# Machine Learning

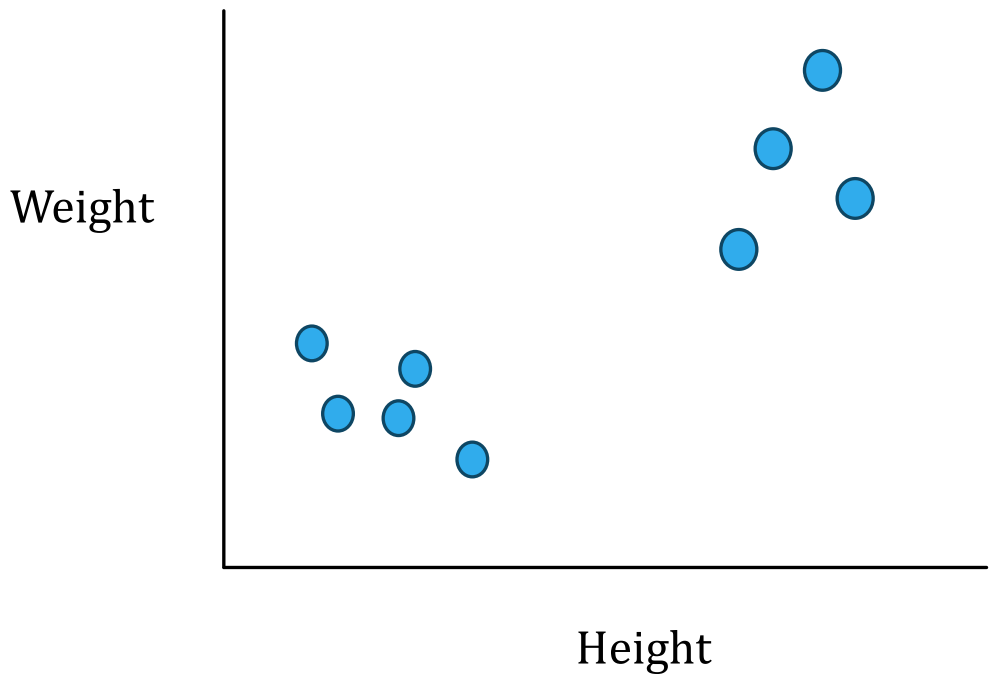
*Angshuman Paul*

*Assistant Professor*

*Department of Computer Science & Engineering*

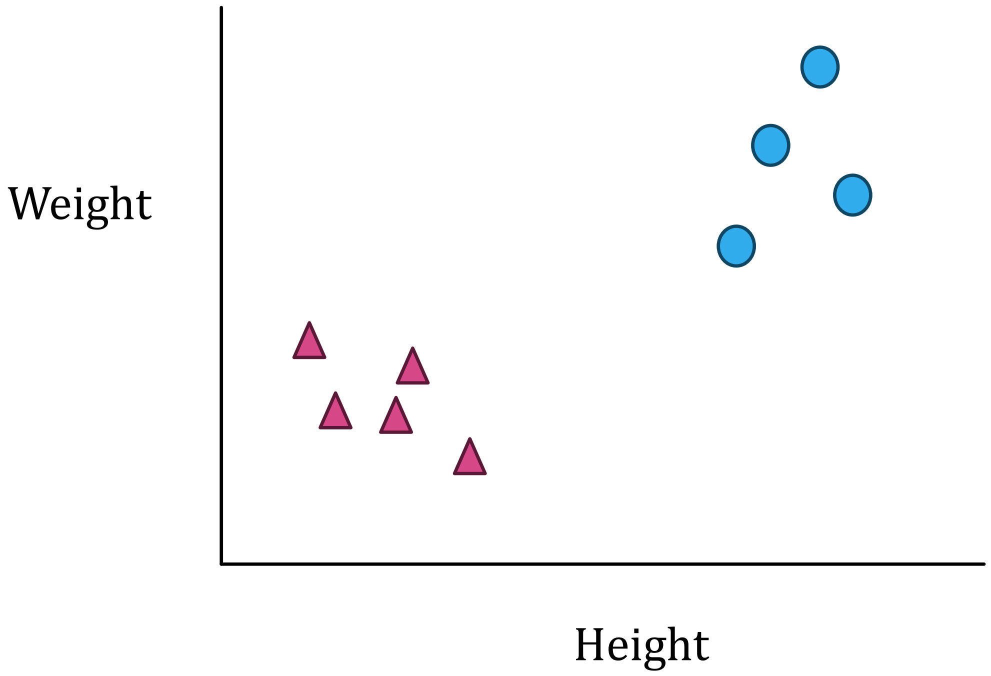
# Clustering

# What is Clustering?




I plot some sample data
of a few animals

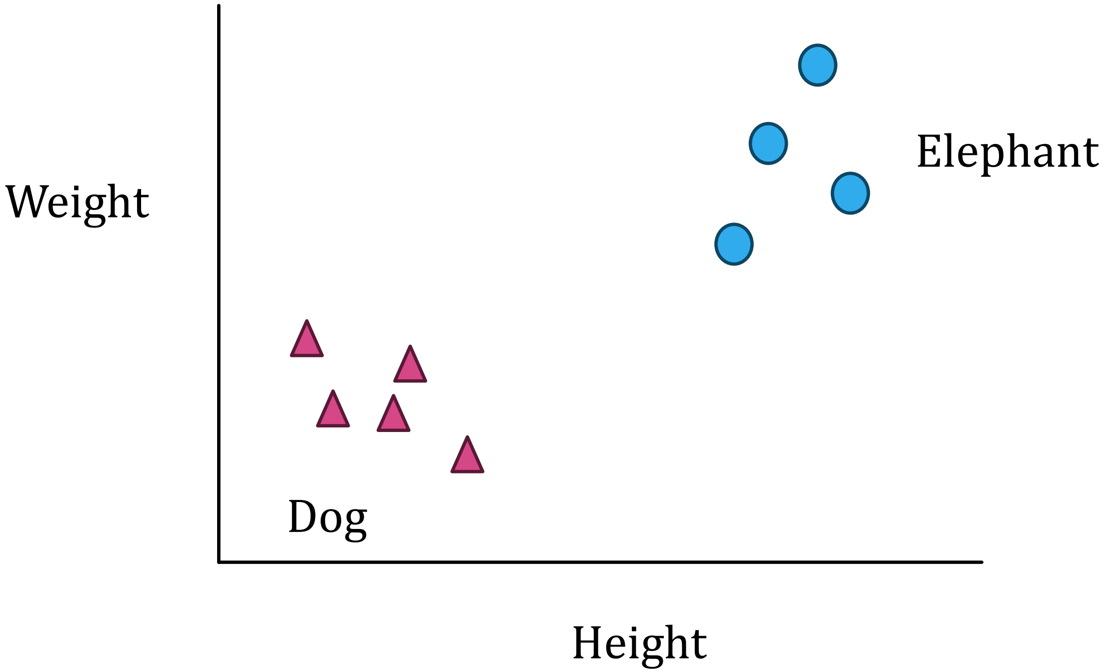
# What is Clustering?




I can clearly see two groups here

In clustering, we create groups of samples in such a way that the samples in a group are more similar to each other than the samples in other groups

# Classification



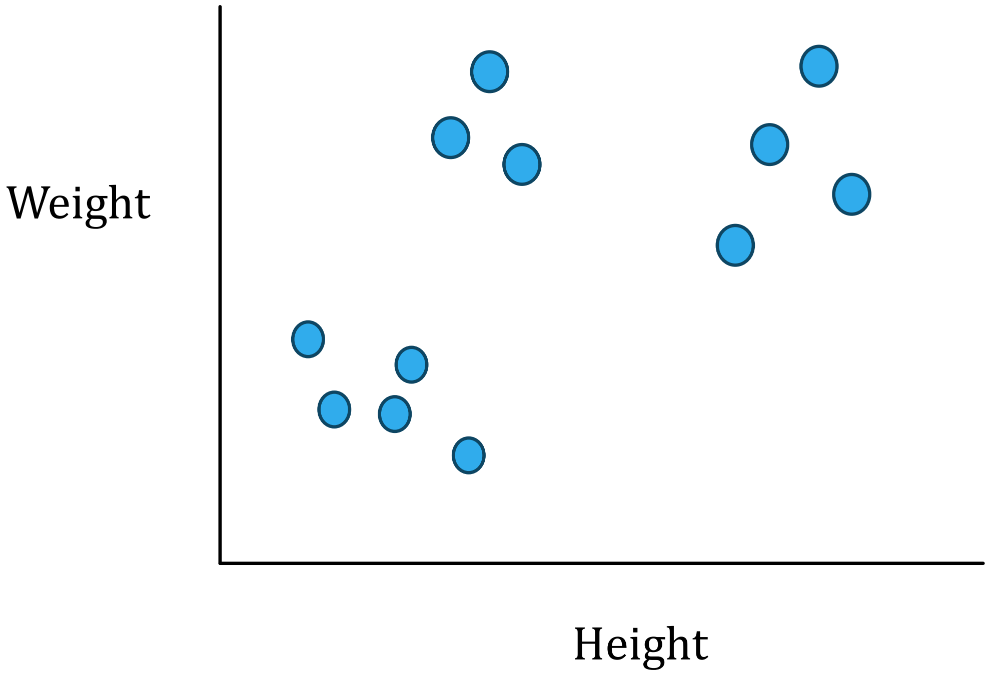

In supervised learning, we assign labels to each group

# Clustering Techniques

- Centroid models: k-Means, k-medoids
- Graph-based models: Spectral Clustering
- Distribution models: Gaussian Mixture Models
- Density models: DBSCAN
- Connectivity Based: Hierarchical Clustering
- Neural models: Self-organizing map

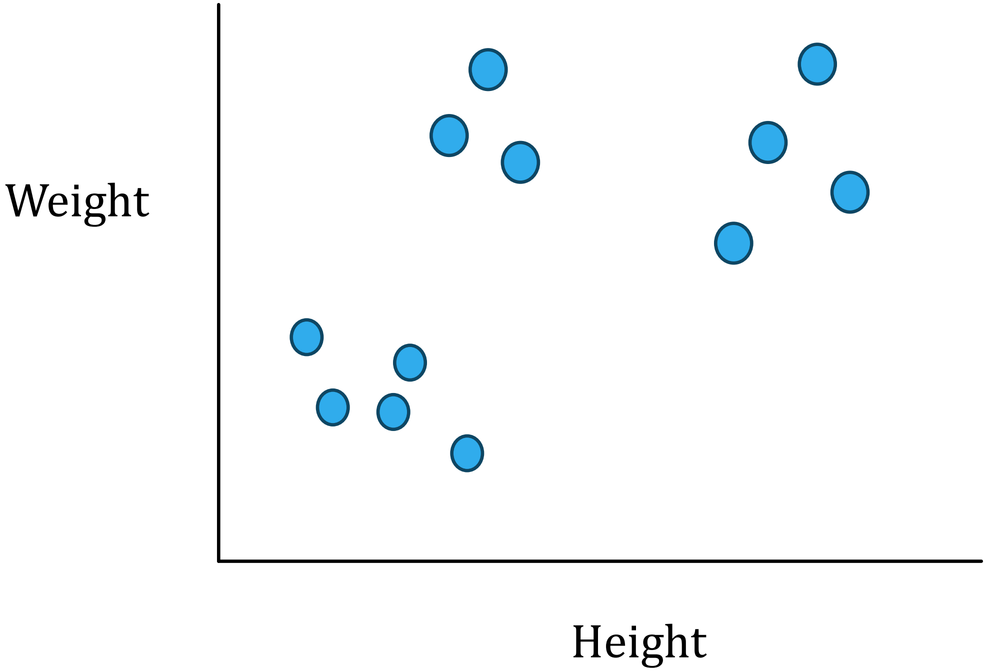
# Clustering Techniques: K-means

- Take the data points



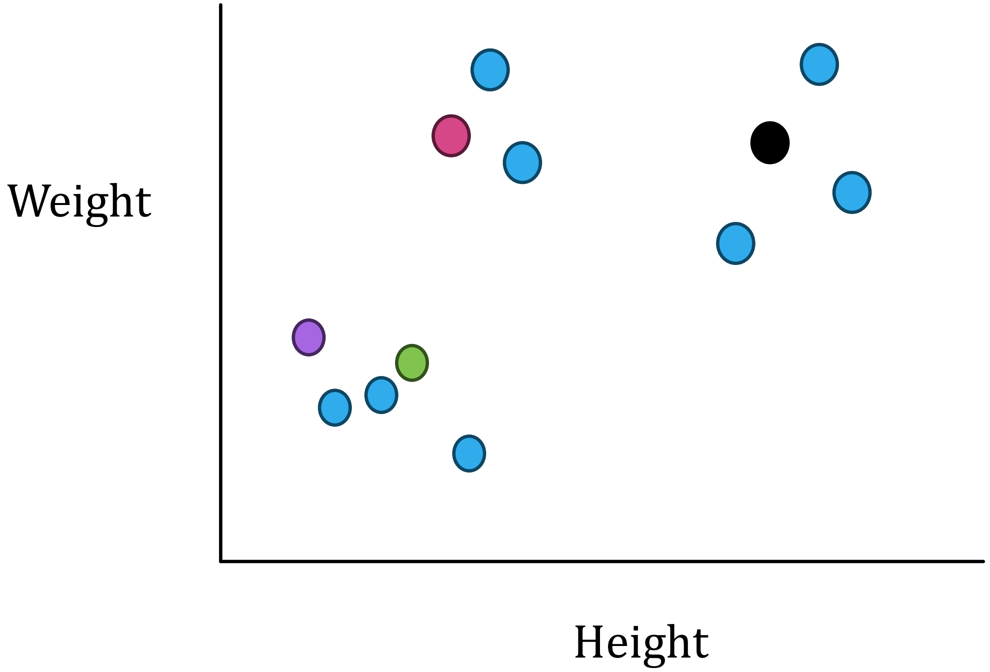
# Clustering Techniques: K-means

- Let's say, I want $m$ clusters
- Here, let's take $m = 4$



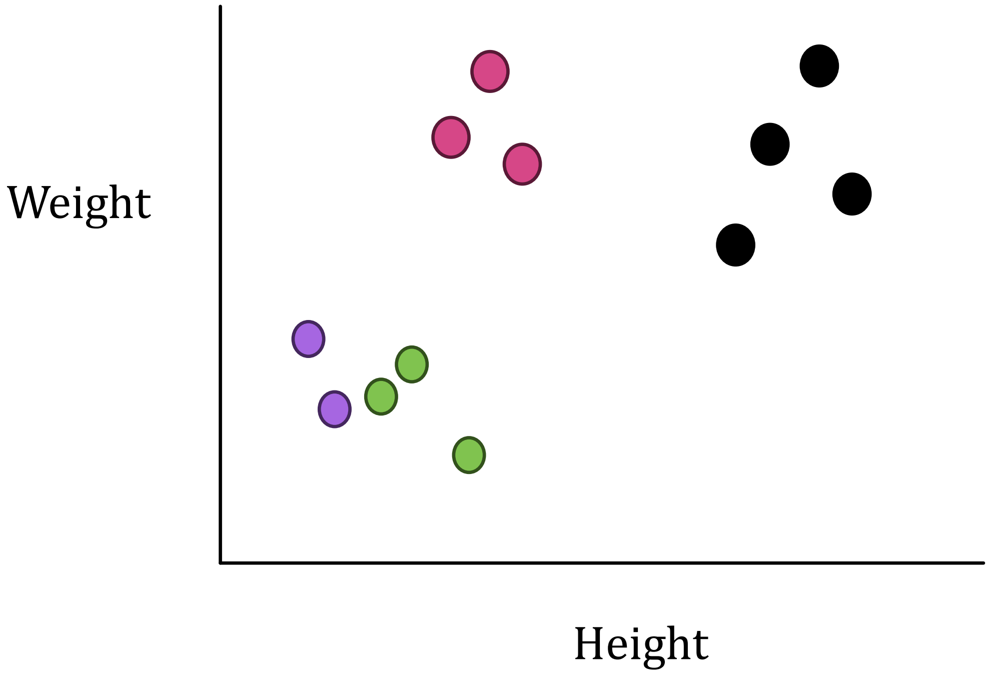
# Clustering Techniques: K-means

- Let's say, I want $m$ clusters
- Here, let's take $m = 4$
- Randomly initialize $m = 4$ cluster centers



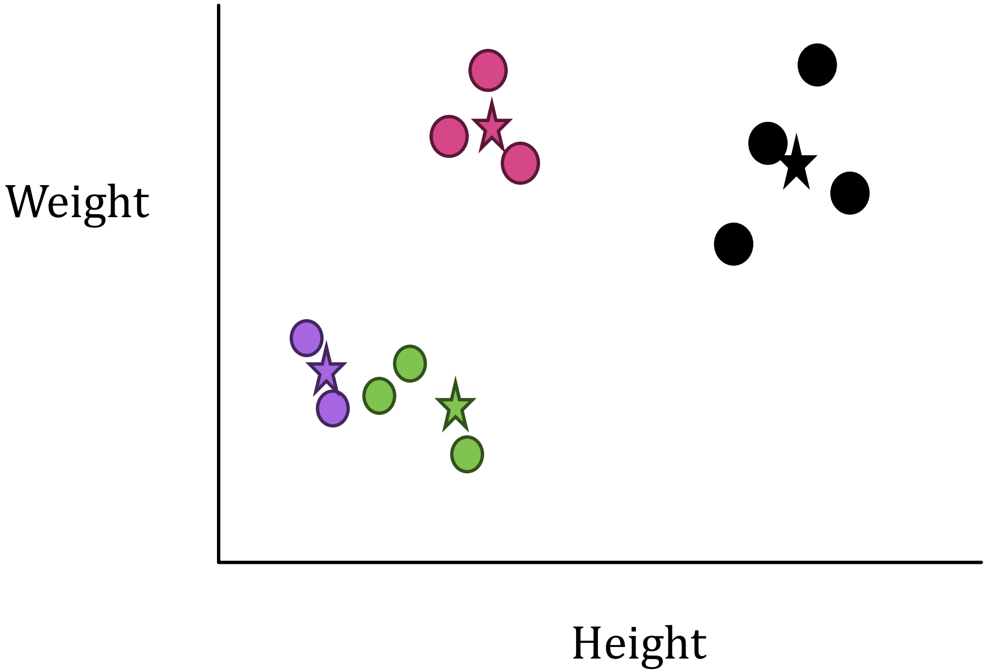
# Clustering Techniques: K-means

- Let's say, I want $m$ clusters
- Here, let's take $m = 4$
- Randomly initialize $m = 4$ cluster centers
- Assign each of the other points to the nearest cluster center



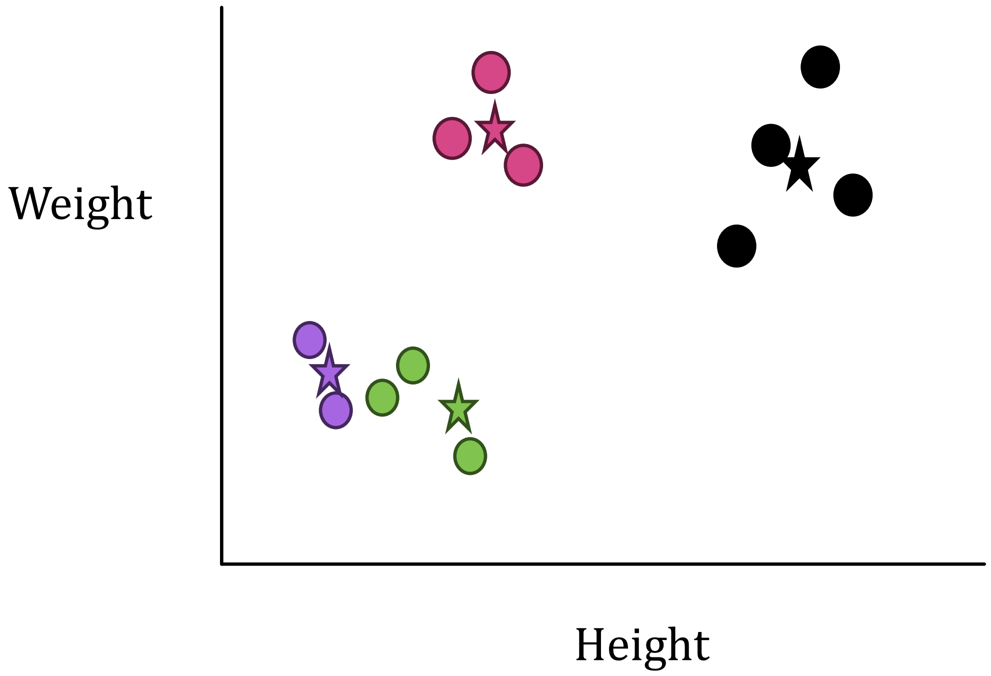
# Clustering Techniques: K-means

- Let's say, I want $m$ clusters
- Here, let's take $m = 4$
- Randomly initialize $m = 4$ cluster centers
- Assign each of the other points to the nearest cluster center
- Recalculate cluster mean



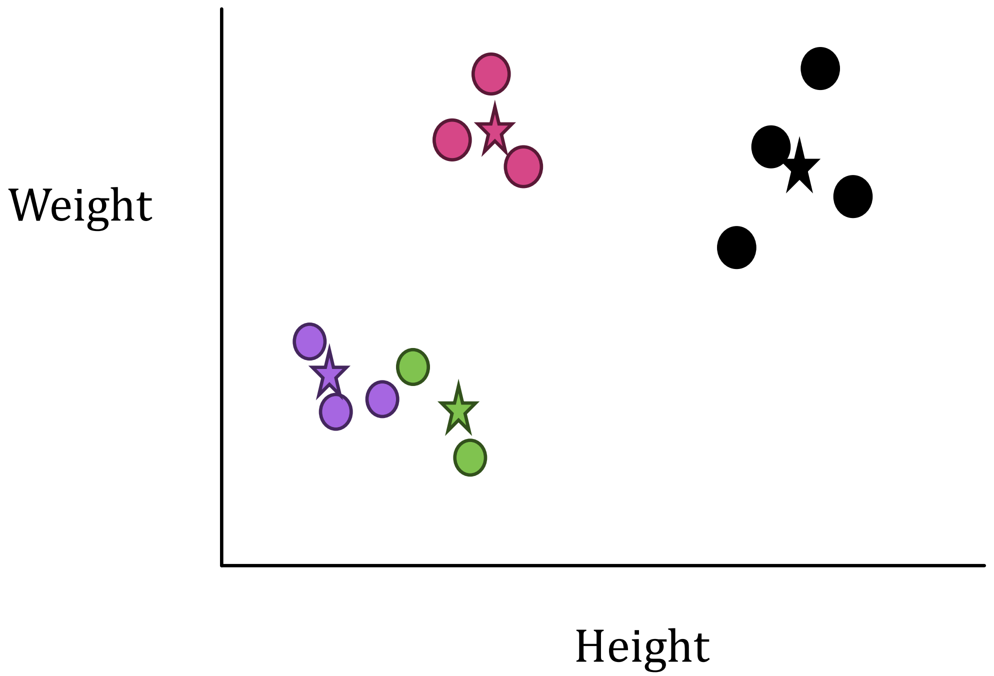
# Clustering Techniques: K-means

- Let's say, I want $m$ clusters
- Here, let's take $m = 4$
- Randomly initialize $m = 4$ cluster centers
- Assign each of the other points to the nearest cluster center
- Recalculate cluster mean
- Reassign points based on updated mean



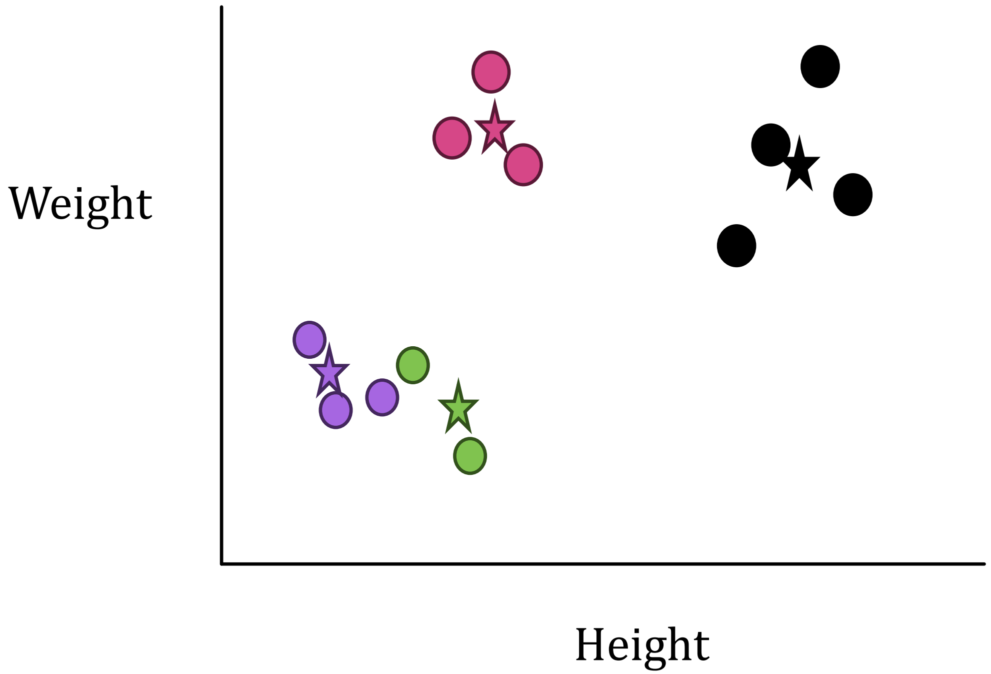
# Clustering Techniques: K-means

- Let's say, I want $m$ clusters
- Here, let's take $m = 4$
- Randomly initialize $m = 4$ cluster centers
- Assign each of the other points to the nearest cluster center
- Recalculate cluster mean
- Reassign points based on updated mean



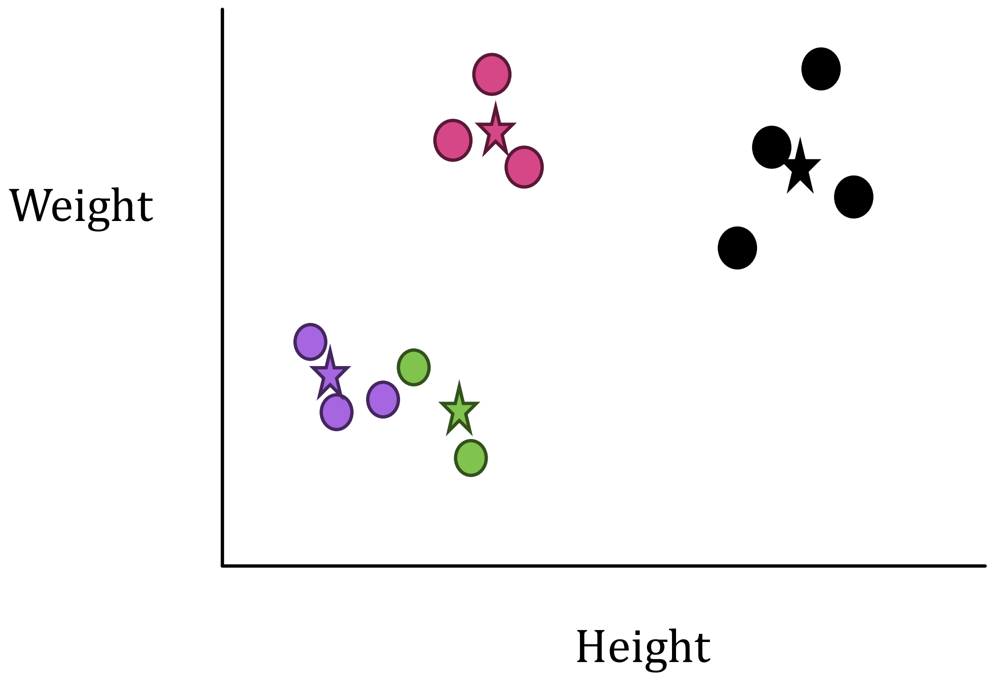
# Clustering Techniques: K-means

1. Let's say, I want $m$ clusters
2. Here, let's take $m = 4$
3. Randomly initialize $m = 4$ cluster centers
4. Assign each of the other points to the nearest cluster center
5. Recalculate cluster mean
6. Reassign points based on updated mean
7. Go to step 5 if there is any change in the assignment. Otherwise, stop



# Clustering Techniques: K-means

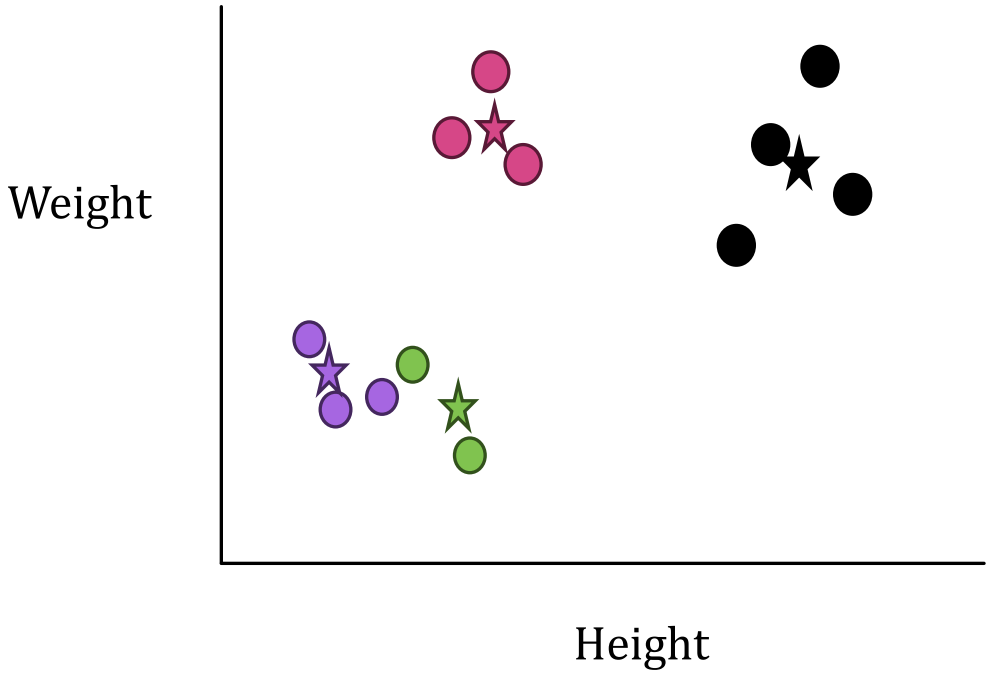
1. Let's say, I want $m$ clusters
2. Here, let's take $m = 4$
3. Randomly initialize $m = 4$ cluster centers
4. Assign each of the other points to the nearest cluster center
5. Recalculate cluster mean
6. Reassign points based on updated mean
7. Go to step 5 if there is any change in the assignment. Otherwise, stop




**Not guaranteed to converge**

# Clustering Techniques: K-means

Assignment: How to find K?



# K-means: Limitations

- Not guaranteed to converge
- Difficult to kind K
- Sensitive to outliers
- May be significantly affected by initialization

# Clustering Techniques: K-medoids

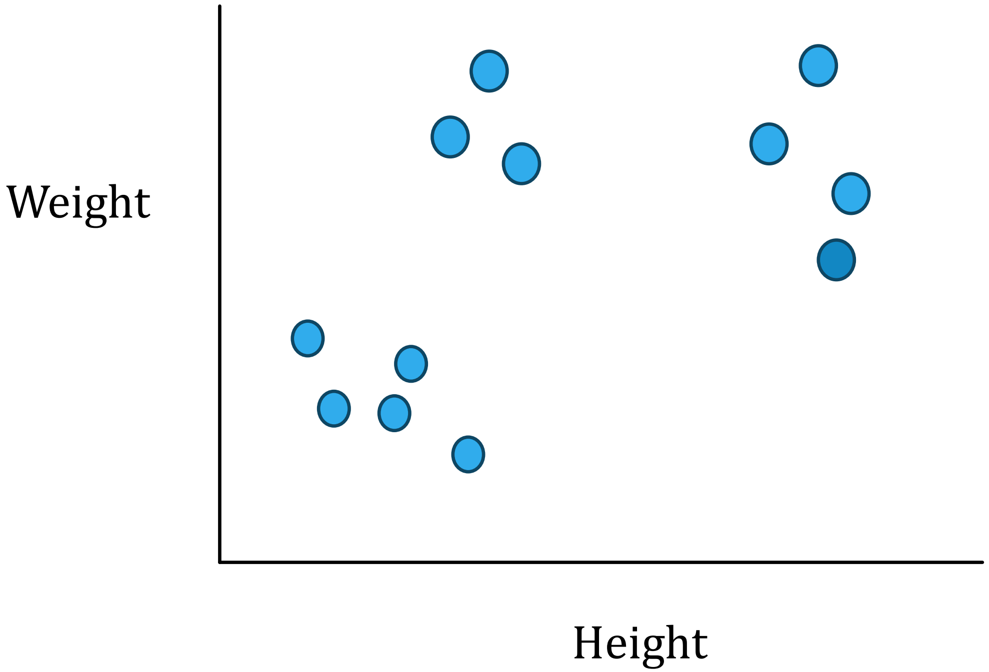
- A Medoid is a point in the cluster from which the sum of distances to other data points is minimal

- A Medoid is a point in the cluster from which dissimilarities with all the other points in the clusters are minimal.

- Instead of centroids as reference points in K-Means algorithms, the K-Medoids algorithm takes a Medoid as a reference point

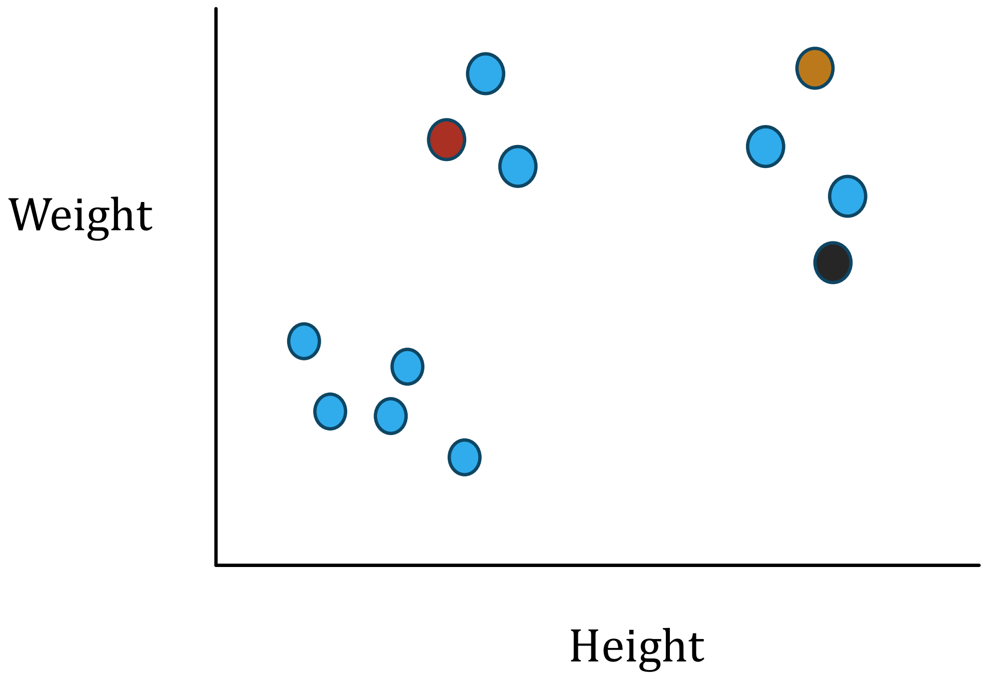
# Clustering Techniques: K-medoids

- Take the data points



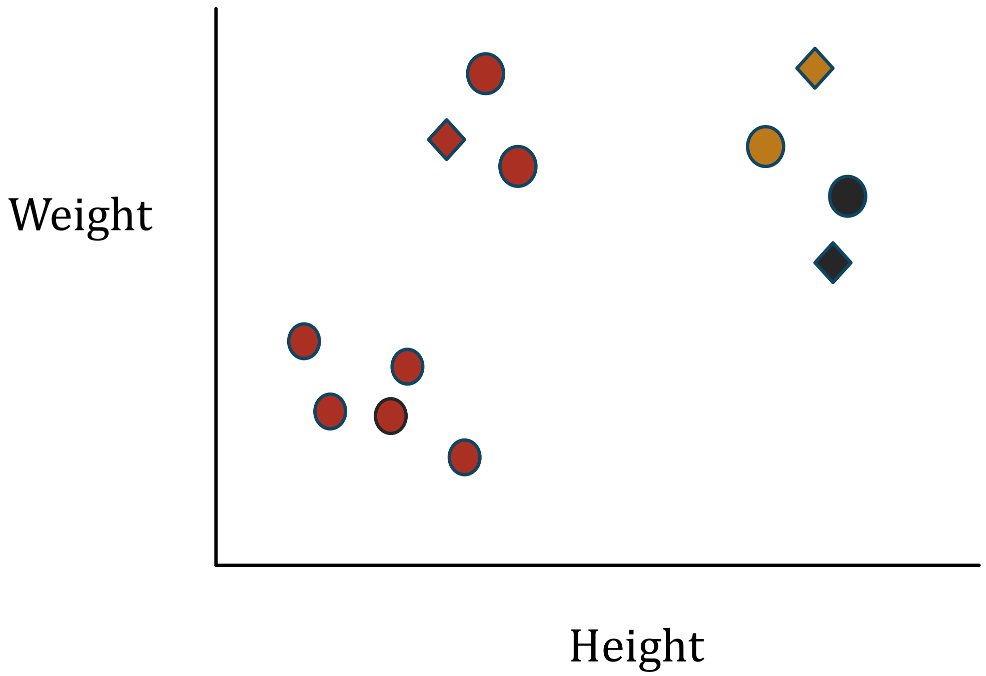
# Clustering Techniques: K-medoids

- Take the data points
- Randomly assign medoids
- For each non-medoid data point, find the nearest medoid and assign the data point to the corresponding cluster



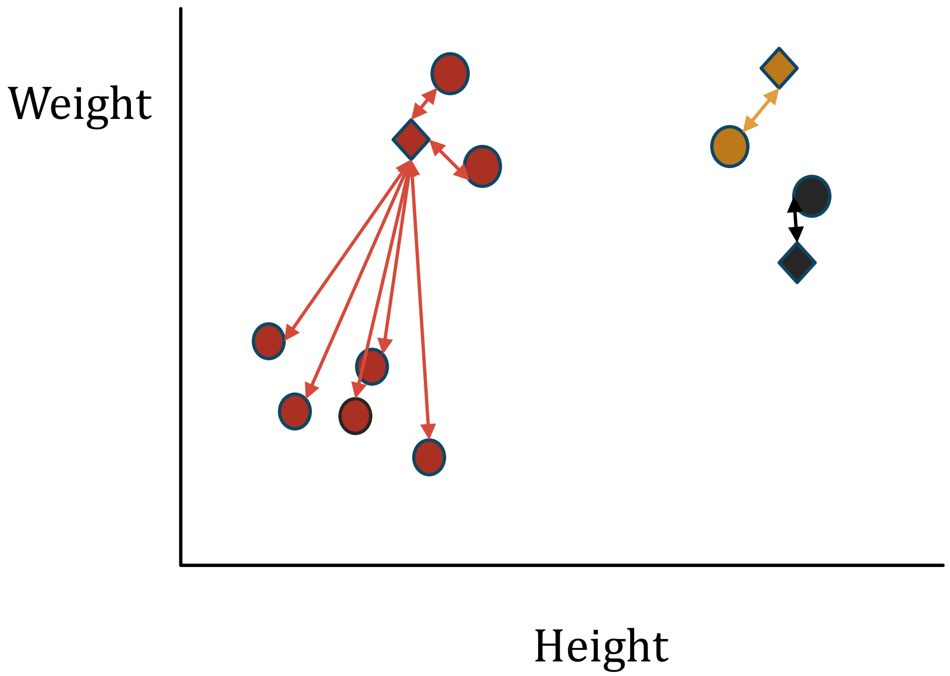
# Clustering Techniques: K-medoids

- Take the data points
- Randomly assign medoids
- For each non-medoid data point, find the nearest medoid and assign the data point to the corresponding cluster
- Calculate the distance of every data point from the corresponding medoid. Sum of all these distances is called the cost



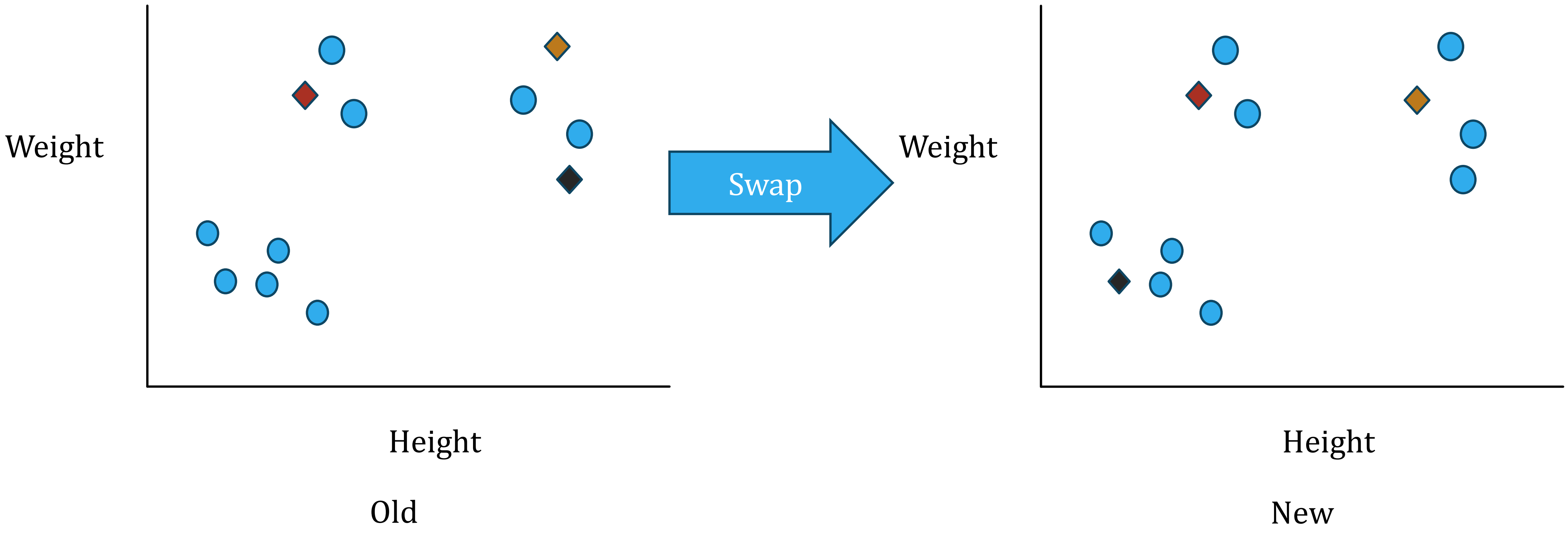
# Clustering Techniques: K-medoids

- Take the data points
- Randomly assign medoids
- For each non-medoid data point, find the nearest medoid and assign the data point to the corresponding cluster
- Calculate the distance of every data point from the corresponding medoid. Sum of all these distances is called the cost
    - Cost= total red distance + total yellow distance + total black distance



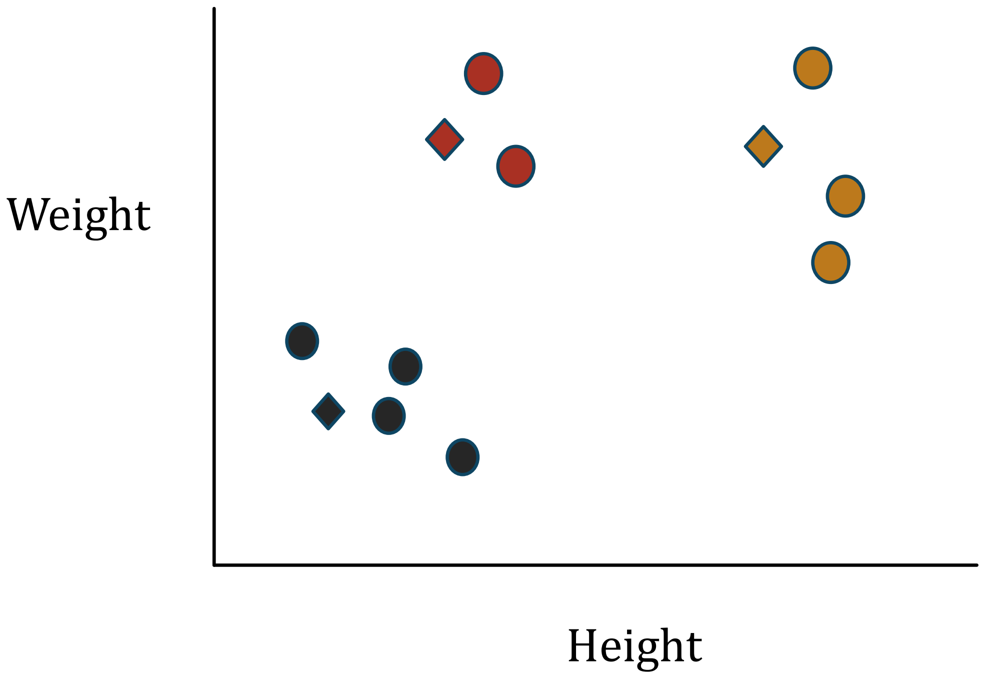
# Clustering Techniques: K-medoids

- Randomly swap the medoids



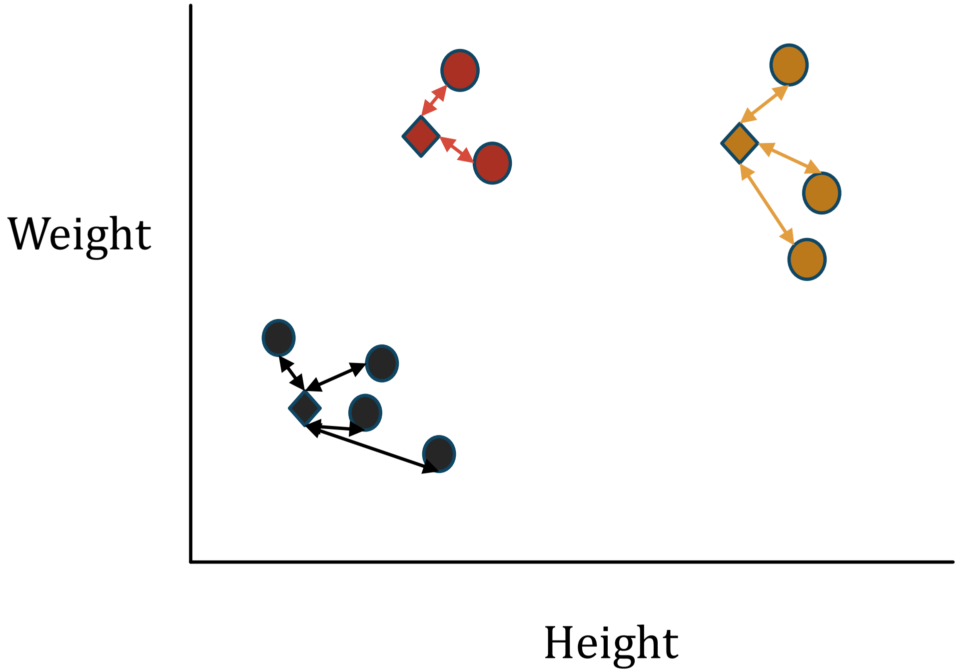
# Clustering Techniques: K-medoids

- With the new medoids, create the clusters using nearest medoids



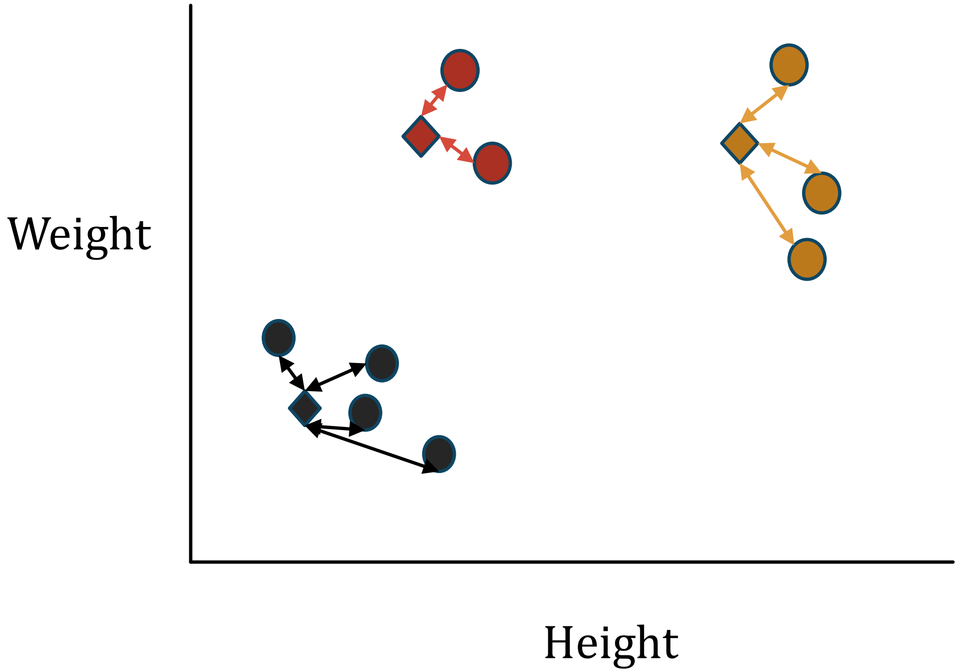
# Clustering Techniques: K-medoids

- With the new medoids recalculate the cost
- Calculate the distance of every data point from the corresponding medoid. Sum of all these distances is called the cost
    - Cost= total red distance + total yellow distance + total black distance



# Clustering Techniques: K-medoids

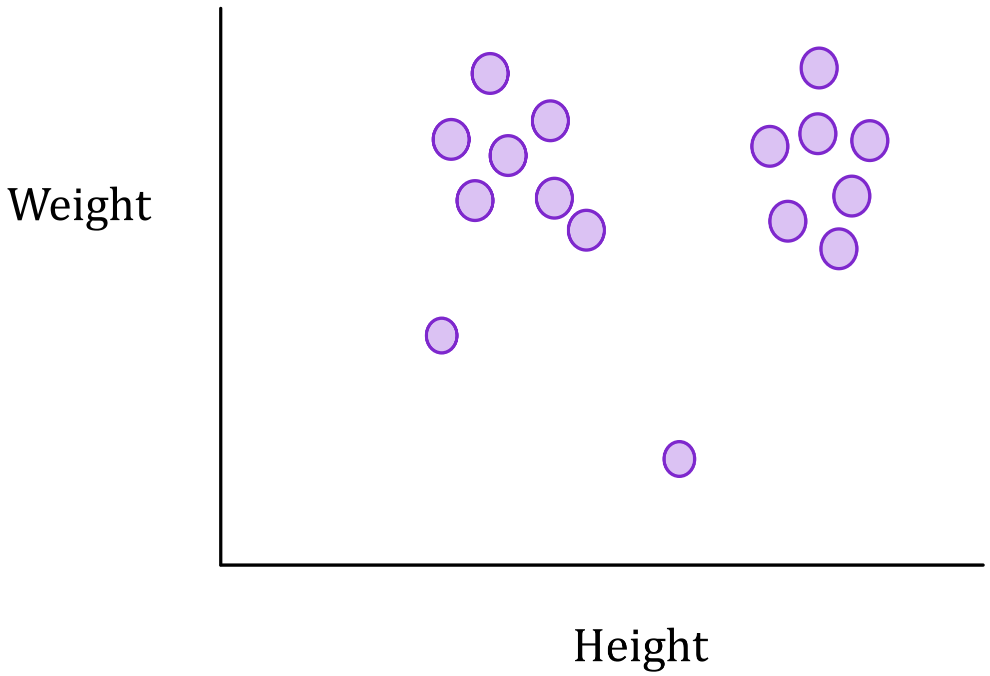
- With the new medoids recalculate the cost
- Calculate the distance of every data point from the corresponding medoid. Sum of all these distances is called the cost
  - Cost= total red distance + total yellow distance + total black distance
- If new cost > old cost, discard the new medoids and go back to the old medoids. The algorithm converges.
  - Else, keep the new medoids and redo the random swapping



# Clustering Techniques: K-medoid

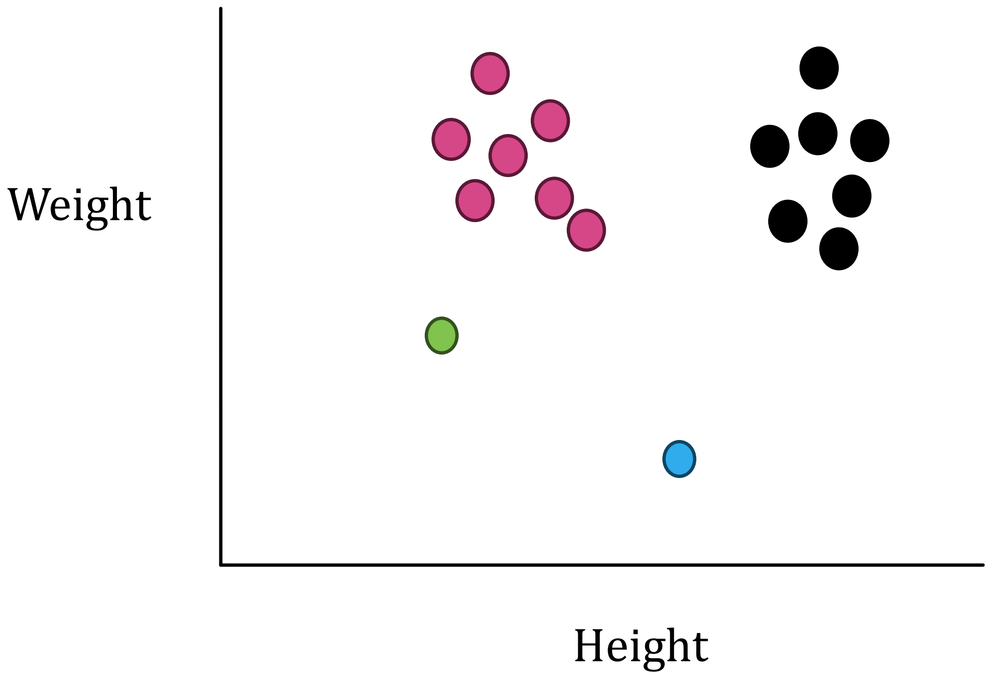
1. **Select $k$ random points from the dataset.** Select $k$ random points from the dataset as the initial medoids. The medoids that are chosen are used to define the initial $k$ clusters.
2. **Assign data points to the cluster of the nearest medoid.** Assign each non-medoid to the cluster corresponding to the closest medoid.
3. **Calculate the total sum of distances of data points from their assigned medoids for each medoid.** Calculate the cost. Cost is given by the sum of the distances from a data point to the assigned (nearest) medoid.
4. **Swap a non-medoid point with a medoid point and recalculate the cost.** Swap a non-medoid point with the medoids and repeat step 2 and 3 to calculate the cost with the new medoids.
5. **Undo the swap if the recalculated cost with the new medoid exceeds the previous cost.** Check if the cost with new medoids is more than the cost with the old medoids. If that is the case, undo the swap, and the algorithm converges. Otherwise, go to step 4.

# Clustering Techniques: Use of Density



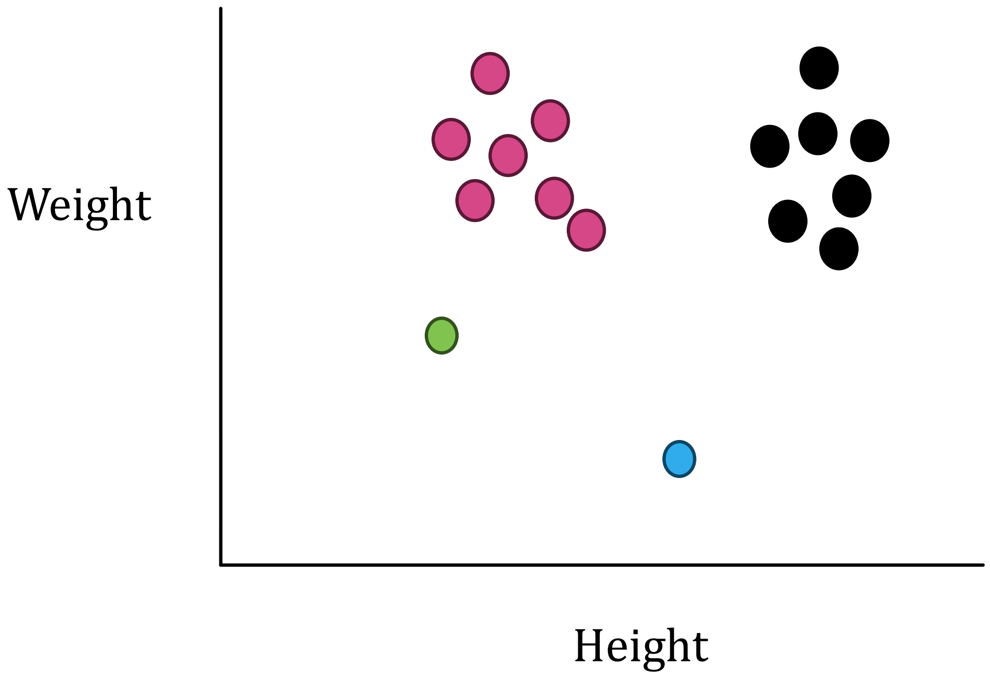

Can we say which points should form clusters?

# Clustering Techniques: Use of Density




How did we do this visually?

# Clustering Techniques: Use of Density

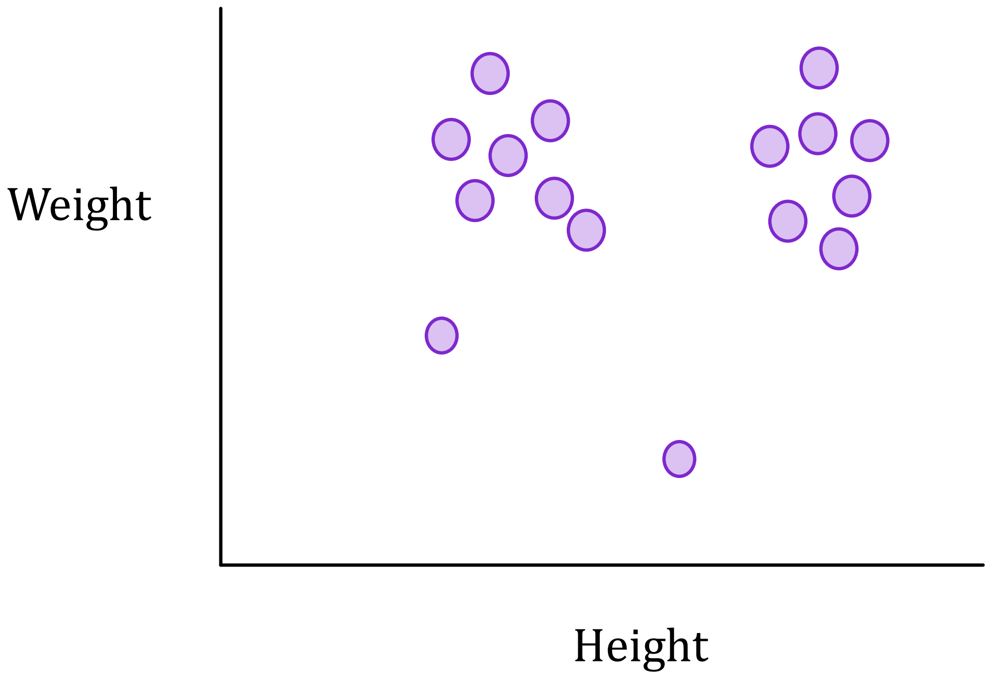



How did we do this visually?

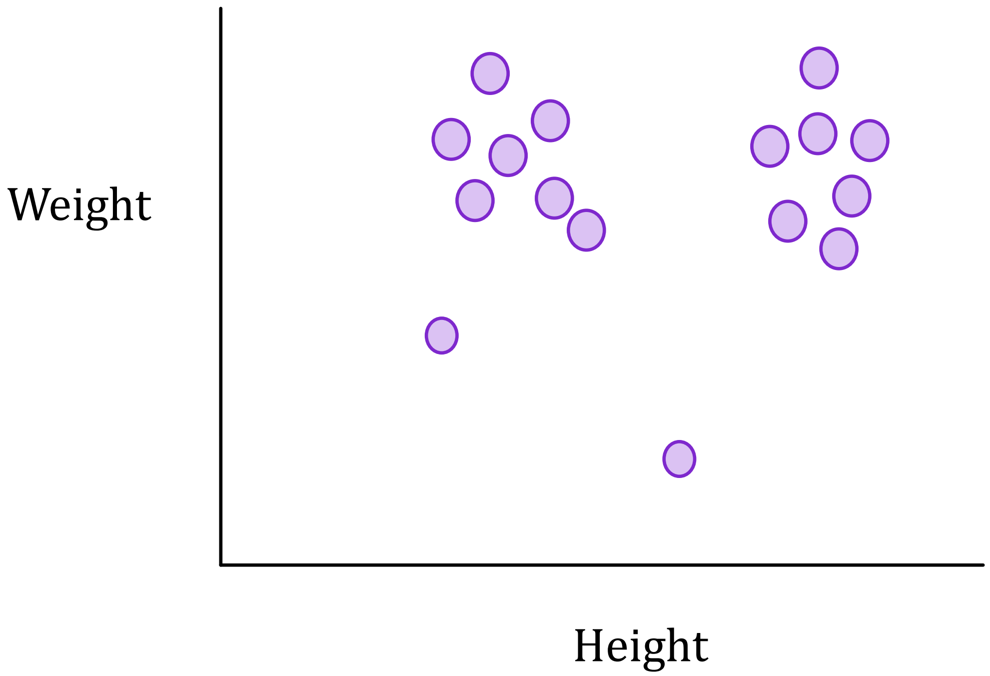
Based on density

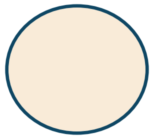
# Clustering Techniques: DBSCAN

DBSCAN: Density-Based Spatial Clustering of Applications with Noise



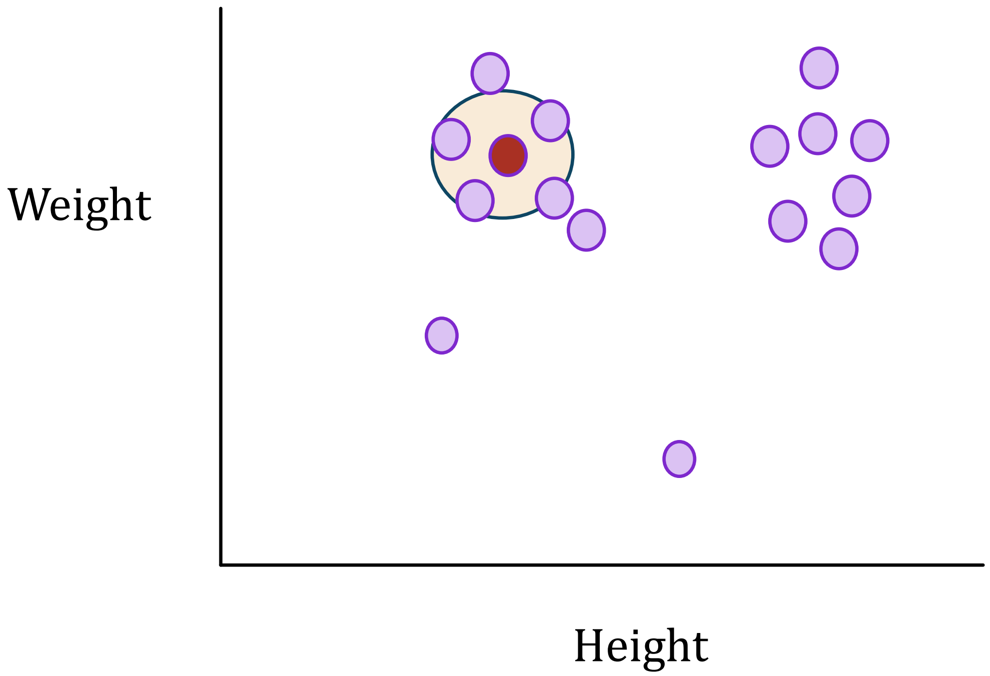
# Clustering Techniques: DBSCAN



Take a disc of specific size (parameter)

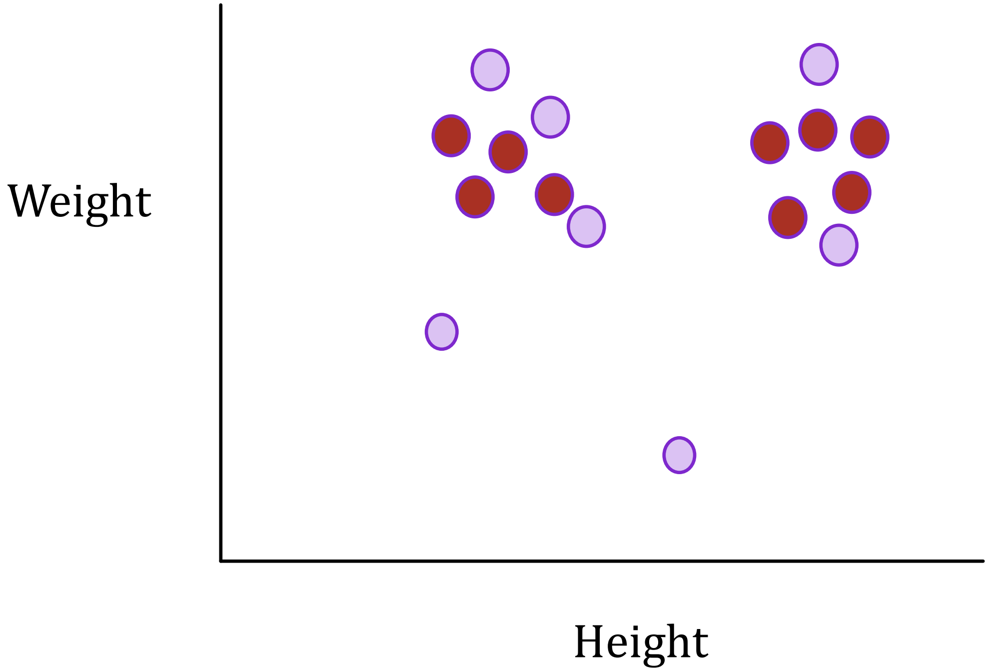
# Clustering Techniques: DBSCAN




Take a disc of specific size (parameter)

Put it across every point. If there are $m$ number of points within the disc when positioned on point $x_i$, the point $x_i$ will be called a core point. $m$ is a parameter.

# Clustering Techniques: DBSCAN

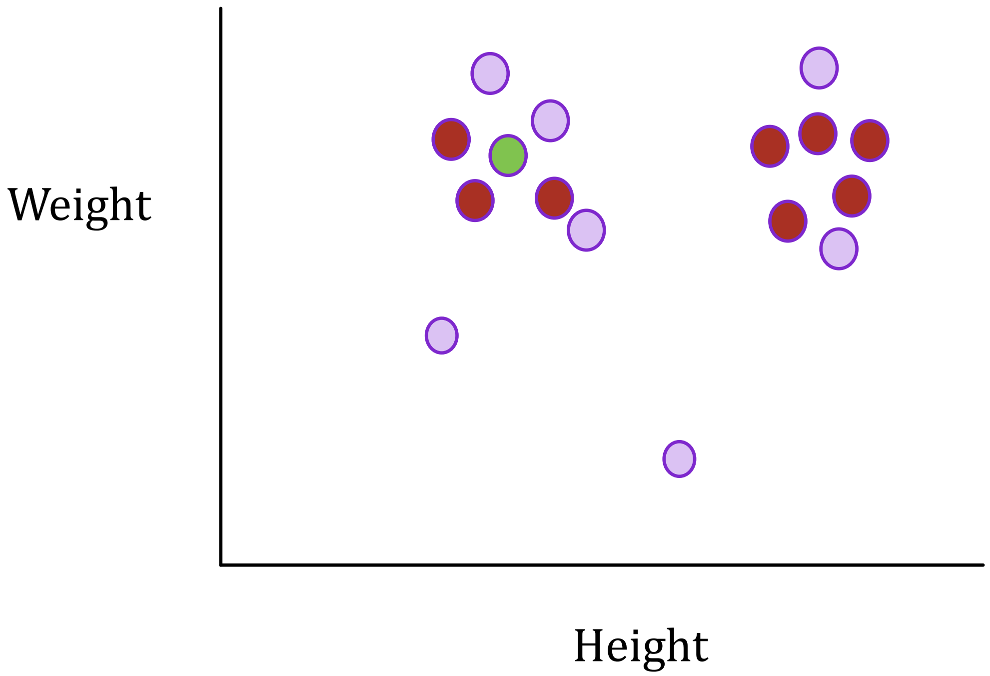



Take a disc of specific size (parameter)

Put it across every point. If there are $m$ number of points within the disc when positioned on point $x_i$, the point $x_i$ will be called a core point. $m$ is a parameter.

Red: core points

# Clustering Techniques: DBSCAN



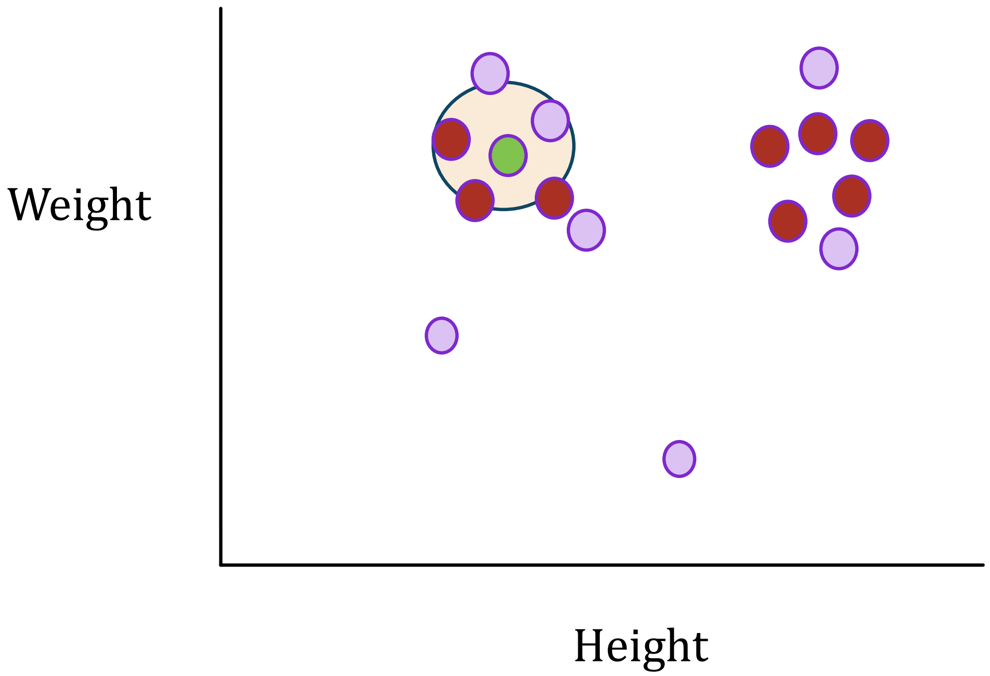

Take a disc of specific size (parameter)

Put it across every point. If there are $m$ number of points within the disc when positioned on point $x_i$, the point $x_i$ will be called a core point. $m$ is a parameter.

Red: core points

Assign a cluster label in a core point (green)

# Clustering Techniques: DBSCAN




Take a disc of specific size (parameter)

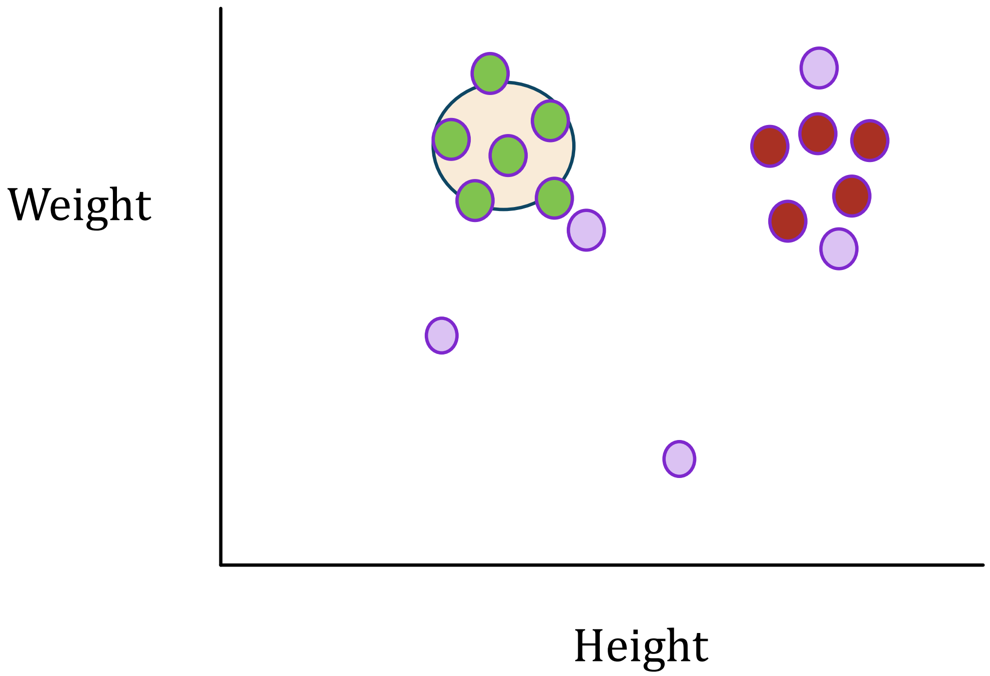
Put it across every point. If there are $m$ number of points within the disc when positioned on point $x_i$, the point $x_i$ will be called a core point. $m$ is a parameter.

Red: core points

Assign a cluster label in a core point (green)

Put every point within the disk inside the first cluster

# Clustering Techniques: DBSCAN




Take a disc of specific size (parameter)

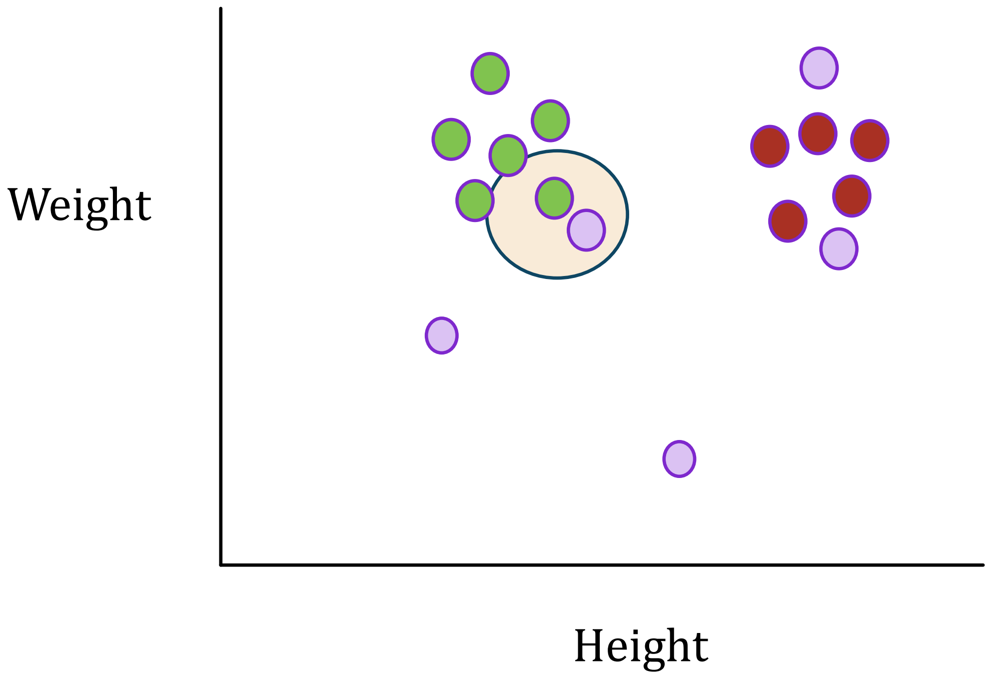
Put it across every point. If there are $m$ number of points within the disc when positioned on point $x_i$, the point $x_i$ will be called a core point. $m$ is a parameter.

Red: core points

Assign a cluster label in a core point (green)

Put every point within the disk inside the first cluster

# Clustering Techniques: DBSCAN




Take a disc of specific size (parameter)

Put it across every point. If there are $m$ number of points within the disc when positioned on point $x_i$, the point $x_i$ will be called a core point. $m$ is a parameter.

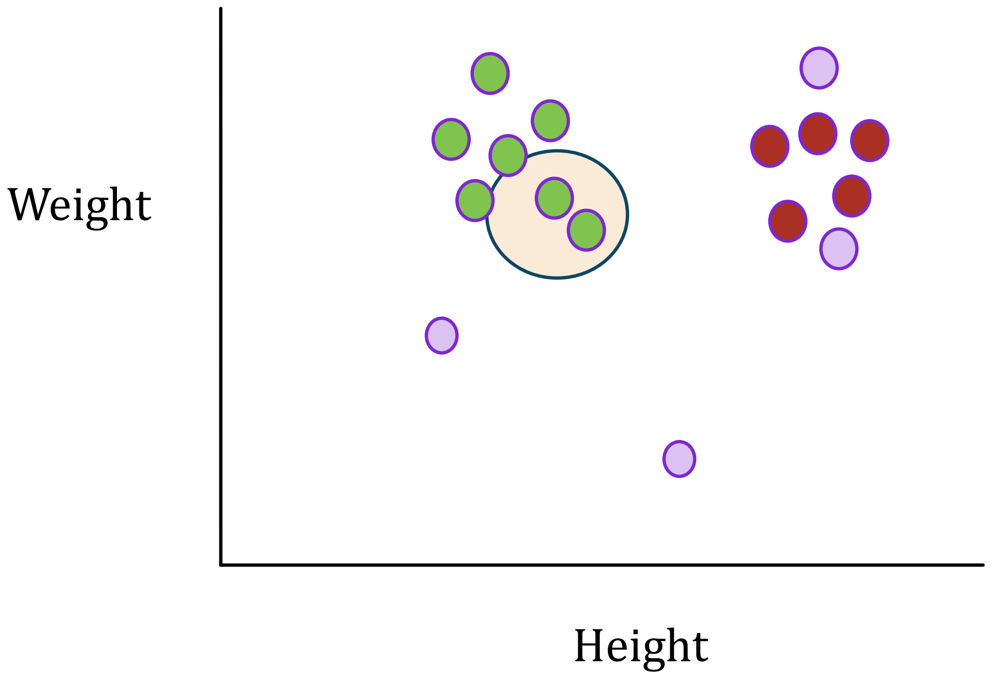
Red: core points

Assign a cluster label in a core point (green)

Put every point within the disk inside the first cluster

Apply the disk from every point in the cluster

# Clustering Techniques: DBSCAN




Take a disc of specific size (parameter)

Put it across every point. If there are $m$ number of points within the disc when positioned on point $x_i$, the point $x_i$ will be called a core point. $m$ is a parameter.

Red: core points

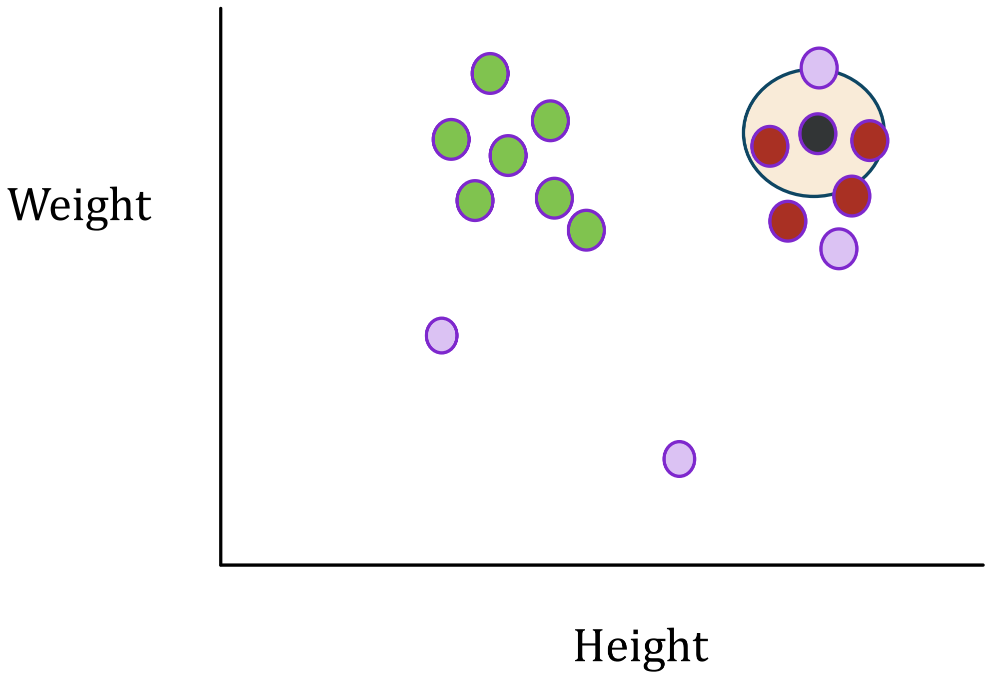
Assign a cluster label in a core point (green)

Put every point within the disk inside the first cluster

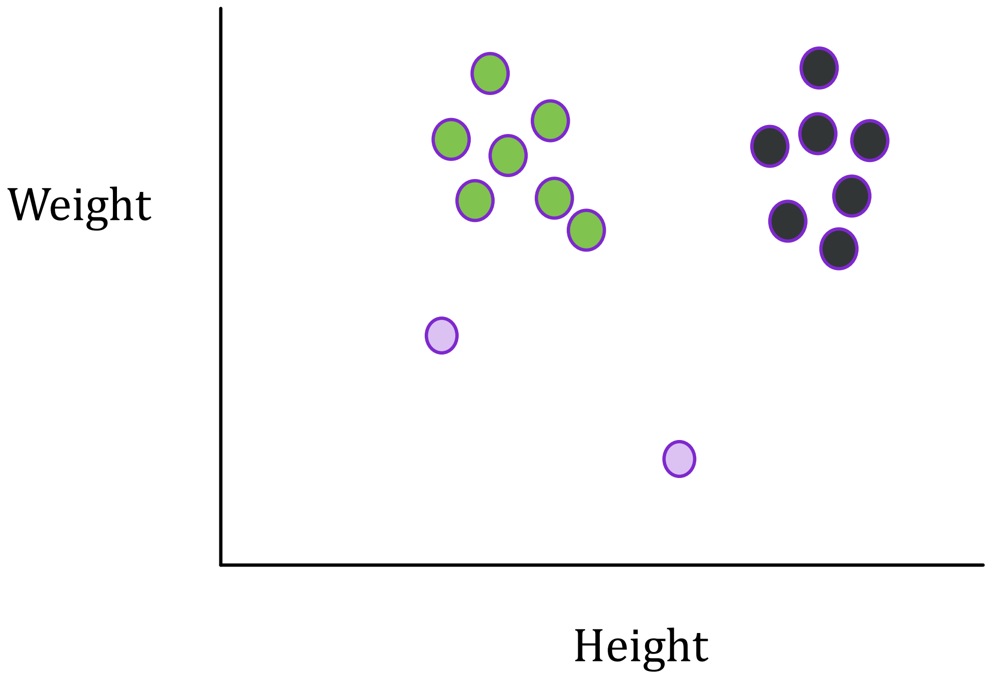
Apply the disk from every core point in the cluster

# Clustering Techniques: DBSCAN

Repeat it for other core points

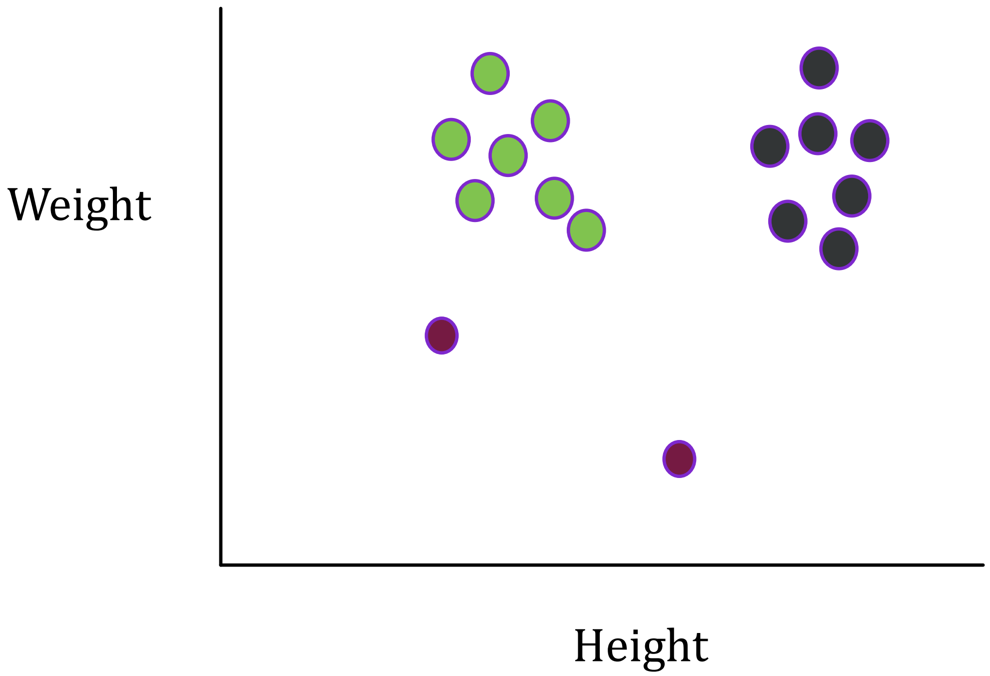


# Clustering Techniques: DBSCAN




Repeat it for other core points

# Clustering Techniques: DBSCAN

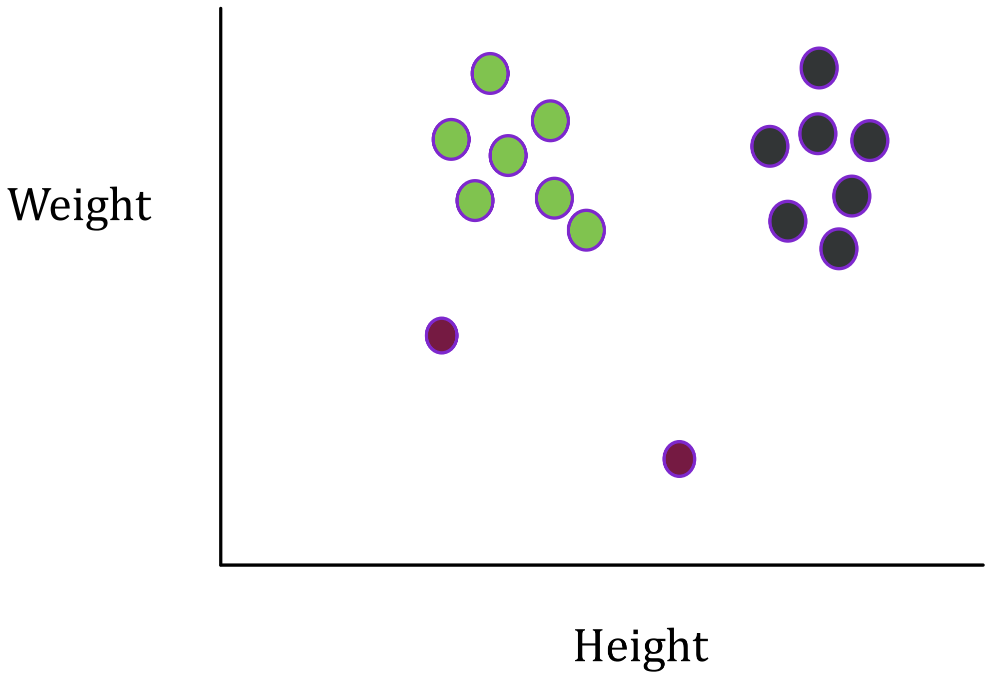



Repeat it for other core points

When we are done with all core points and left with only non-core points that can't be added to any clusters, we are done.

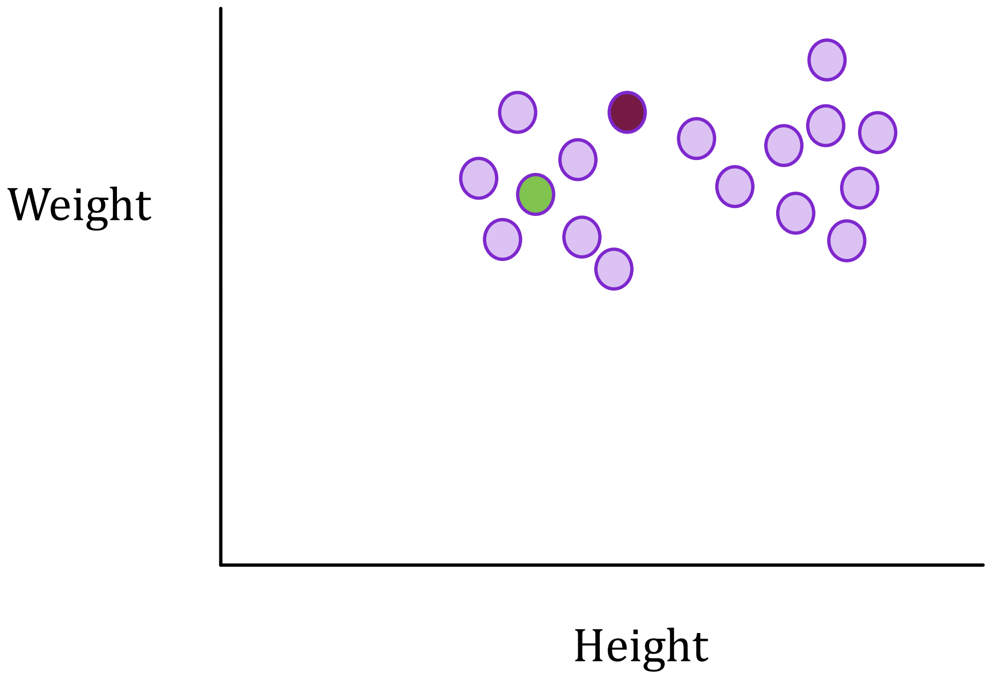
These non-core points are called outliers.

# DBSCAN: Advantage




We don’t need to know the number of clusters apriori

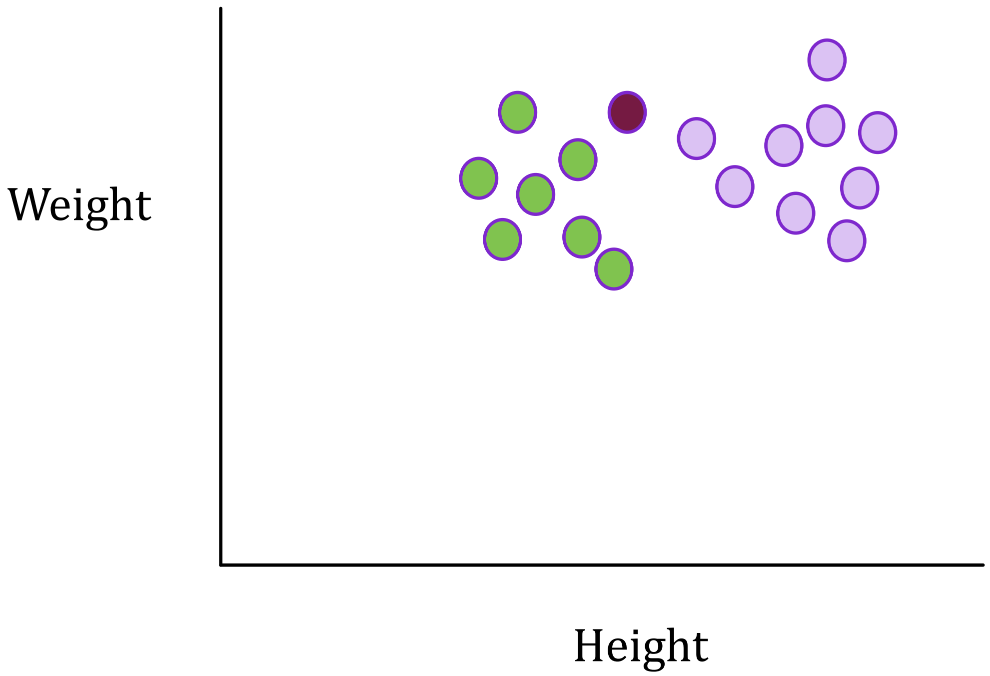
# DBSCAN: Disadvantage




The red point is a non-core point. So it can’t expand a cluster.

Suppose, at start, the green point is chosen for expansion.

# DBSCAN: Disadvantage

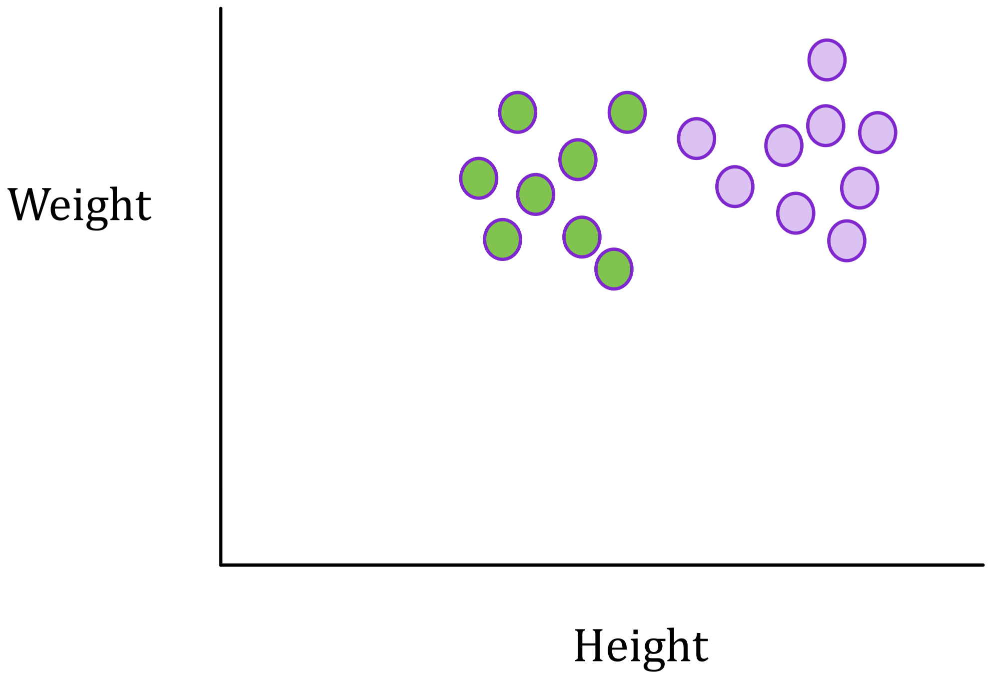



The red point is a non-core point. So it can’t expand a cluster.

Suppose, at start, the green point is chosen for expansion.

The red  point will get included in the first cluster

# DBSCAN: Disadvantage

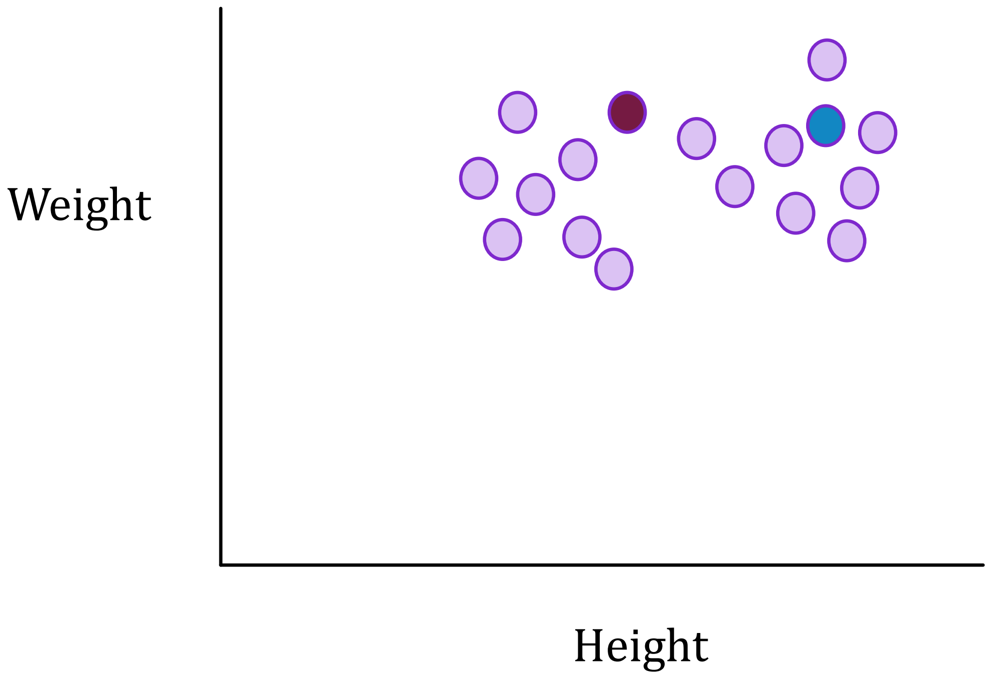



The red point is a non-core point. So it can’t expand a cluster.

Suppose, at start, the green point is chosen for expansion.

The red  point will get included in the first cluster

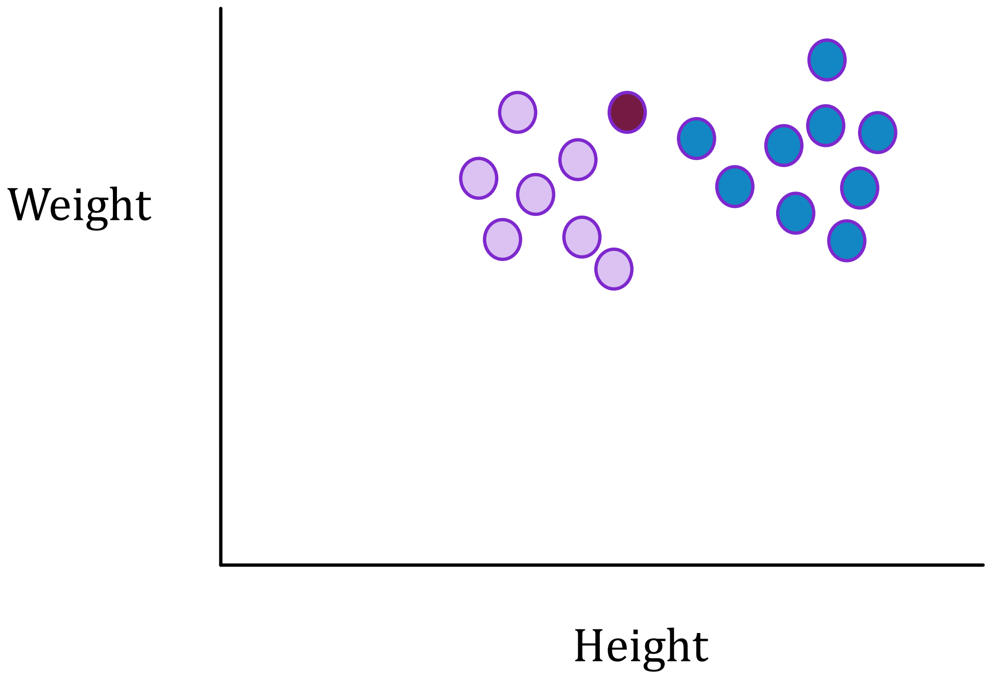
# DBSCAN: Disadvantage




The red point is a non-core point. So it can’t expand a cluster.

But, at start, if the blue point is chosen for expansion,

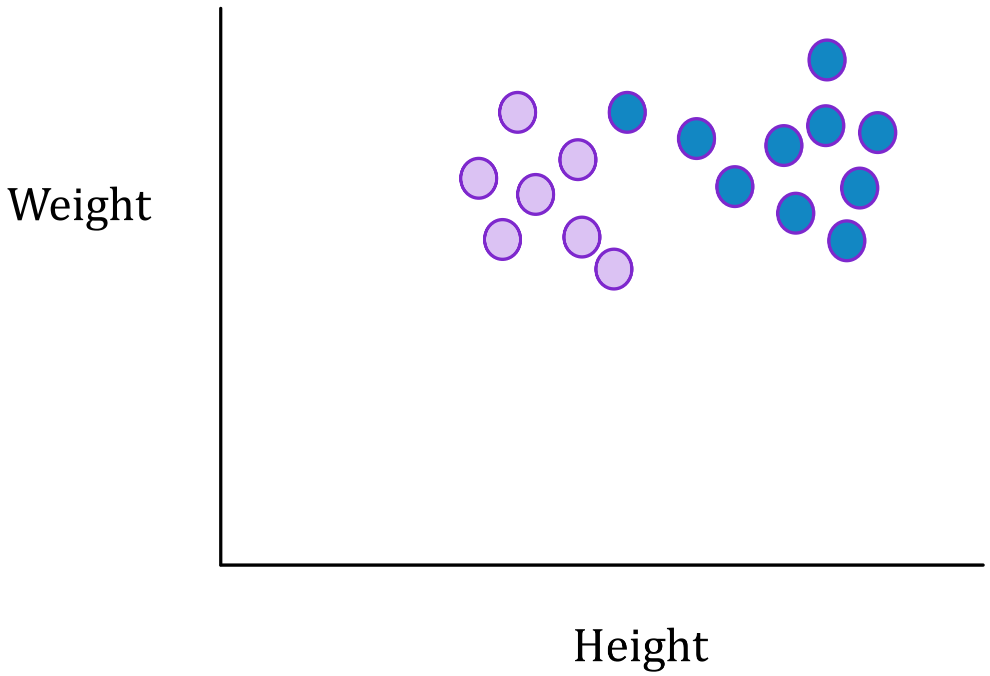
# DBSCAN: Disadvantage




The red point is a non-core point. So it can't expand a cluster.

But, at start, if the blue point is chosen for expansion, the red point will get included in the second cluster

# DBSCAN: Disadvantage




The red point is a non-core point. So it can’t expand a cluster.

But, at start, if the blue point is chosen for expansion, the red point will get included in the second cluster

# Clustering Techniques: DBSCAN

- Density-Based Spatial Clustering of Applications with Noise
- Let $S = \{x_1, x_2, \dots, x_n\}$ be a set of points to be clustered
- The goal is to identify points that form groups (nested clusters)

# Clustering Techniques: DBSCAN

- Let $S = \{x_1, x_2, \dots, x_n\}$ be a set of points to be clustered

1. Choose $m > 0$ and $r$
2. Let $A_i$ be the set of points that lies withing a disc of radius $r$ from $x_i$. Do it for every $x_i$
3. If $A_i < m$, we will not consider this $x_i$ for our calculation
   - Points other than these points are called core points
4. Take union of $A_i$ and $A_j$ if $A_i \cap A_j \neq \phi$
   - Go on doing it no more union operation is possible