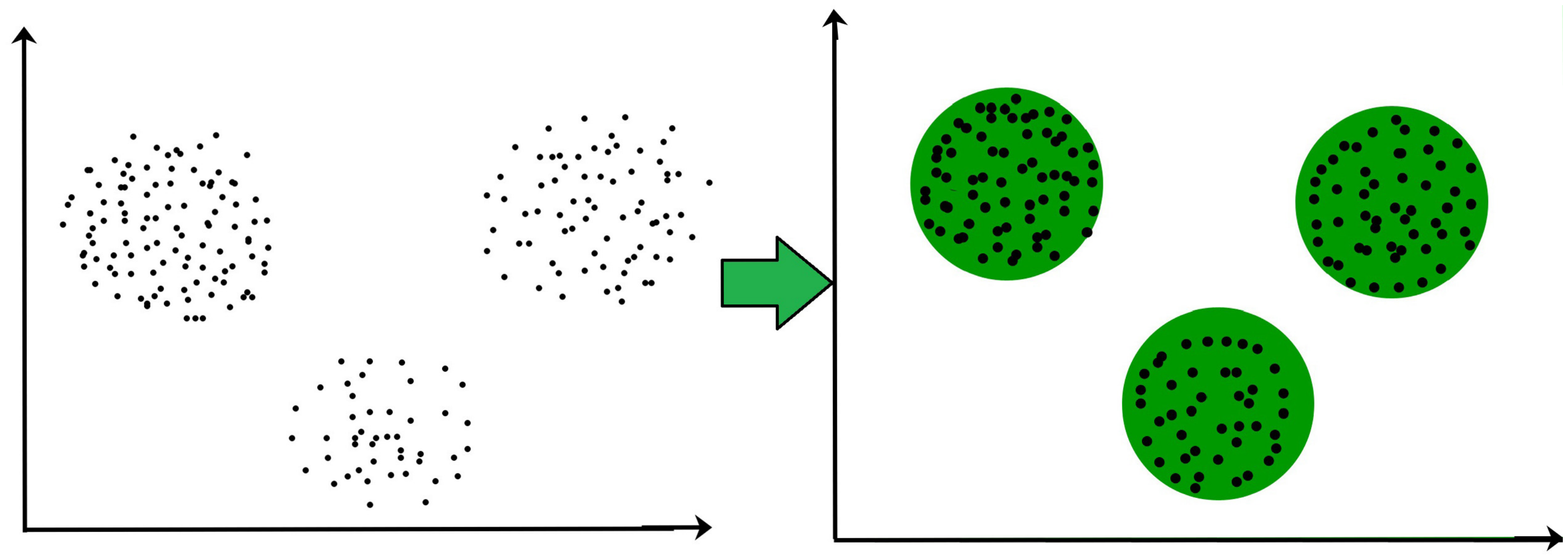
Artificial Intelligence

Taro Sekiyama

National Institute of Informatics (NII)
sekiyama@nii.ac.jp

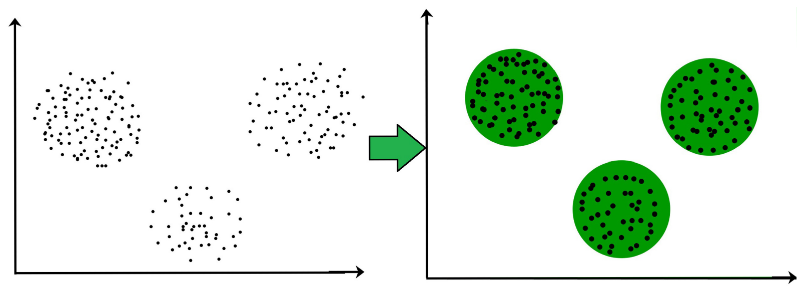
Unsupervised learning

- Goal: learning patterns of datasets ***without*** knowing correct answers



https://www.geeksforgeeks.org/clustering-in-machine-learning/

Task



Cluster analysis

- Grouping similar data points
- Applications include:
  - Marketing
    - Helpful for advertisement to identify customer groups having different preferences
  - Medicine
    - Useful to find diseases from symptoms

Task

Feature transformation/extraction

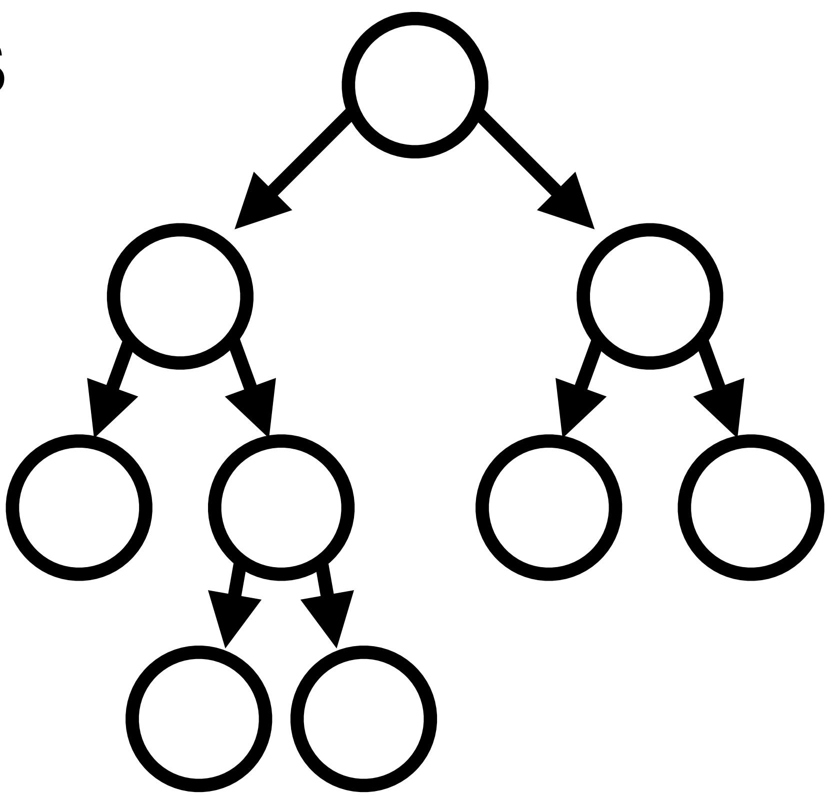
- Finding (transformation to) the most informative features of data points
- Applications include:
  - Understanding and visualization of data
  - Dimensionality reduction
    - Reducing the number of features
    - Contributing to improvement of accuracy, speed-up of training, and efficient memory usage

Agenda

- Cluster analysis
  - K-means
  - Hierarchical clustering
- Feature transformation
  - Principal component analysis (PCA)

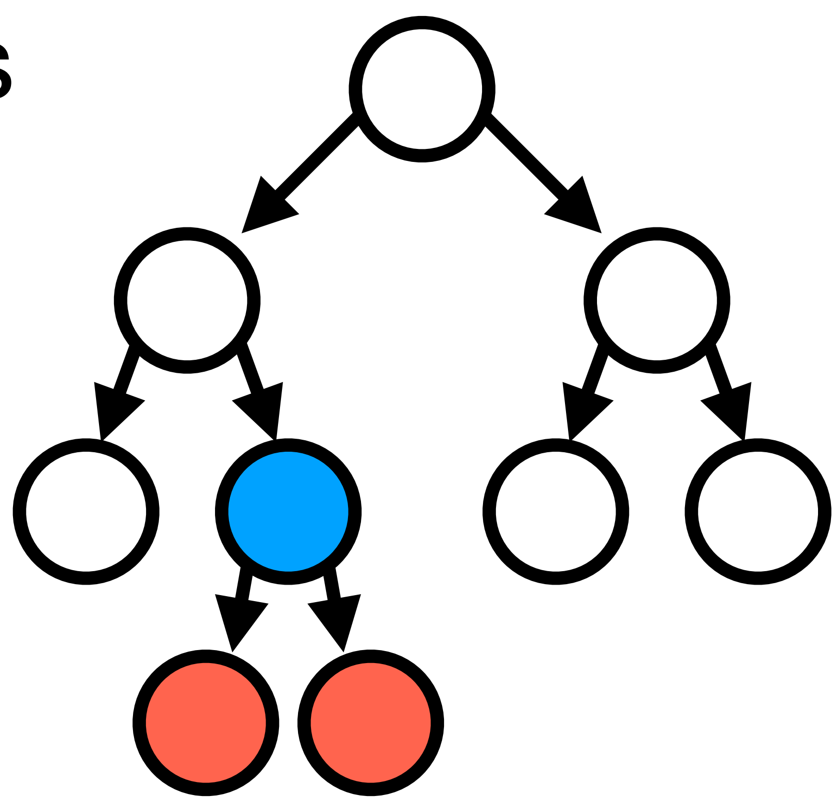
Hierarchical clustering

- Constructing dendrograms (tree diagrams)
- Input: $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^n$
- Output: a dendrogram where:
  - Leaf nodes are data points
  - Internal nodes are clusters containing data points below it



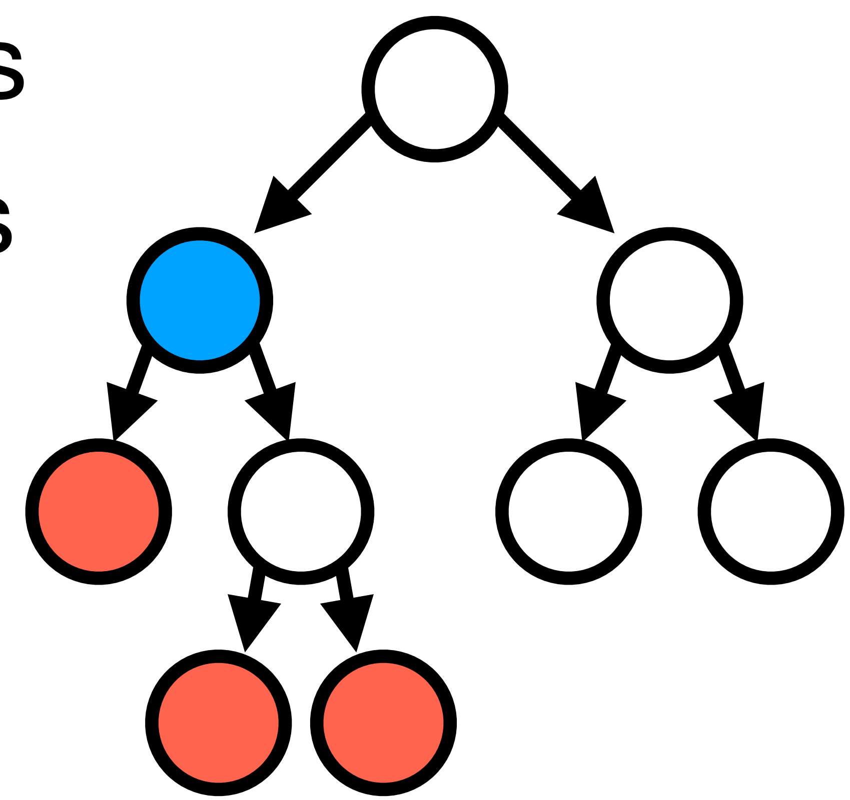
Hierarchical clustering

- Constructing dendrograms (tree diagrams)
- Input: $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^n$
- Output: a dendrogram where:
  - Leaf nodes are data points
  - Internal nodes are clusters containing data points below it



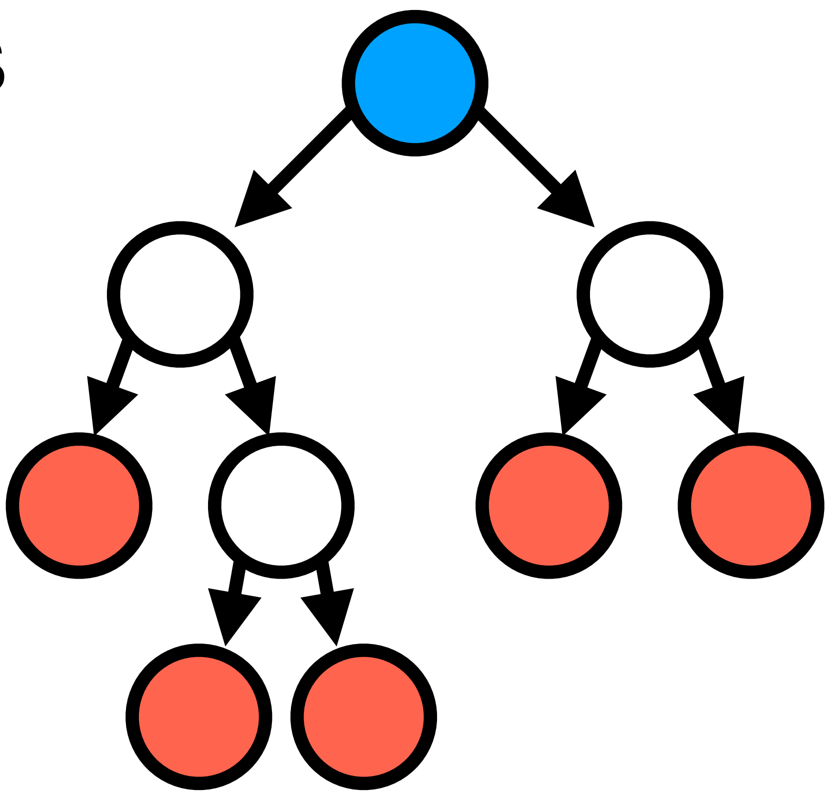
Hierarchical clustering

- Constructing dendrograms (tree diagrams)
- Input: $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^n$
- Output: a dendrogram where:
  - Leaf nodes are data points
  - Internal nodes are clusters containing data points below it



Hierarchical clustering

- Constructing dendrograms (tree diagrams)
- Input: $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^n$
- Output: a dendrogram where:
  - Leaf nodes are data points
  - Internal nodes are clusters containing data points below it

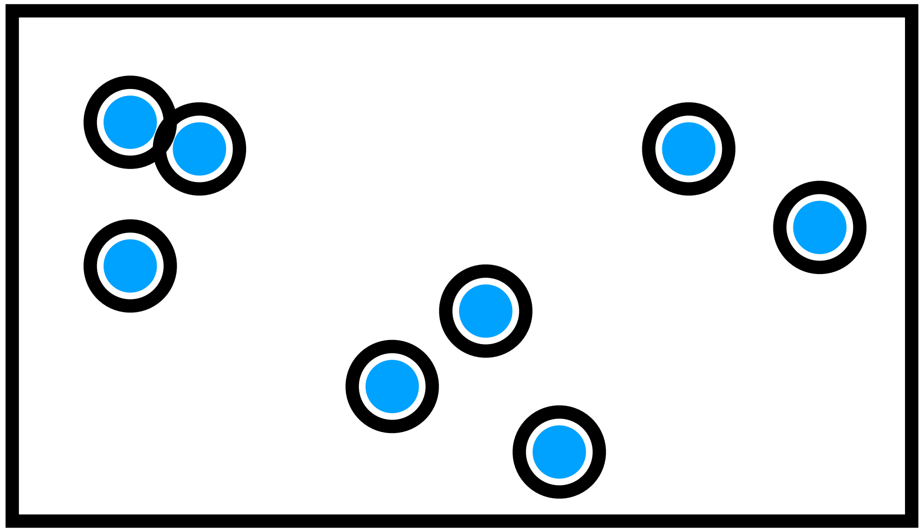


Algorithms

- Agglomerative (bottom-up) clustering
- Divisive (top-down) clustering
  - Not presented in this lecture

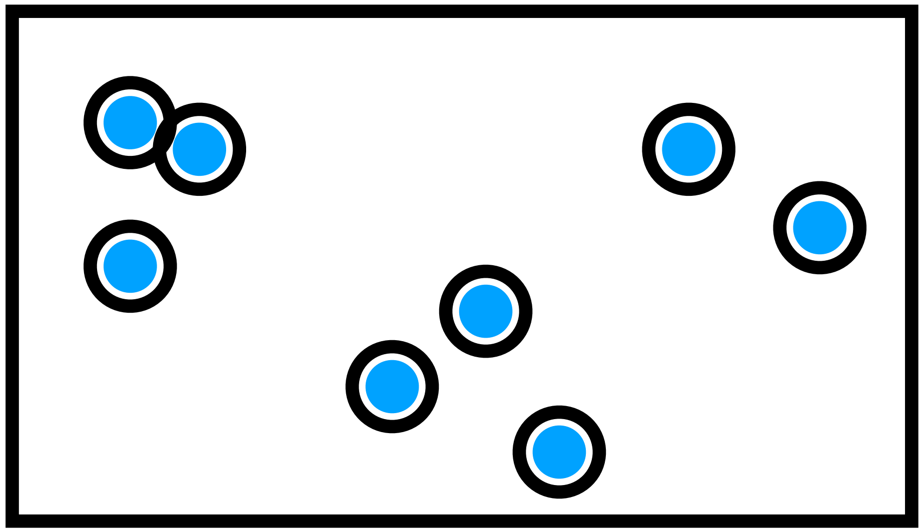
Agglomerative clustering

- Let each data point be a single cluster



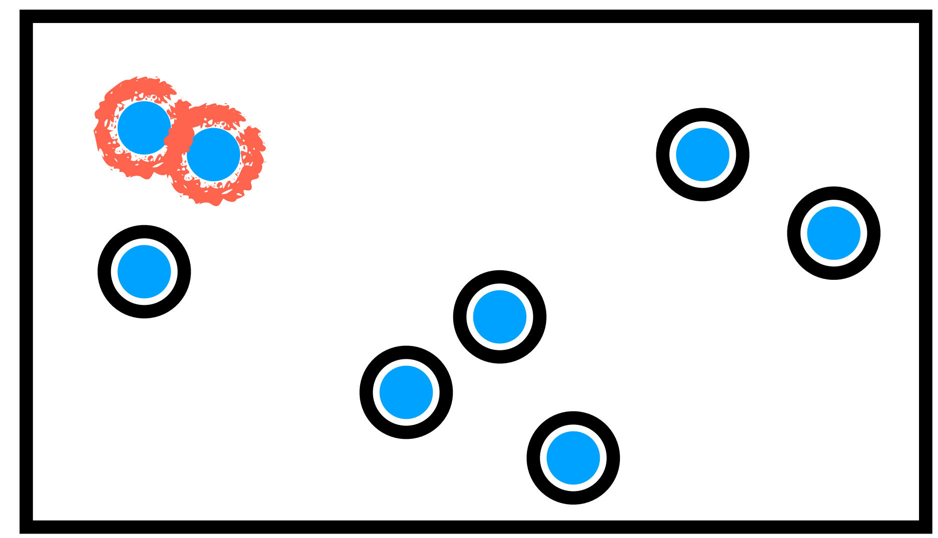
Agglomerative clustering

- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
  1. Find two clusters nearest to each other



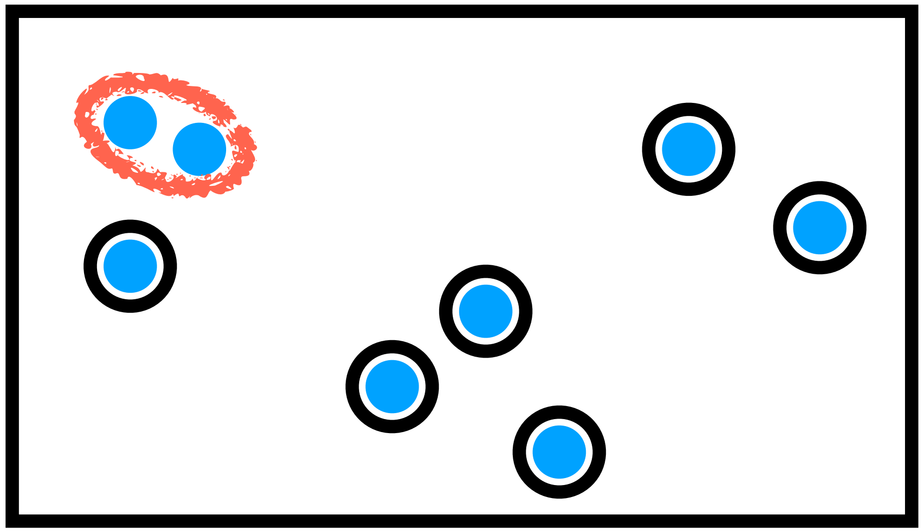
Agglomerative clustering

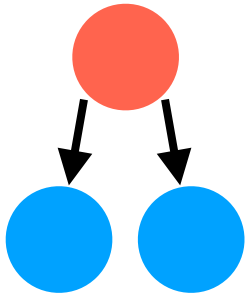
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
  1. Find two clusters nearest to each other
  2. Merge them and make a new cluster



Agglomerative clustering

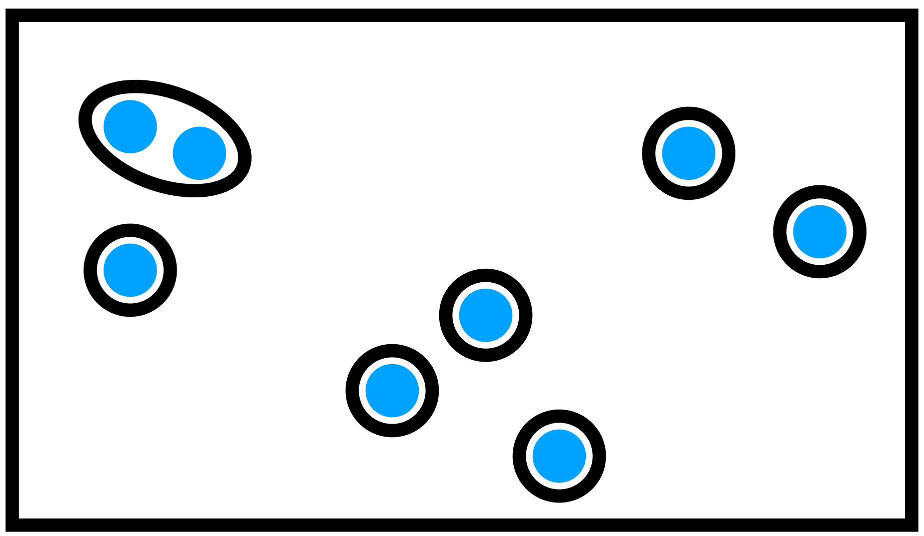
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
    1. Find two clusters nearest to each other
    2. Merge them and make a new cluster

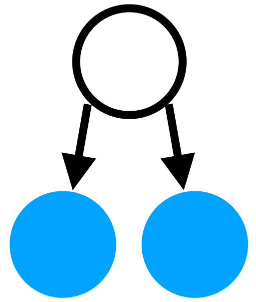




Agglomerative clustering

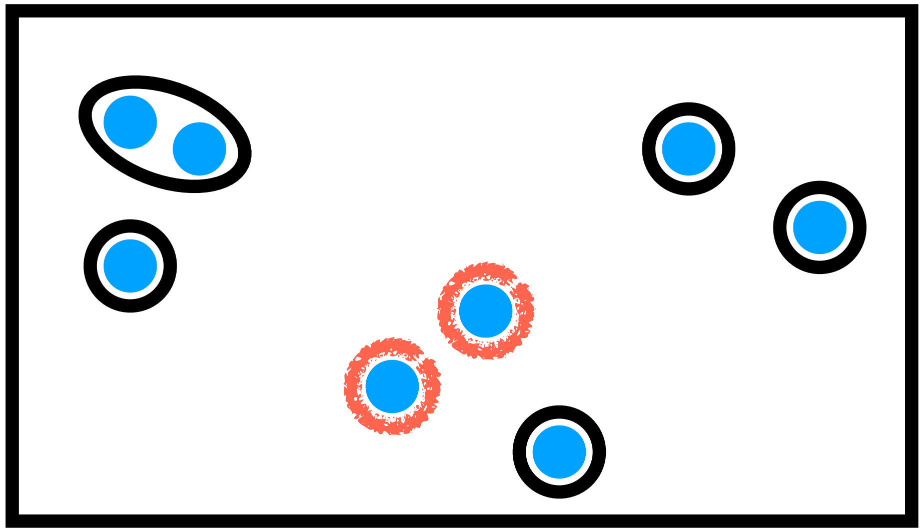
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
    1. Find two clusters nearest to each other
    2. Merge them and make a new cluster

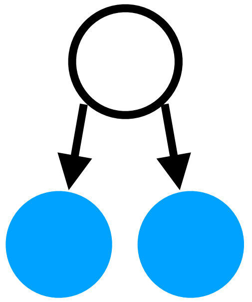




Agglomerative clustering

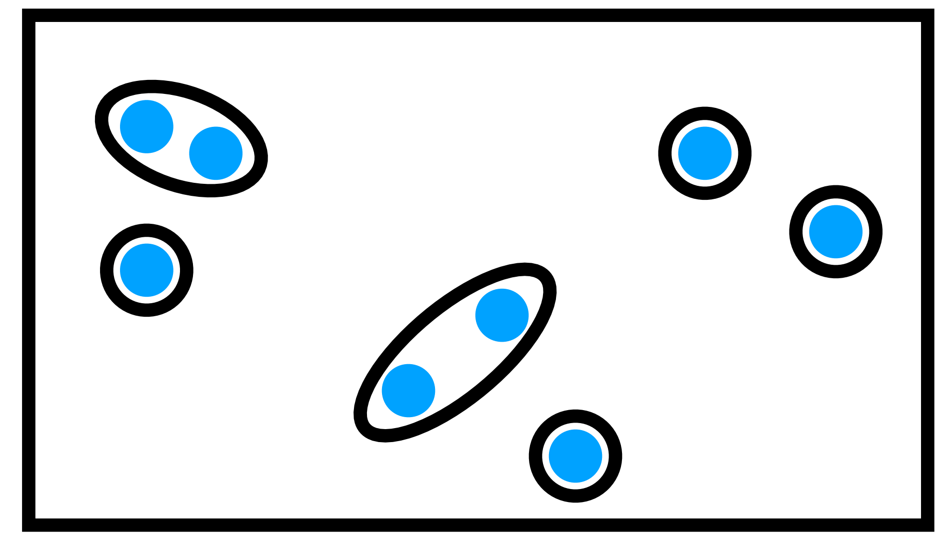
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
  1. Find two clusters nearest to each other
  2. Merge them and make a new cluster

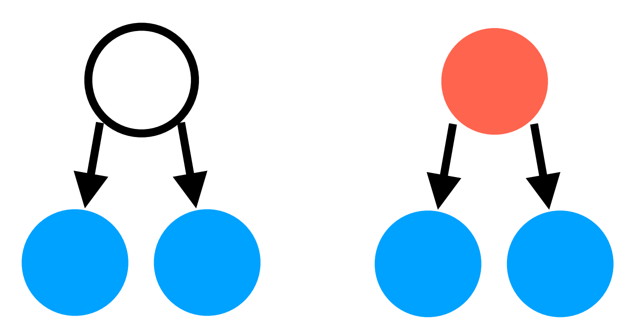




Agglomerative clustering

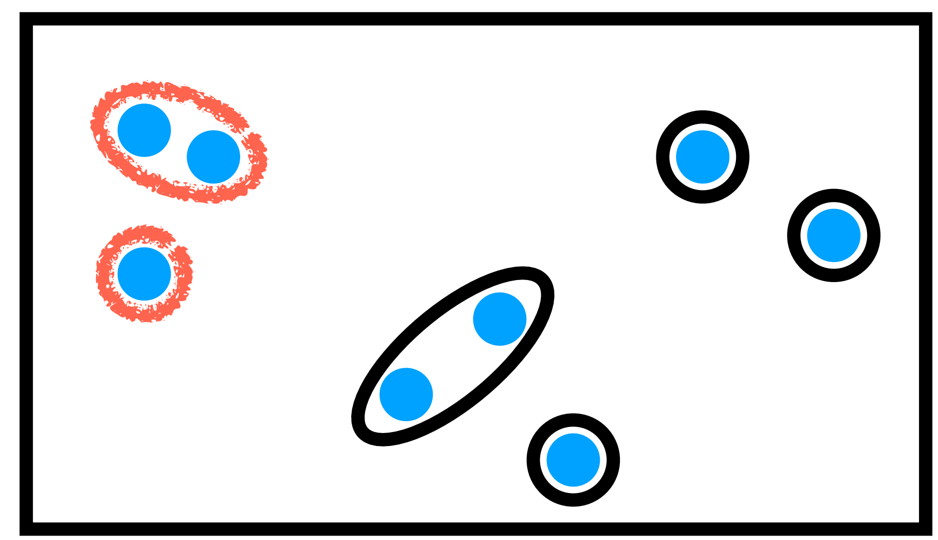
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
  1. Find two clusters nearest to each other
  2. Merge them and make a new cluster

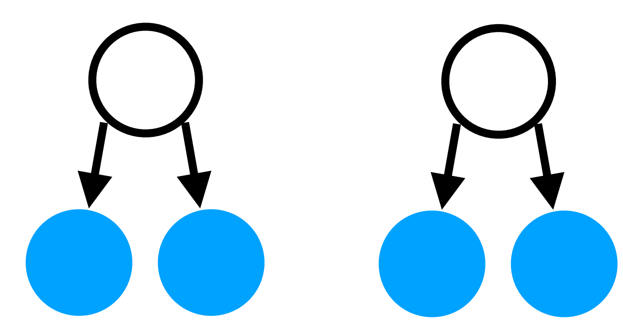




Agglomerative clustering

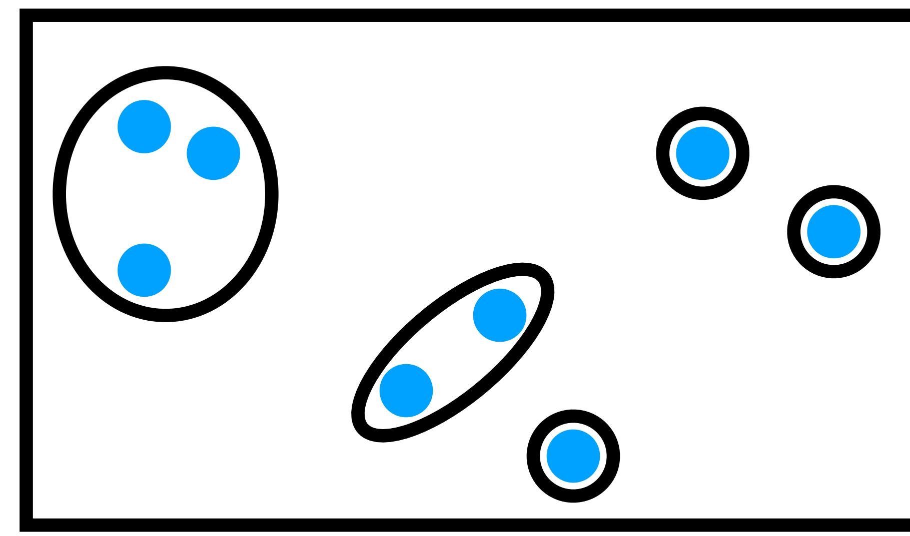
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
    1. Find two clusters nearest to each other
    2. Merge them and make a new cluster

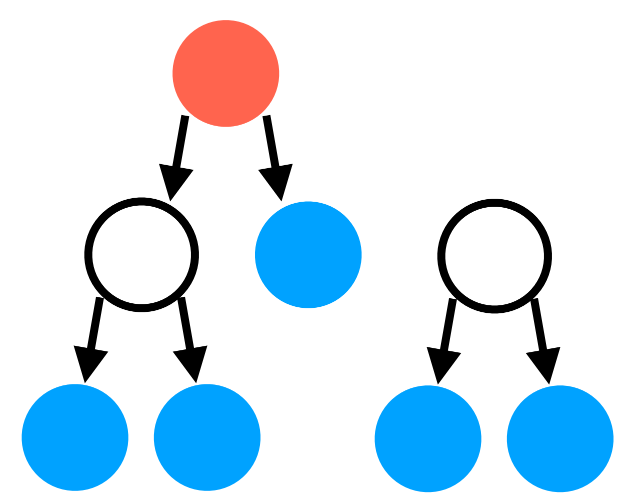




Agglomerative clustering

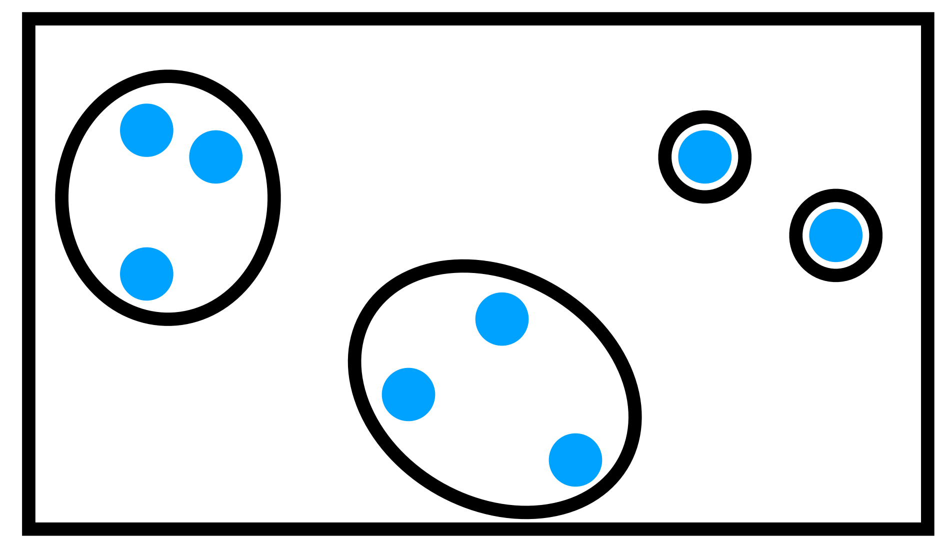
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
  1. Find two clusters nearest to each other
  2. Merge them and make a new cluster

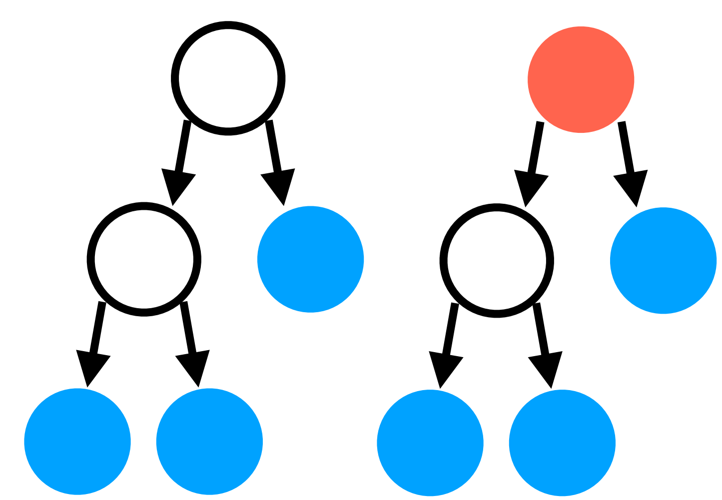




Agglomerative clustering

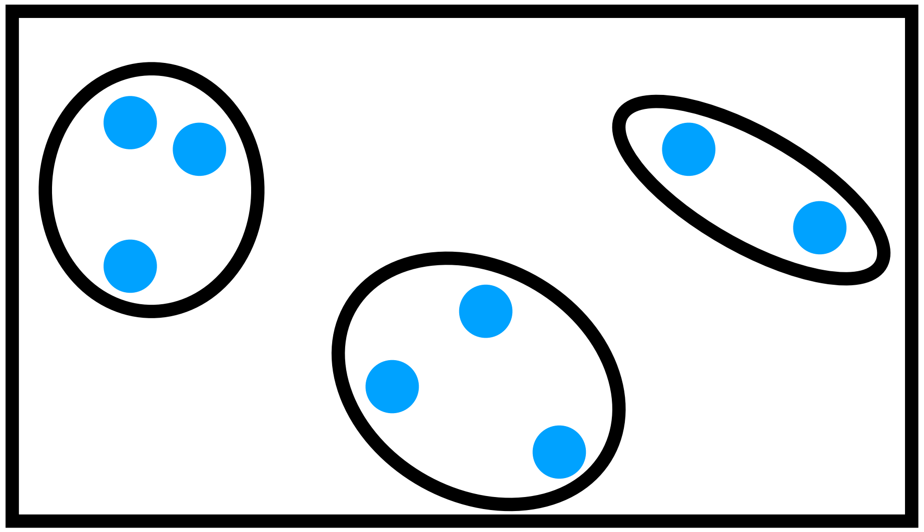
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
    1. Find two clusters nearest to each other
    2. Merge them and make a new cluster

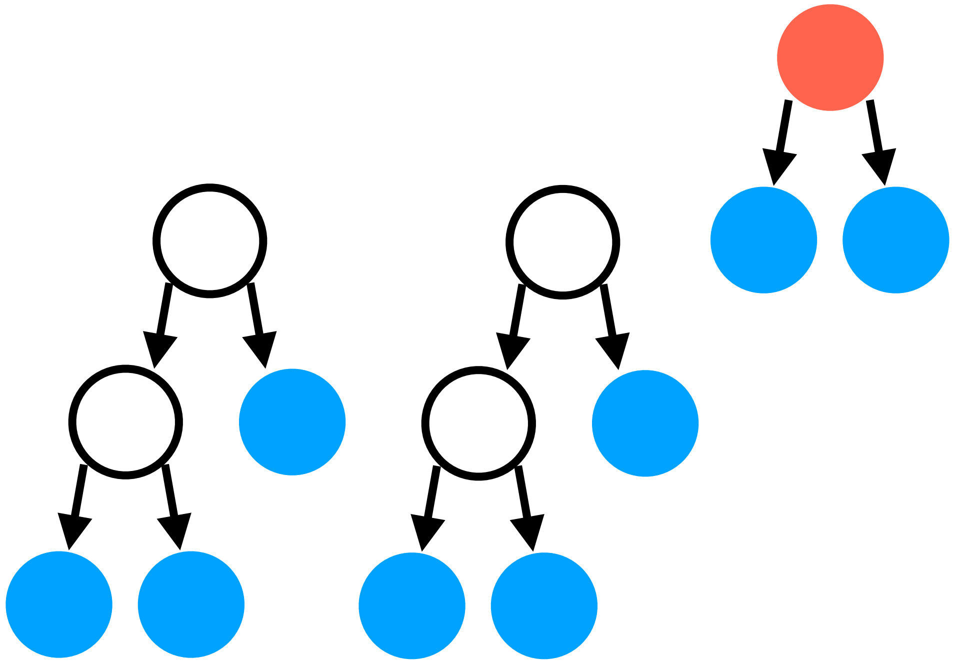




Agglomerative clustering

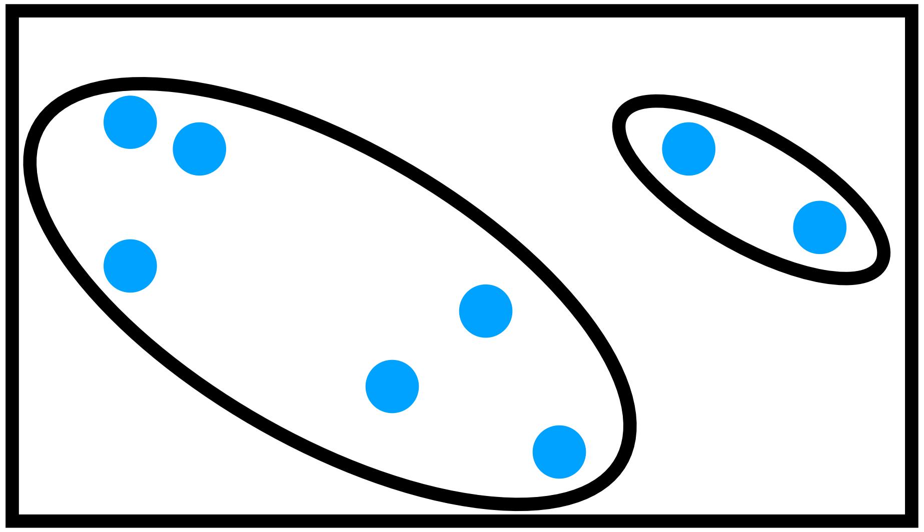
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
    1. Find two clusters nearest to each other
    2. Merge them and make a new cluster

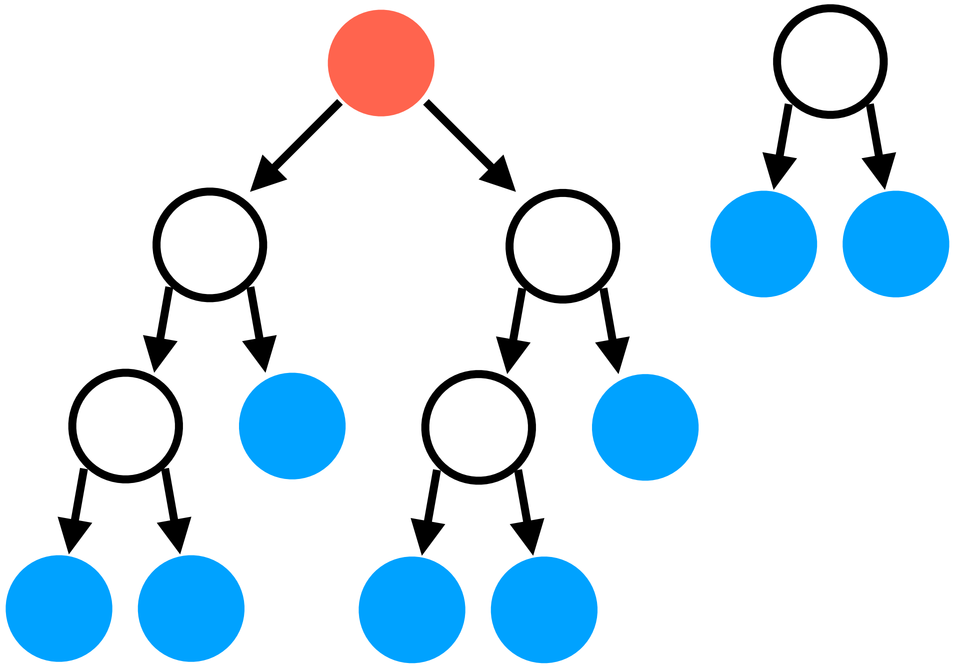




Agglomerative clustering

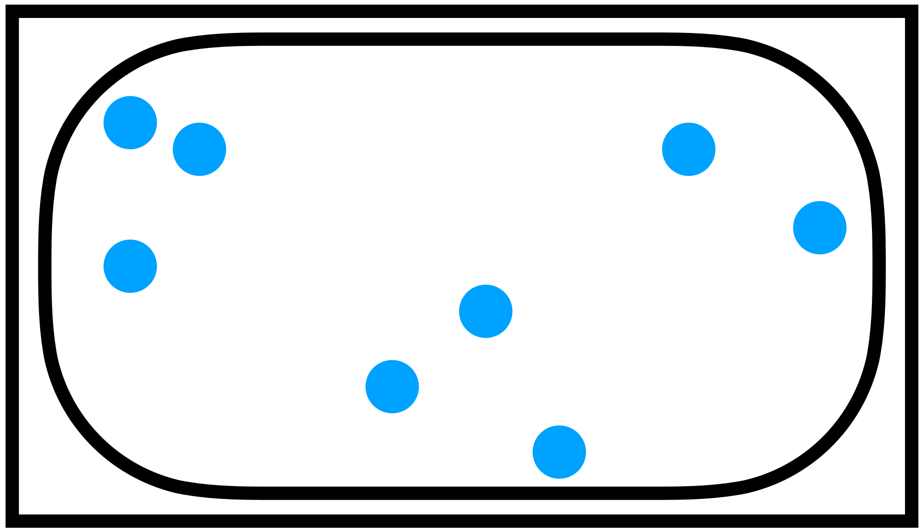
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
    1. Find two clusters nearest to each other
    2. Merge them and make a new cluster

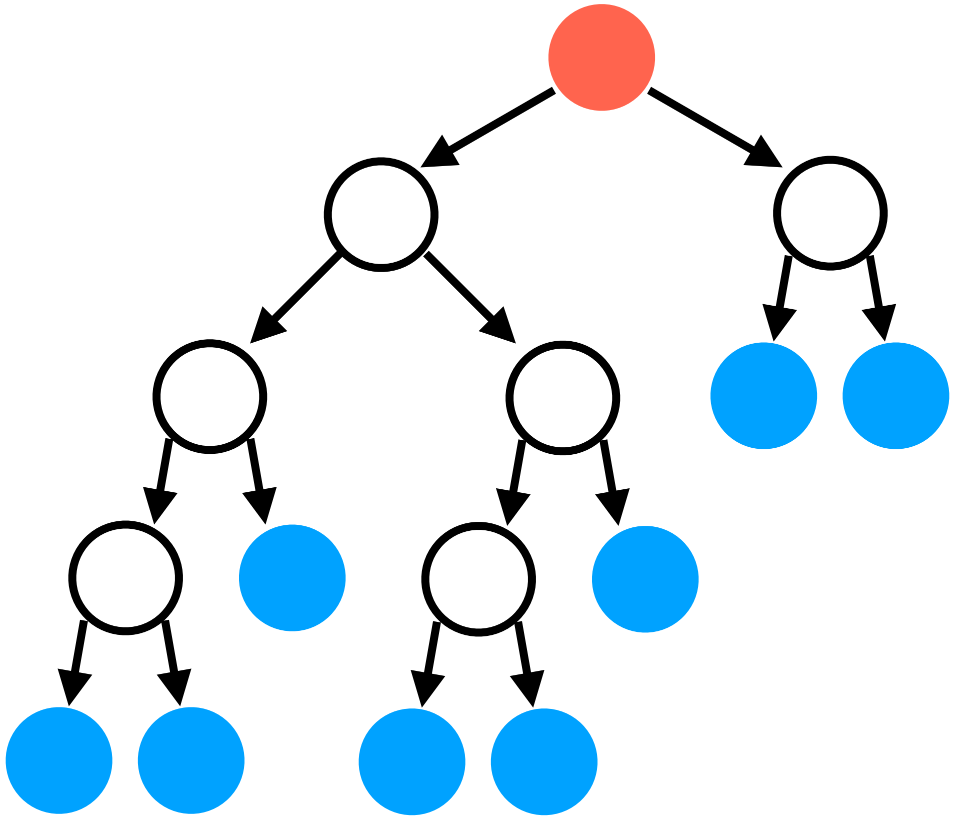




Agglomerative clustering

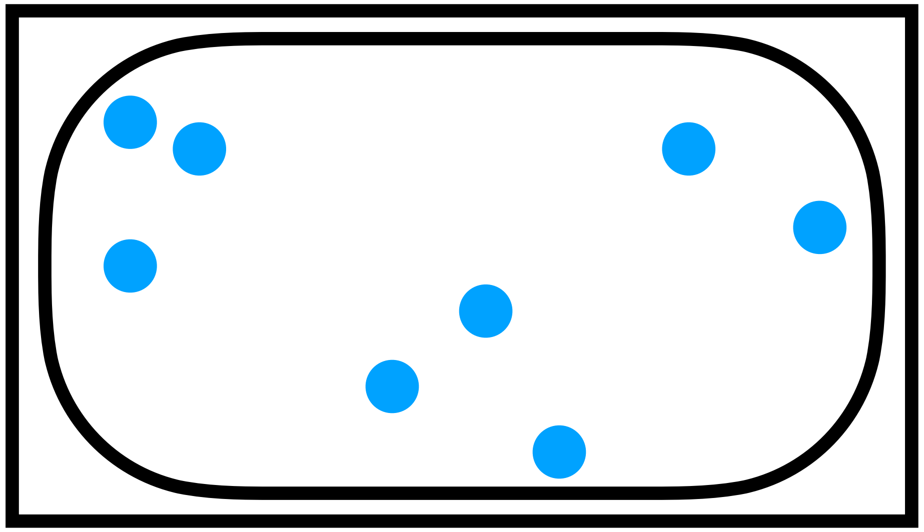
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
    1. Find two clusters nearest to each other
    2. Merge them and make a new cluster

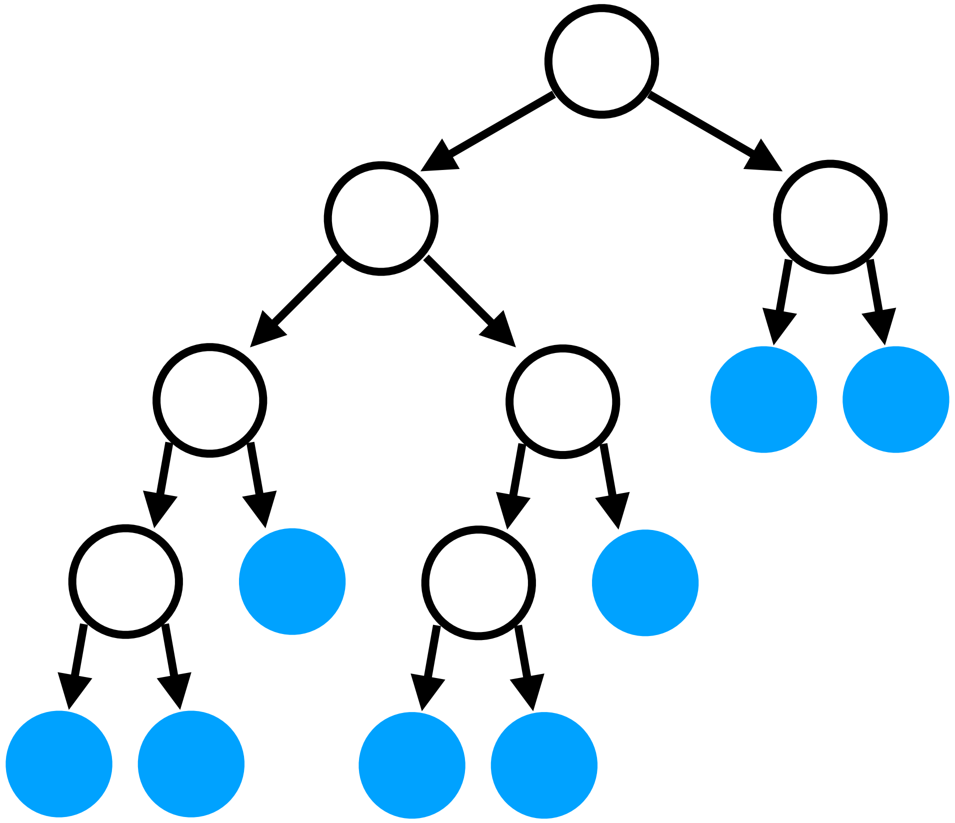




Agglomerative clustering

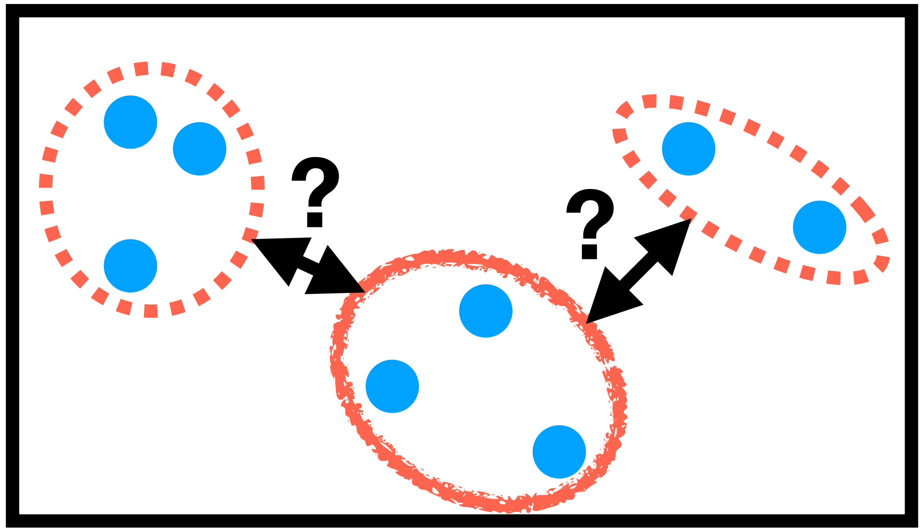
- Let each data point be a single cluster
- Repeat two steps until we get the cluster containing all the points
  1. Find two clusters nearest to each other
  2. Merge them and make a new cluster





Importance of distance

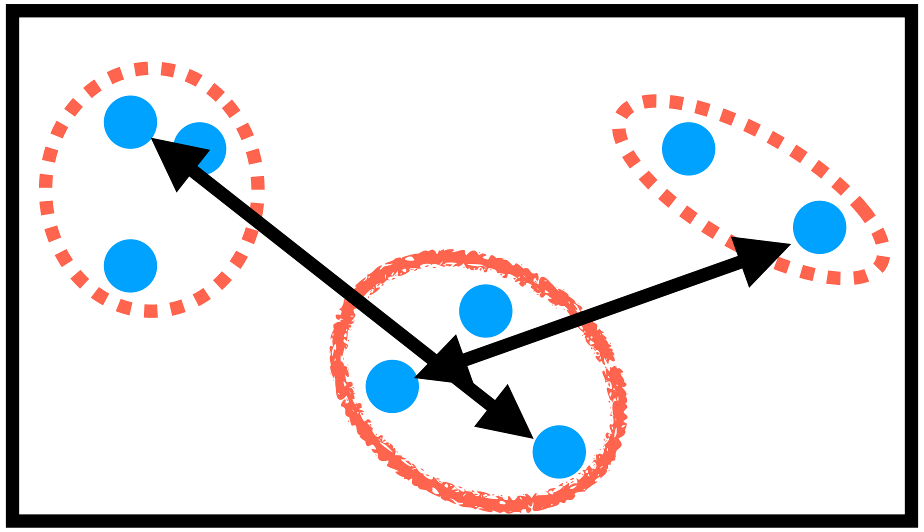
- Distance determines which clusters are merged
- May be influential on the final clustering results



Cluster distance (linkage criterion)

- Complete linkage

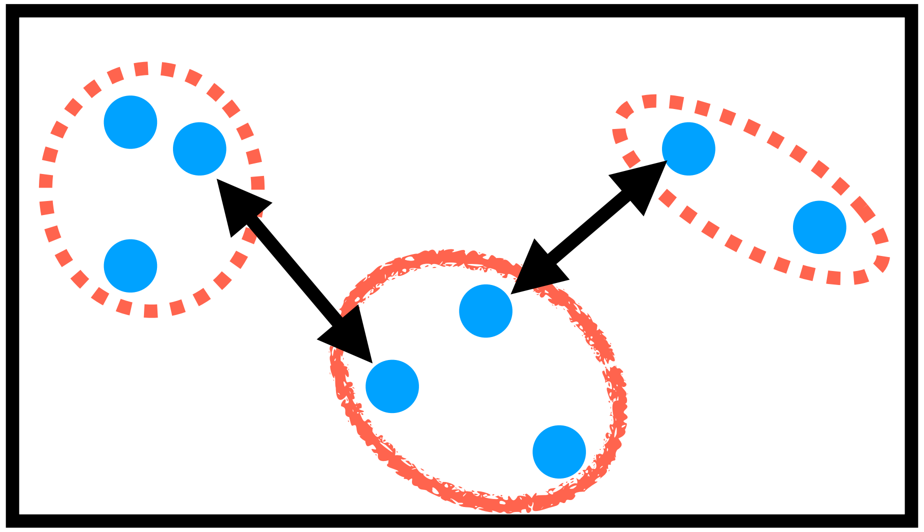
$$d(c_1, c_2) = \max_{x_1 \in c_1, x_2 \in c_2} \|c_1 - c_2\|$$



Cluster distance (linkage criterion)

- Single linkage

$$d(c_1, c_2) = \min_{x_1 \in c_1, x_2 \in c_2} \|c_1 - c_2\|$$



Distance

- Distance between data points $\|x_1 - x_2\|$
  $(x_1 = (x_{11}, \ldots, x_{1m}), x_2 = (x_{21}, \ldots, x_{2m}) \in \mathbb{R}^m)$
  - Euclid distance

$$\|x_1 - x_2\| = \sqrt{(x_{11} - x_{21})^2 + \cdots + (x_{1m} - x_{2m})^2}$$

  - Manhattan distance

$$\|x_1 - x_2\| = \sum |x_{1i} - x_{2i}|$$

  - etc.

Advantages

- Easy to implement (for the agglomerative version)
- Possible to make clusters flexibly
  - Specifying the number of clusters
  - Specifying the number of data points in a single cluster
  - etc.

Disadvantages

- Not scaling to huge datasets
  - Time complexity: $O(n^2)$
- Difficult to decide how to make clustering especially for huge datasets

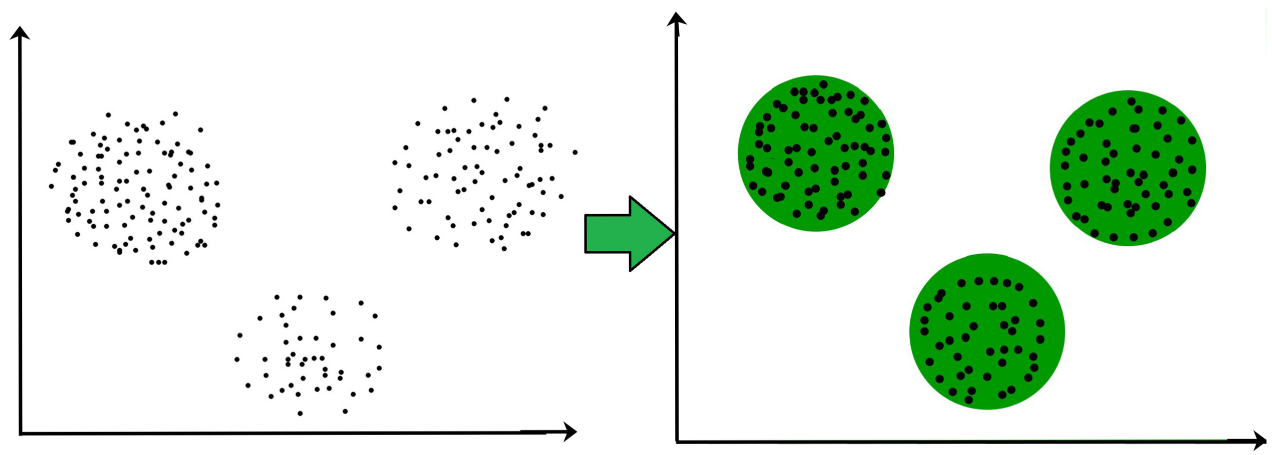
Agenda

- **Cluster analysis**
  - Hierarchical clustering
  - **K-means**
- Feature transformation
  - Principal component analysis (PCA)

K-means

- Finding K clusters from given data points
- Input: $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^n$
- Output: $c_1, \ldots, c_K \subseteq X$ s.t.
  1. $c_i \cap c_j = \emptyset$ for any i, j $(i \neq j)$

K = 3



K-means

- Finding K clusters from given data points
- Input: $X = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^n$
- Output: $c_1, \ldots, c_K \subseteq X$ s.t.
  1. $c_i \cap c_j = \emptyset$ for any i, j ($i \neq j$)
  2. $\sum_i \sum_{x_i \in c_i} \|x_i - m_i\|^2$ is minimized

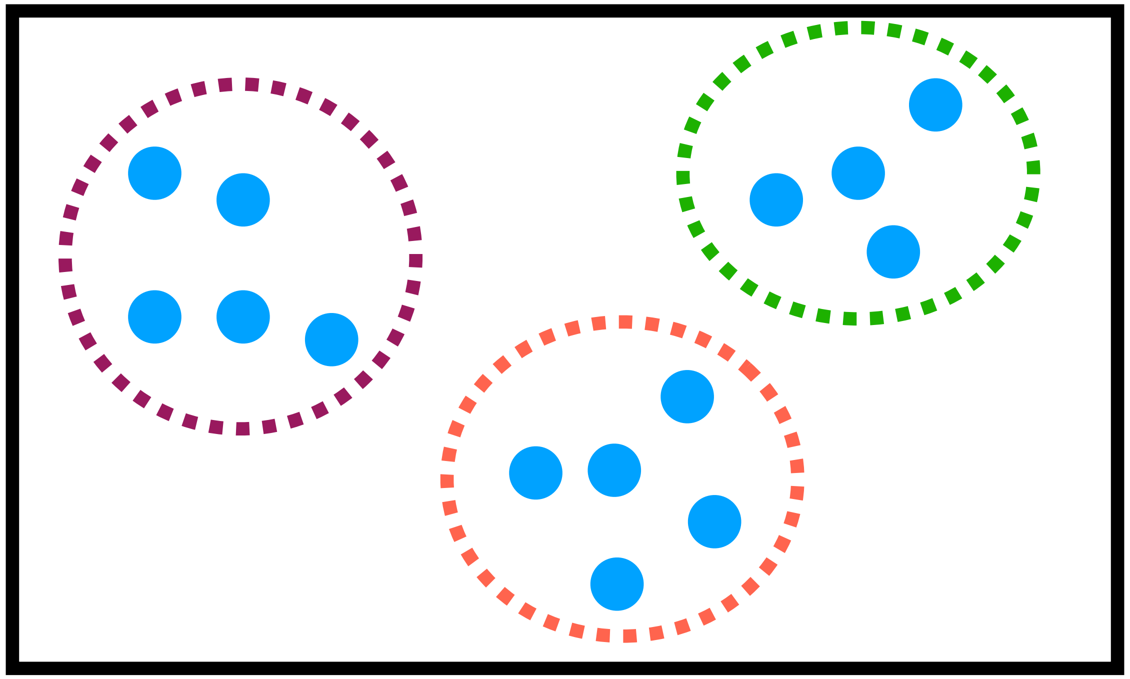
     (m_i is the mean of c_i)

Problem

- Solving the problem is NP-hard
- We need a heuristic
  - Ex: Lloyd’s algorithm

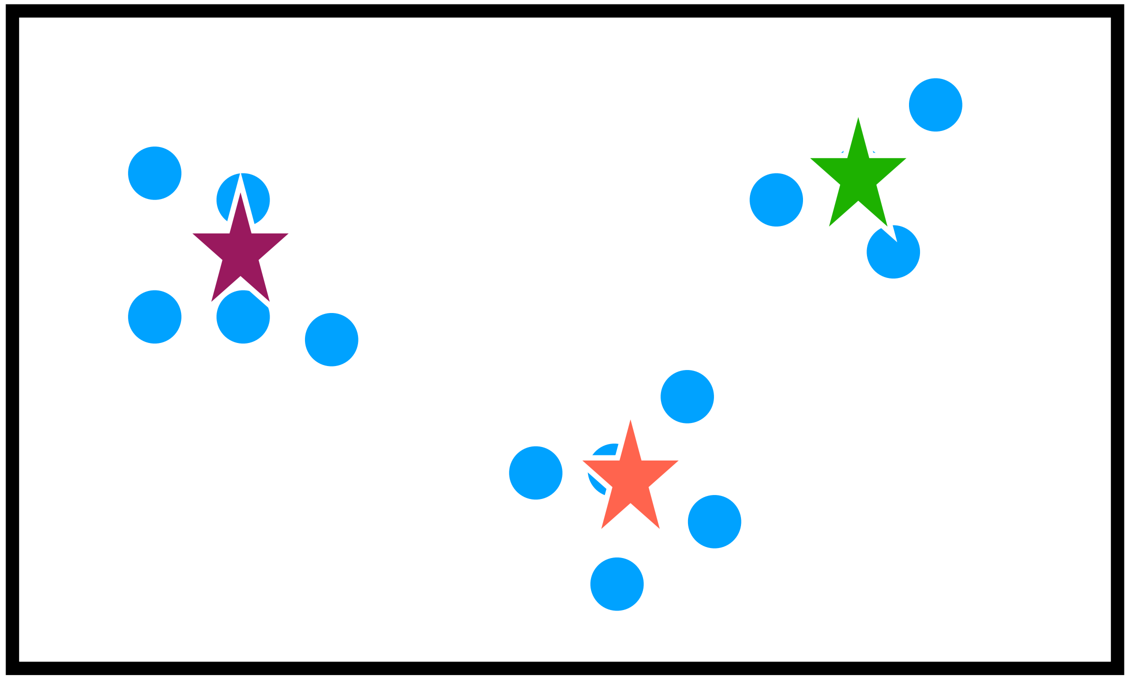
Idea of Lloyd's algorithm

- Estimating center locations of each clusters



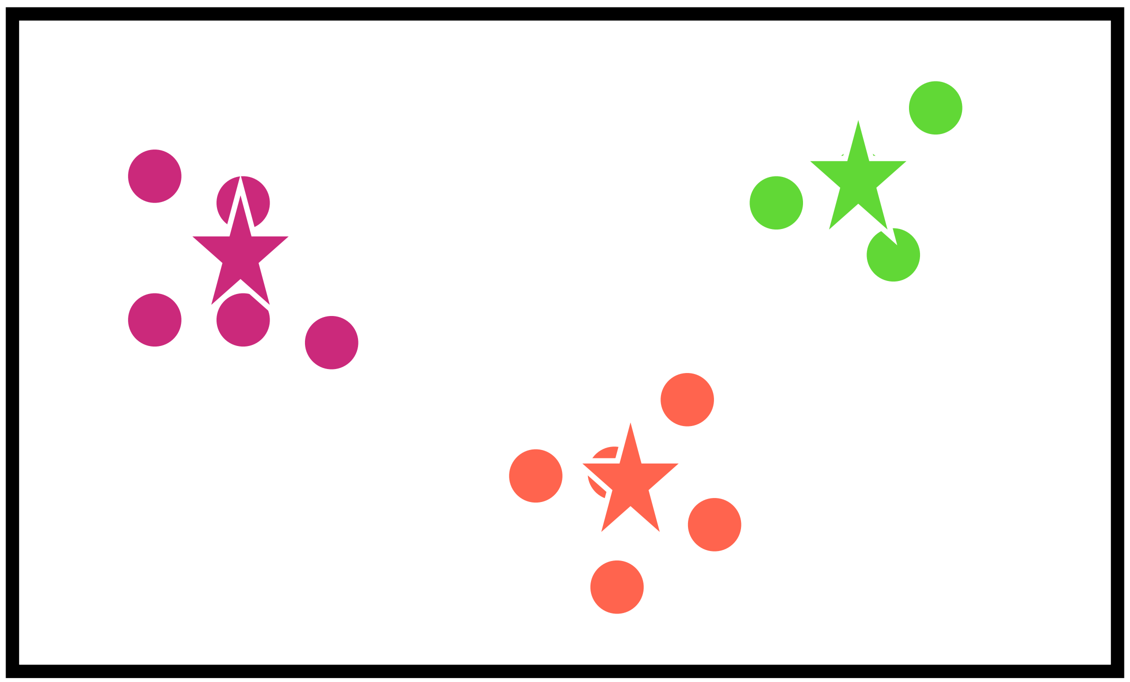
Idea of Lloyd's algorithm

- Estimating center locations of each clusters



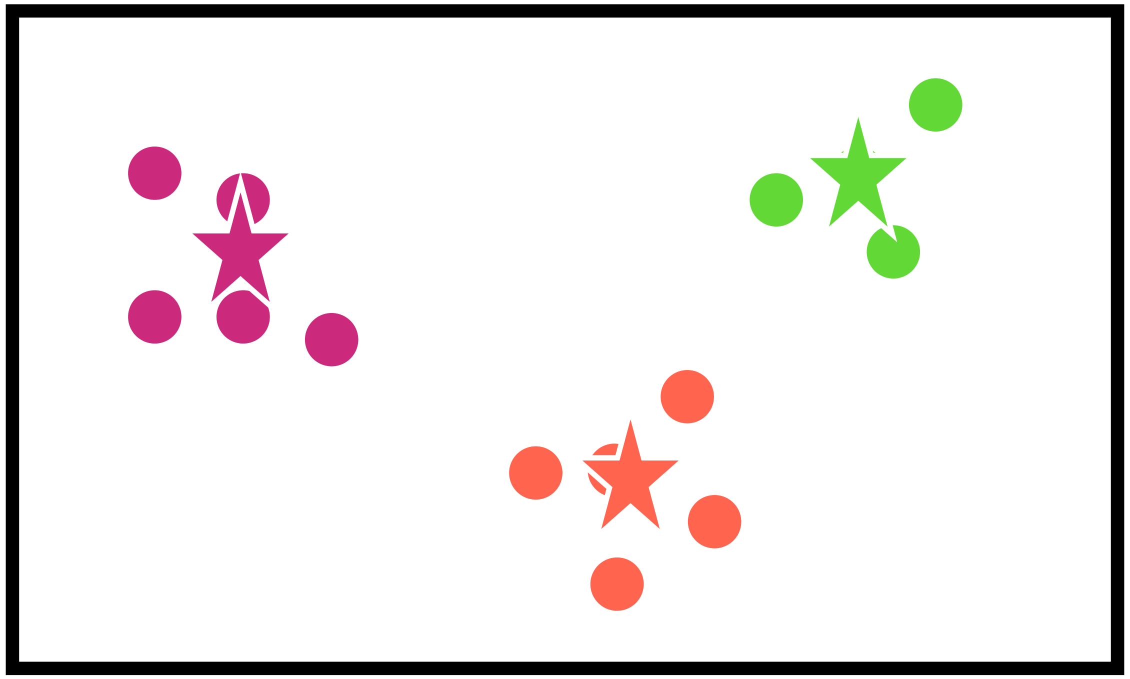
Idea of Lloyd’s algorithm

- Estimating center locations of each clusters



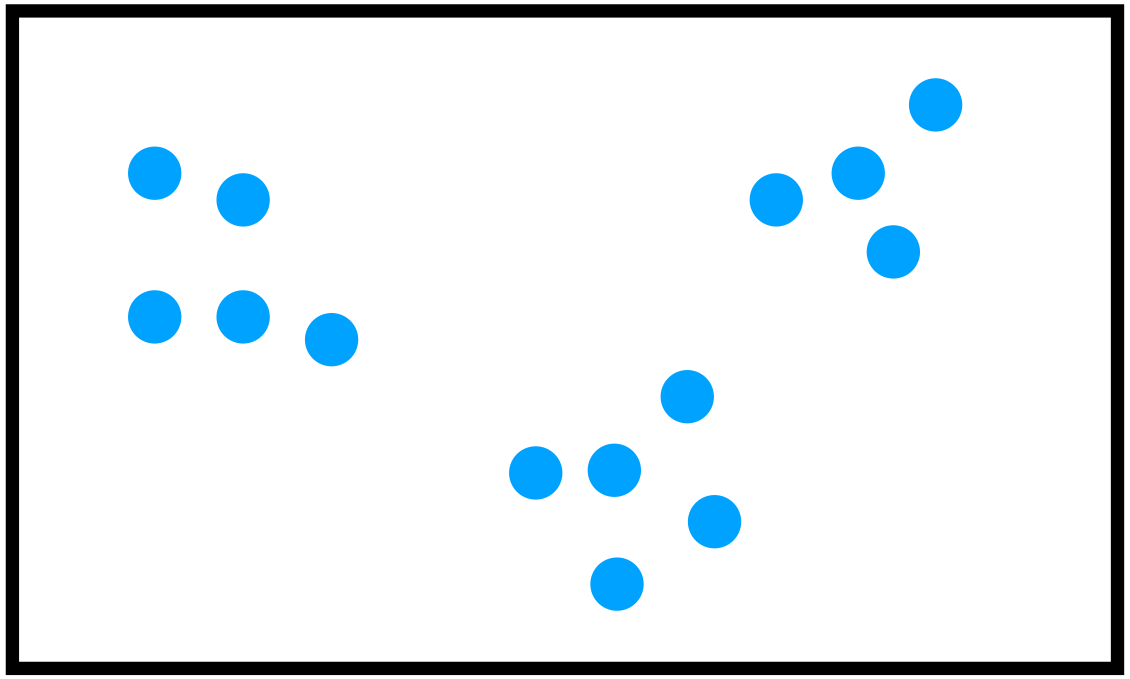
Idea of Lloyd's algorithm

- Estimating center locations of each clusters
  - The locations are called ***centroids***



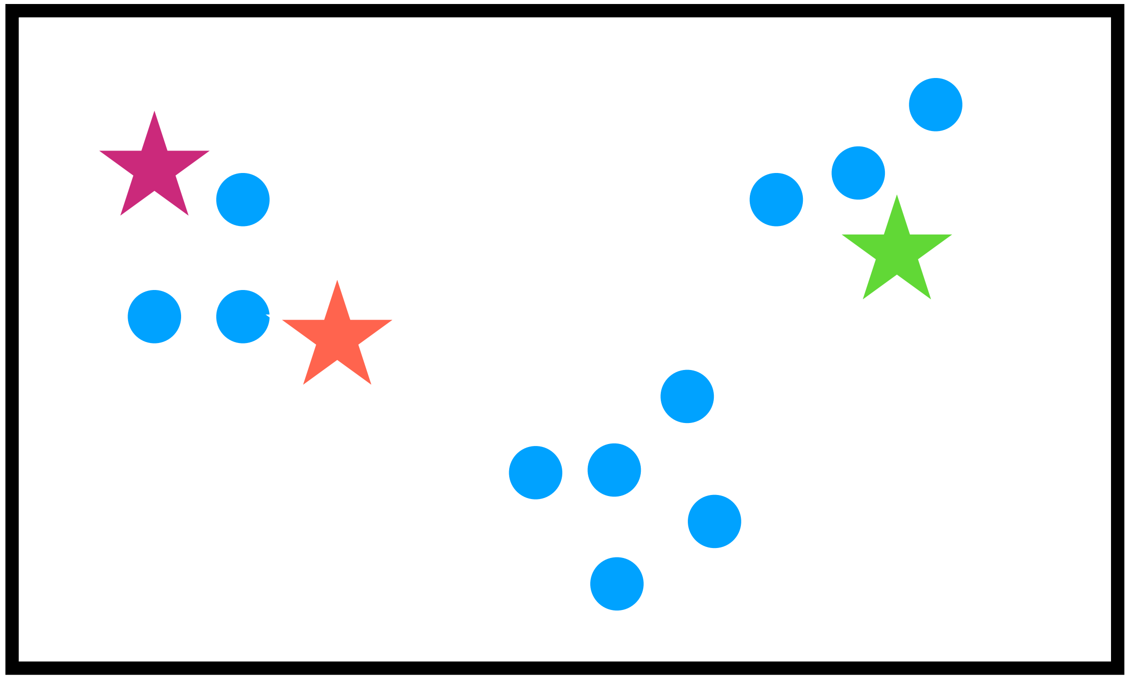
Lloyd's algorithm

1. Select K centroids $m_1, \ldots, m_K$ from given data points at random



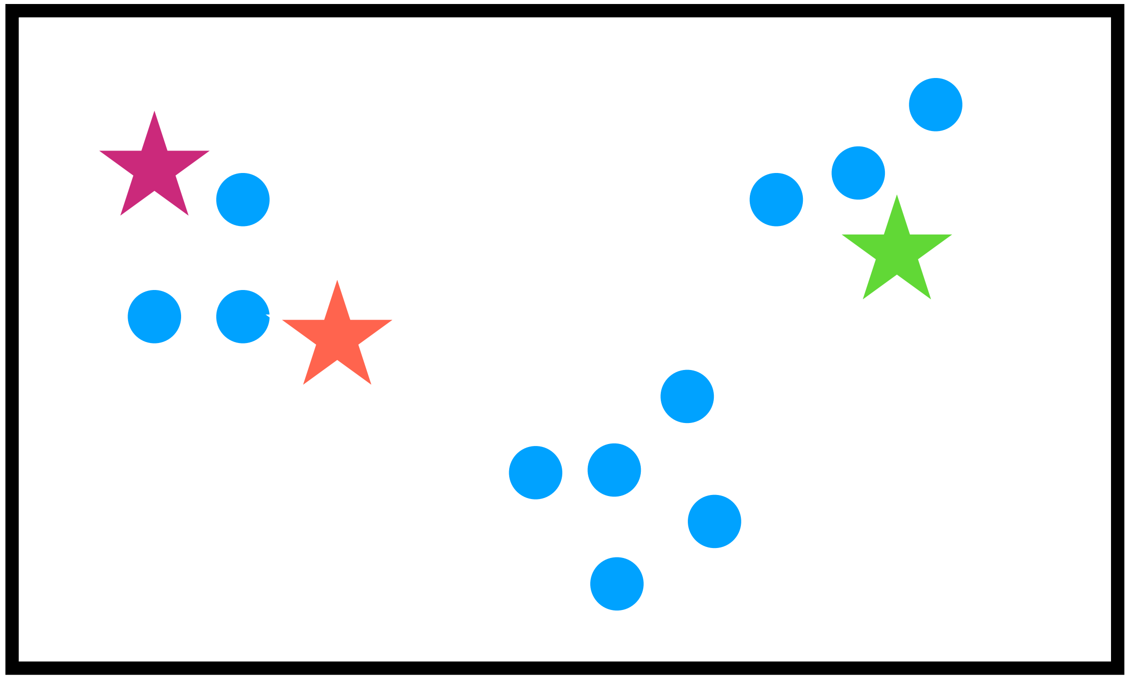
Lloyd's algorithm

1. Select K centroids $m_1, \ldots, m_K$ from given data points at random



Lloyd's algorithm

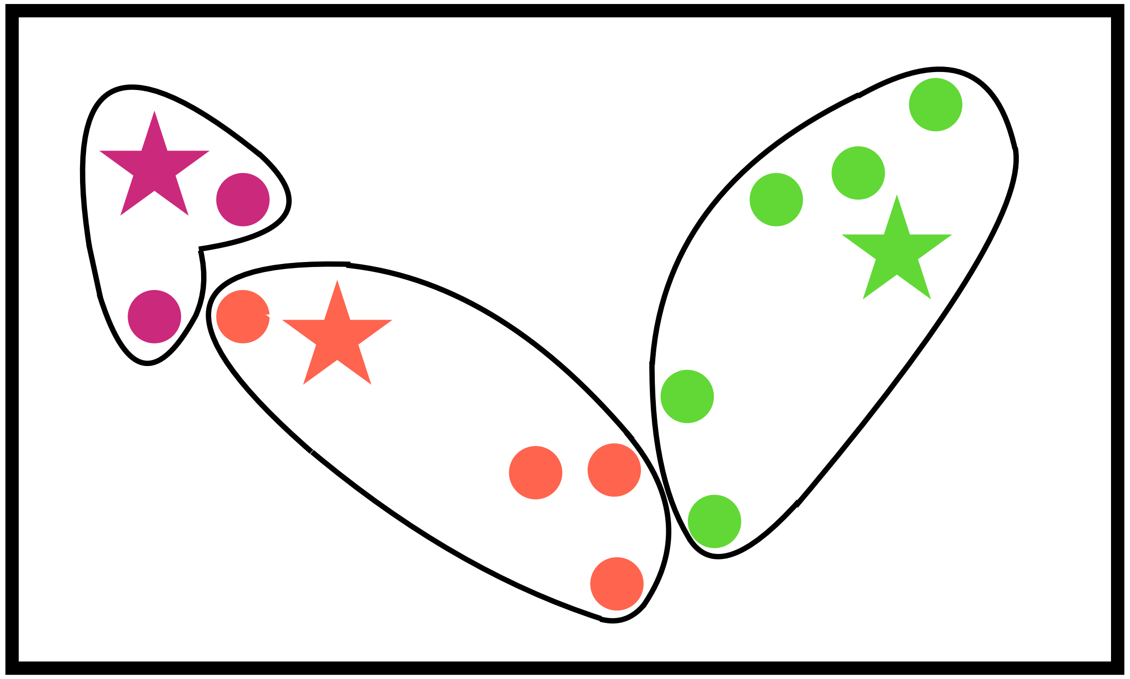
1. Select K centroids $m_1, \ldots, m_K$ from given data points at random
2. Construct clusters $c_1, \ldots, c_K$ s.t. each c_i collects data points nearest to m_i



$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$

Lloyd's algorithm

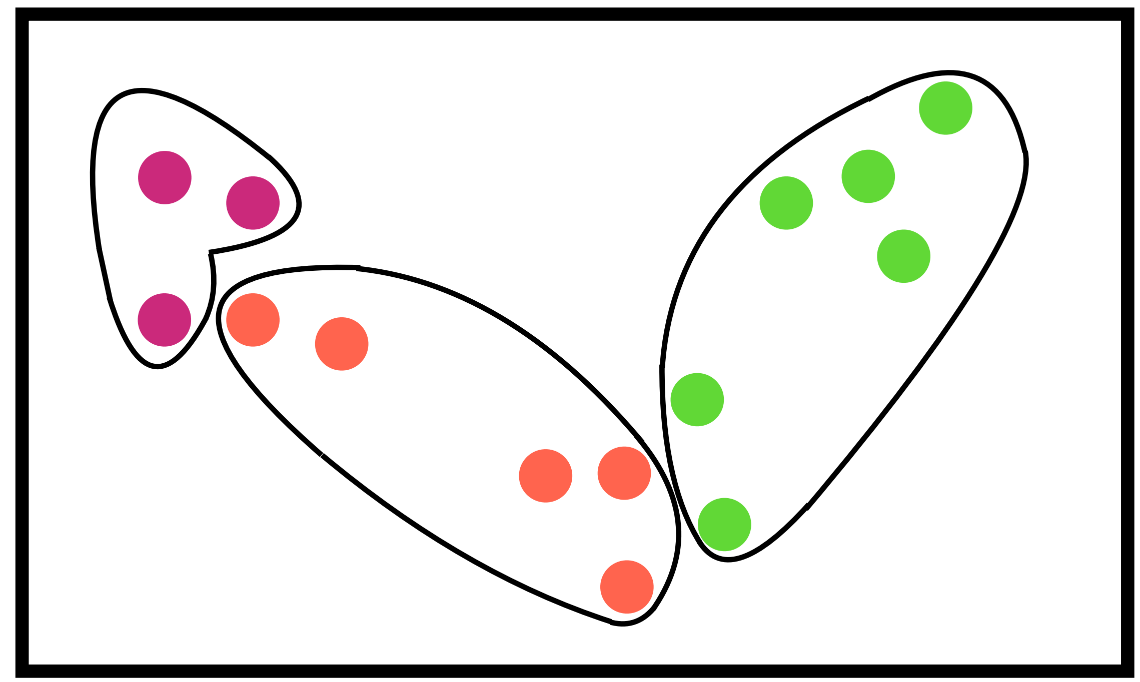
1. Select K centroids $m_1, \ldots, m_K$ from given data points at random
2. Construct clusters $c_1, \ldots, c_K$ s.t. each c_i collects data points nearest to m_i



$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$

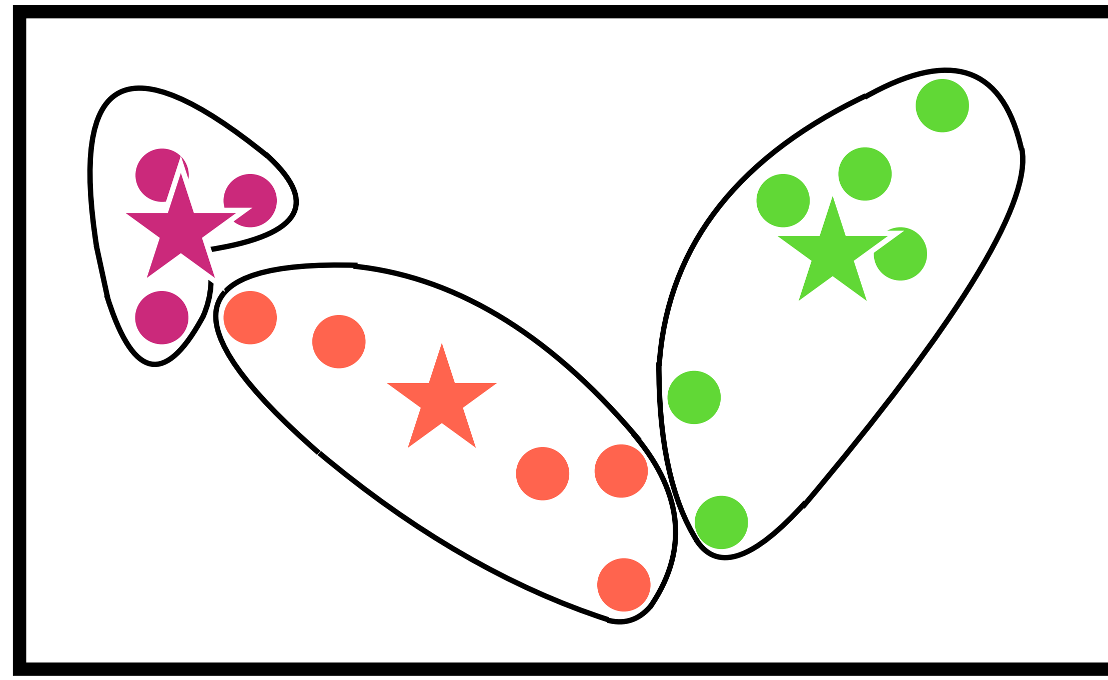
Lloyd's algorithm

1. Select K centroids $m_1, \ldots, m_K$ from given data points at random
2. Construct clusters $c_1, \ldots, c_K$ s.t. each c_i collects data points nearest to m_i



$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$

Lloyd's algorithm

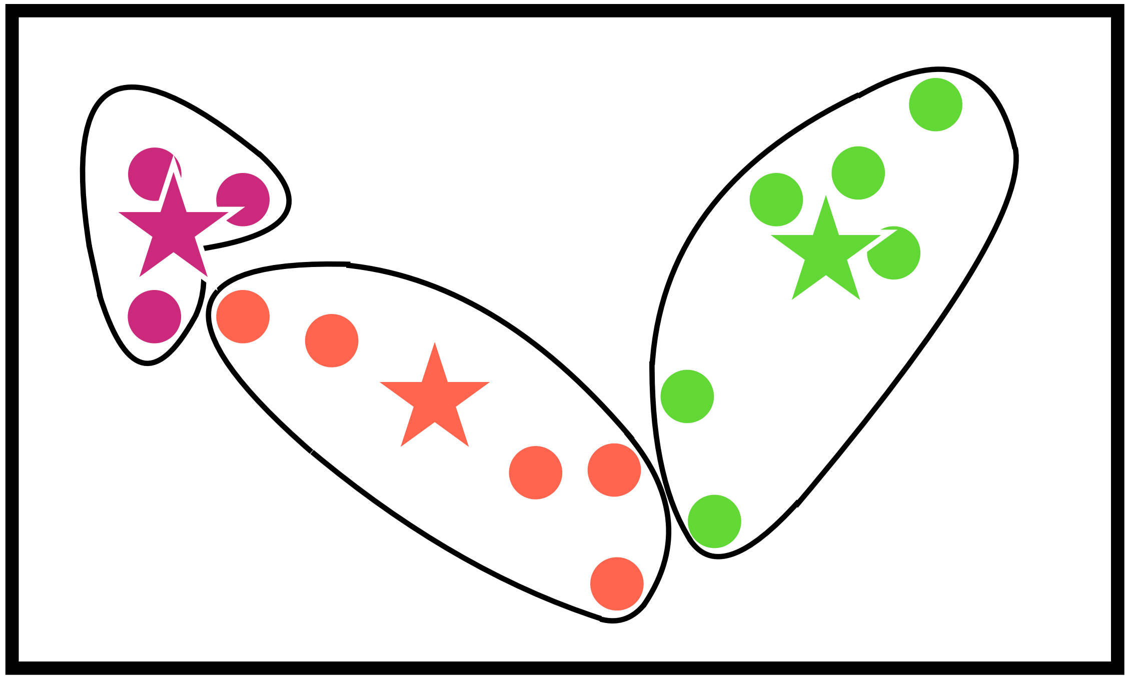


1. Select K centroids $m_1, \ldots, m_K$ from given data points at random
2. Construct clusters $c_1, \ldots, c_K$ s.t. each c_i collects data points nearest to m_i

$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$

3. Update the centroids $m_i = \dfrac{1}{|c_i|} \sum_{x \in c_i} x$

Lloyd's algorithm

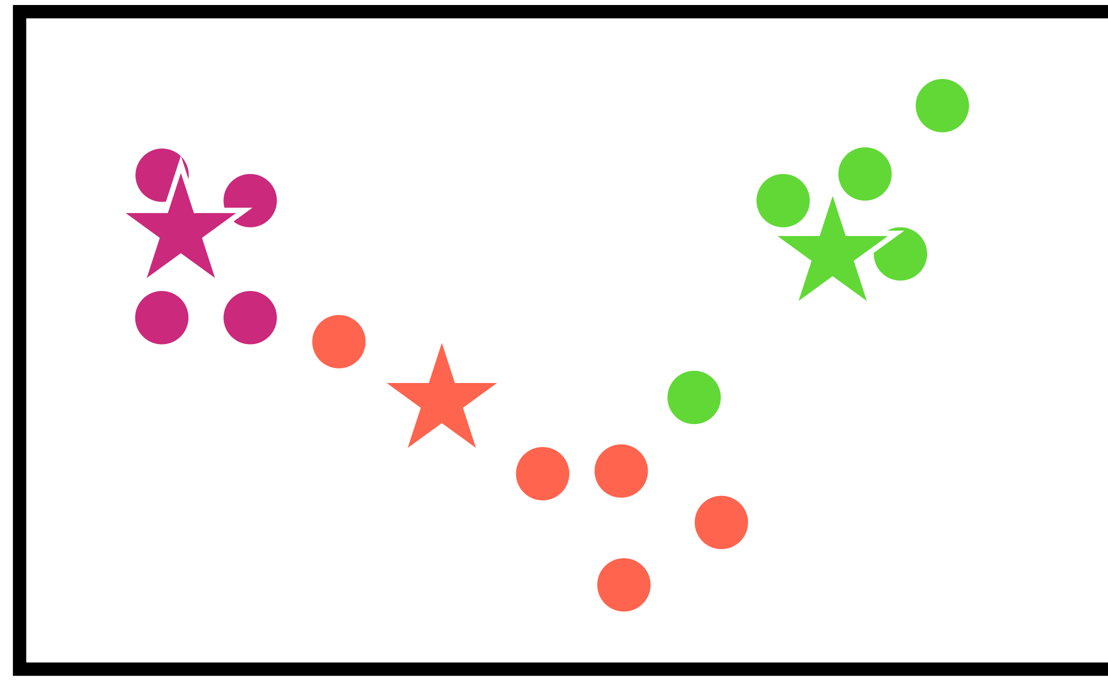


1. Select K centroids $m_1, \ldots, m_K$ from given data points at random
2. Construct clusters $c_1, \ldots, c_K$ s.t. each c_i collects data points nearest to m_i

$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$

3. Update the centroids $m_i = \dfrac{1}{|c_i|} \sum_{x \in c_i} x$
4. Repeat Steps 2 & 3 until the clusters are no longer updated

Lloyd's algorithm



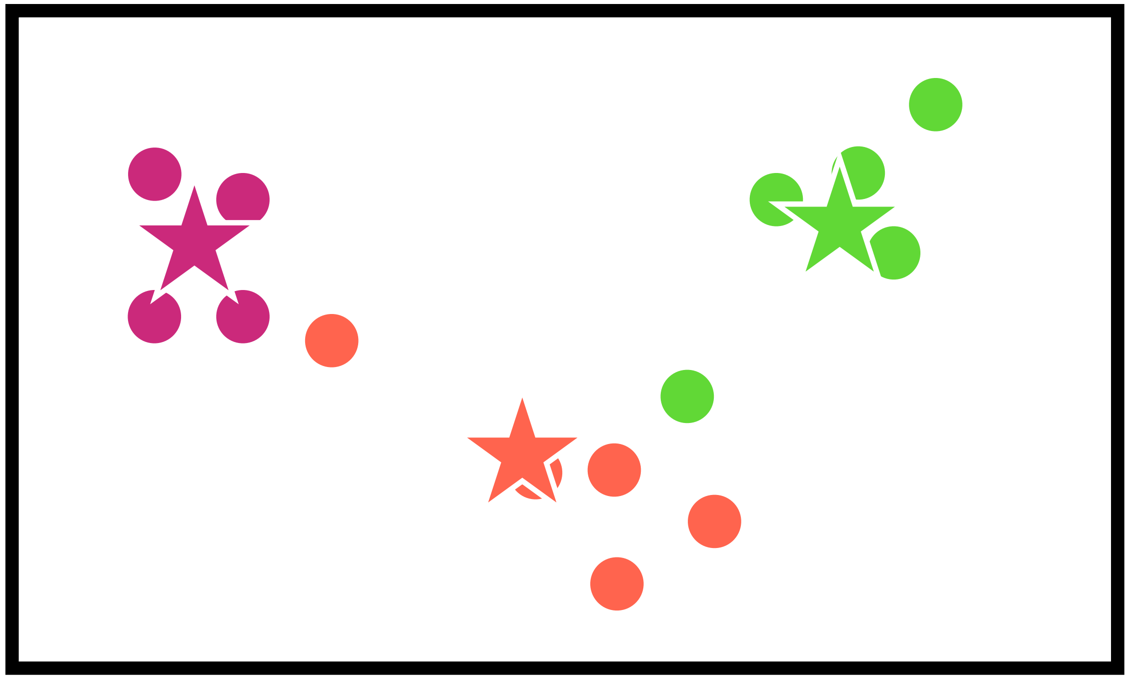
1. Select K centroids $m_1, \ldots, m_K$ from given data points at random
2. Construct clusters $c_1, \ldots, c_K$ s.t. each c_i collects data points nearest to m_i
$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$
3. Update the centroids $m_i = \dfrac{1}{|c_i|} \sum_{x \in c_i} x$
4. Repeat Steps 2 & 3 until the clusters are no longer updated

Lloyd's algorithm

1. Select K centroids $m_1, \ldots, m_K$ from given data points at random
2. Construct clusters $c_1, \ldots, c_K$ s.t. each c_i collects data points nearest to m_i

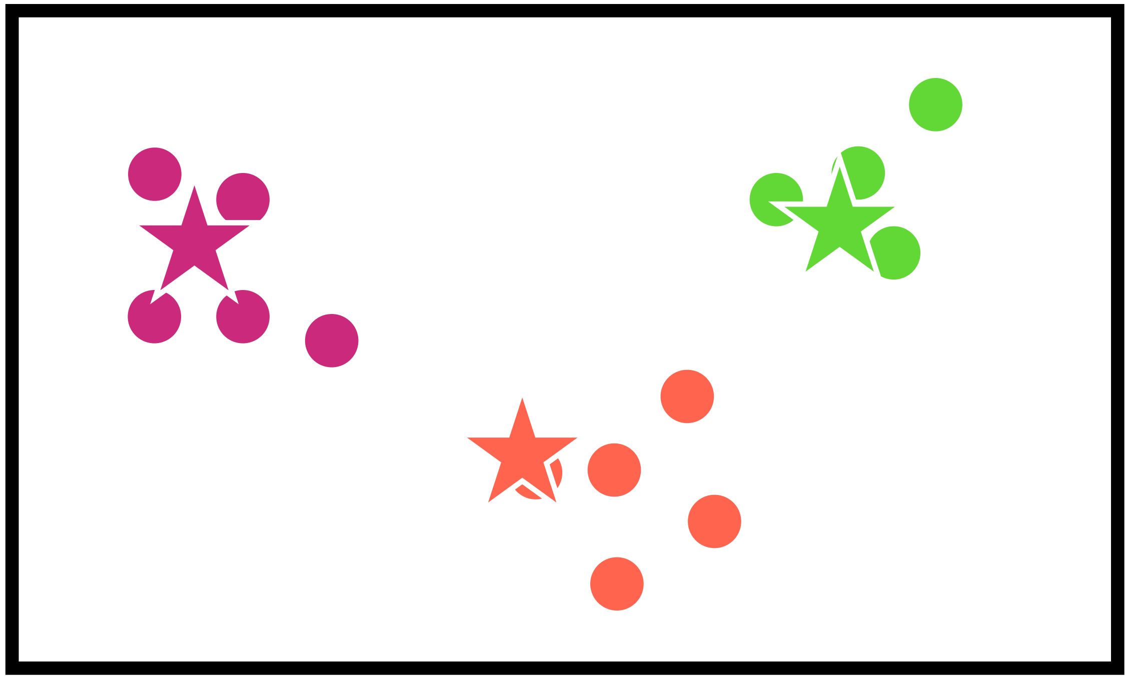
$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$

3. Update the centroids $m_i = \dfrac{1}{|c_i|} \sum_{x \in c_i} x$
4. Repeat Steps 2 & 3 until the clusters are no longer updated



Lloyd's algorithm

1. Select K centroids $m_1, \ldots, m_K$ from given data points at random
2. Construct clusters $c_1, \ldots, c_K$ s.t. each c_i collects data points nearest to m_i

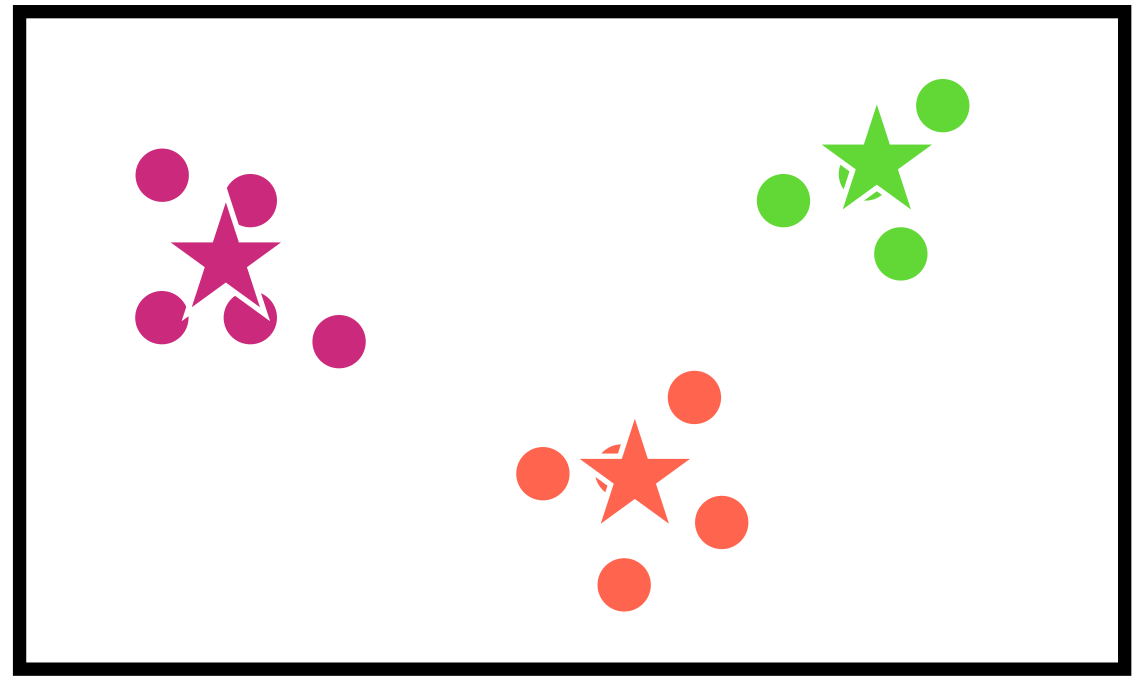


$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$

3. Update the centroids $m_i = \dfrac{1}{|c_i|} \sum_{x \in c_i} x$

4. Repeat Steps 2 & 3 until the clusters are no longer updated

Lloyd's algorithm



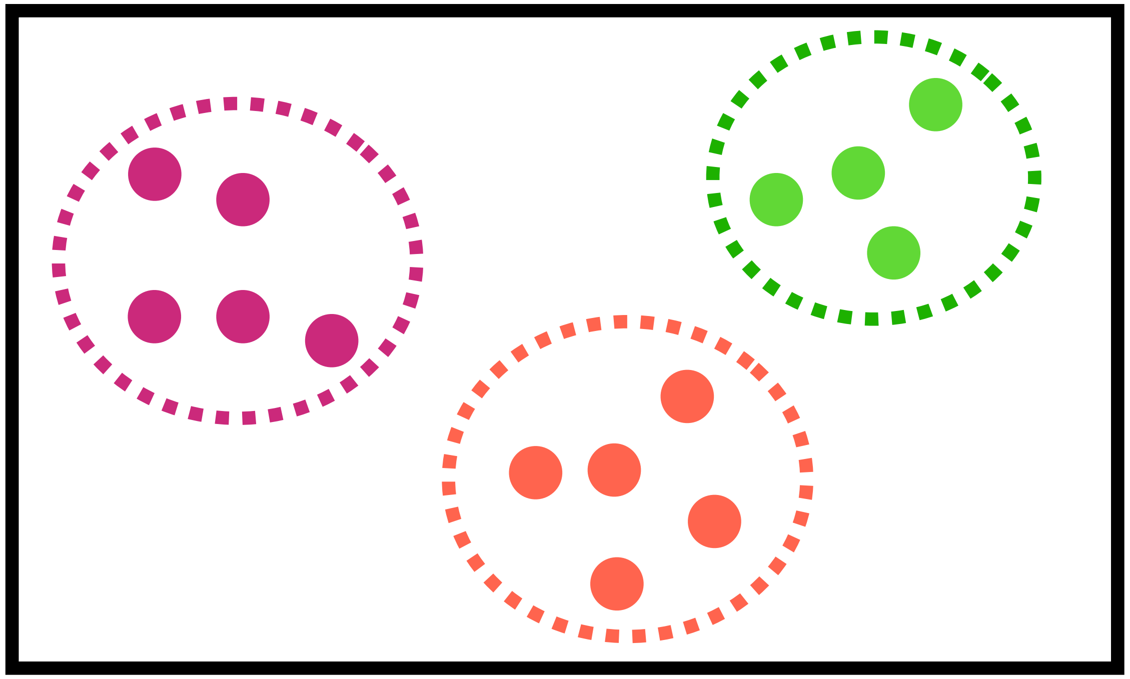
1. Select K centroids $m_1, \dots, m_K$ from given data points at random
2. Construct clusters $c_1, \dots, c_K$ s.t. each c_i collects data points nearest to m_i

$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$

3. Update the centroids $m_i = \dfrac{1}{|c_i|} \sum_{x \in c_i} x$
4. Repeat Steps 2 & 3 until the clusters are no longer updated

Lloyd's algorithm

1. Select K centroids $m_1, \ldots, m_K$ from given data points at random



2. Construct clusters $c_1, \ldots, c_K$ s.t. each c_i collects data points nearest to m_i

$$c_i = \{x \in X \mid \forall j \leq k, \|x - m_i\| \leq \|x - m_j\|\}$$

3. Update the centroids $m_i = \dfrac{1}{|c_i|} \sum_{x \in c_i} x$

4. Repeat Steps 2 & 3 until the clusters are no longer updated

Advantages of Lloyd's algorithm

- Easy to implement
- Fast
  - Time complexity is $O(n^2)$
  - Empirically known that it works *as if* its time complexity is $O(n)$

Disadvantages of Lloyd's algorithm

- Difficult to choose K
  - Users have to decide the number of clusters in advance
- Dependence on initial centroids
  - Globally optimal clusters may not be found

Agenda

- Cluster analysis
  - Hierarchical clustering
  - K-means
- **Feature transformation**
  - **Principal component analysis (PCA)**

Principal components analysis

- Transforming features into informative ones, called ***principal components (PCs)***
- Transformation is performed by linear combination
    - Function F mapping m-dimensional points to PCs is expressed by

    $$F(x_1, \ldots, x_m) = a_1 x_1 + a_2 x_2 + \cdots + a_m x_m$$

    using parameters $a_1, \ldots, a_m$

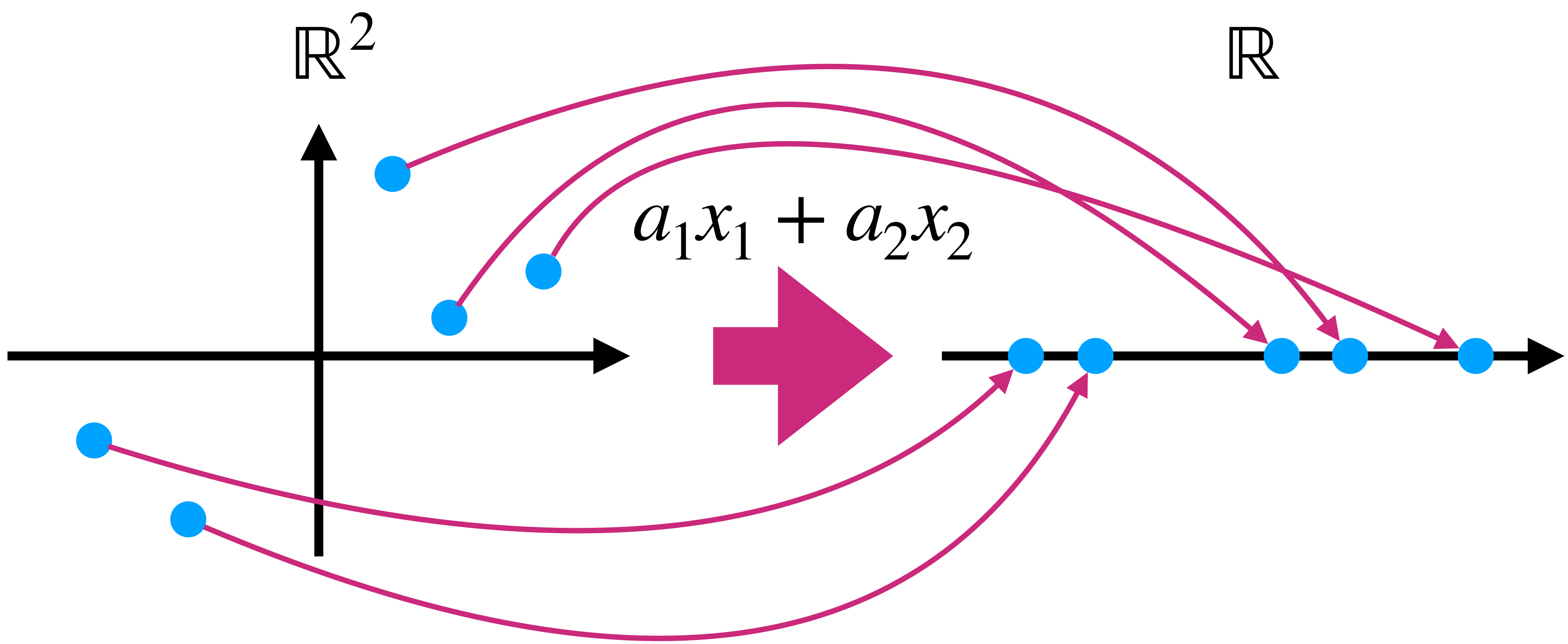
Problem

What parameters $a_1, \ldots, a_m$ *do make* F *return the most informative PCs?*

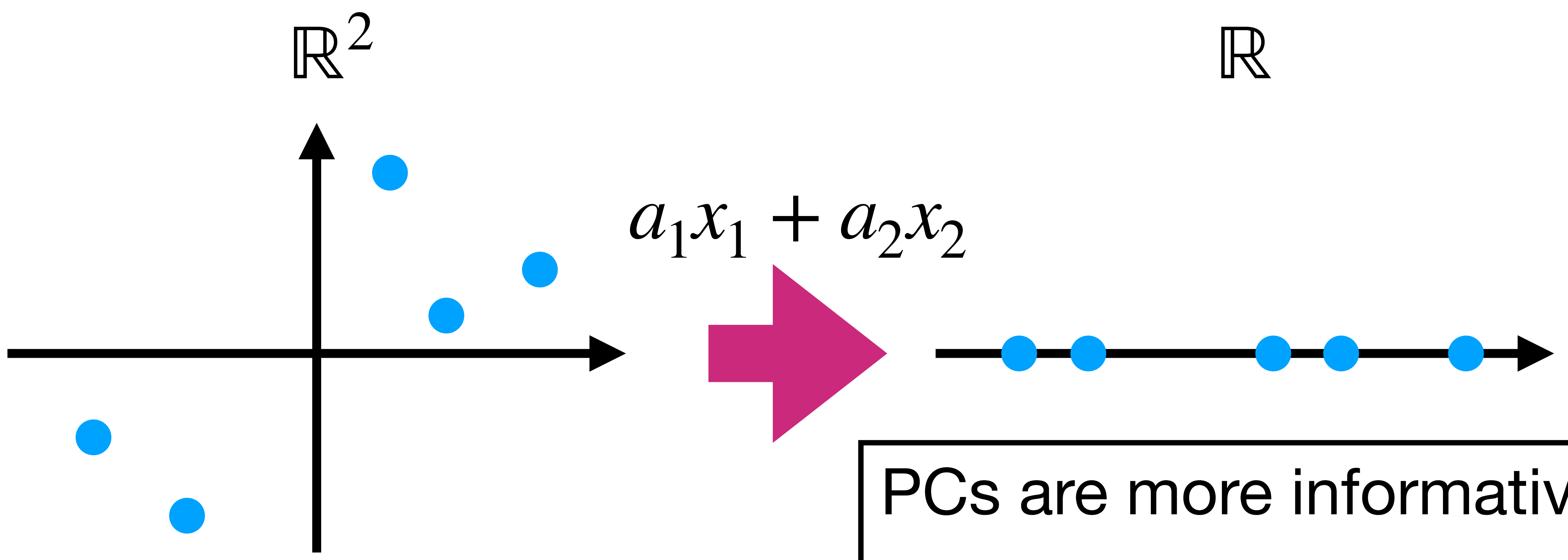
Q1. What does “informative” mean?

Q2. How do we calculate the parameters?

Transformation by linear combination



Transformation by linear combination

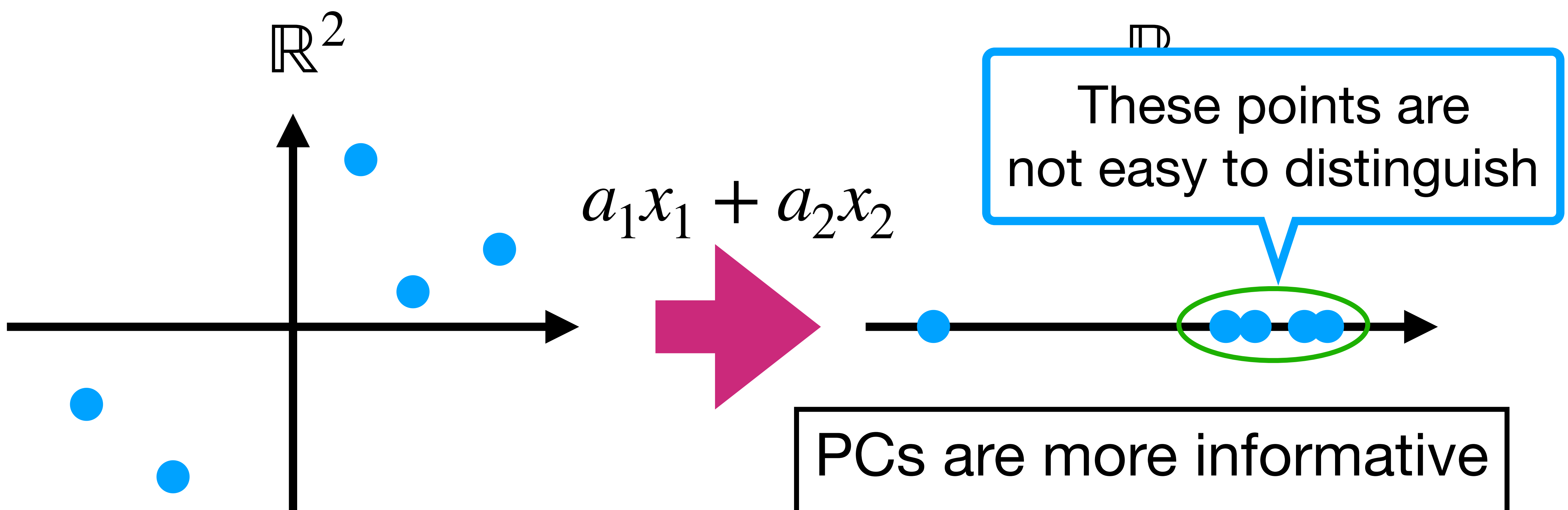


PCs are more informative

=

The PCs are ***easier to distinguish***

Transformation by linear combination

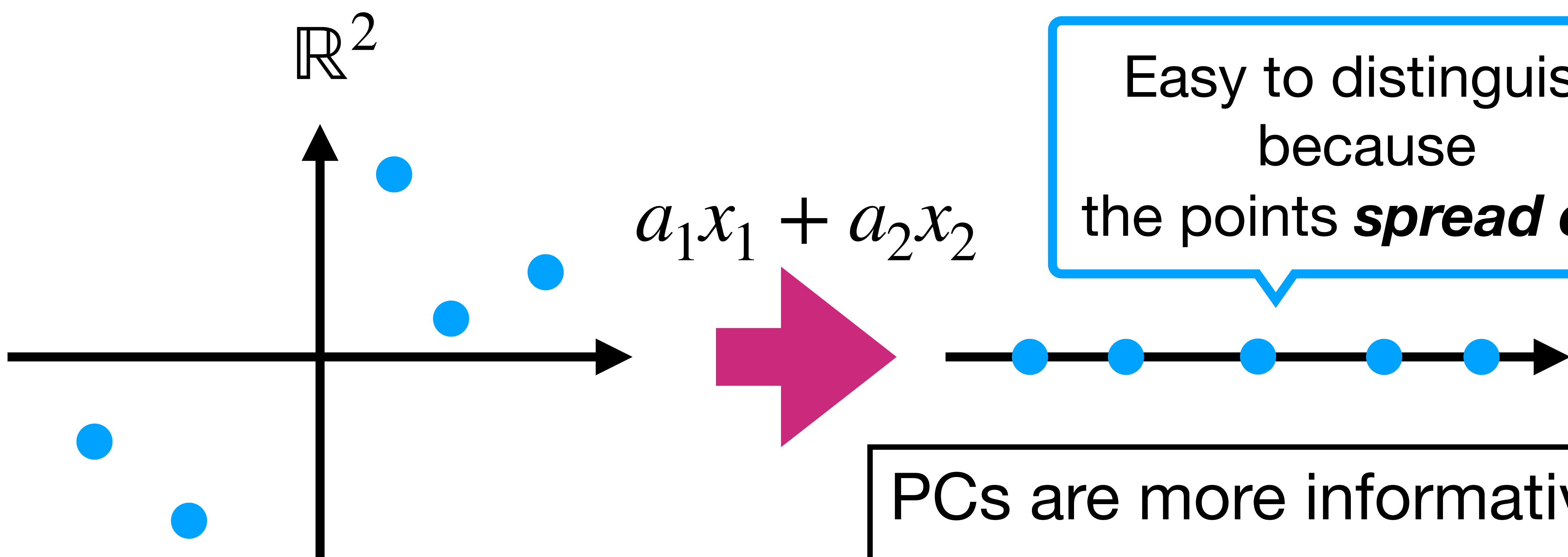


PCs are more informative

=

The PCs are ***easier to distinguish***

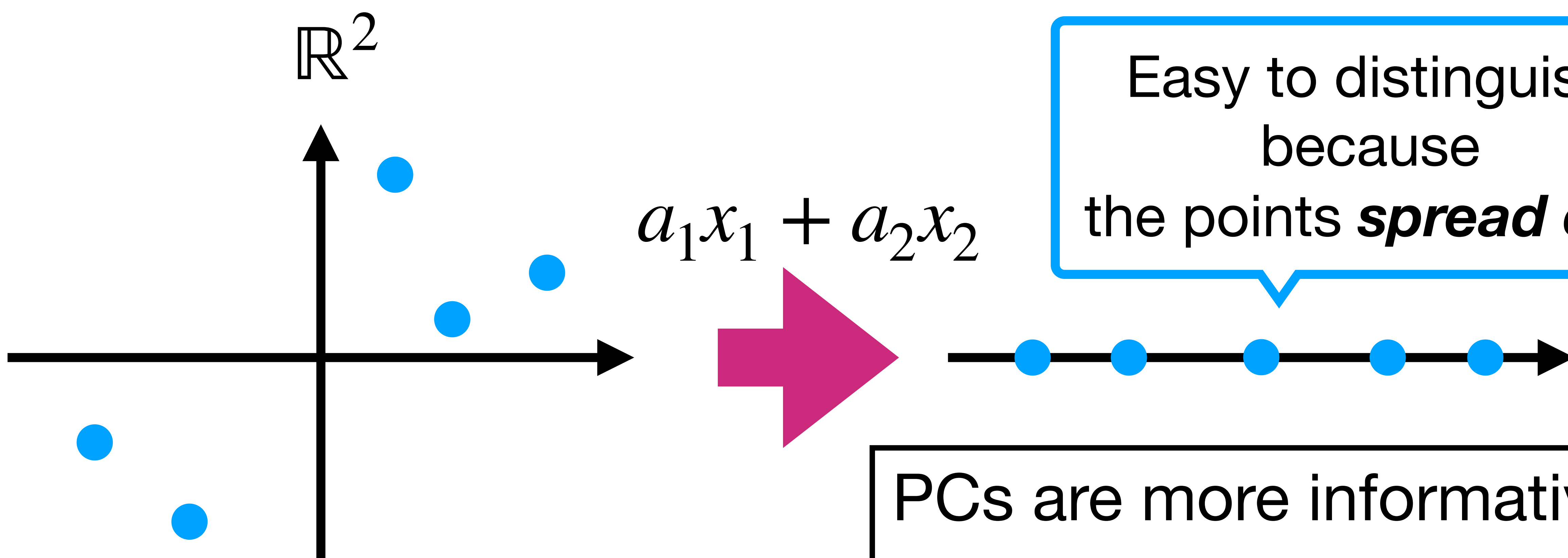
Transformation by linear combination



Easy to distinguish
because
the points ***spread out***

PCs are more informative
=
The PCs are ***easier***
to distinguish

Transformation by linear combination



Easy to distinguish because the points ***spread out***

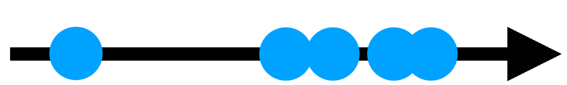
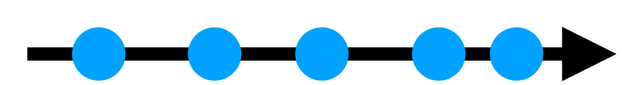
PCs are more informative
=
The PCs spread out ***more widely***

Variance

- A statistical metric to measure how data points spread out
- The variance $\mathbf{Var}(X)$ of $X \subseteq \mathbb{R}$ is defined by

$$\frac{1}{|X|}\sum_{x \in X}(x-m)^2$$

where m is the mean of X $\left(=\frac{1}{|X|}\sum_{x \in X} x\right)$

The variance of  $<$ **The variance of** 

Goal

- Finding parameters $a_1, \ldots, a_m$ s.t.

  $\mathbf{Var}(\{F(x) \mid x \in X \subseteq \mathbb{R}^m\})$ is maximized

  (= the variance of the PCs $F(x)$ for the data points x in X is maximized)

Problem

The variance can be arbitrarily large by taking large parameters

Goal

- Finding parameters $a_1, \ldots, a_m$ s.t.

  $\mathbf{Var}(\{F(x) \mid x \in X \subseteq \mathbb{R}^m\})$ is maximized

  (= the variance of the PCs $F(x)$ for the data points x in X is maximized)

  under the condition that $a_1^2 + \cdots + a_m^2 = 1$

Problem

What parameters $a_1, \ldots, a_m$ *do make* F *return the most informative PCs?*

Q1. What does “informative” mean?

Q2. How do we calculate the parameters?

Problem

What parameters $a_1, \ldots, a_m$ *do make* F *return the most informative PCs?*

Q1. What does "informative" mean?

Q2. How do we calculate the parameters?

- Using Language multipliers
- C.f. *"Pattern Recognition and Machine Learning" Chapter 12.1* for detail

The second and further PCs

- The PCs returned by F usually lost some information from original features
- The 2nd, 3rd, 4th, …, and m-th informative PCs are needed to recover the original features in $\mathbb{R}^m$ completely
- In general, the i-th informative PCs are obtained as F but they should be independent of the $(i-1)$-th and earlier informative PCs