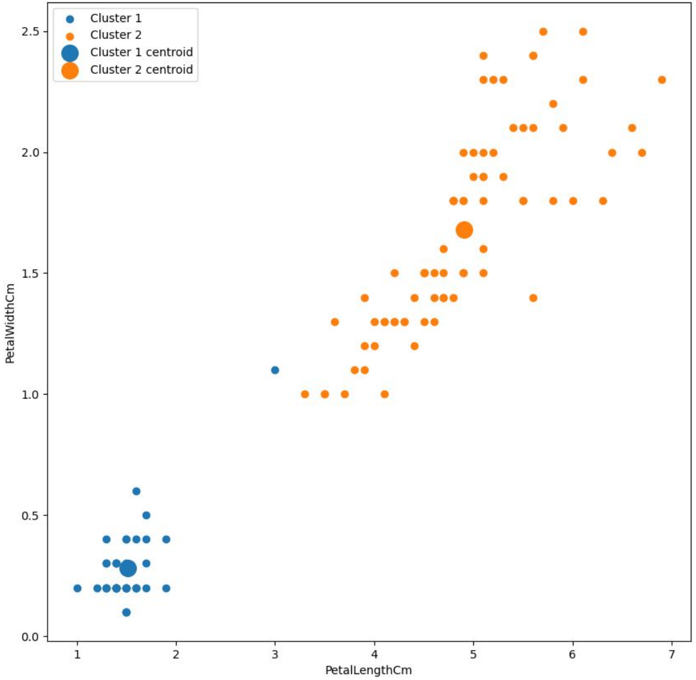
Left: *k*=2

Right: *k*=3

Neither is more or less correct than the other.



Evaluating *k*-means clustering

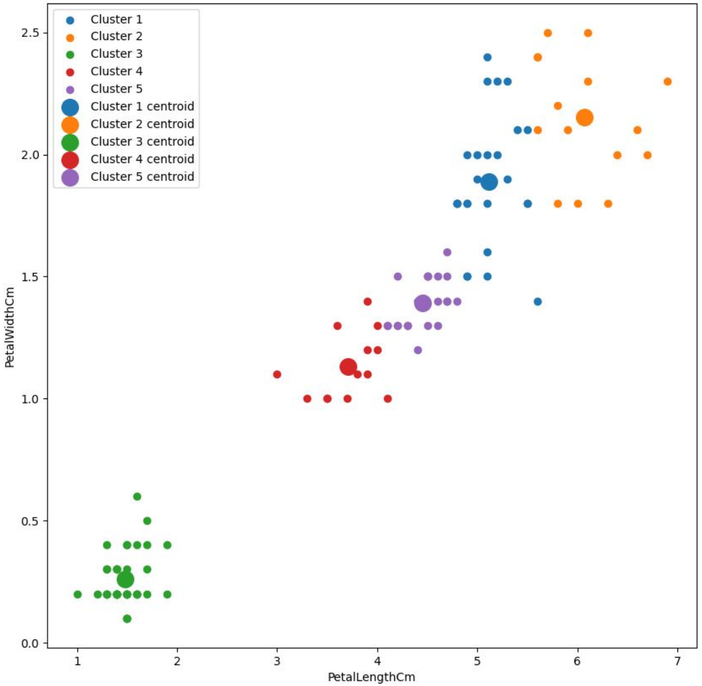


Left: k=2

Right: k=5

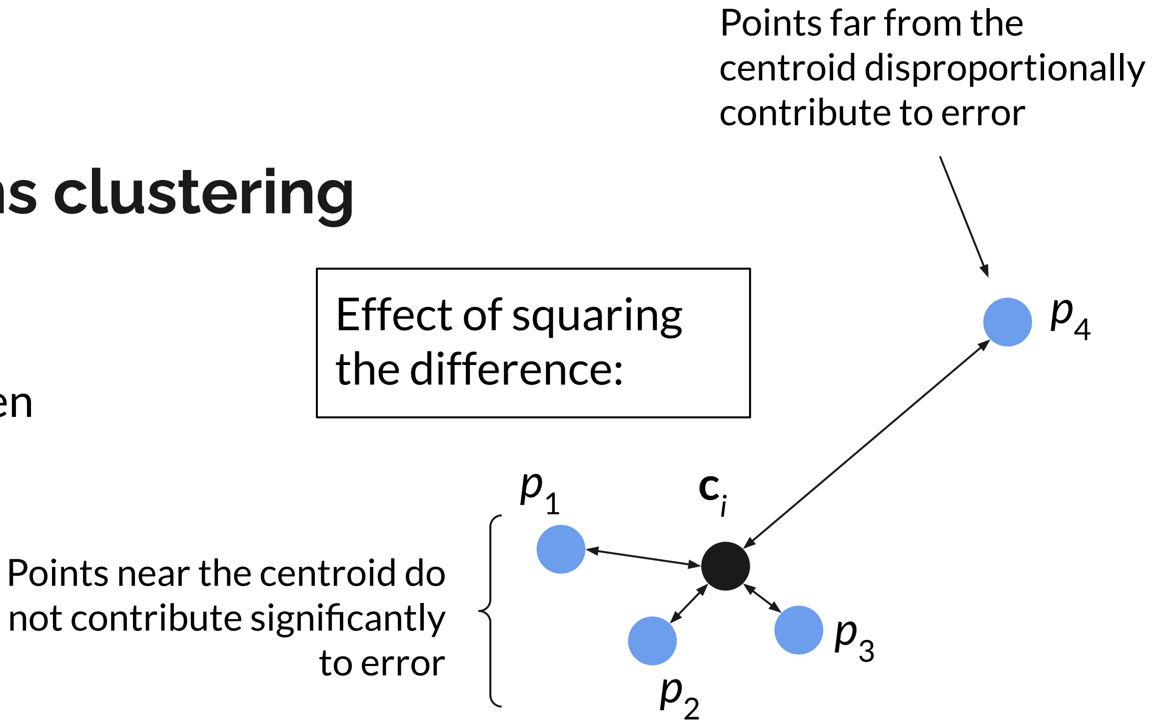
Which is a better clustering?

Intuition says the left, because there are visibly not 5 distinct groups of points. How to calculate numerically?

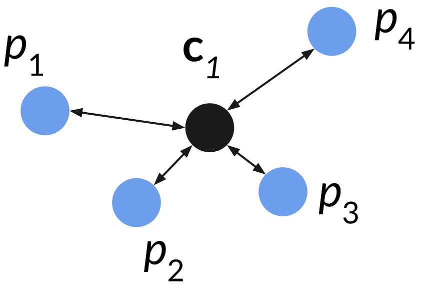


Evaluating *k*-means clustering

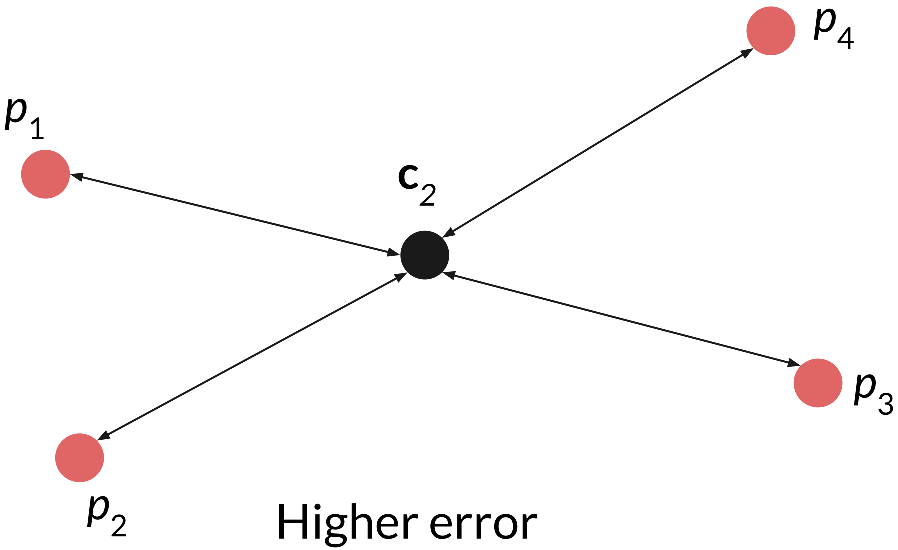
Idea: **sum of squared error** between centroids and their nearest points.



Evaluating *k*-means clustering

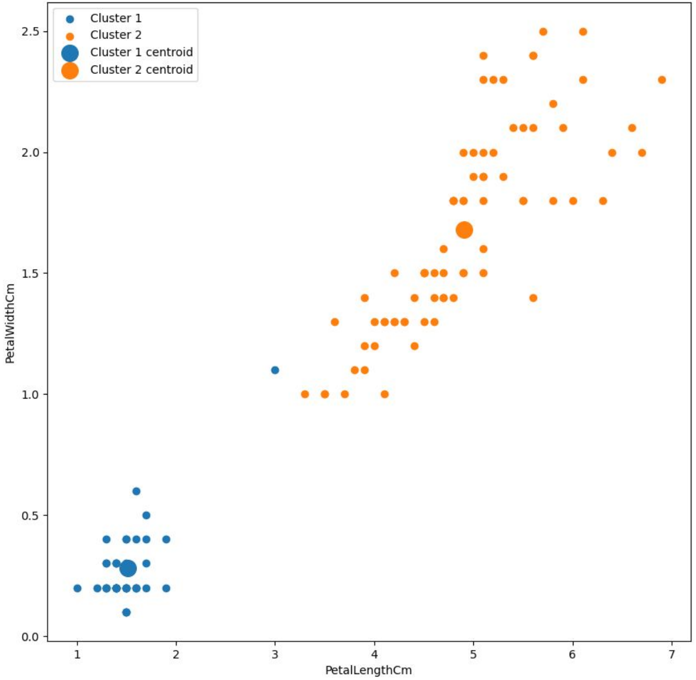


Lower error



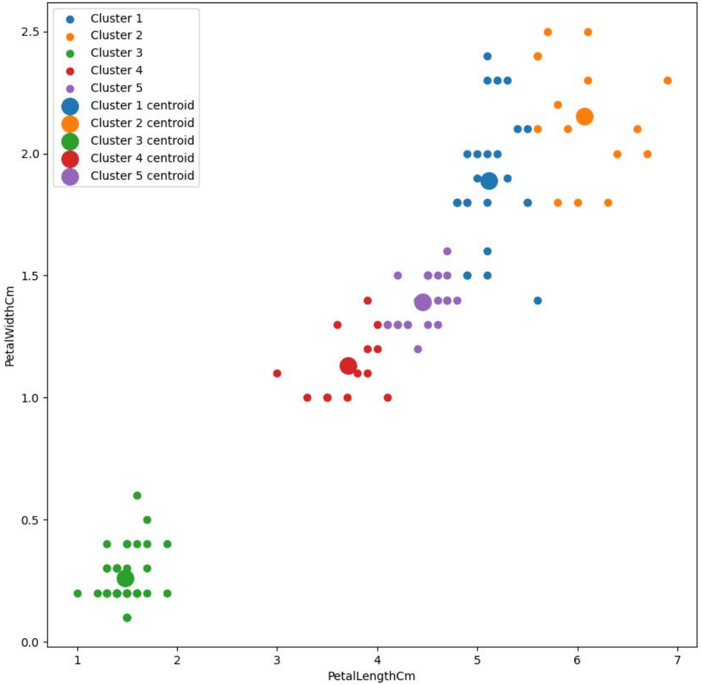
Higher error

Evaluating *k*-means clustering



k=2: Higher overall error, because clusters are very spread out. Most points are far from the centroid.

k=5: Lower overall error, because clusters are very compact. Most points are near the centroid.



Evaluating *k*-means clustering

Error associated with each value of *k* should be plotted.

Look for an "elbow": a place where the error stops decreasing as much. This is usually a good starting point for *k*.