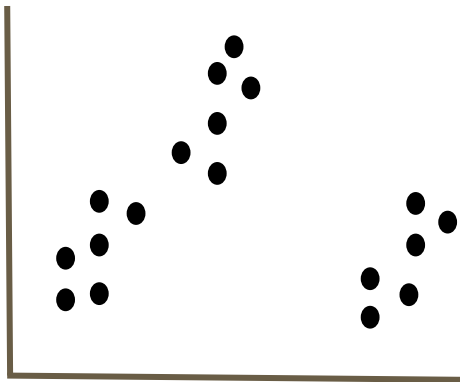
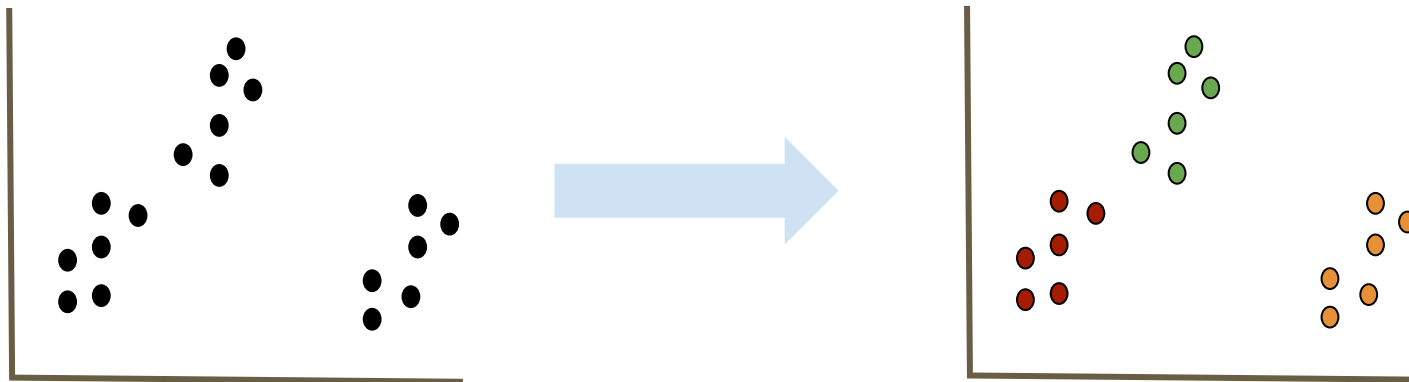

Clustering - Kmeans

— Boston University CS 506 - Lance Galletti —

What is a Clustering



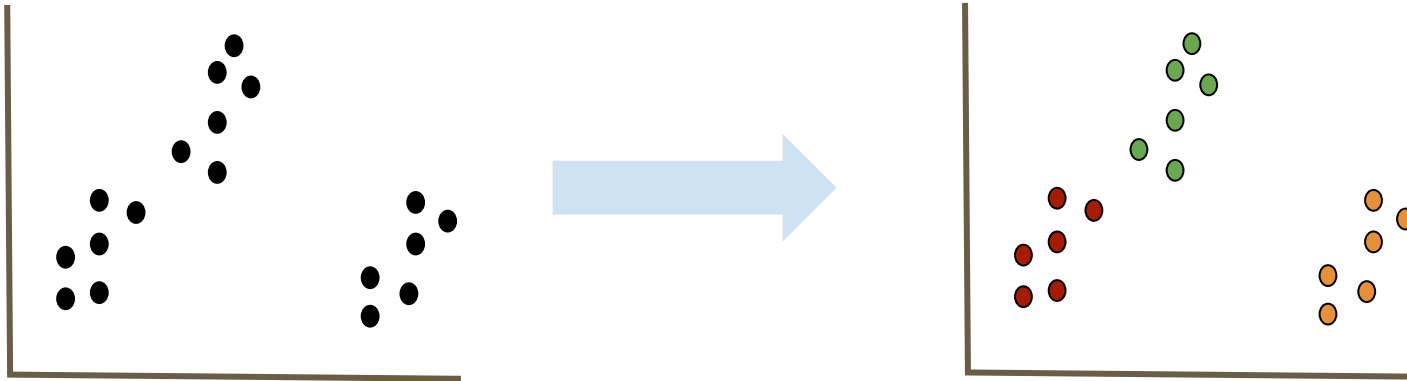
What is a Clustering



What is a Clustering

A clustering is a grouping / assignment of objects (data points) such that objects in the same group / cluster are:

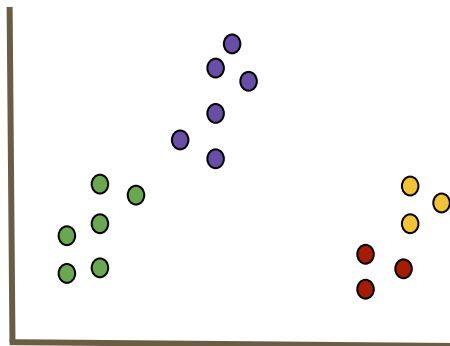
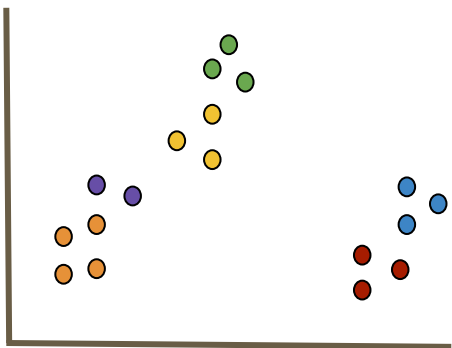
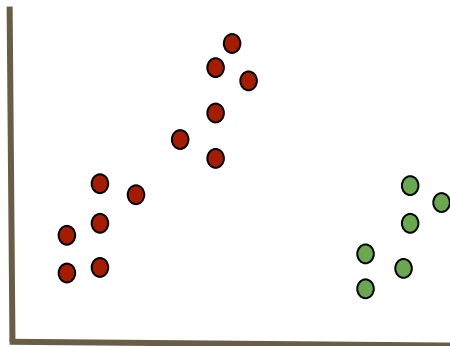
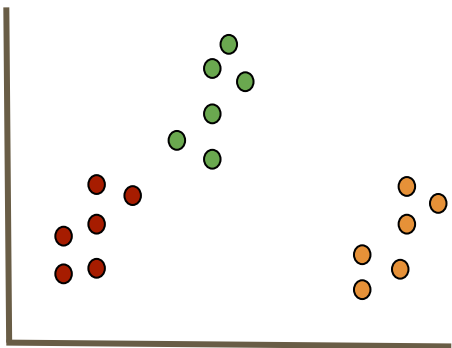
- similar to one another
- dissimilar to objects in other groups



Applications

- Outlier detection / anomaly detection
 - Data Cleaning / Processing
 - Credit card fraud, spam filter etc.
- Feature Extraction
- Filling Gaps in your data
 - Using the same marketing strategy for similar people
 - Infer probable values for gaps in the data (similar users could have similar hobbies, likes / dislikes etc.)

Clusters can be Ambiguous



Types of Clusterings

Partitional

Each object belongs to exactly one cluster

Hierarchical

A set of nested clusters organized in a tree

Density-Based

Defined based on the local density of points

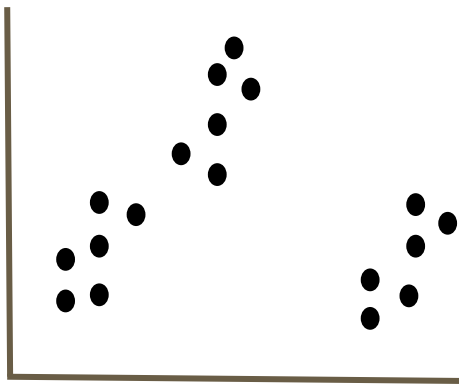
Soft Clustering

Each point is assigned to every cluster with a certain probability

Partitional Clustering

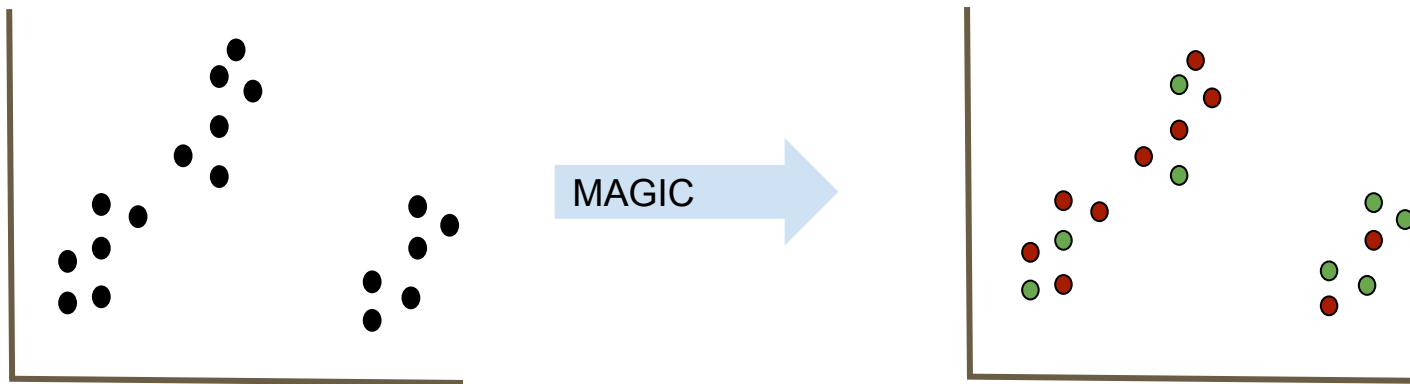
Partitional Clustering

Goal: partition dataset into k partitions



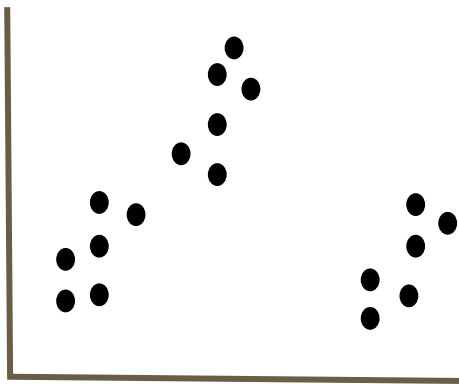
Partitional Clustering

Goal: partition dataset into k partitions



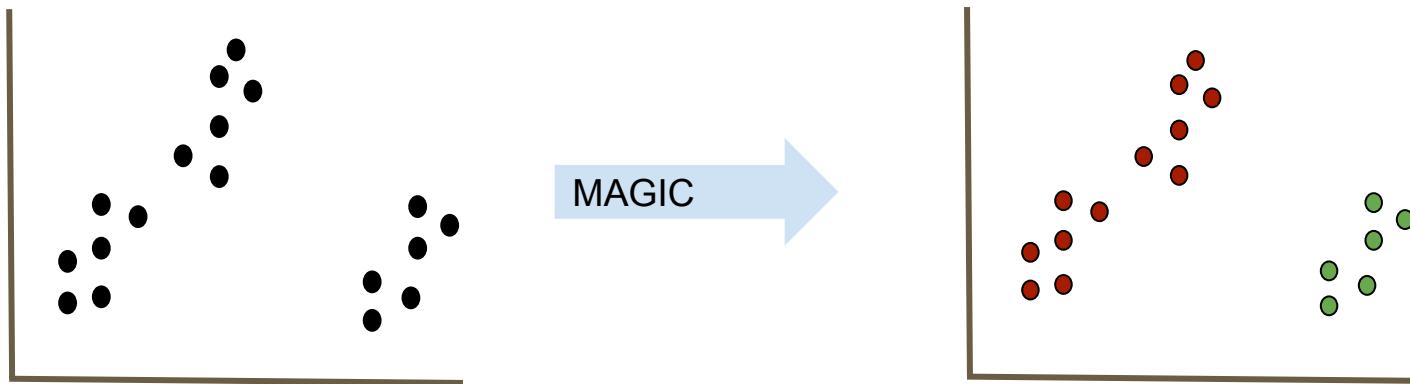
Partitional Clustering

Goal: partition dataset into k partitions

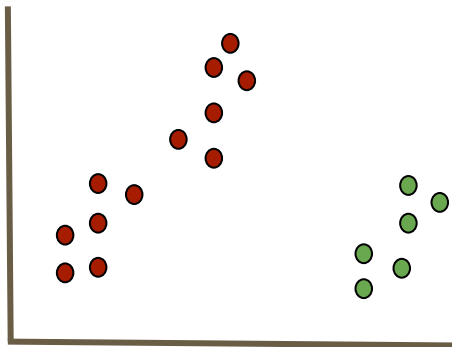


Partitional Clustering

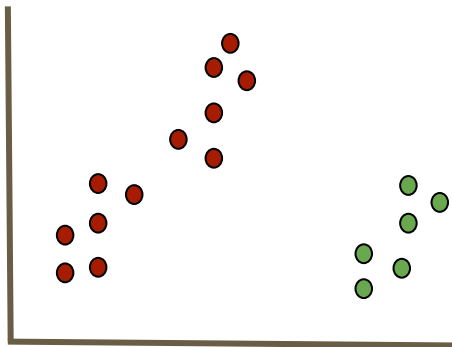
Goal: partition dataset into k partitions



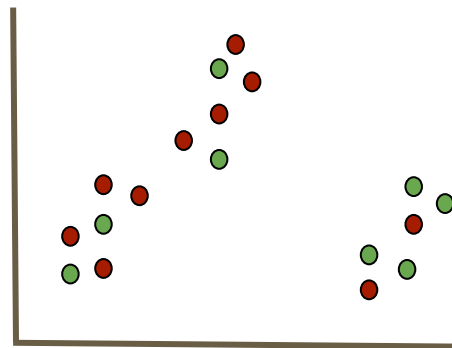
Partitional Clustering



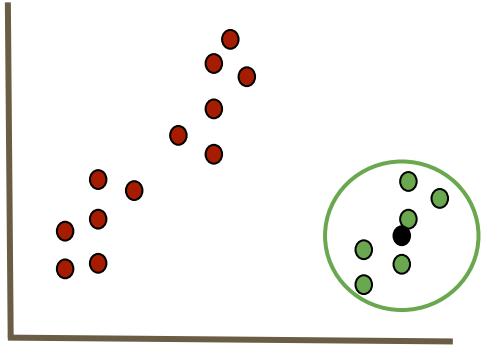
Partitional Clustering



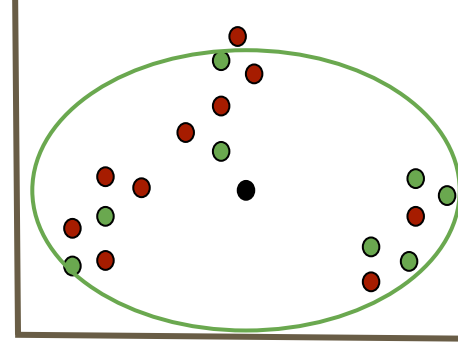
VS



Example

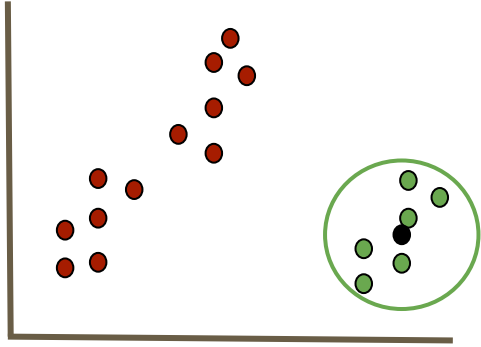


VS

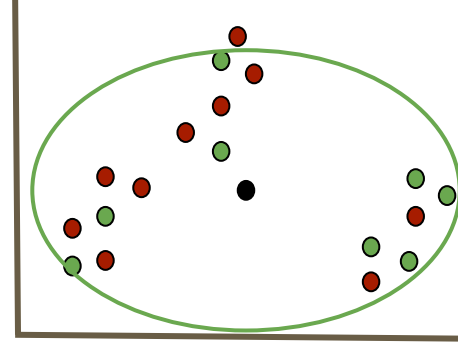


Given a distance function \mathbf{d} , we can find points (not necessarily part of our dataset) for each cluster called **centroids** that are at the center of each cluster.

Example



VS



Q: When **d** is Euclidean, what is the **centroid** (also called **center of mass**) of **m** points $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$?

A: The mean / average of the points

Example



VS

Looking at the sum of the distances of points in a cluster to its centroid also captures the “spread” (variance) of a cluster

$$\sum_i^k \sum_{x \in C_i} d(x, \mu_i)^2$$

Mean of cluster i (points to μ_i)

Cluster i (points to C_i)

Cost Function

- Way to evaluate and compare solutions
- Hope: can find some algorithm that find solutions that make the cost small

Q: Can you suggest a cost function to use for partitional clustering?

$$\sum_i^k \sum_{x \in C_i} d(x, \mu_i)^2$$

K-means

Given $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ our dataset and \mathbf{k}

Find \mathbf{k} points $\{\mu_1, \dots, \mu_k\}$ that minimize the **cost function**:

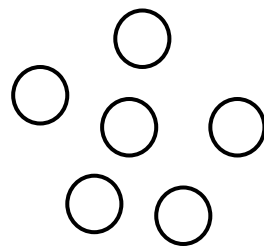
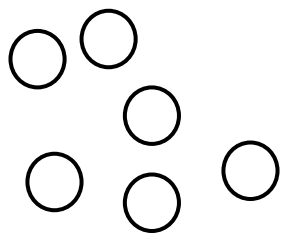
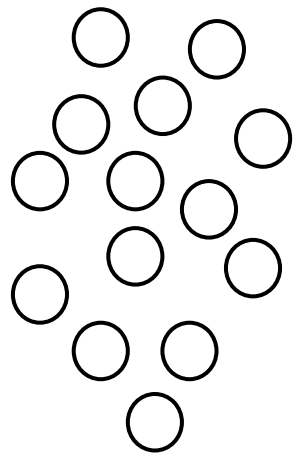
$$\sum_i^k \sum_{x \in C_i} d(x, \mu_i)^2$$

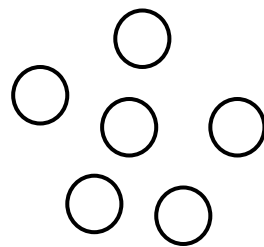
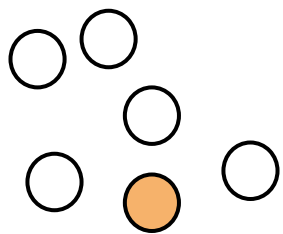
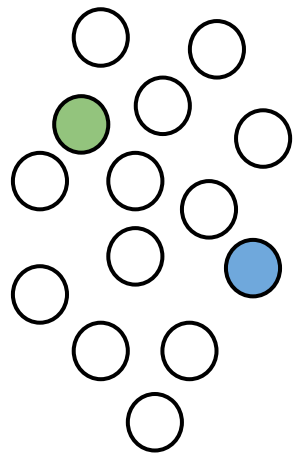
When $\mathbf{k}=1$ and $\mathbf{k}=n$ this is easy. Why?

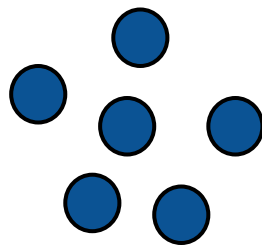
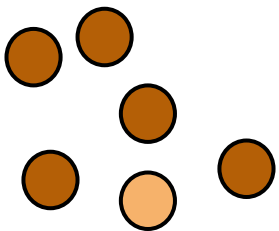
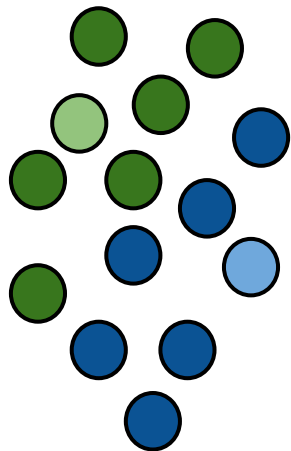
When \mathbf{x}_i lives in more than 2 dimensions, this is a very difficult (**NP-hard**) problem

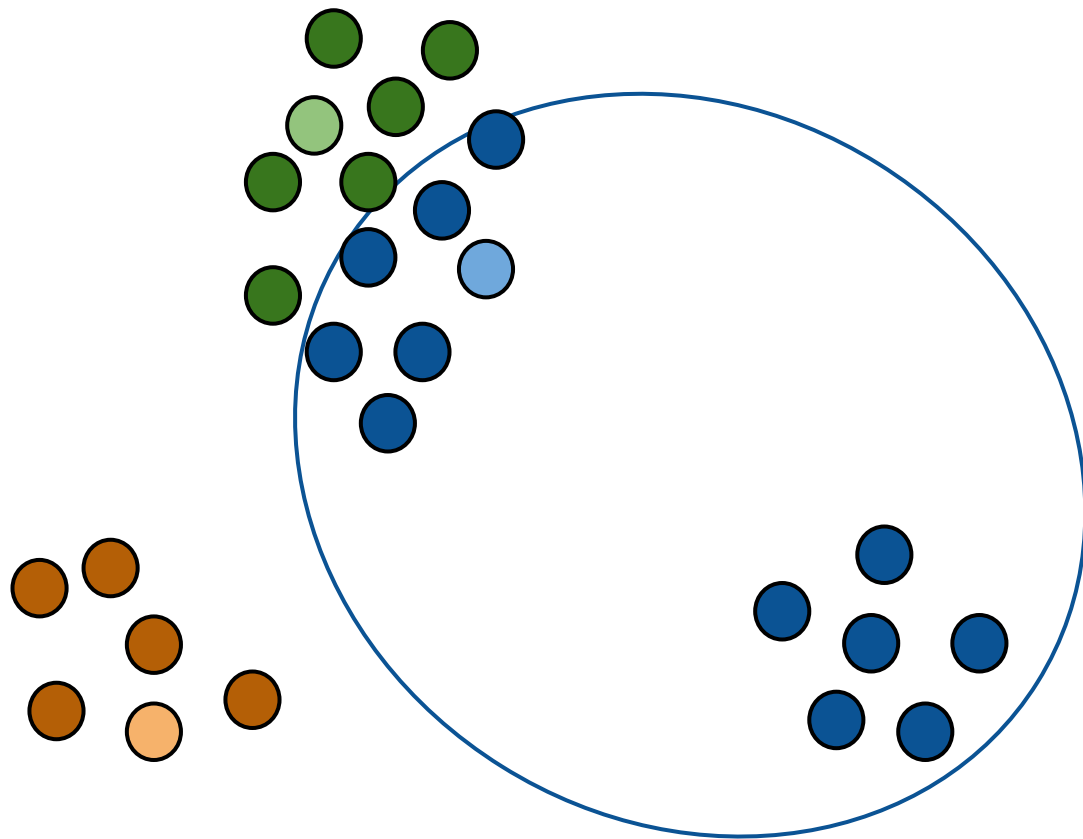
K-means - Lloyd's Algorithm

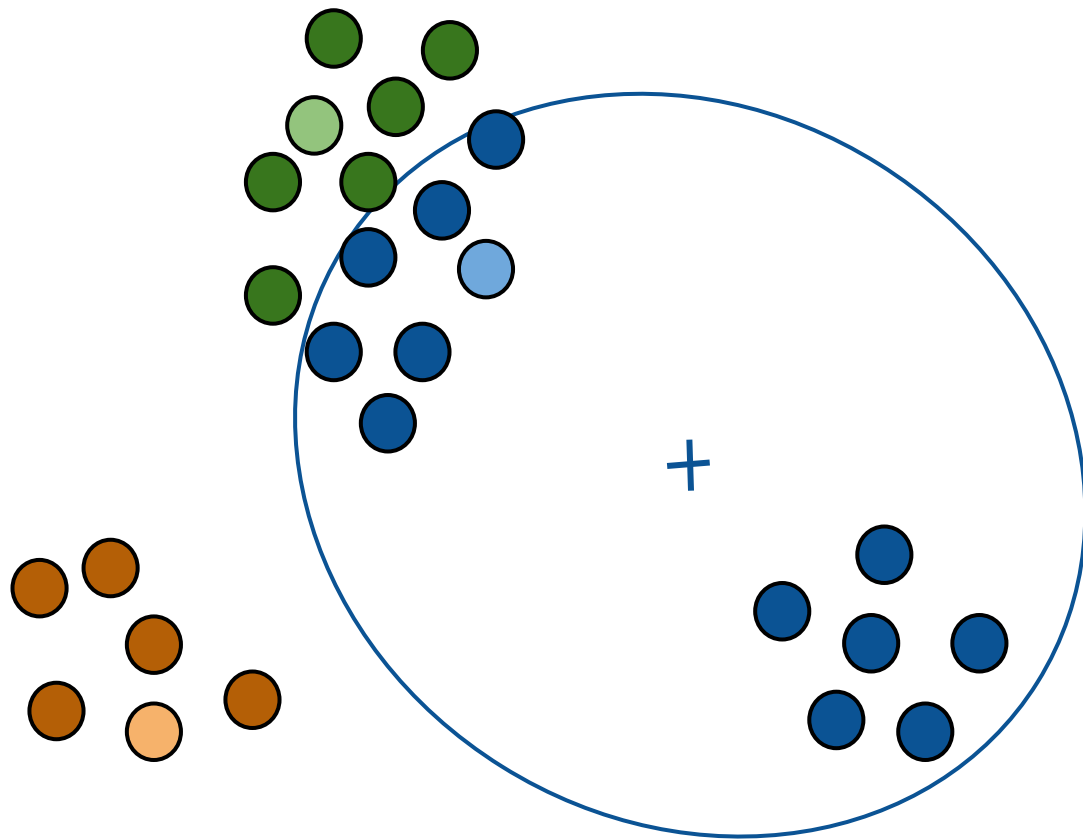
1. Randomly pick k centers $\{\mu_1, \dots, \mu_k\}$
2. Assign each point in the dataset to its closest center
3. Compute the new centers as the means of each cluster
4. Repeat 2 & 3 until convergence

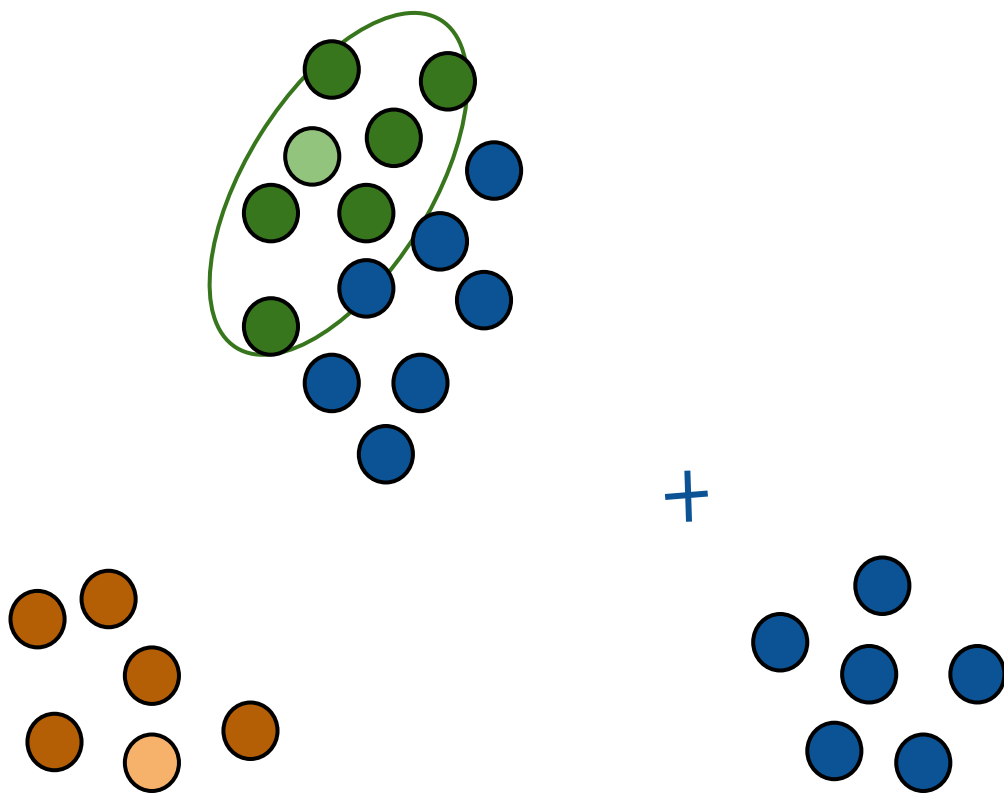


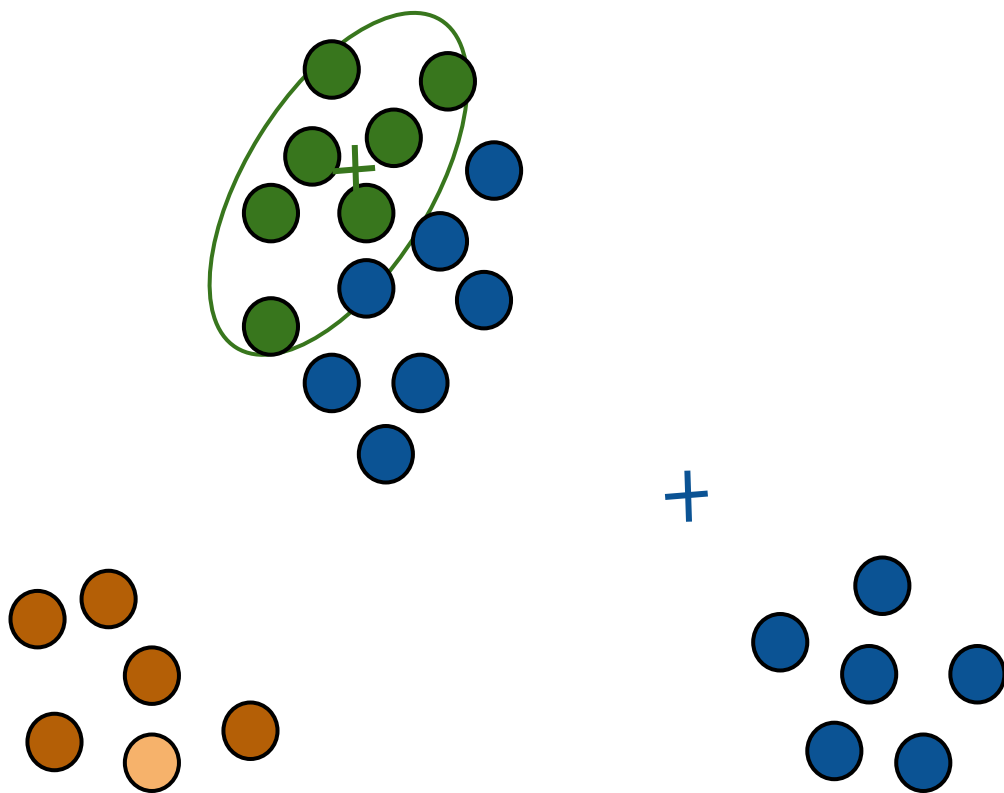


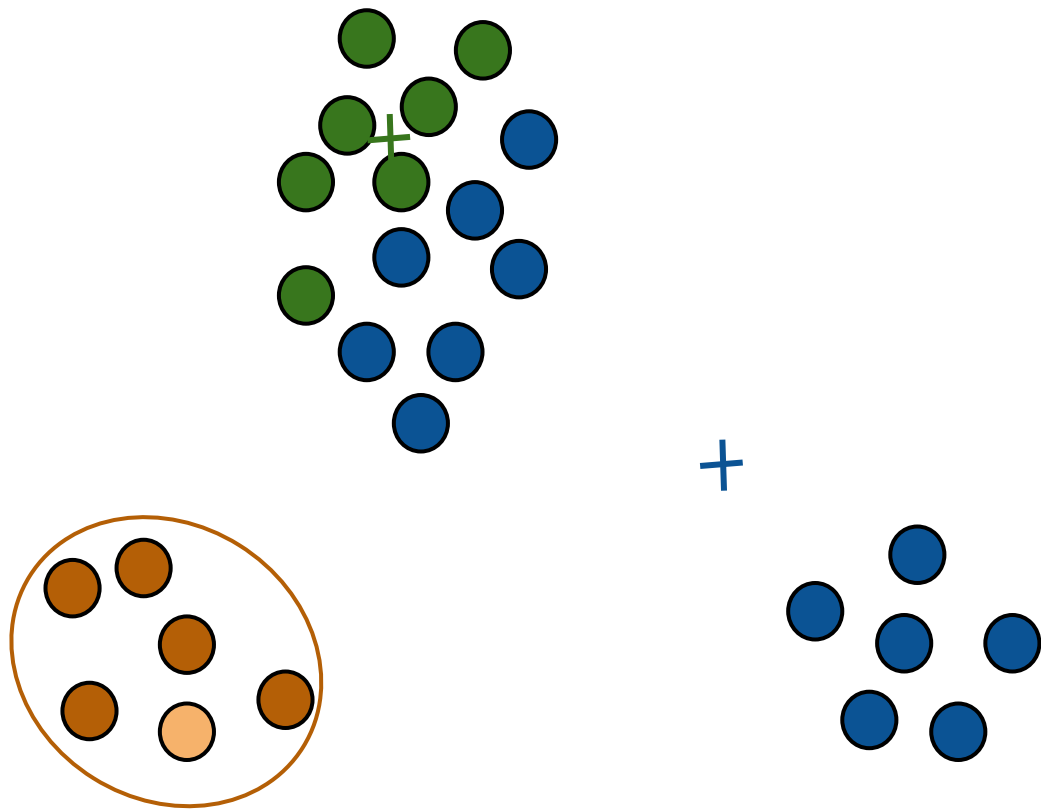


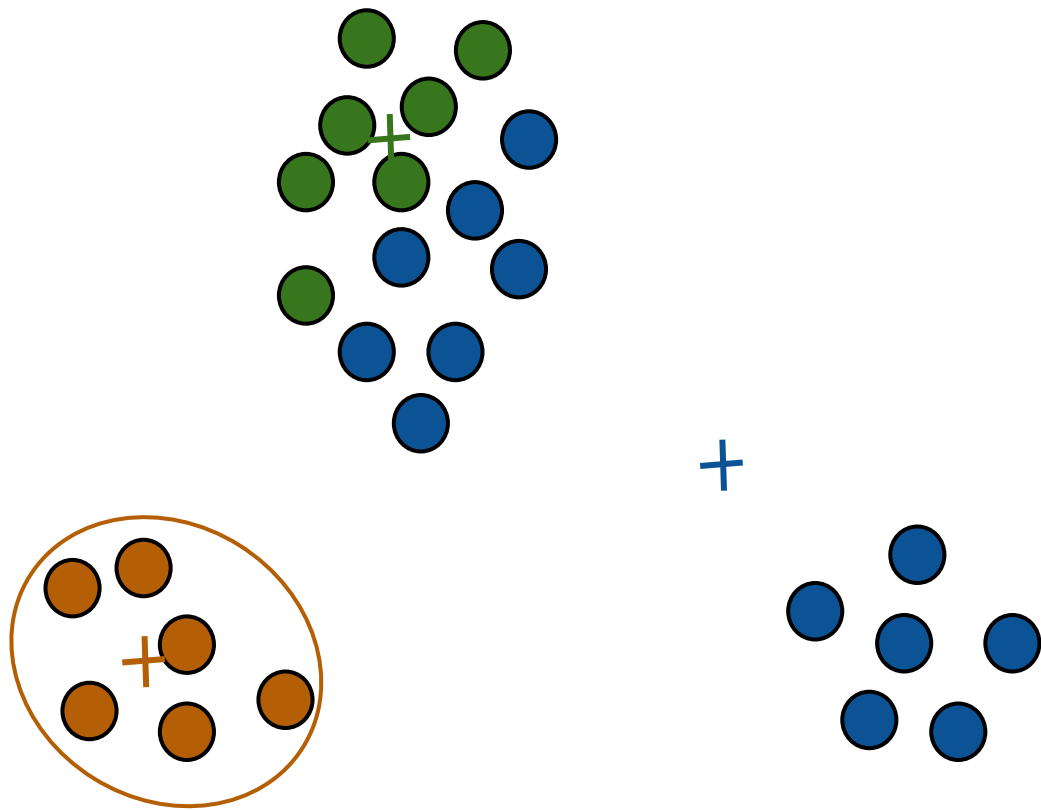


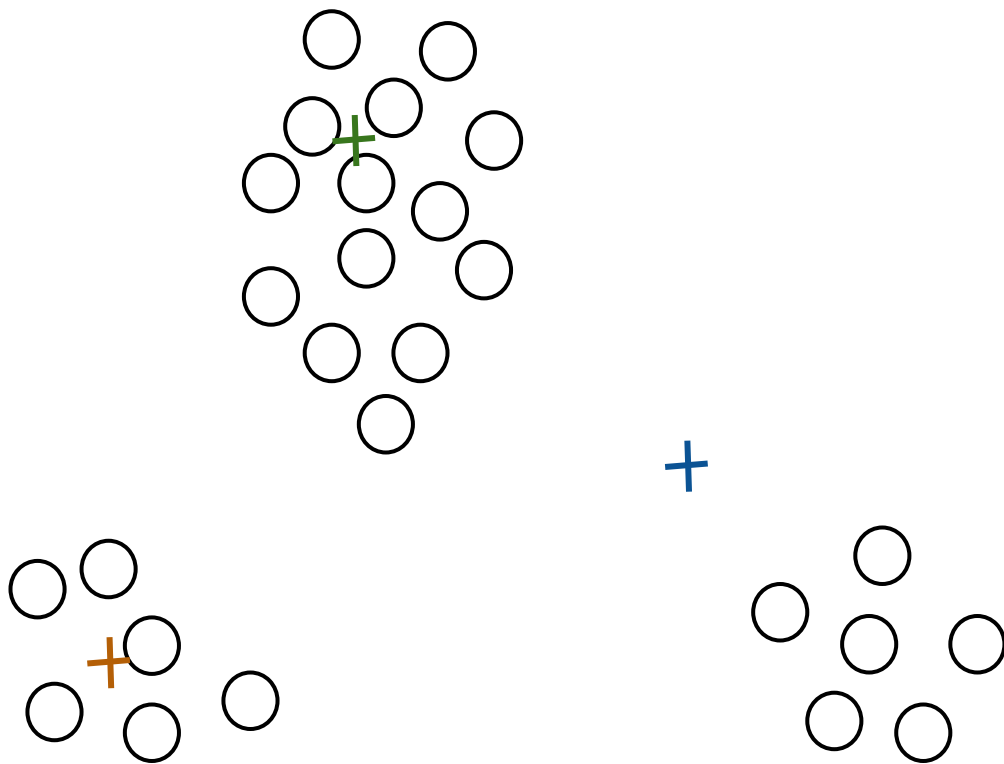


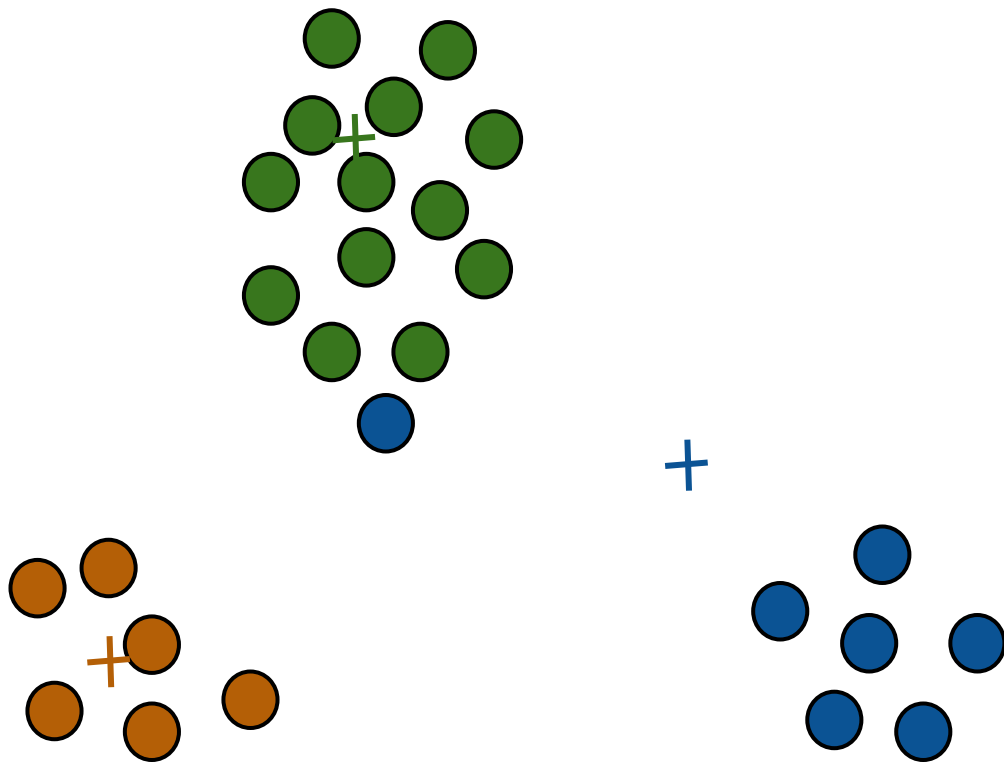


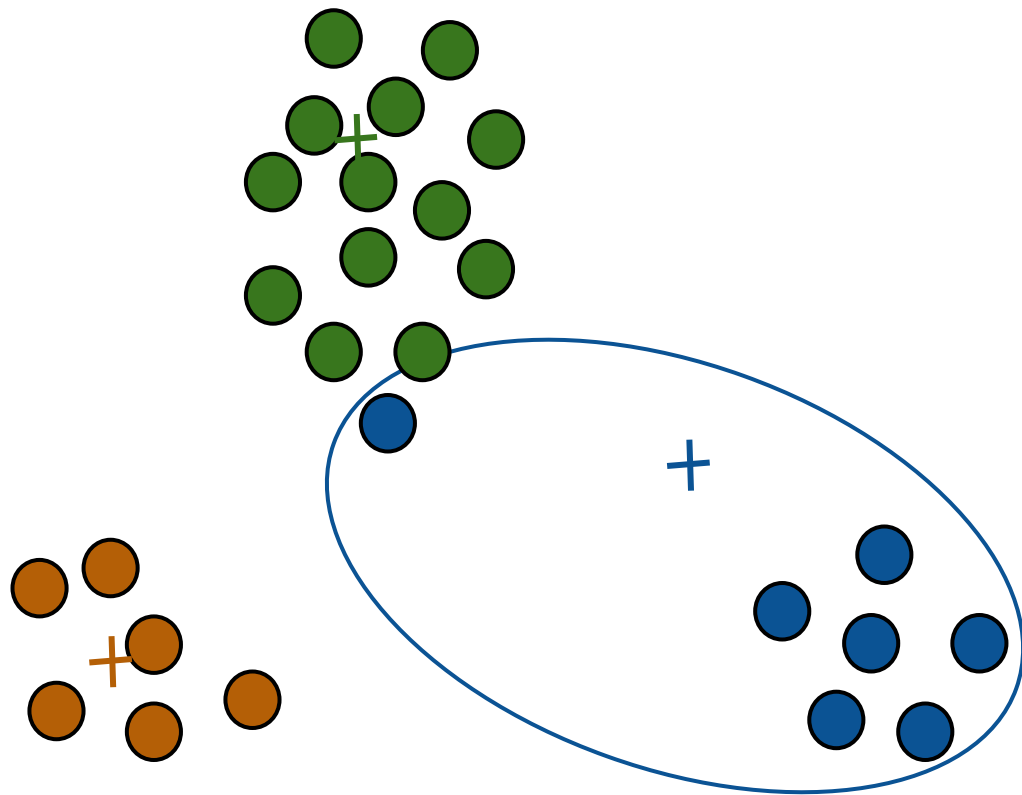


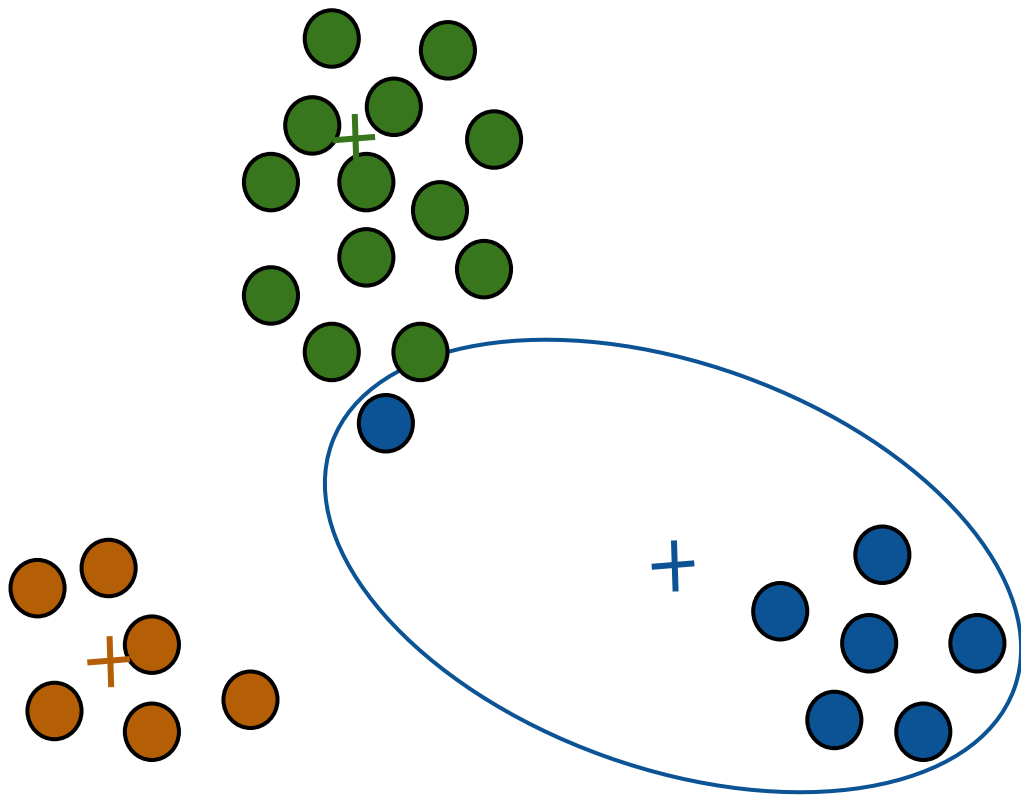


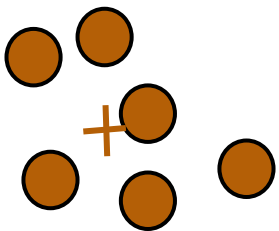
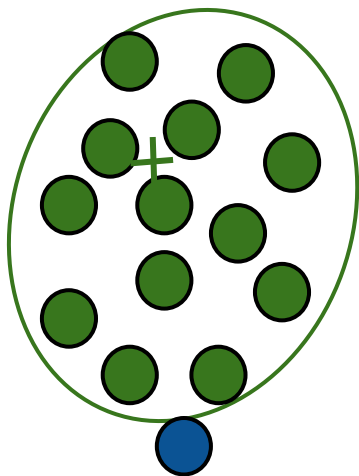




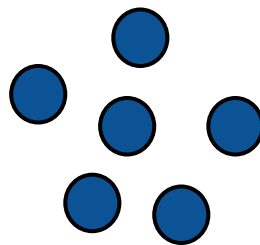


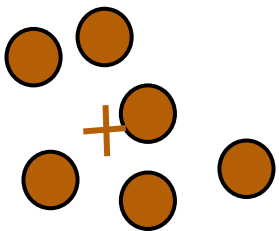
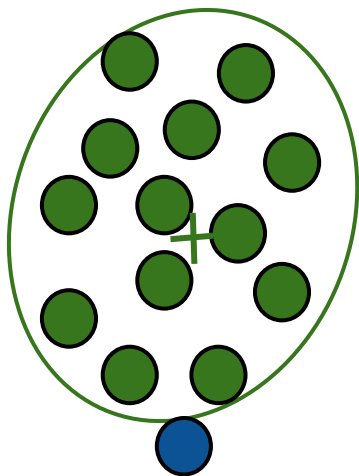




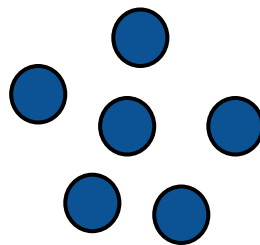


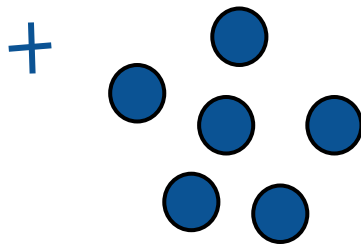
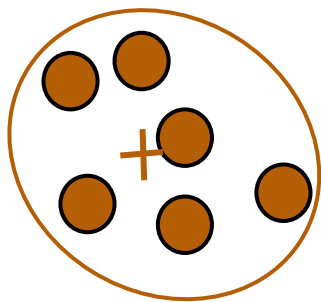
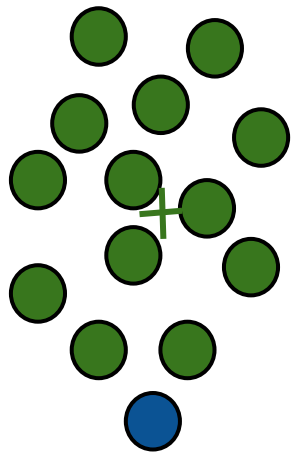
+

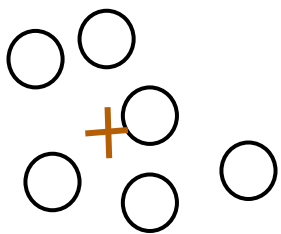
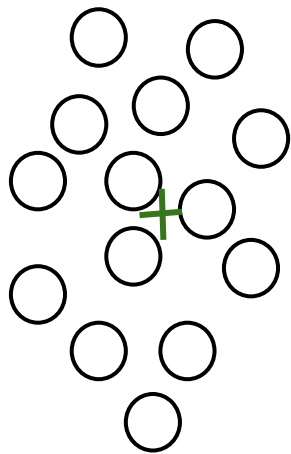




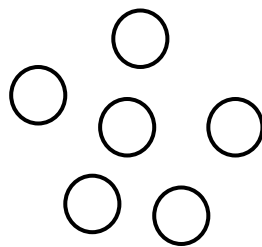
+

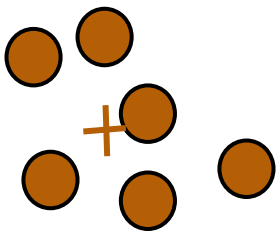
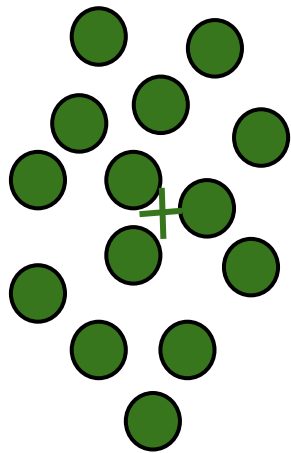




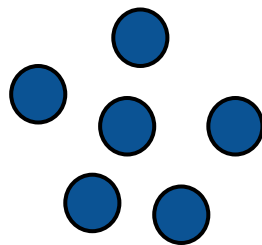


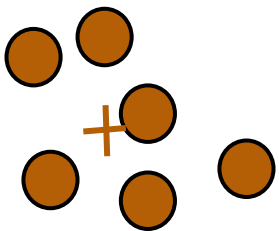
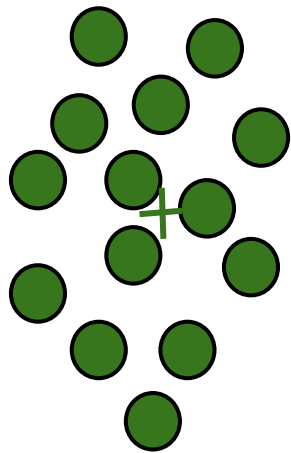
+



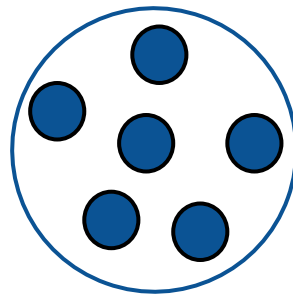


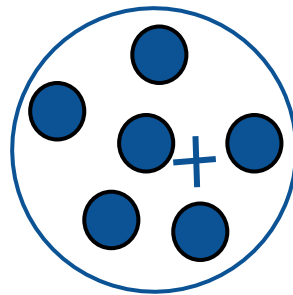
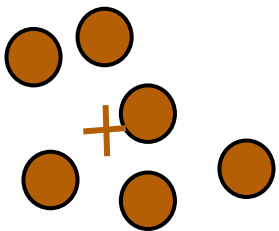
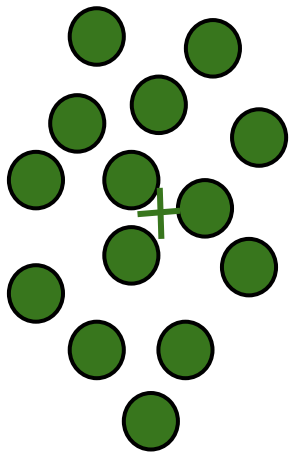
+

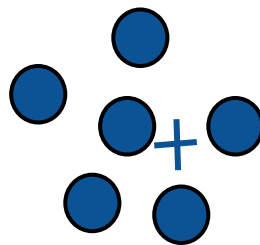
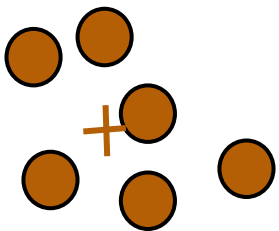
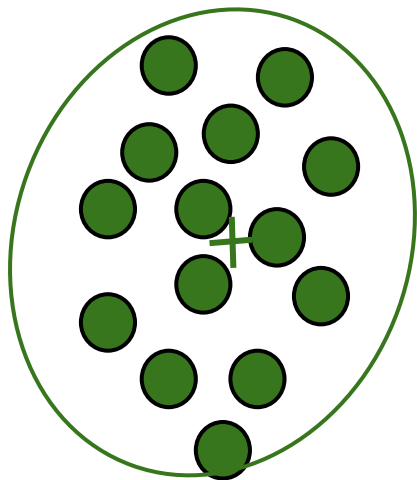


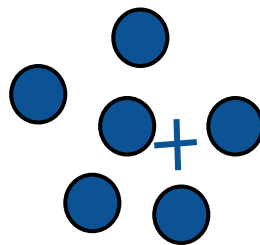
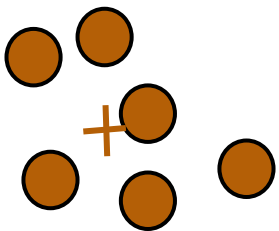
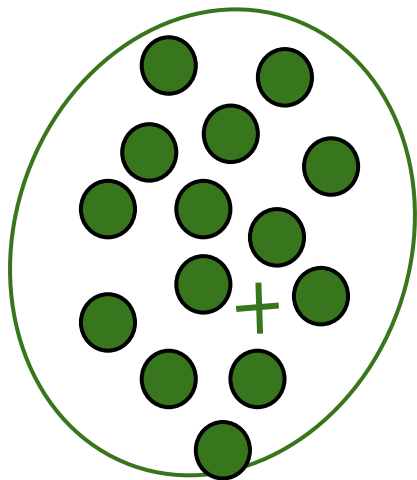


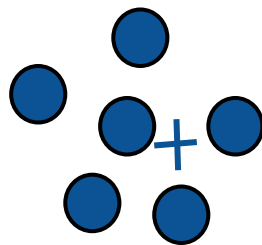
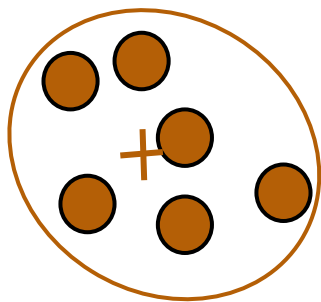
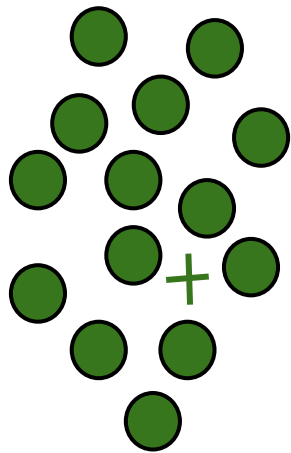
+

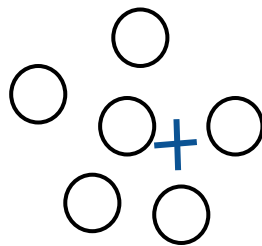
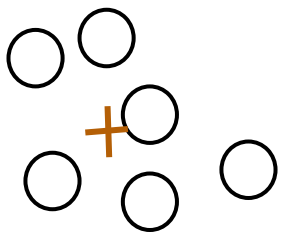
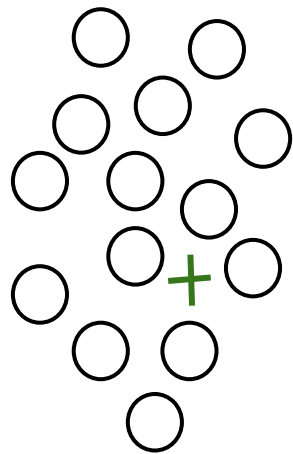


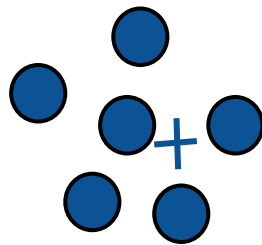
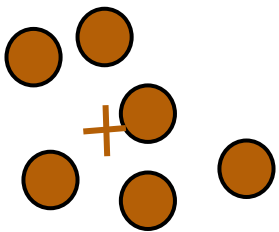
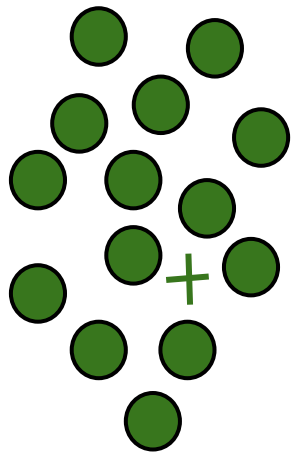


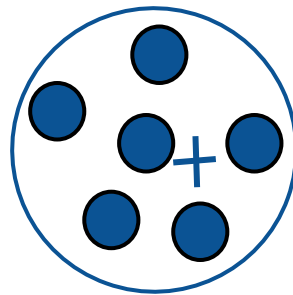
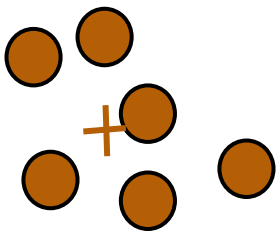
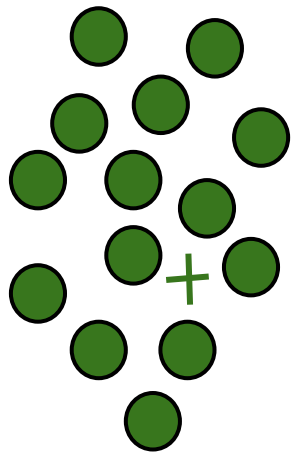


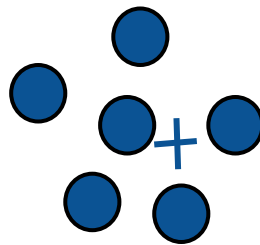
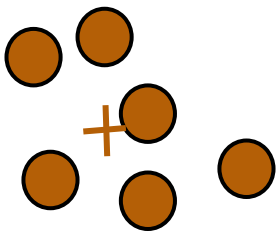
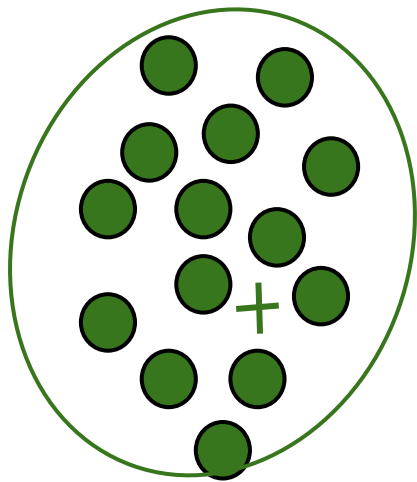


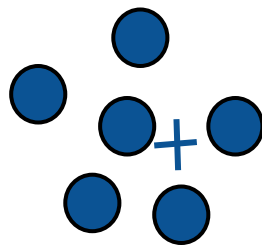
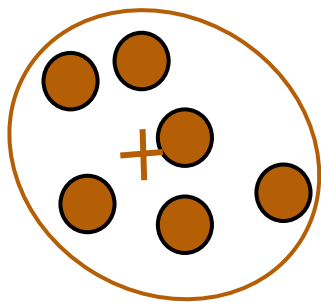
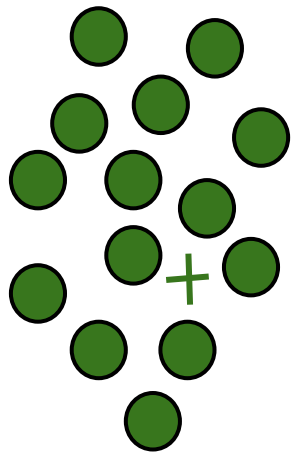


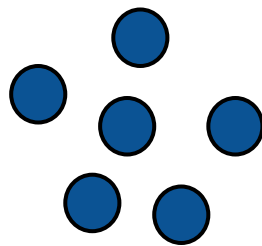
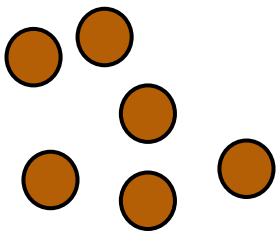
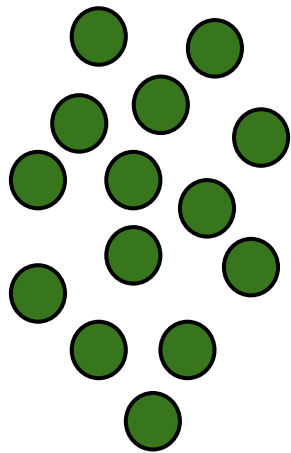




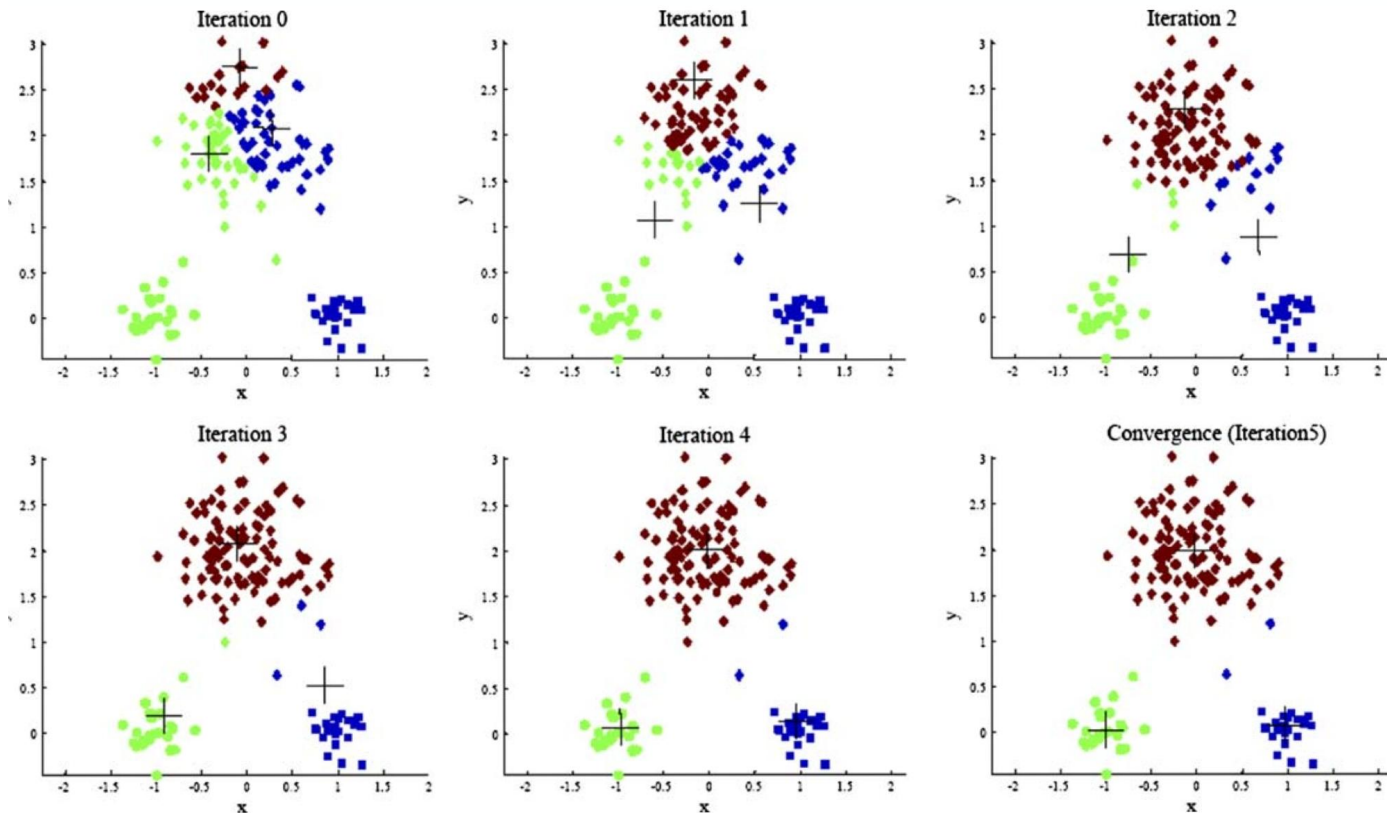








K-means - Lloyd's Algorithm



Worksheet - 5min

Please do a) -> d) of the worksheet with the person sitting next to you.

Worksheet - 5min

Share your answers with the group next to you. Discuss / debate if you have different answers.

K-means - Lloyd's Algorithm

Will this algorithm always converge?

Proof (by contradiction): Suppose it does not converge. Then, either:

1. The minimum of the cost function is only reached in the limit (i.e. after an infinite number of iterations).

Impossible because we are iterating over a finite set of partitions

1. The algorithm gets stuck in a cycle / loop

Impossible since this would require having a clustering that has a lower cost than itself and we know:

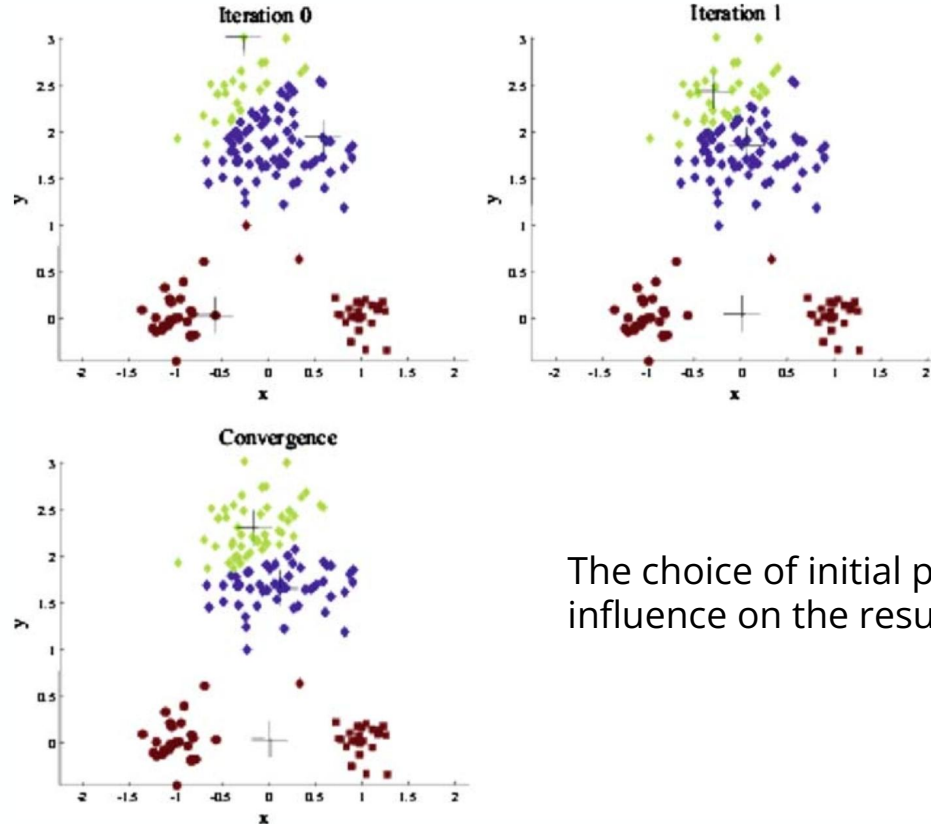
- If $\text{old} \neq \text{new}$ clustering then the cost has improved
- If $\text{old} = \text{new}$ clustering then the cost is unchanged

Conclusion: Lloyd's Algorithm always converges!

K-means - Lloyd's Algorithm

Will this always converge to the optimal solution?

K-means - Lloyd's Algorithm



The choice of initial points has a large influence on the resulting clustering

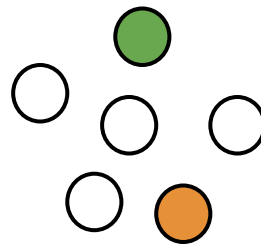
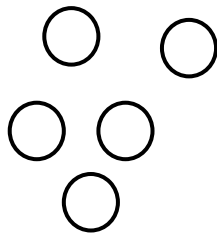
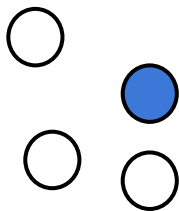
K-means - Initialization

One solution: Run Lloyd's algorithm multiple times and choose the result with the lowest cost.

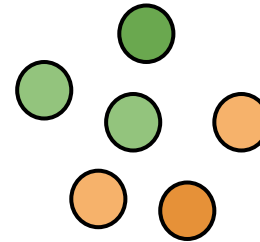
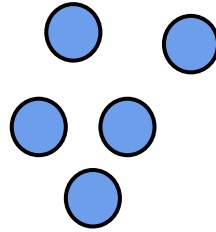
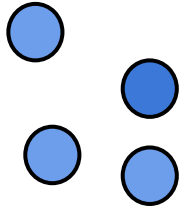
This can still lead to bad results because of randomness.

Another solution: Try different initialization methods

K-means - Random

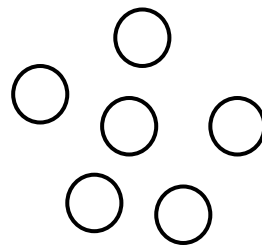
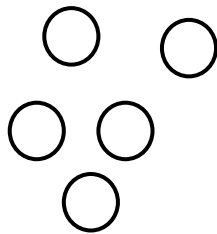
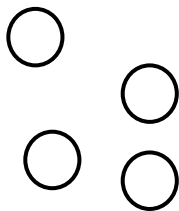


K-means - Random

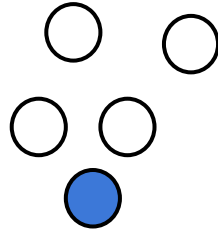
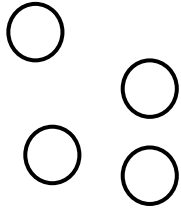


Starting with initialization points too close to each other may be problematic

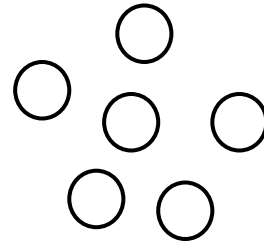
K-means - Farthest First Traversal



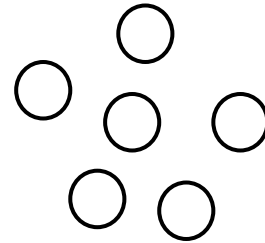
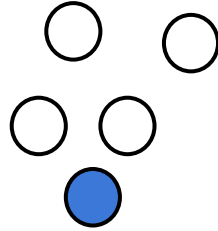
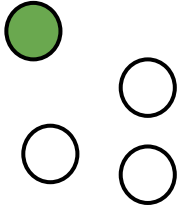
K-means - Farthest First Traversal



Pick the first center at random

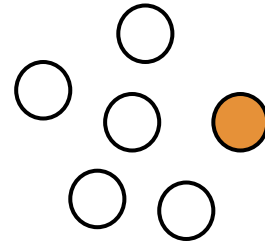
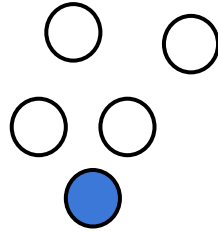
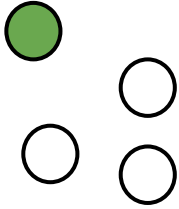


K-means - Farthest First Traversal



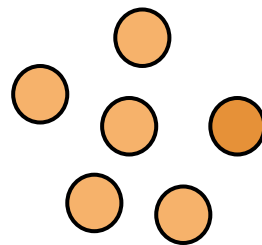
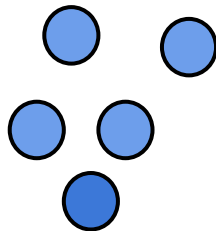
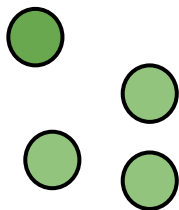
Pick the next center to be the point farthest from all previous

K-means - Farthest First Traversal

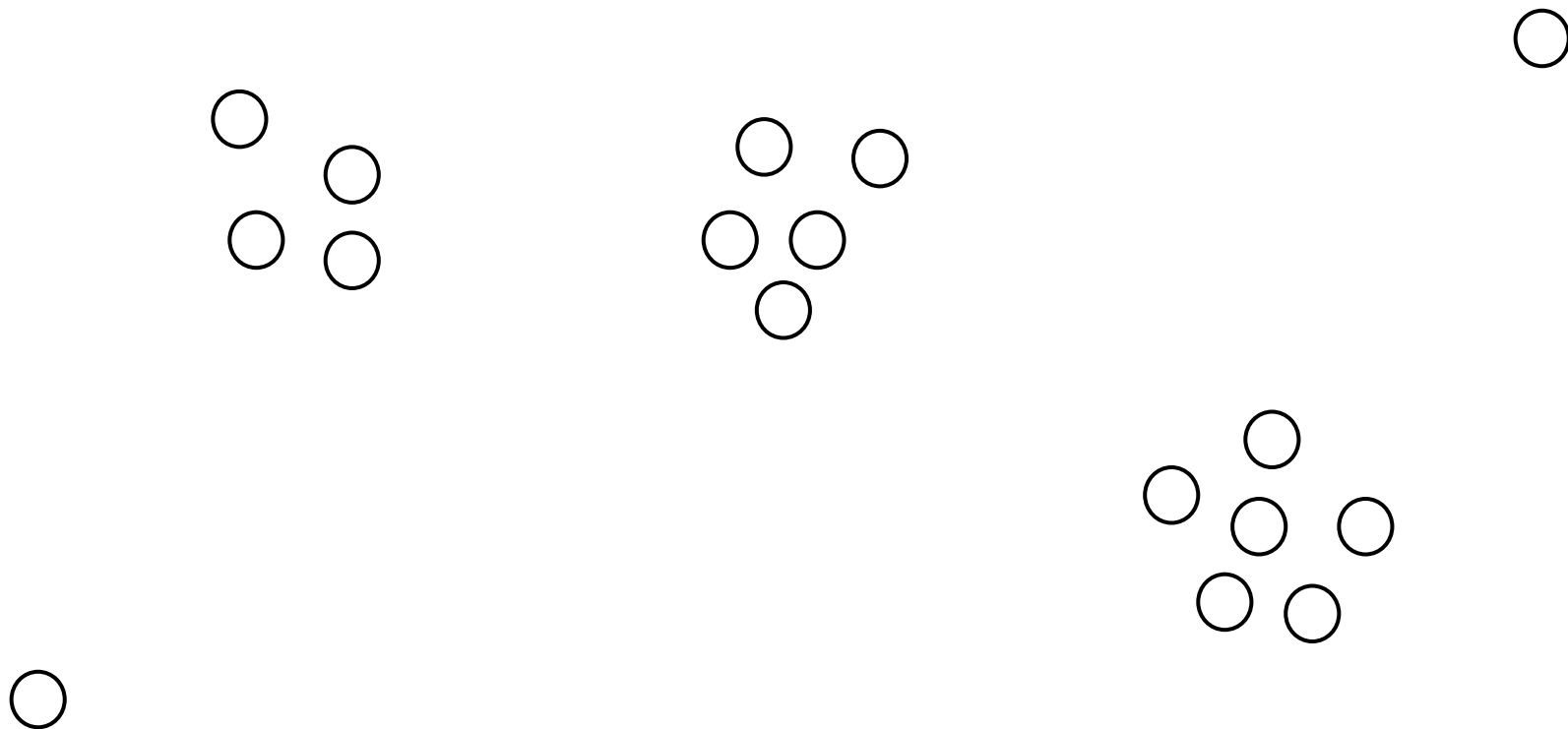


Pick the next center to be the point farthest from all previous

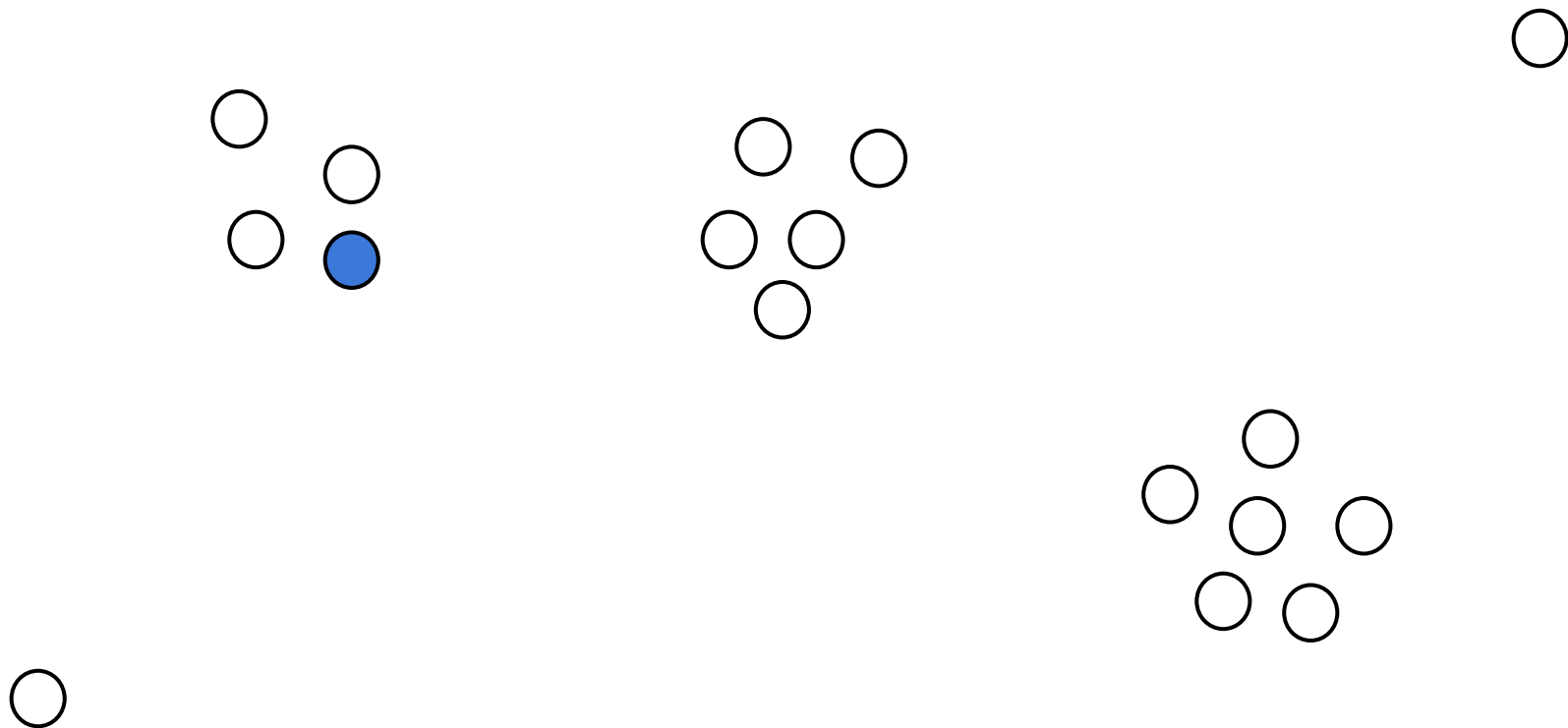
K-means - Farthest First Traversal



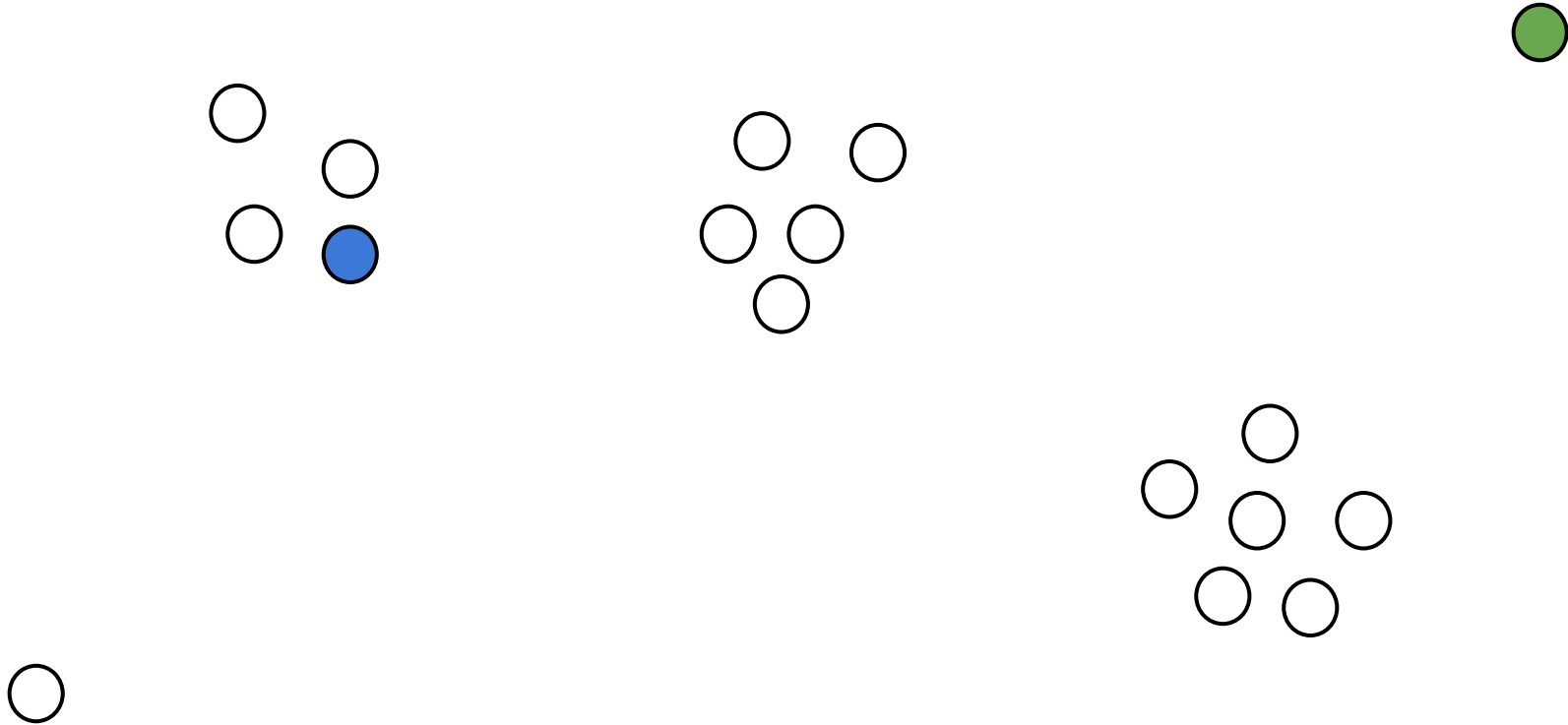
K-means - FFT and outliers



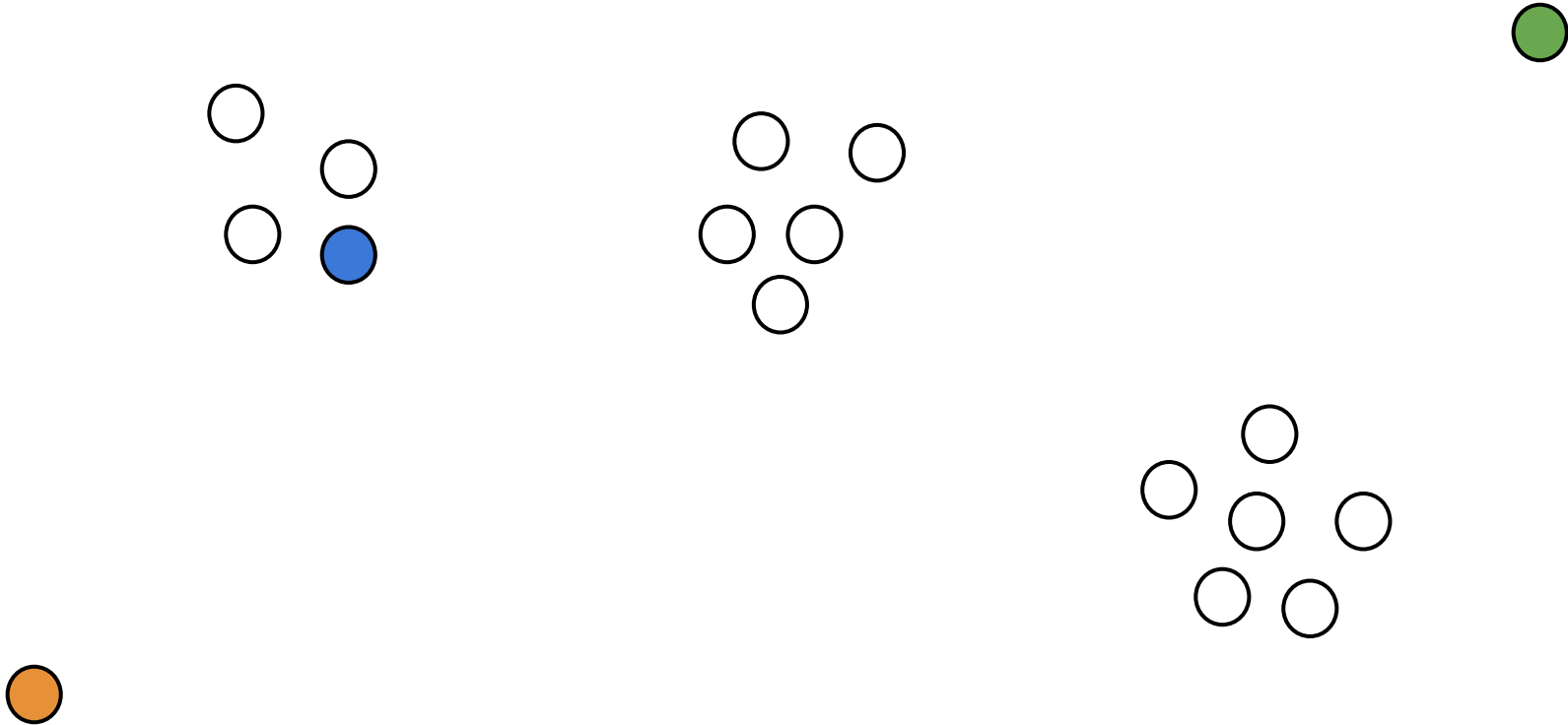
K-means - FFT and outliers



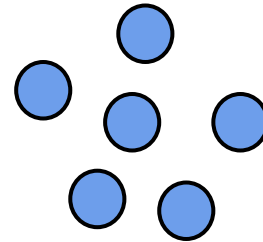
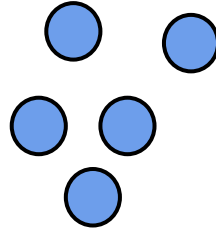
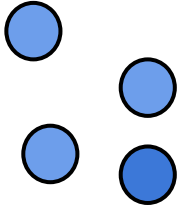
K-means - FFT and outliers



K-means - FFT and outliers



K-means - FFT and outliers



Random might have worked better here



K-means++

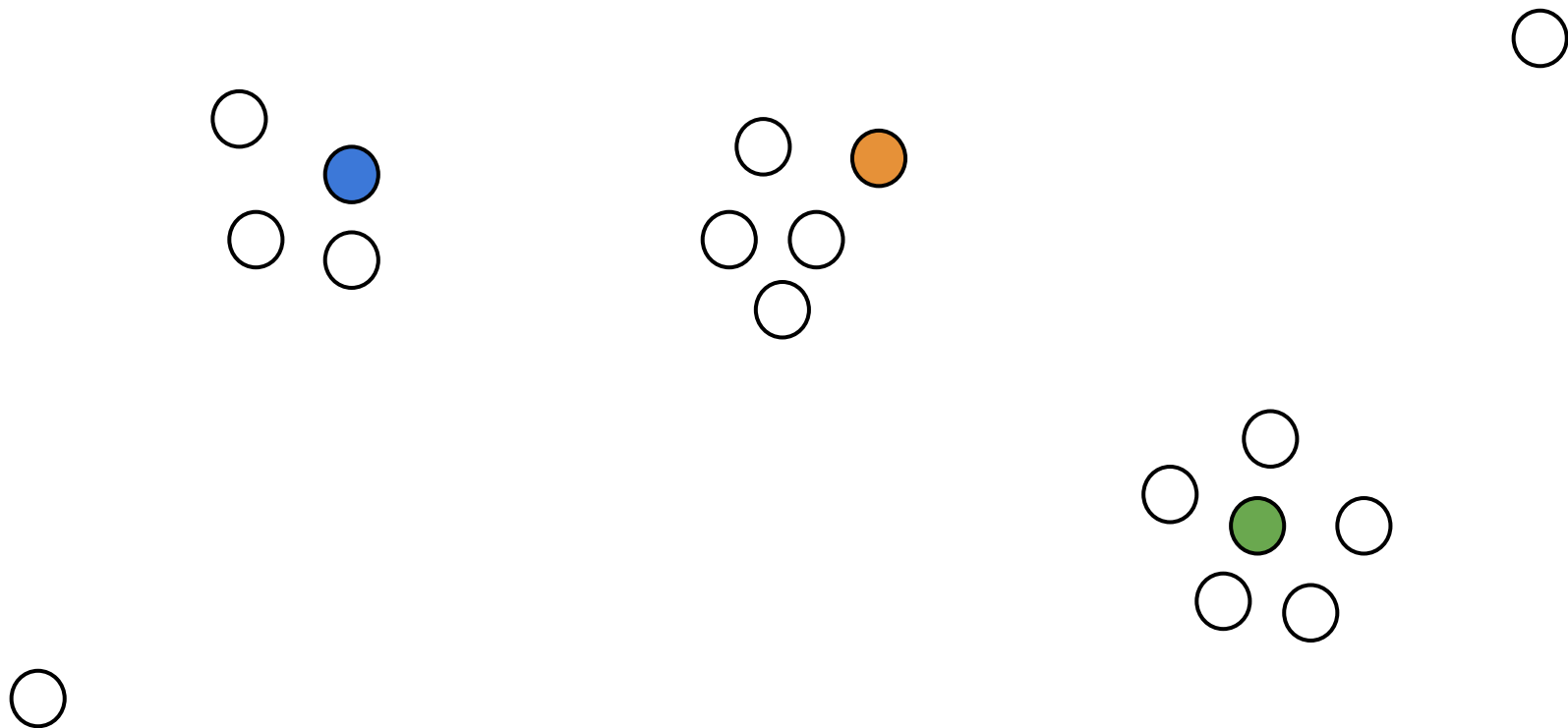
Initialize with a combination of the two methods:

1. Start with a random center
2. Let $\mathbf{D}(\mathbf{x})$ be the distance between \mathbf{x} and the centers selected so far.
Choose the next center with probability proportional to $\mathbf{D}(\mathbf{x})^a$

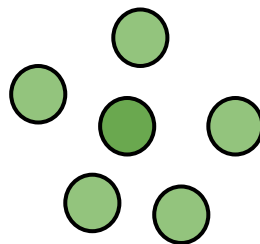
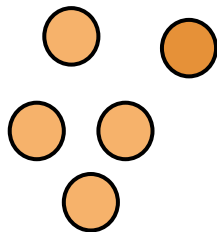
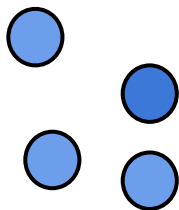
When:

- $\mathbf{a} = \mathbf{0}$: random initialization (all points have equal probability)
- $\mathbf{a} = \infty$: farthest first traversal
- $\mathbf{a} = 2$: K-means++

K-means++



K-means++



No reason to use k-means over
k-means++

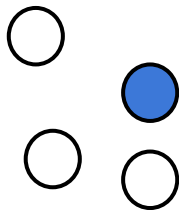


K-means++

Suppose we are given a black box that will generate a uniform random number between 0 and any **N**. How can we use this black box to select points with probability proportional to **D(x)^a**?

K-means++

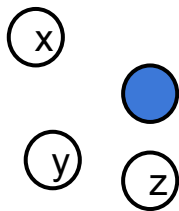
Suppose we are given a black box that will generate a uniform random number between 0 and any \mathbf{N} . How can we use this black box to select points with probability proportional to $\mathbf{D}(\mathbf{x})^a$?



K-means++

Suppose we are given a black box that will generate a uniform random number between 0 and any **N**. How can we use this black box to select points with probability proportional to $D(\mathbf{x})^2$?

Let's set $a = 2$



$$D(\mathbf{x})^2 = 3^2 = 9$$

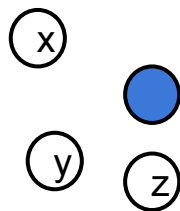
$$D(\mathbf{y})^2 = 2^2 = 4$$

$$D(\mathbf{z})^2 = 1^2 = 1$$

K-means++

Suppose we are given a black box that will generate a uniform random number between 0 and any **N**. How can we use this black box to select points with probability proportional to $D(\mathbf{x})^2$?

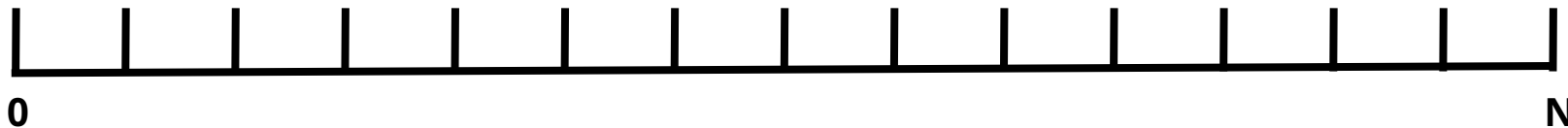
Let's set $a = 2$



$$D(\mathbf{x})^2 = 3^2 = 9$$

$$D(\mathbf{y})^2 = 2^2 = 4$$

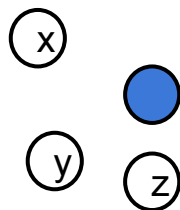
$$D(\mathbf{z})^2 = 1^2 = 1$$



K-means++

Suppose we are given a black box that will generate a uniform random number between 0 and any **N**. How can we use this black box to select points with probability proportional to $D(\mathbf{x})^2$?

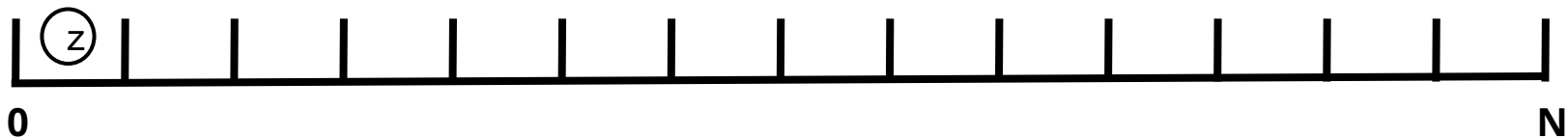
Let's set $a = 2$



$$D(\mathbf{x})^2 = 3^2 = 9$$

$$D(\mathbf{y})^2 = 2^2 = 4$$

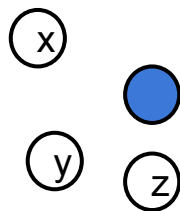
$$D(\mathbf{z})^2 = 1^2 = 1$$



K-means++

Suppose we are given a black box that will generate a uniform random number between 0 and any **N**. How can we use this black box to select points with probability proportional to $D(\mathbf{x})^2$?

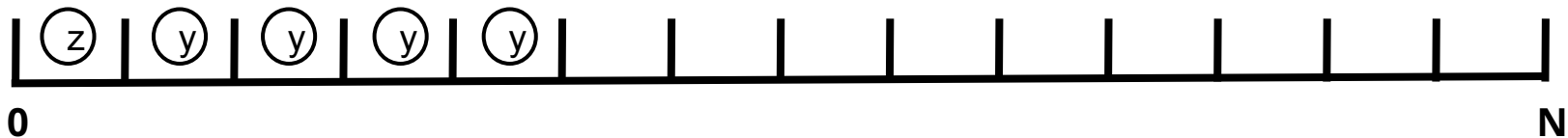
Let's set $a = 2$



$$D(\mathbf{x})^2 = 3^2 = 9$$

$$D(\mathbf{y})^2 = 2^2 = 4$$

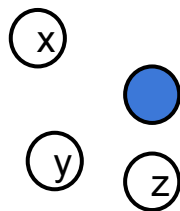
$$D(\mathbf{z})^2 = 1^2 = 1$$



K-means++

Suppose we are given a black box that will generate a uniform random number between 0 and any **N**. How can we use this black box to select points with probability proportional to $D(\mathbf{x})^2$?

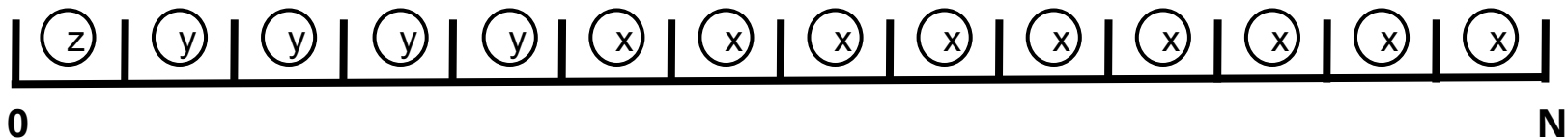
Let's set $a = 2$



$$D(\mathbf{x})^2 = 3^2 = 9$$

$$D(\mathbf{y})^2 = 2^2 = 4$$

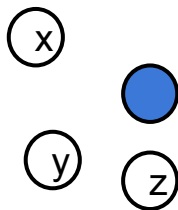
$$D(\mathbf{z})^2 = 1^2 = 1$$



K-means++

Suppose we are given a black box that will generate a uniform random number between 0 and any **N**. How can we use this black box to select points with probability proportional to $D(\mathbf{x})^2$?

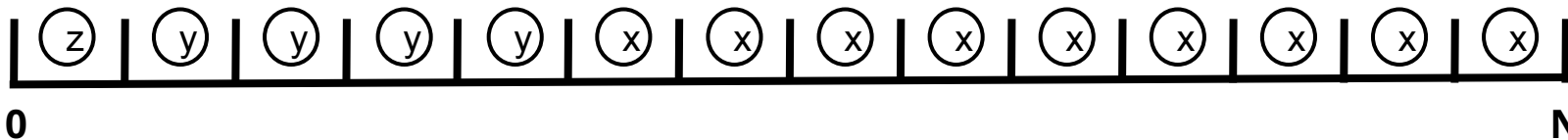
Let's set $a = 2$



$$D(\mathbf{x})^2 = 3^2 = 9$$

$$D(\mathbf{y})^2 = 2^2 = 4$$

$$D(\mathbf{z})^2 = 1^2 = 1$$

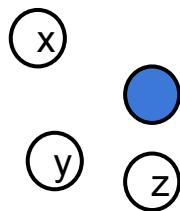


$$\begin{aligned} &= D(\mathbf{x})^2 + D(\mathbf{y})^2 \\ &\quad + D(\mathbf{z})^2 = 14 \end{aligned}$$

K-means++

Suppose we are given a black box that will generate a uniform random number between 0 and any **N**. How can we use this black box to select points with probability proportional to $D(\mathbf{x})^2$?

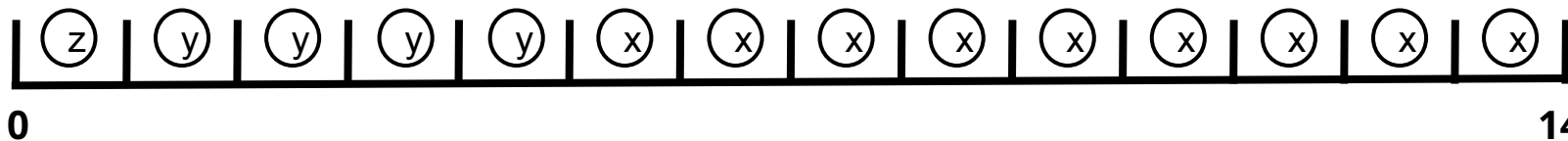
Let's set $a = 2$



$$D(\mathbf{x})^2 = 3^2 = 9$$

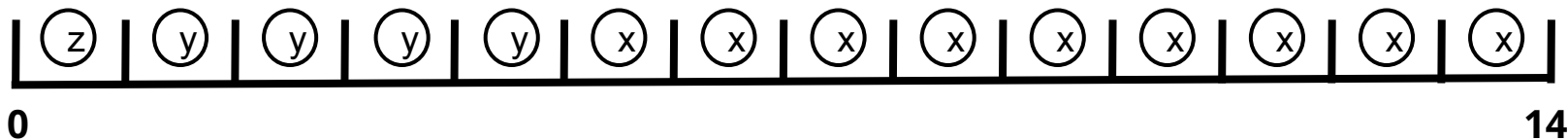
$$D(\mathbf{y})^2 = 2^2 = 4$$

$$D(\mathbf{z})^2 = 1^2 = 1$$



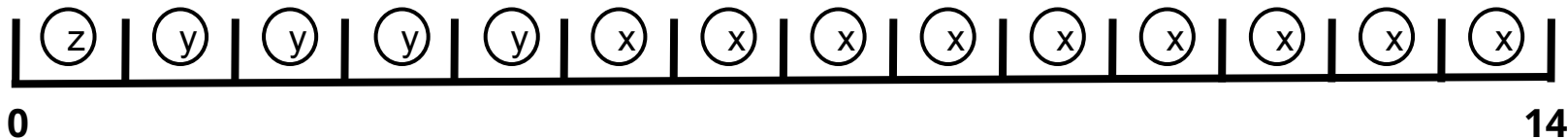
K-means++

Q: the black box returns "12" as the random number generated. Which point do we choose for the next center (x, y, or z) ?



K-means++

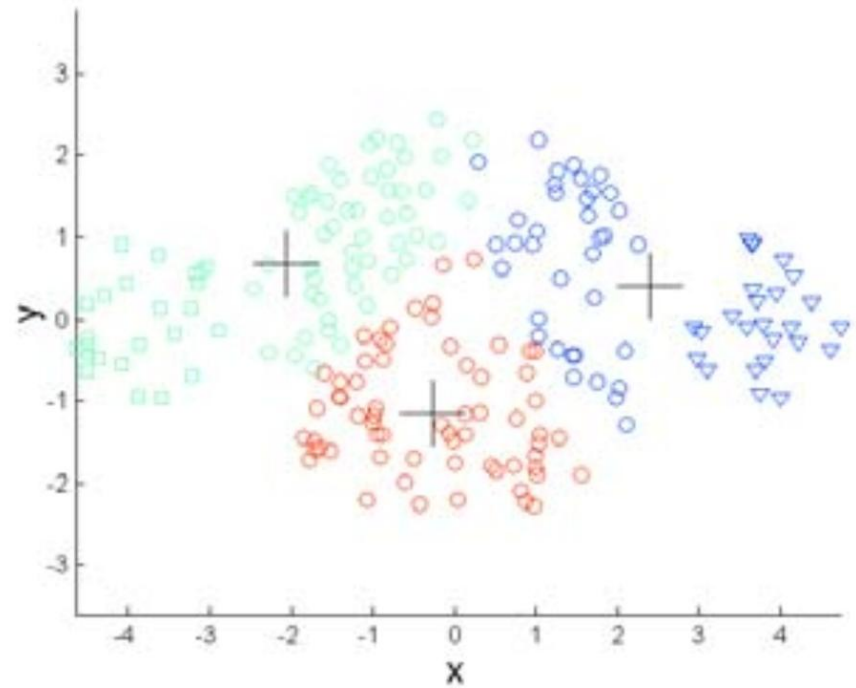
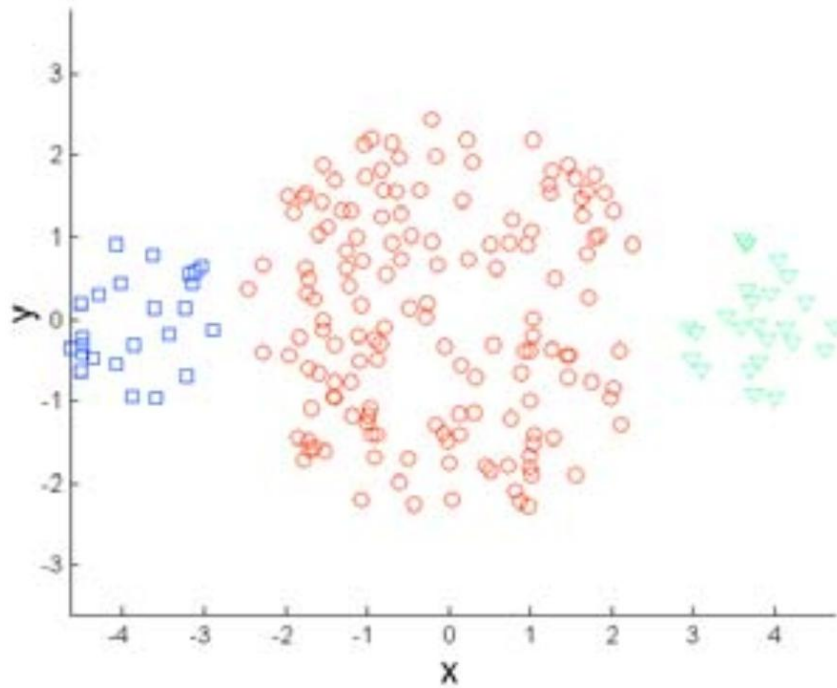
Q: the black box returns “4” as the random number generated. Which point do we choose for the next center (x, y, or z) ?



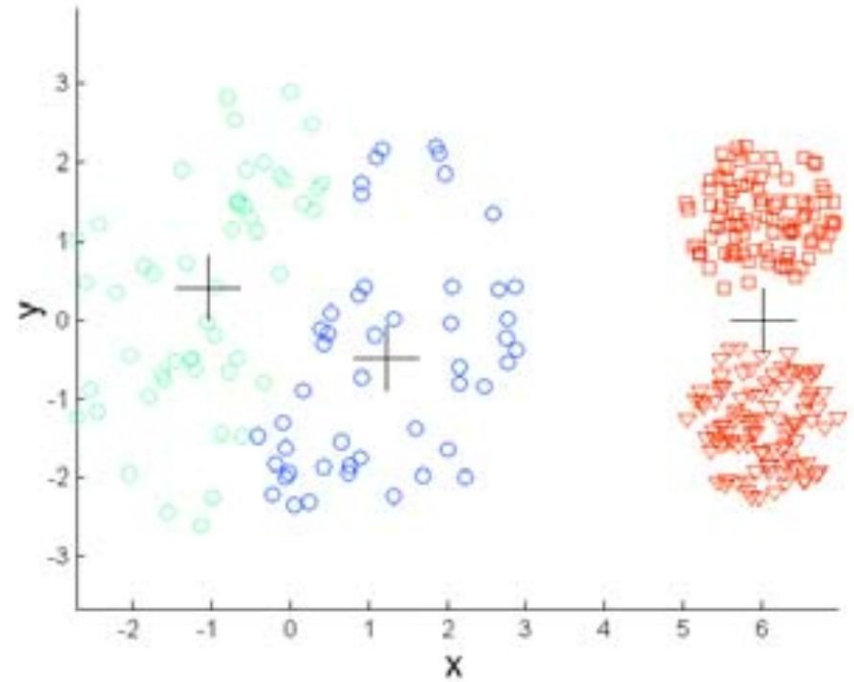
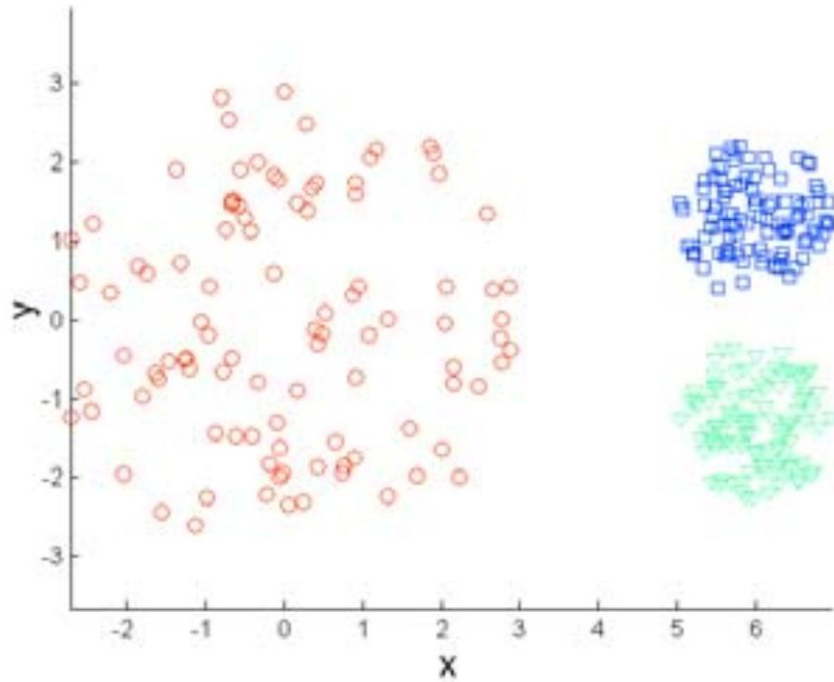
K-means++

What happens if the black box can only generate numbers between 0 and 1?

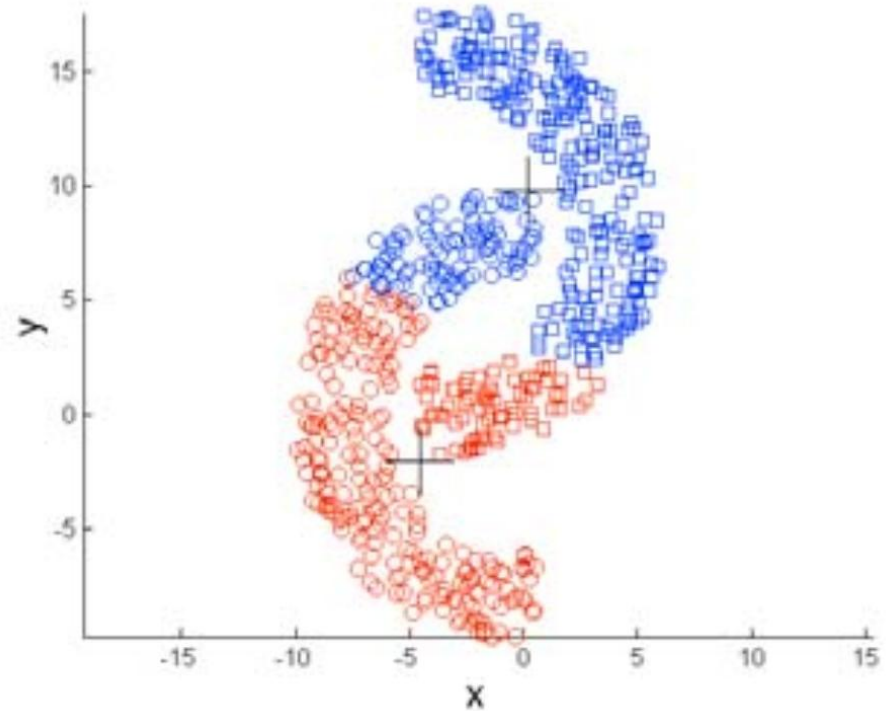
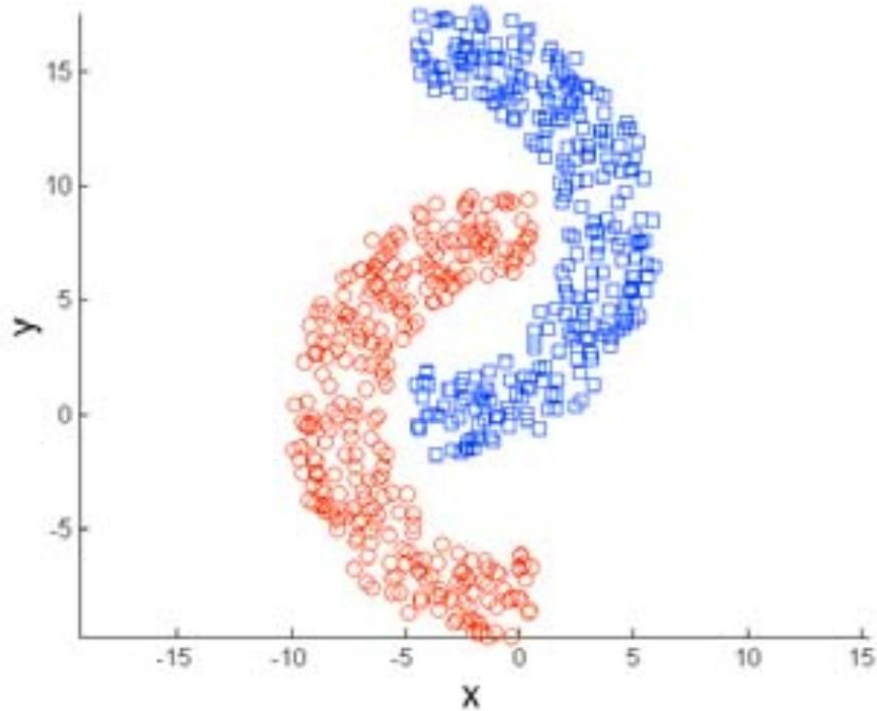
K-means / K-means++ Limitations



K-means / K-means++ Limitations



K-means / K-means++ Limitations



How to choose the right k?

1. Iterate through different values of k (elbow method)
2. Use empirical / domain-specific knowledge
Example: Is there a known approximate distribution of the data? (K-means is good for spherical gaussians)
3. Metric for evaluating a clustering output

Evaluation

Recall our goal: Find a clustering such that

- **Similar** data points are in the **same cluster**
- **Dissimilar** data points are in **different clusters**

Evaluation

Recall our goal: Find a clustering such that

- **Similar** data points are in the **same cluster** ✓
- **Dissimilar** data points are in **different clusters**

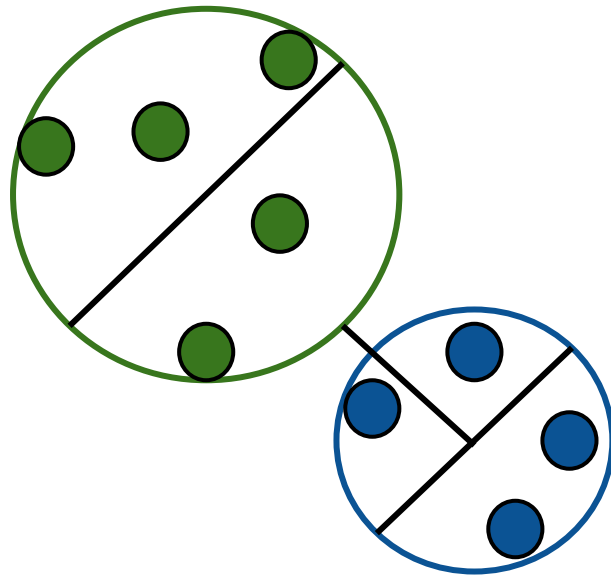
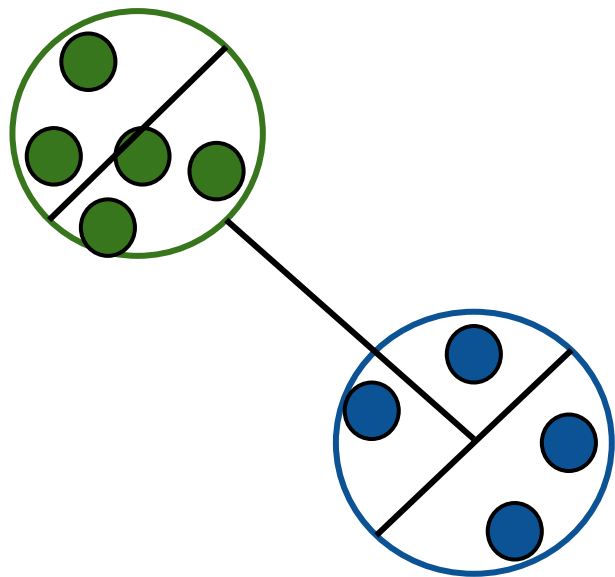
Evaluation

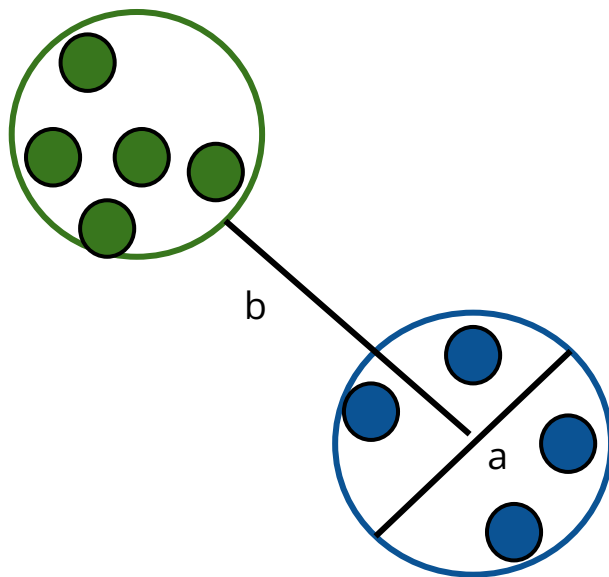
K-means cost function tells us the within-cluster distances between points will be small overall.

But what about the intra-cluster distance? Are the clusters we created far?
How far? Relative to what?

Discuss - 5min

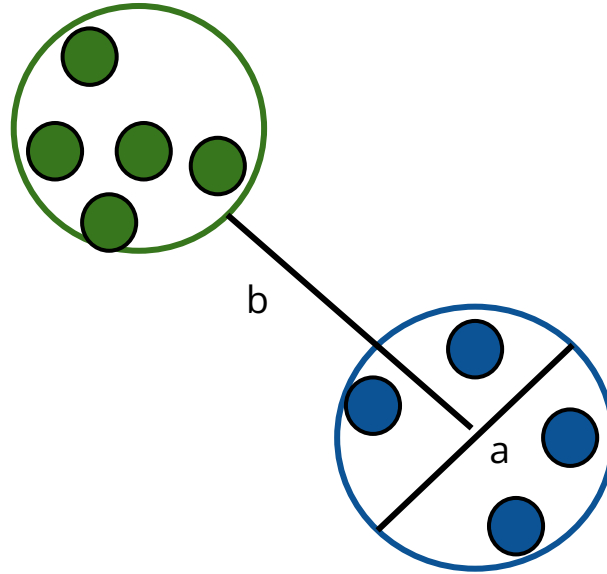
Define a few metrics that you might care about when evaluating a clustering.



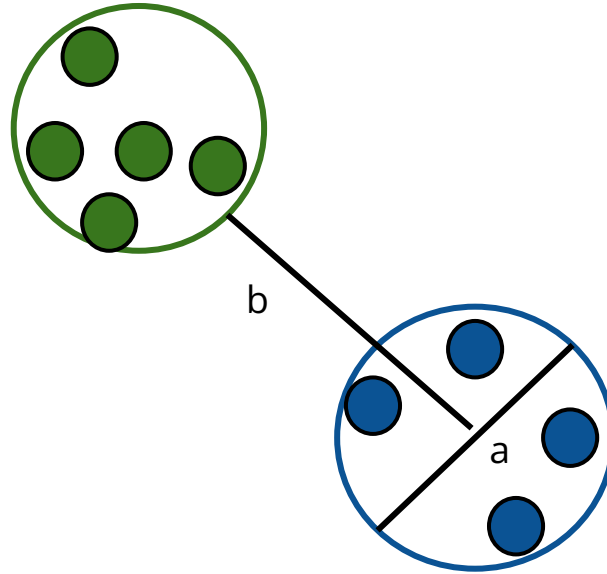


a: average within-cluster distance

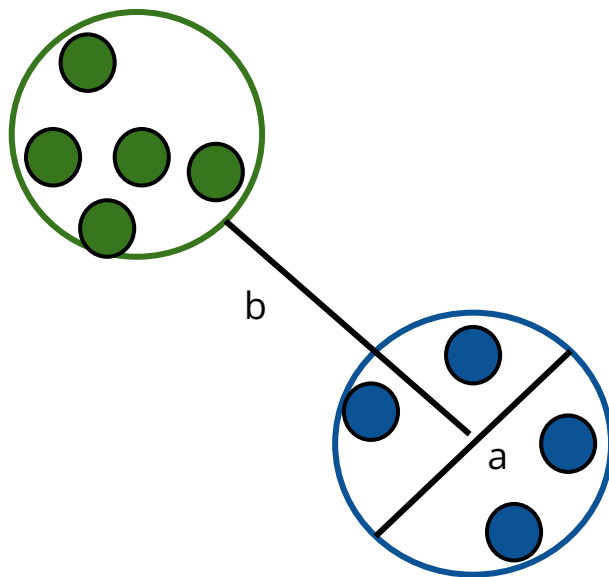
b: average intra-cluster distance



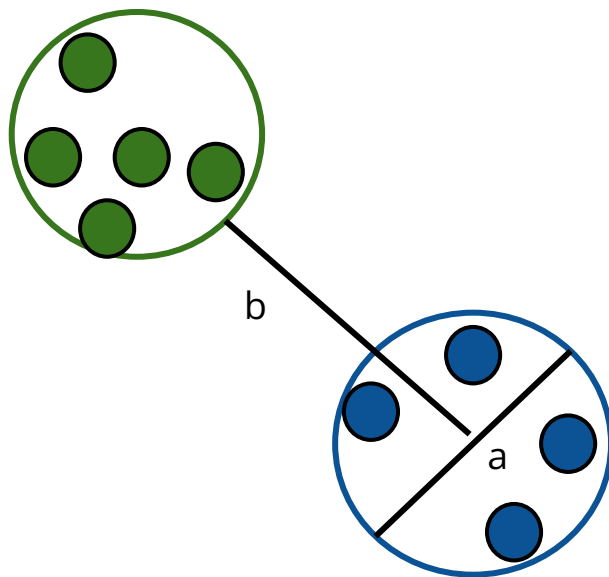
What does it mean for $(b - a)$ to be 0?



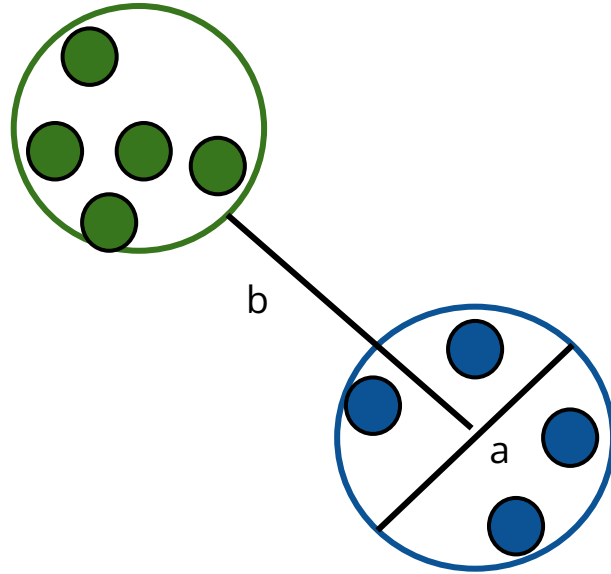
What does it mean for $(b - a)$ to be large?



What does it mean for $(b - a)$ to be negative?



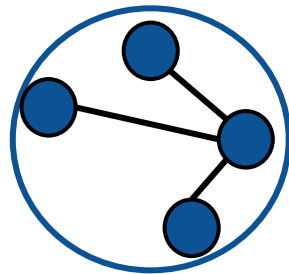
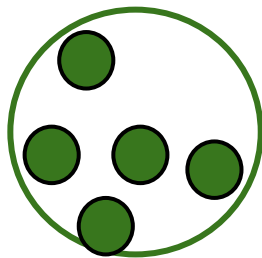
Should we compare $(b - a)$ to some other value, in order to get a sense of how representative that average value is overall?



$$(b - a) / \max(a, b)$$

Silhouette Scores

For each data point i :
 a_i : mean distance from point i to every other point in its cluster

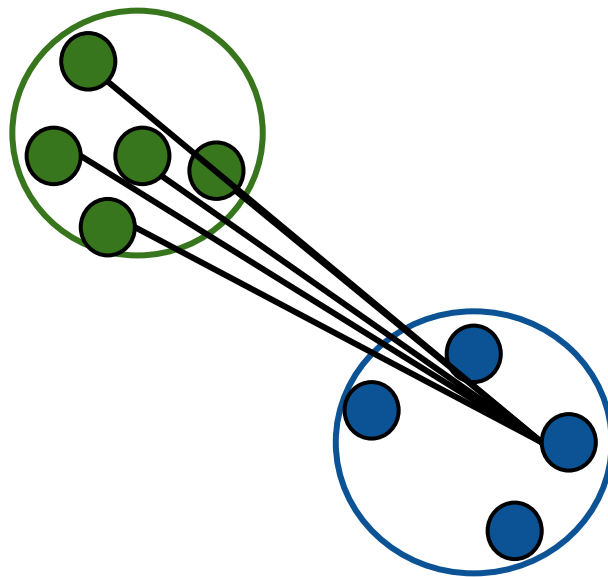


Silhouette Scores

For each data point i :

a_i : mean distance from point i to every other point in its cluster

b_i : smallest mean distance from point i to every point in another cluster



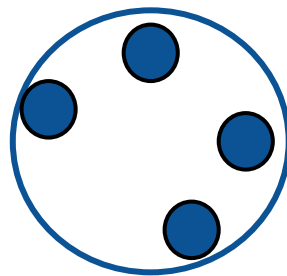
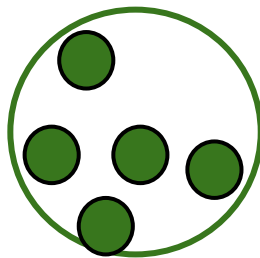
Silhouette Scores

For each data point i :

a_i : mean distance from point i to every other point in its cluster

b_i : smallest mean distance from point i to every point in another cluster

$$s_i = (b_i - a_i) / \max(a_i, b_i)$$



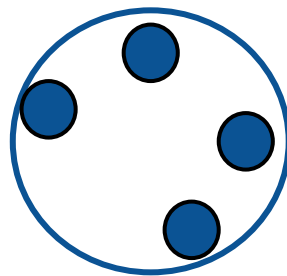
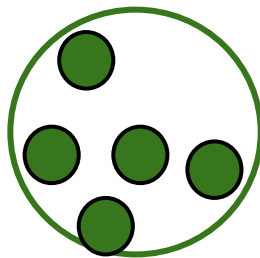
Silhouette Scores

$$s_i = (b_i - a_i) / \max(a_i, b_i)$$

Silhouette score plot

OR

return the mean s_i over the entire dataset as a measure of goodness of fit



Discuss

Q: What is a good silhouette score value?

Q: What is a bad silhouette score value?

Worksheet - answer the last question

K-means Variations

1. K-medians (uses the L_1 norm / manhattan distance)
2. K-medoids (any distance function + the centers must be in the dataset)
3. Weighted K-means (each point has a different weight when computing the mean)