Unsupervised Learning

Stephen G. Ware

CSCI 4525 / 5525

Unsupervised Learning

Sometimes labeled data is not available, but we still want to look for patterns.

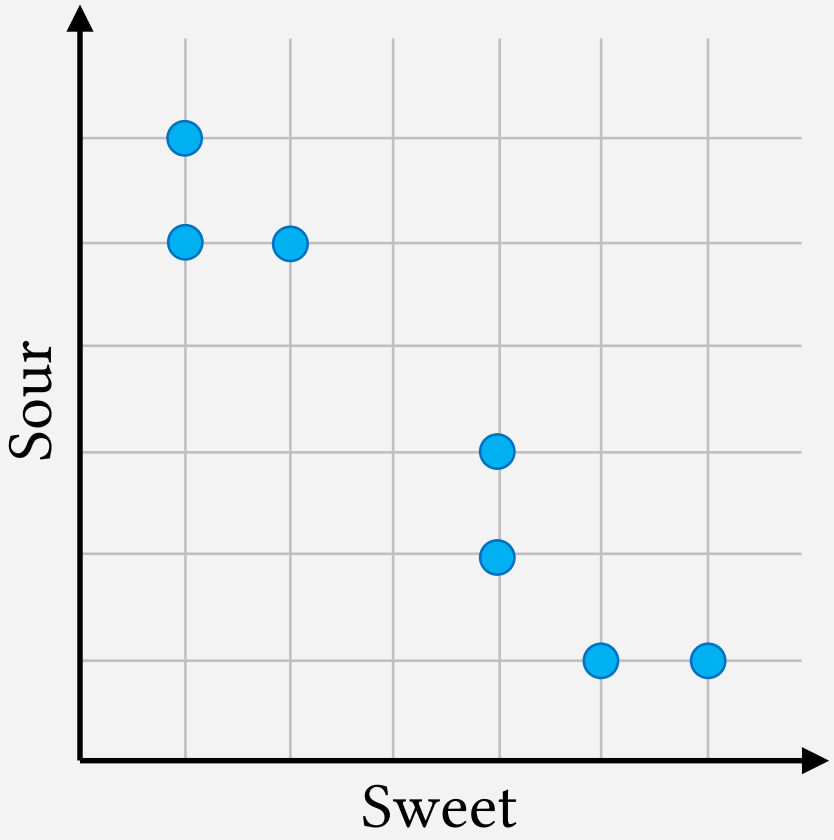
- We don't know if labels exist.
- We know that labels exist, but we don't know which data points have which labels.

Clustering

Clustering is an unsupervised learning technique which gathers similar examples into groups (or clusters). These groups can then be used directly or analyzed post-hoc to determine their significance.

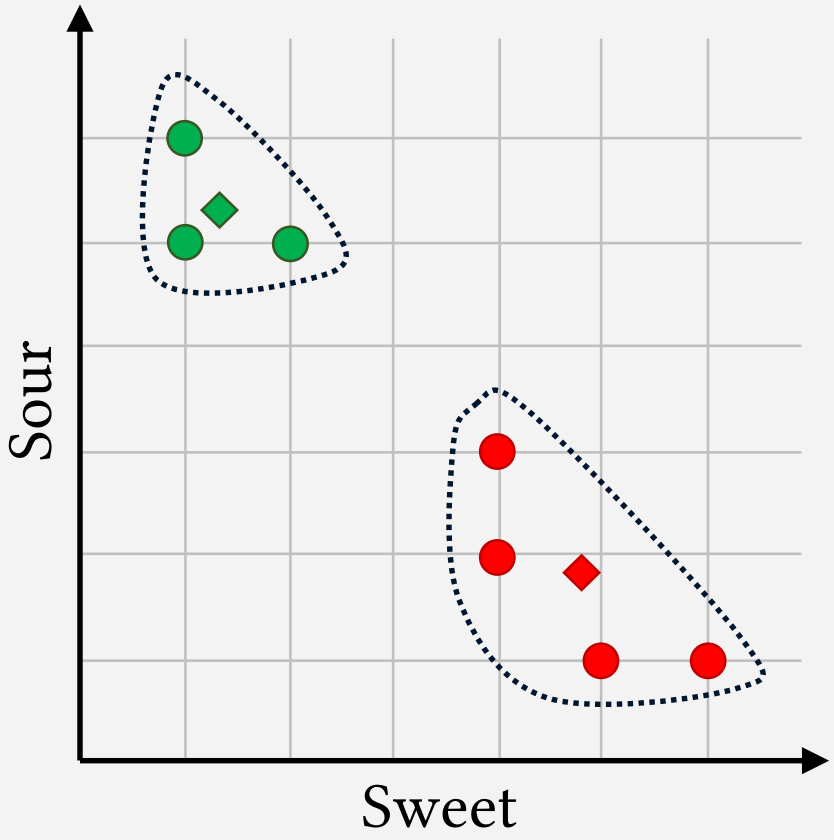
We will consider the simple ***k*-means clustering** technique which automatically discovers k clusters. Again, the value of k is arbitrary, and this technique is often repeated with multiple values of k until satisfactory results are found.

Fruit Data Set



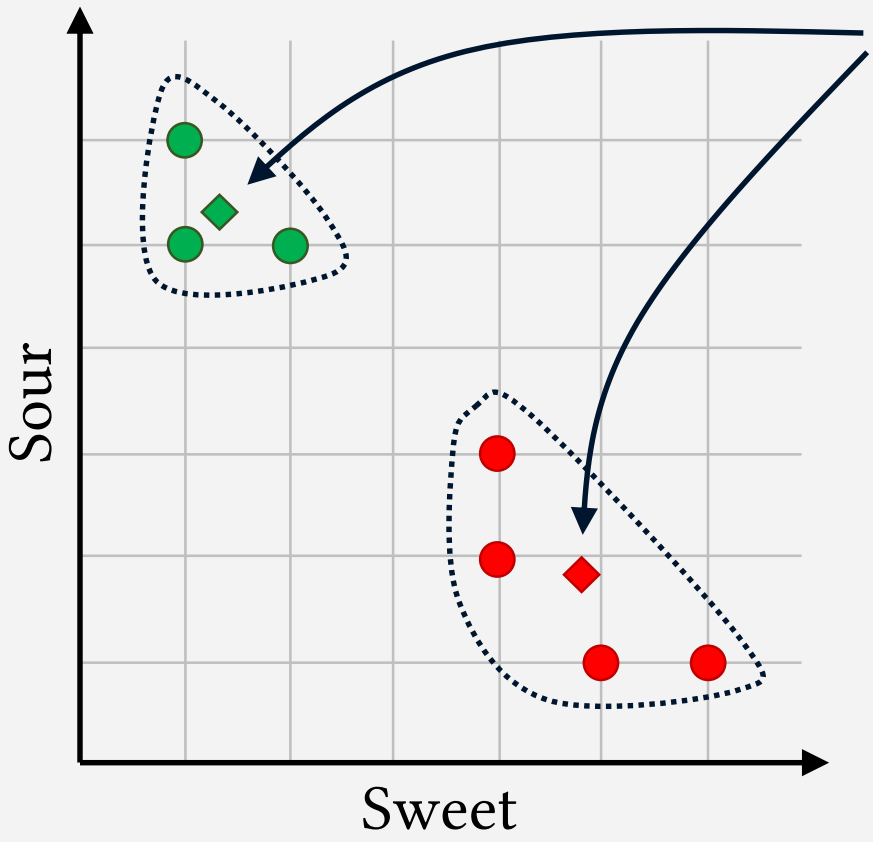
Fruit	Sweet	Sour
Lemon	1	5
Lime	1	6
Grapefruit	2	5
Cantaloupe	4	2
Orange	4	3
Honeydew	5	1
Watermelon	6	1

Fruit Data Set



Fruit	Sweet	Sour
Lemon	1	5
Lime	1	6
Grapefruit	2	5
Cantaloupe	4	2
Orange	4	3
Honeydew	5	1
Watermelon	6	1

Fruit Data Set



The **centroid** of a cluster is the point which minimizes the average distance to all other points in the cluster.

Fruit Data Set

Note how these clusters roughly correspond to the citrus and melon fruits (with Orange misclassified as a melon).

These labels were not provided, so the clusters are just green and red, but we can see their significance after the fact.

Fruit	Sweet	Sour
Lemon	1	5
Lime	1	6
Grapefruit	2	5
Cantaloupe	4	2
Orange	4	3
Honeydew	5	1
Watermelon	6	1

Lloyd's Algorithm

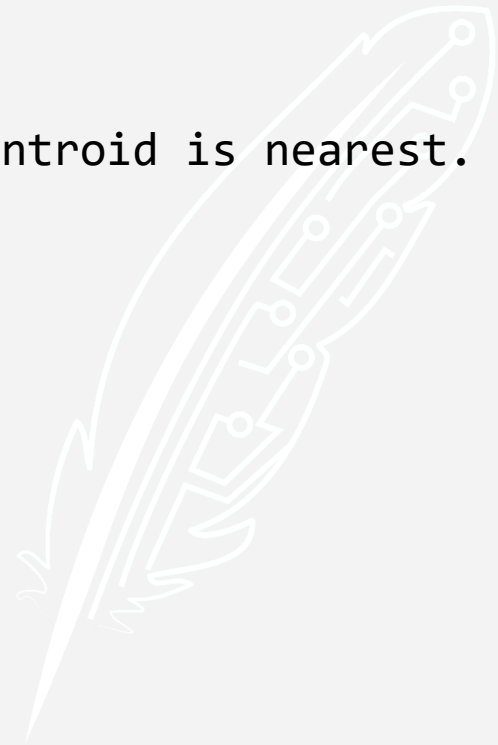
Randomly assign each example to a cluster.

Loop:

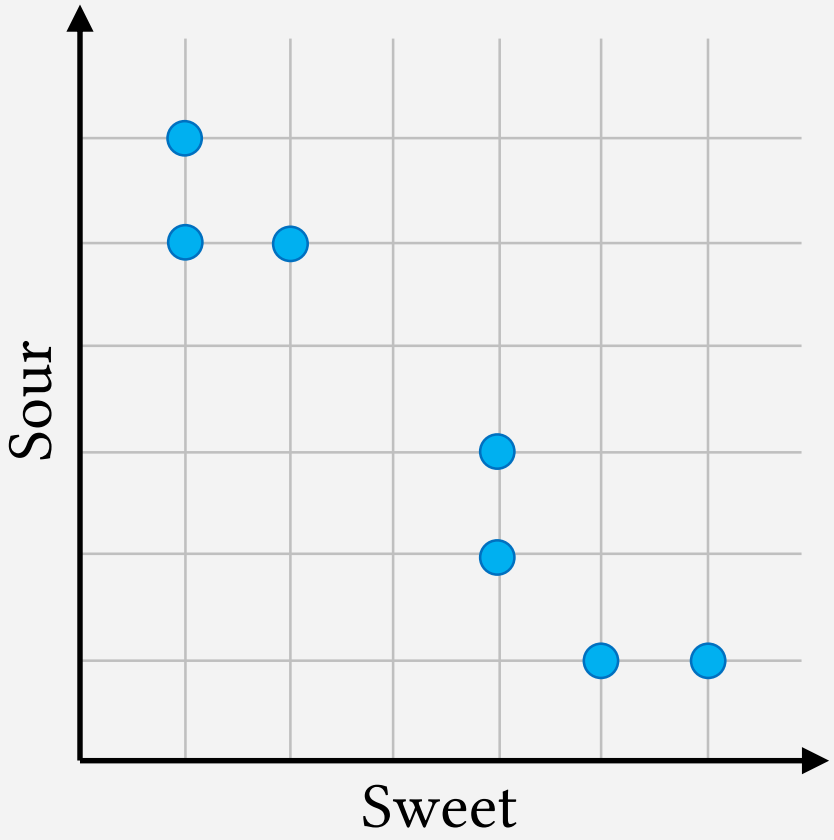
    Recalculate the centroids for each cluster.

    Reassign each point to the cluster whose centroid is nearest.

    If no points have changed clusters, stop.

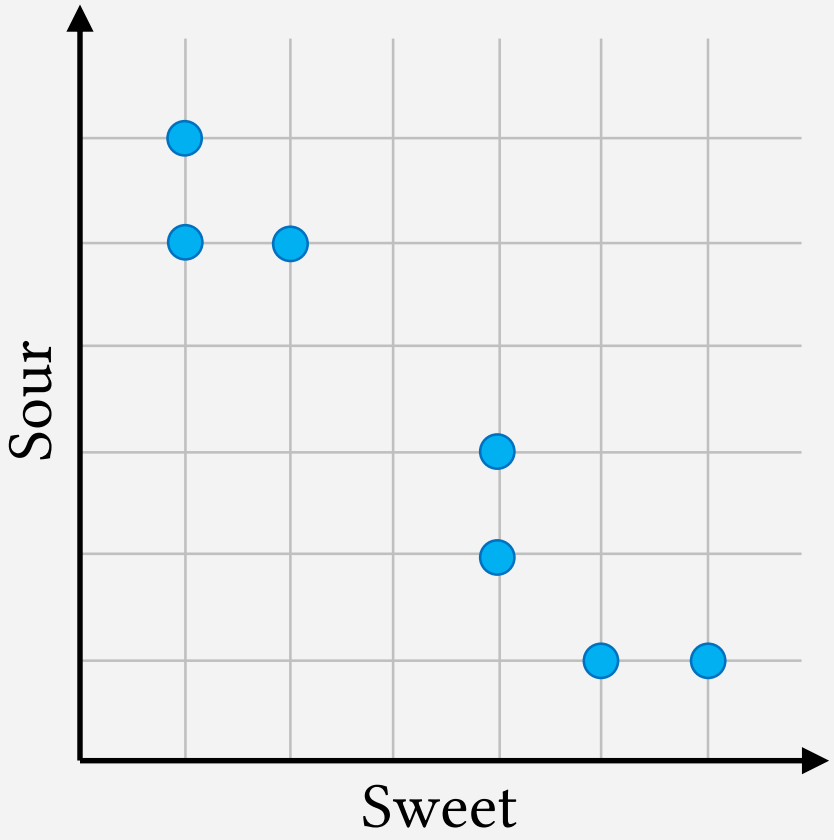


k-Means Example



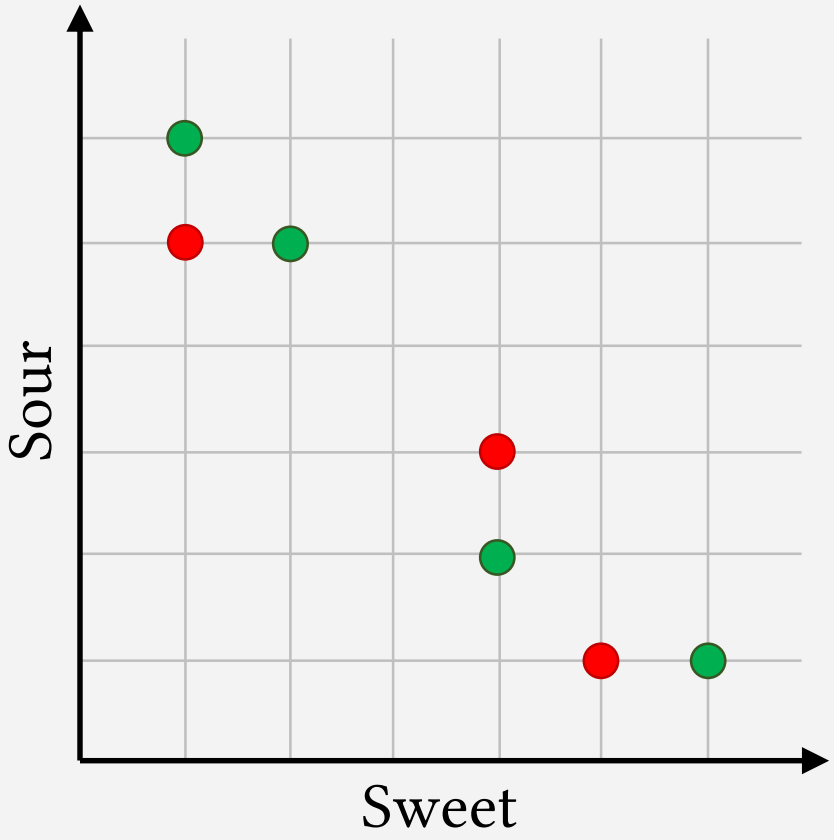
In this example $k = 2$. This means that there will be 2 clusters, labeled green and red.

k-Means Example



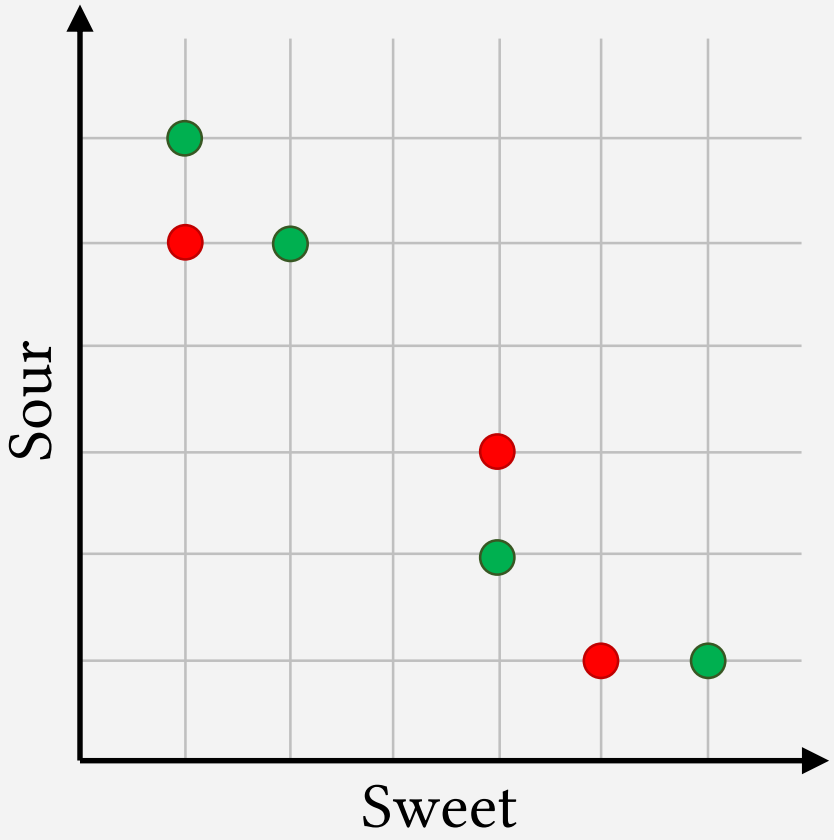
Start by assigning each example to a random cluster.

k-Means Example



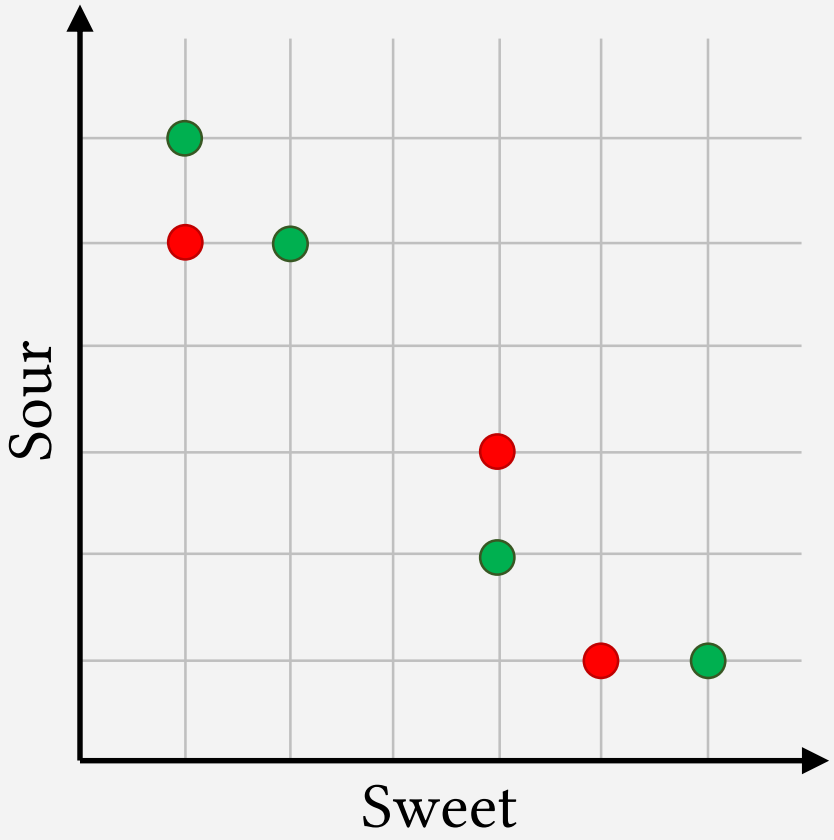
Start by assigning each example to a random cluster.

k-Means Example



Calculate the centroids of each cluster.

k-Means Example



Calculate the centroids of each cluster.

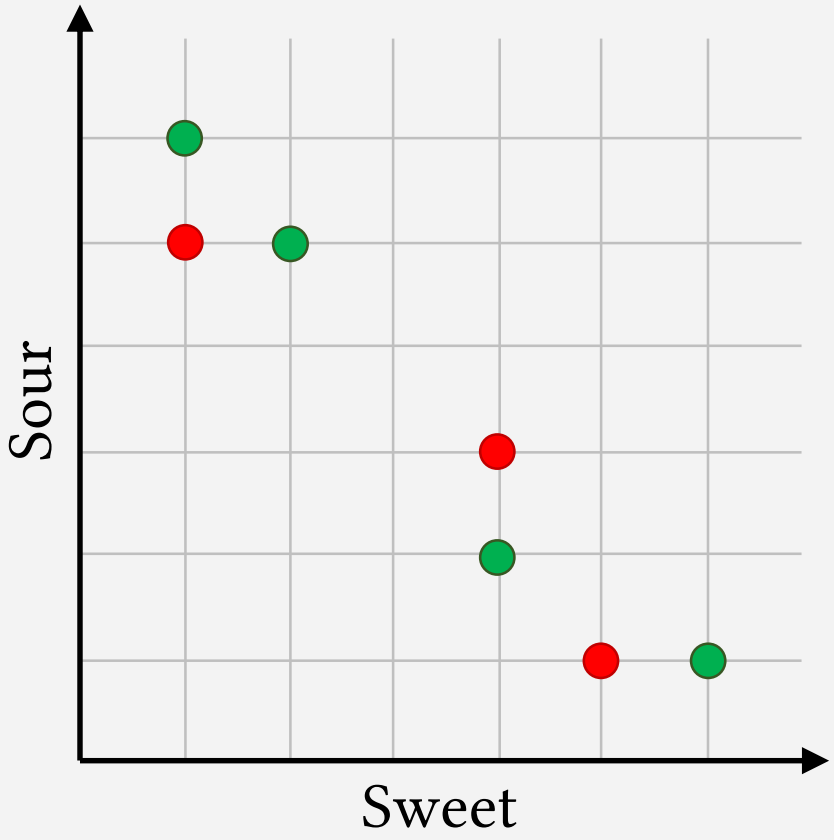
green =

$$\frac{[1,6] + [2,5] + [4,2] + [6,1]}{4}$$

red =

$$\frac{[1,5] + [4,3] + [5,1]}{3}$$

k-Means Example

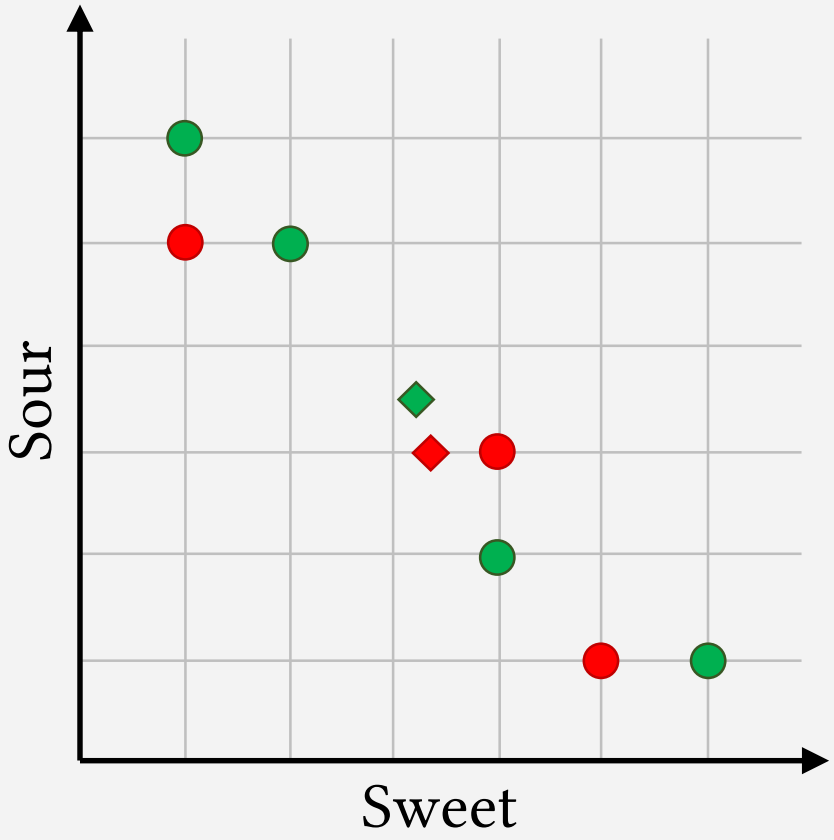


Calculate the centroids of each cluster.

green = $\frac{[13,14]}{4}$

red = $\frac{[10,9]}{3}$

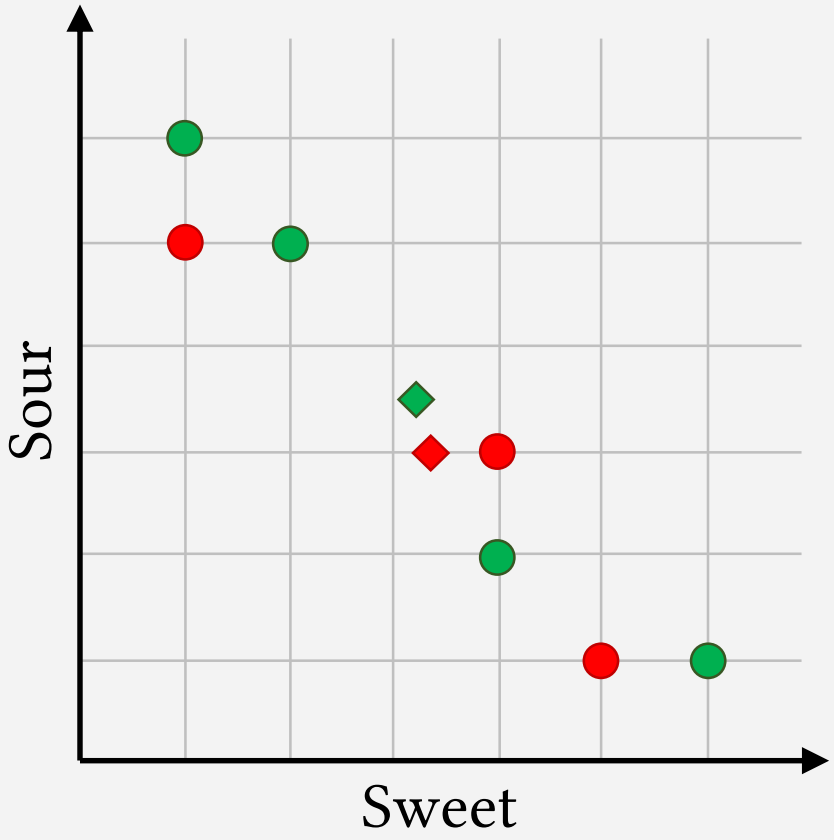
k-Means Example



Calculate the centroids of each cluster.

- Green Centroid = [3.25,3.50]
- Red Centroid = [3.33,3.00]

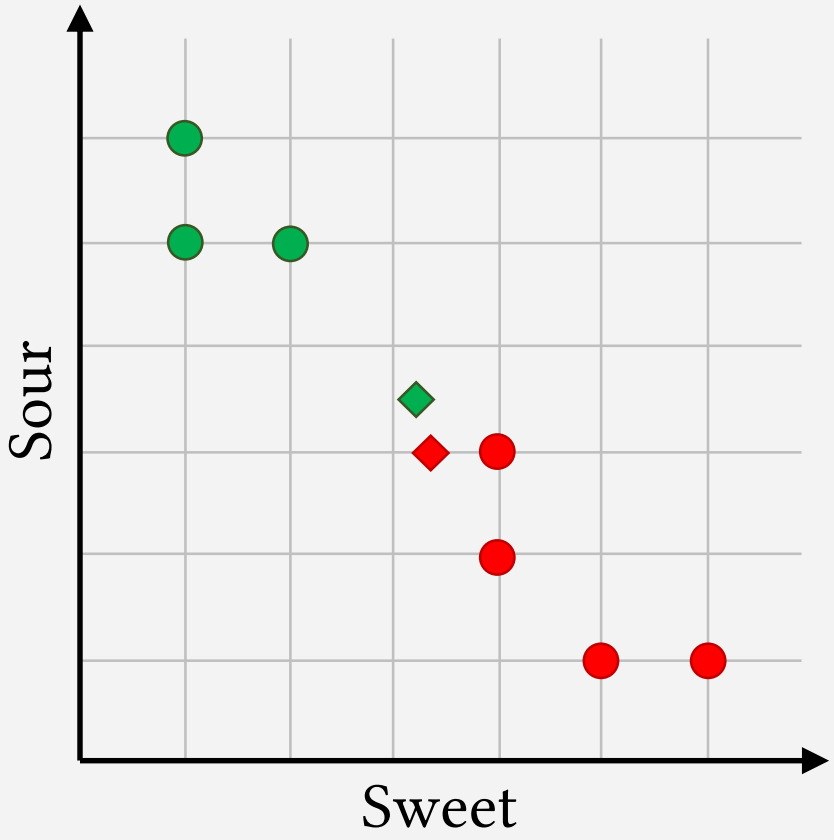
k-Means Example



Reassign each example to the cluster whose centroid is nearest.

- Green Centroid = [3.25,3.50]
- Red Centroid = [3.33,3.00]

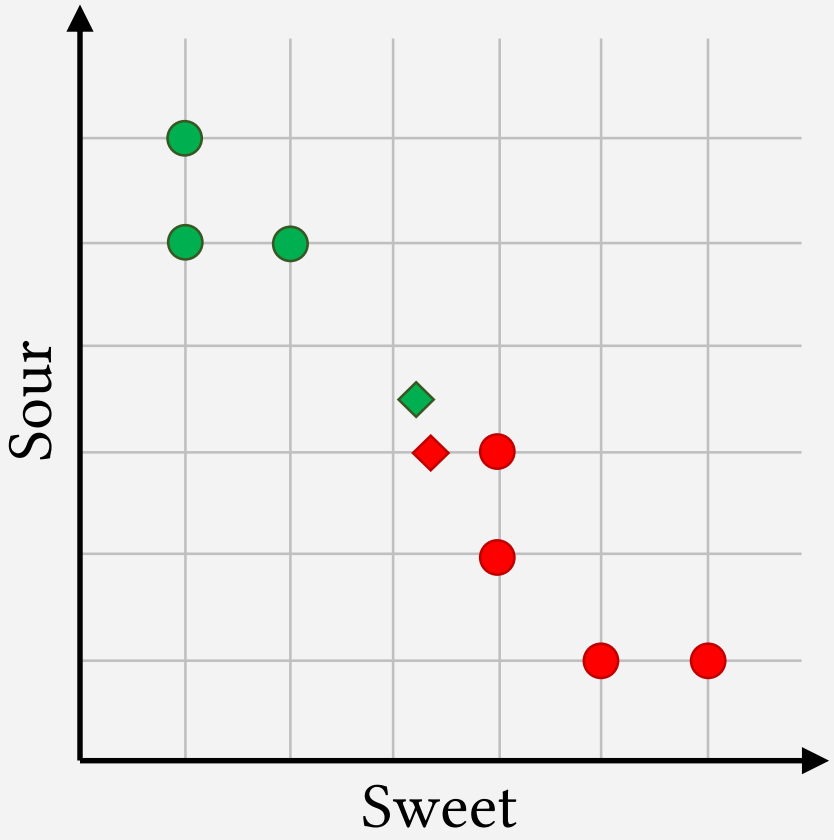
k-Means Example



Reassign each example to the cluster whose centroid is nearest.

- Green Centroid = [3.25,3.50]
- Red Centroid = [3.33,3.00]

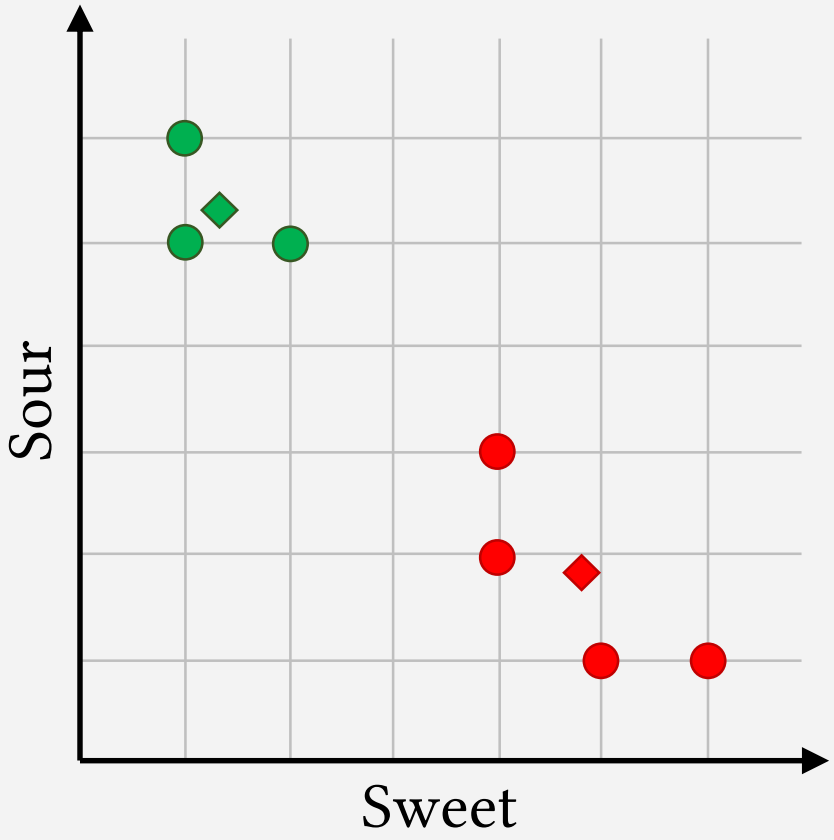
k-Means Example



Recalculate the centroid of each cluster.

- Green Centroid = [3.25,3.50]
- Red Centroid = [3.33,3.00]

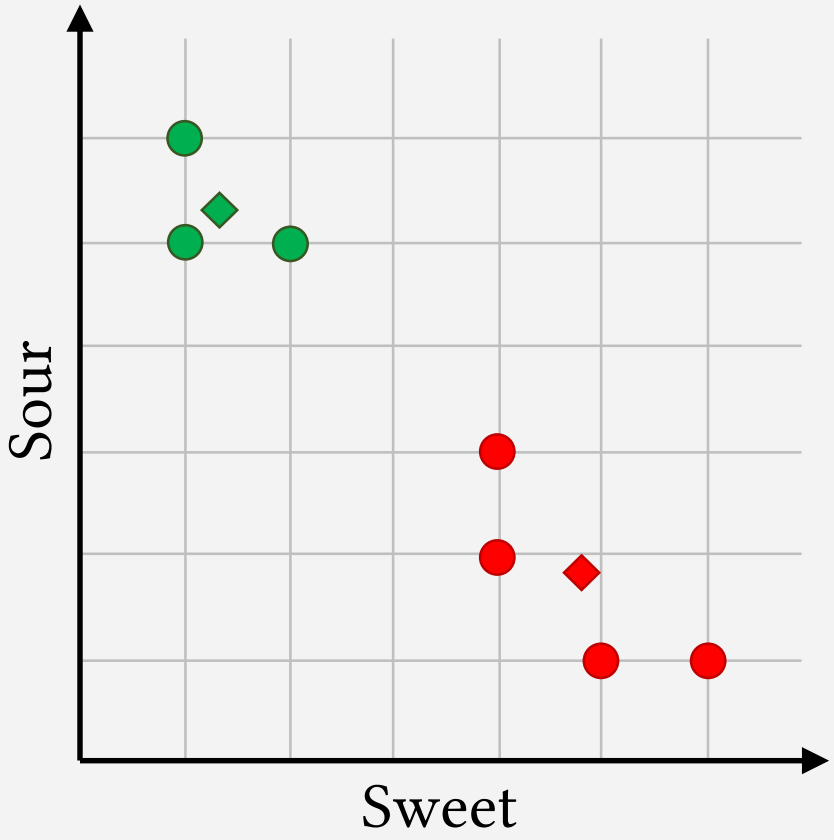
k-Means Example



Recalculate the centroid of each cluster.

- Green Centroid = [1.33,5.33]
- Red Centroid = [4.75,1.75]

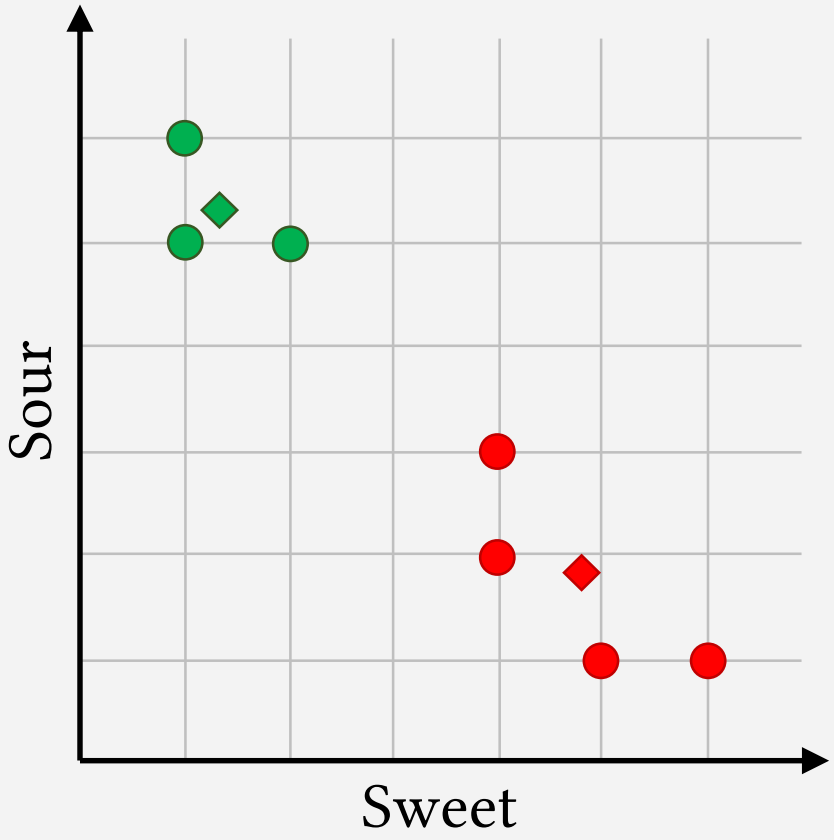
k-Means Example



Reassign each example to the cluster whose centroid is nearest.

- Green Centroid = [1.33,5.33]
- Red Centroid = [4.75,1.75]

k-Means Example



No examples changed clusters, so we are done!

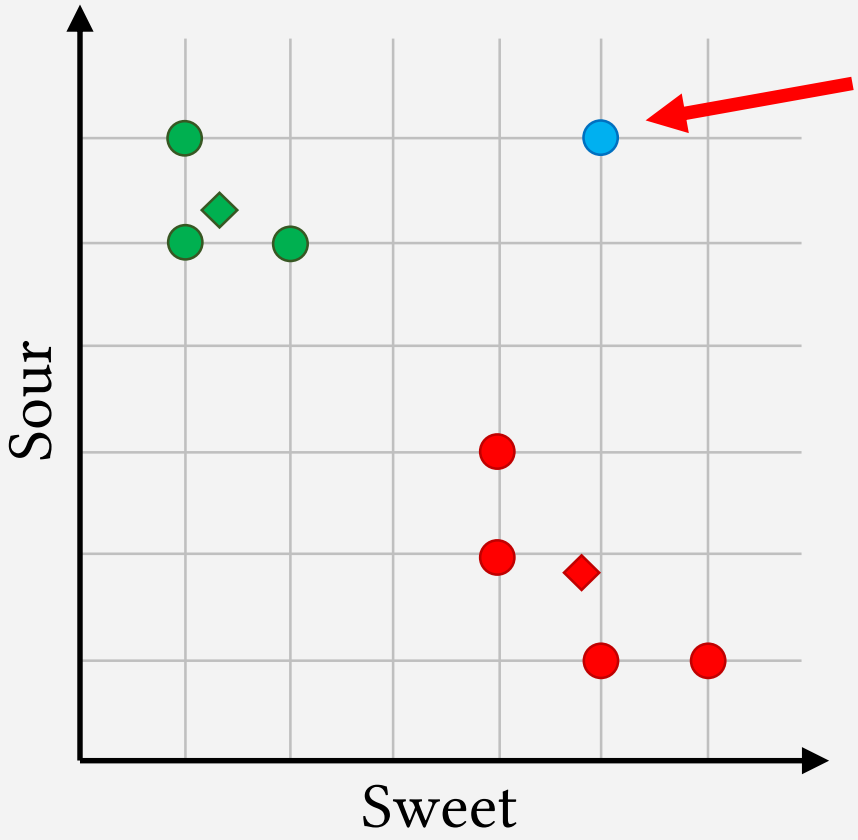
- Green Centroid = [1.33,5.33]
- Red Centroid = [4.75,1.75]

k-Means Problems

Like KNN classification, k-means is sensitive to noise and irrelevant attributes.

There are also some kinds of patterns that k-means clusters poorly.

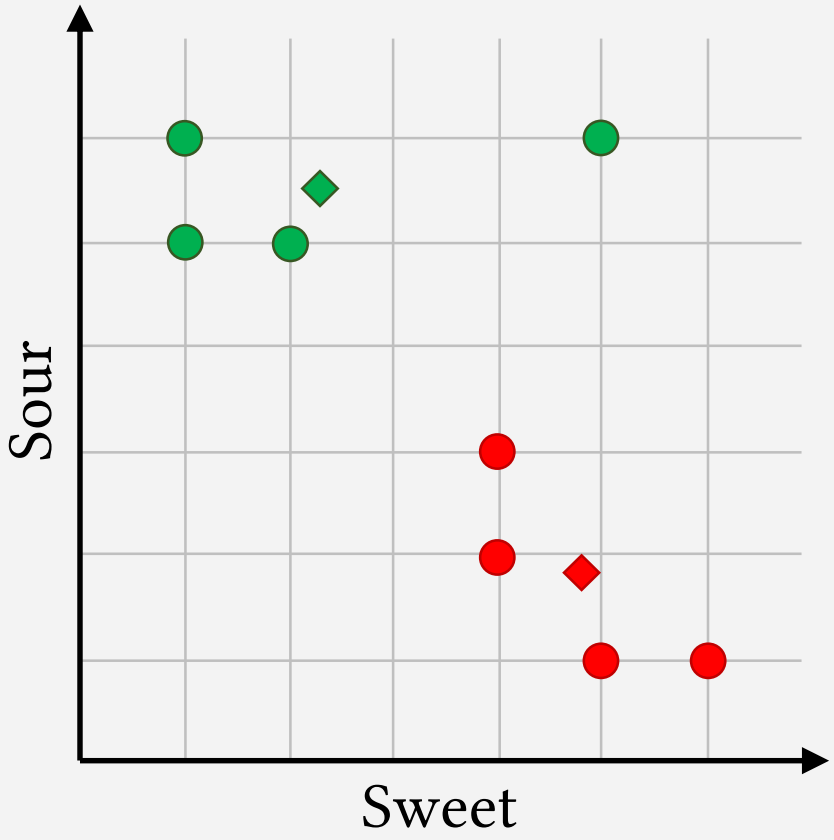
Noise in k-Means



This point isn't close to any centroid, but it has to go somewhere.

- Green Centroid = [2.25,5.50]
- Red Centroid = [4.75,1.75]

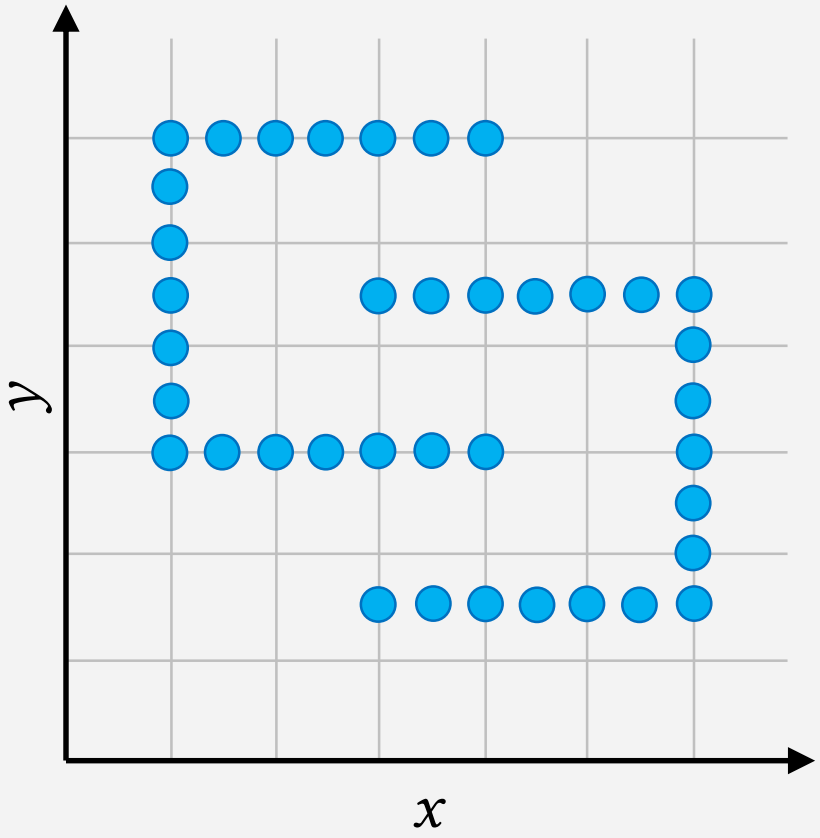
Noise in k-Means



Notice how one outlier affects the centroid.

- Green Centroid = [2.25,5.50]
- Red Centroid = [4.75,1.75]

Shapes in k-Means



An example of non-spherical clusters which k-means has a hard time recognizing.

Density-Based Clustering

In density-based clustering, a cluster is a group of points who are all closer together to one another than to other points.

The **DBSCAN** (Density-Based Spatial Clustering of Applications with Noise) algorithm groups points based on density and is resilient to noise.

DBSCAN

DBSCAN needs two parameters:

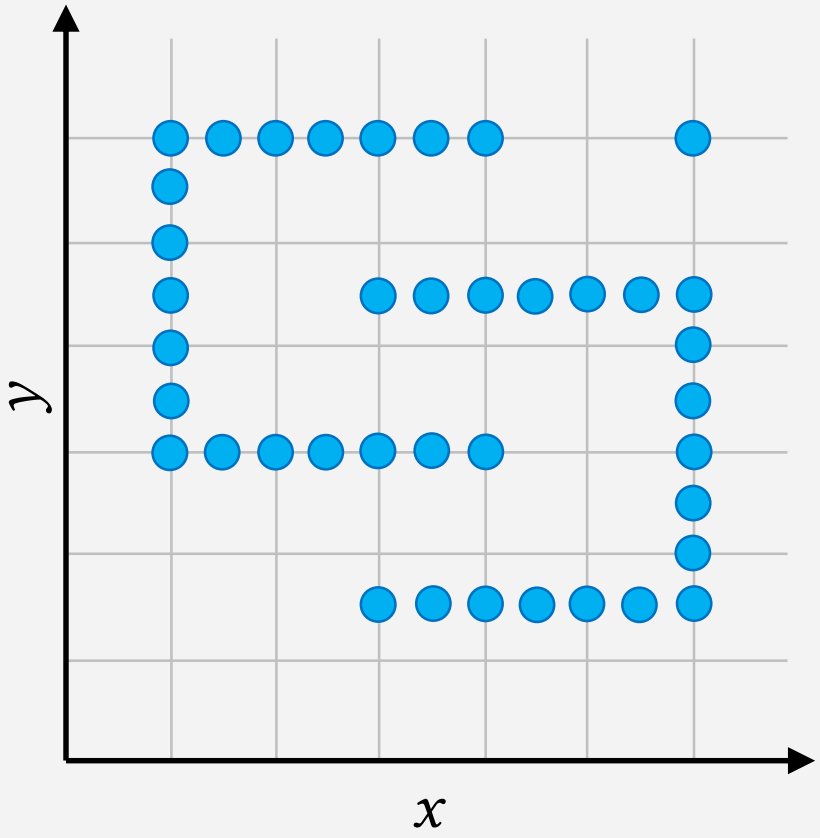
- $\boldsymbol{\varepsilon}$, the radius around a point
- $\boldsymbol{m}$, the minimum number of nearby points

It divides all points into three categories:

- **core points** have m points within ε
- **reachable points** are non-core but near a core
- **outliers** are non-core and not near a core

All points mutually reachable are a cluster.

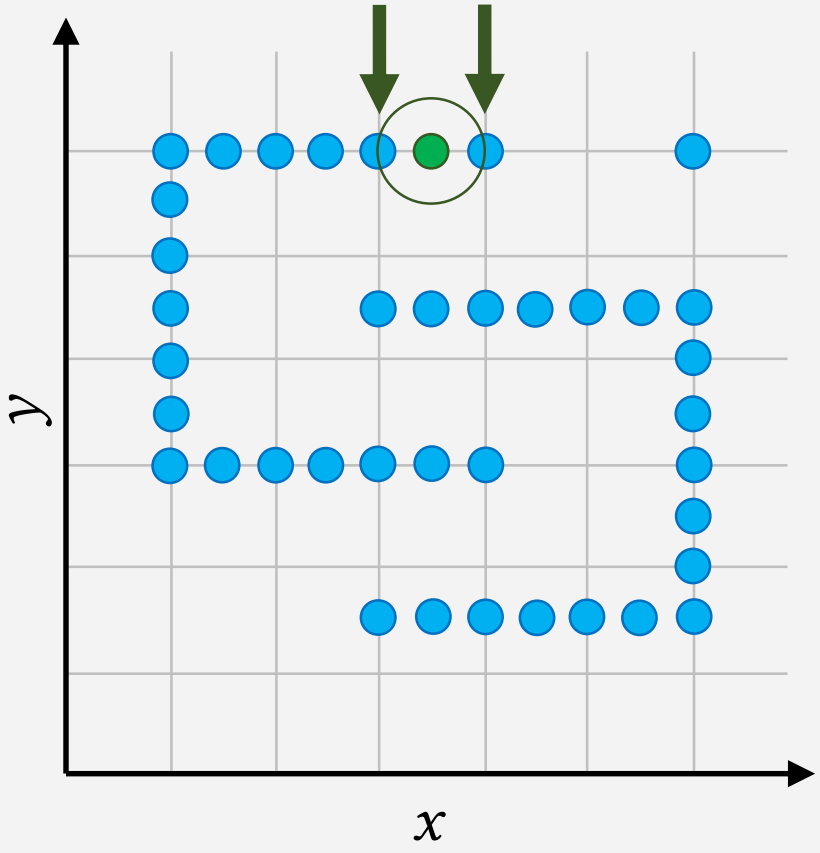
DBSCAN Example



For every point, look for other points that are at least as close as ε.

Parameters: $\varepsilon = 0.5 \quad m = 2$

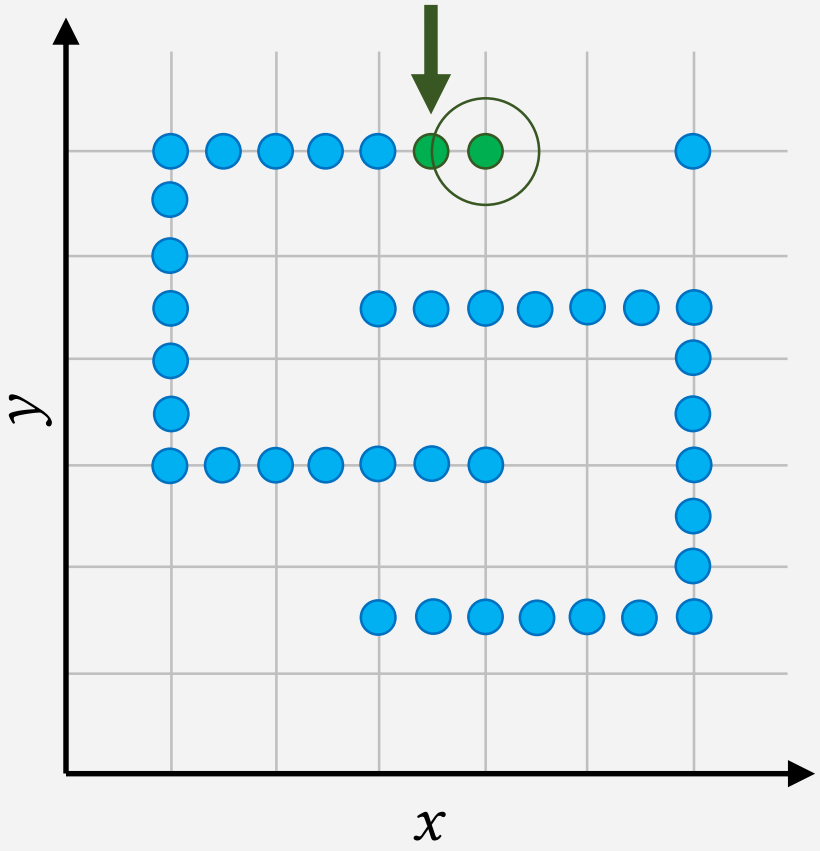
DBSCAN Example



This point has two nearby points, so it is a core.

Parameters: $\varepsilon = 0.5 \quad m = 2$

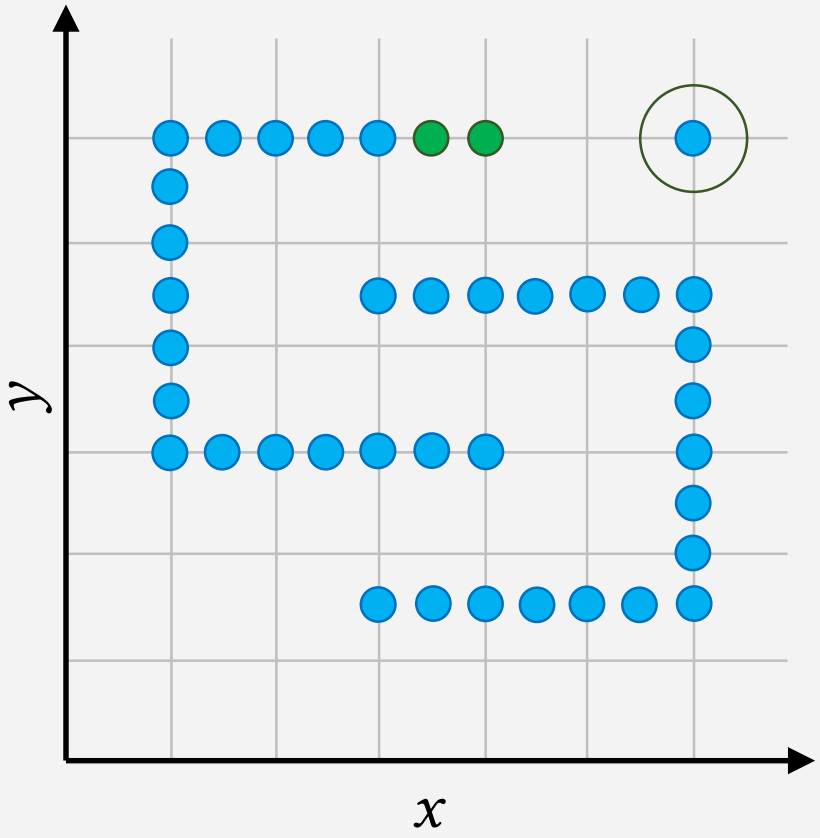
DBSCAN Example



This point has one nearby core point, so it is not a core, but it is reachable from a core.

Parameters: $\varepsilon = 0.5 \quad m = 2$

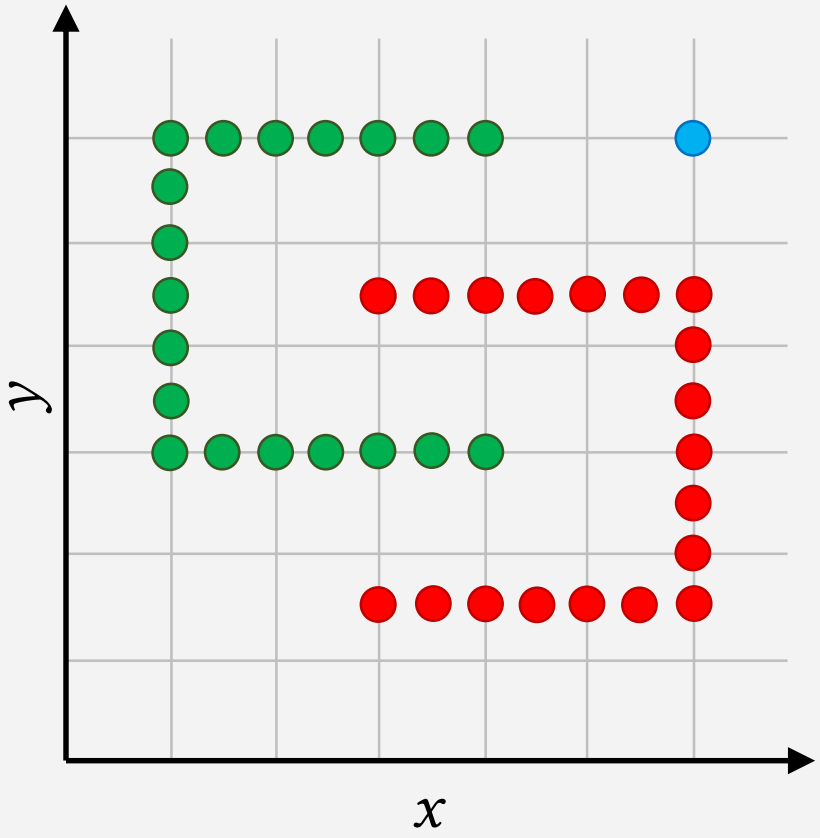
DBSCAN Example



This point has no nearby points, so it is an outlier.

Parameters: $\varepsilon = 0.5 \quad m = 2$

DBSCAN Example



Mutually reachable points define a cluster.

Parameters: $\varepsilon = 0.5 \quad m = 2$

DBSCAN vs. k-Means

- More deterministic (same input gives same output).
- Can find non-spherical clusters.
- Has build-in noise detection.
- Algorithm decides how many clusters there are, instead of requiring us to specify.
- However, we do have to specify ε and m, so there is a tradeoff.