FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

NATURWISSENSCHAFTLICHE
FAKULTÄT

# Unsupervised Learning: Clustering

Lecture “Mathematics of Learning” 2022

Andreas Bärmann
Friedrich-Alexander-Universität Erlangen-Nürnberg

WIRTSCHAFTS
MATHEMATIK

# Rough Differentiation in Learning Methods

Supervised learning:

- predict values of an outcome measure based on a number of input measures (e.g. given some patient data together with label ’has illness’ / ’does not have illness’. New patient data comes in, predict whether he is ill or not.)

# Rough Differentiation in Learning Methods

Supervised learning:

- predict values of an outcome measure based on a number of input measures (e.g. given some patient data together with label ’has illness’ / ’does not have illness’. New patient data comes in, predict whether he is ill or not.)

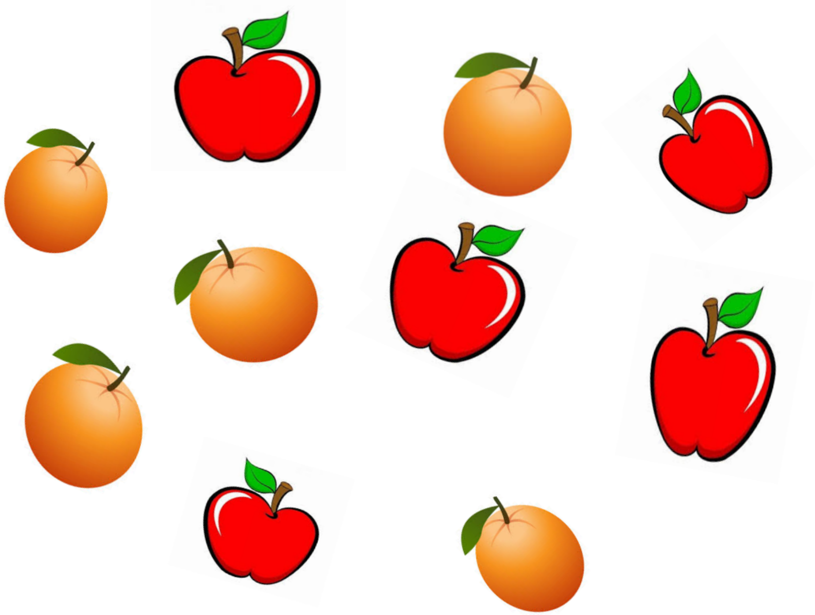
Unsupervised learning:

- no outcome measure is given. Goal: find structures in the data.

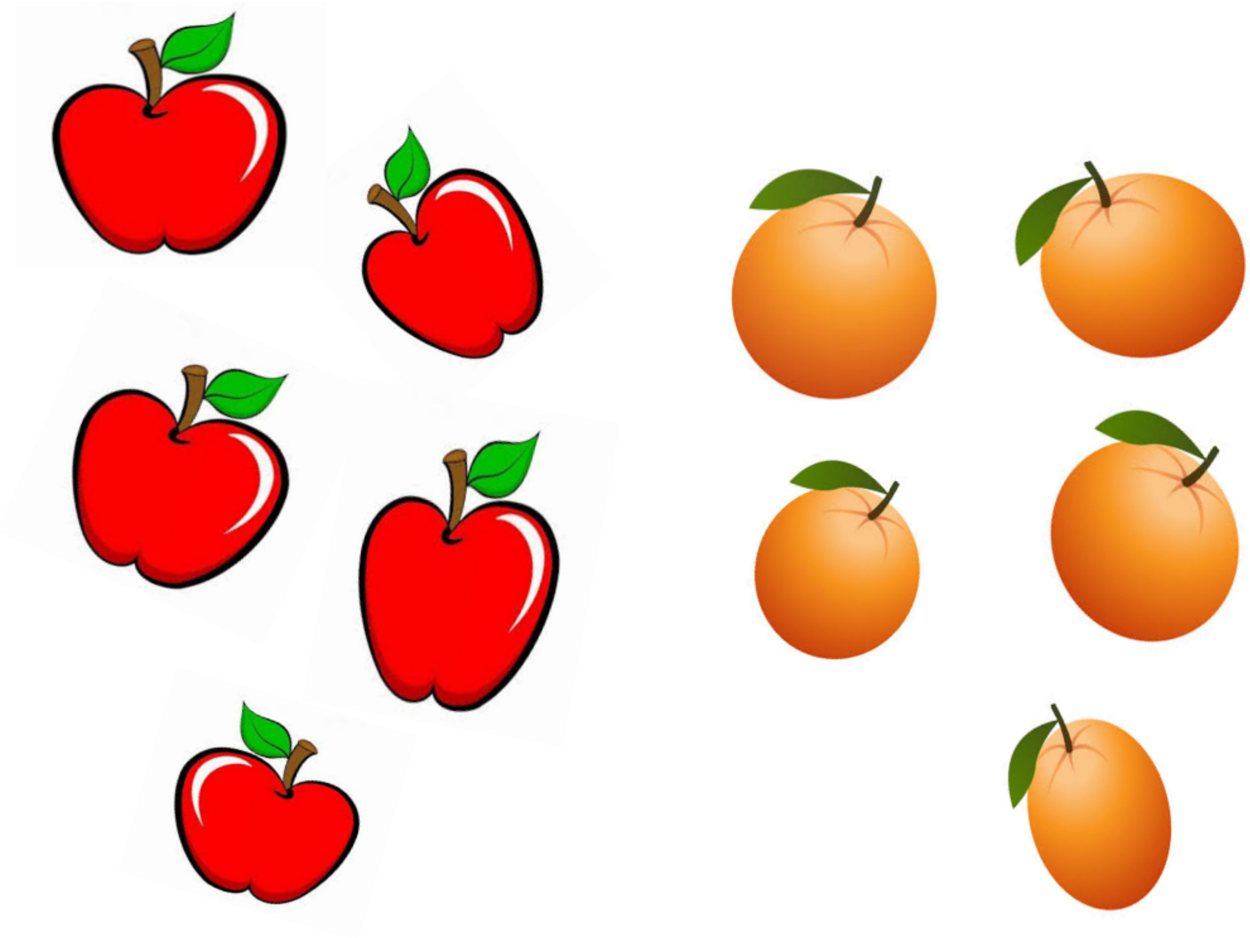
There is also something in between: semi-supervised learning.

# Unsupervised Learning: Clustering of Data

Is there any structure in this data?



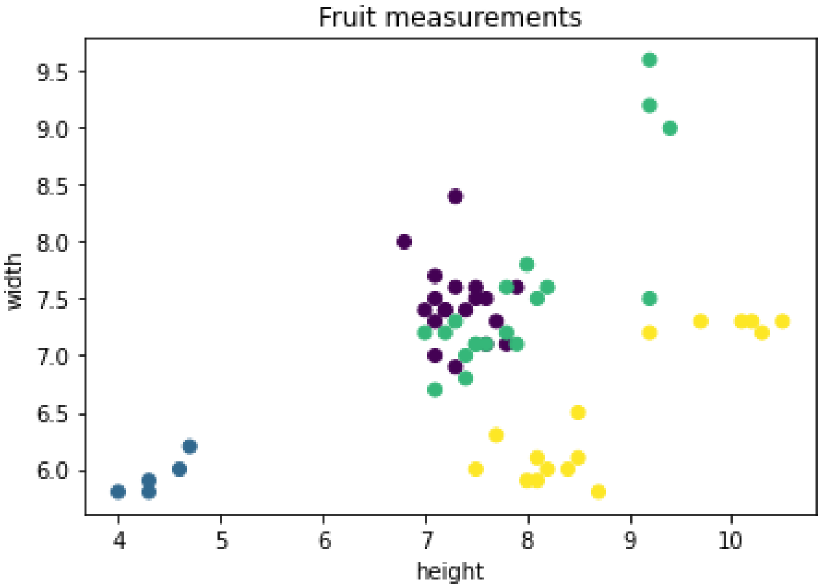
There are **two categories/clusters** such that objects *within* a cluster resemble each other but objects from *different clusters* look different.



# What is clustering?

Given fruit measurement data $(\text{height}_i, \text{width}_i)_{i=1}^N$.

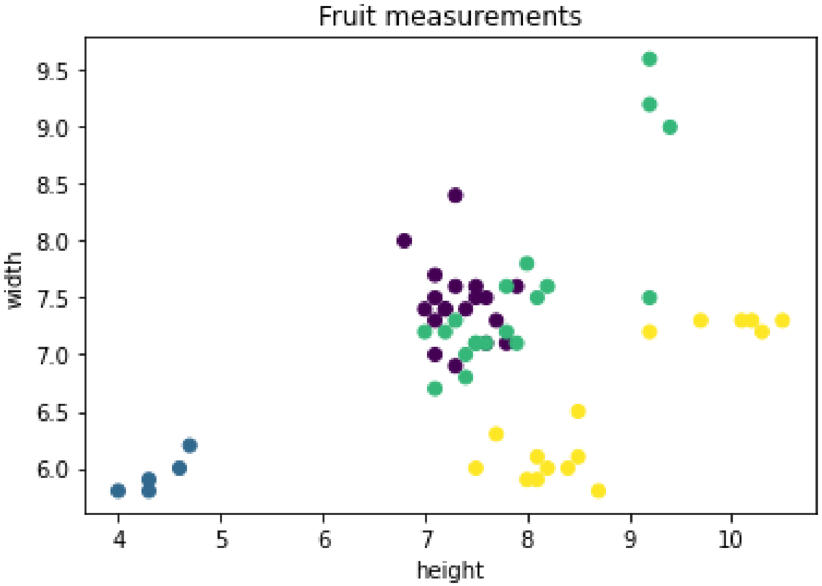
- Visually: There are multiple "categories" of data.



# What is clustering?

Given fruit measurement data $(\text{height}_i, \text{width}_i)_{i=1}^{N}$.

- Visually: There are multiple "categories" of data.
- How can we sort data into categories/clusters?

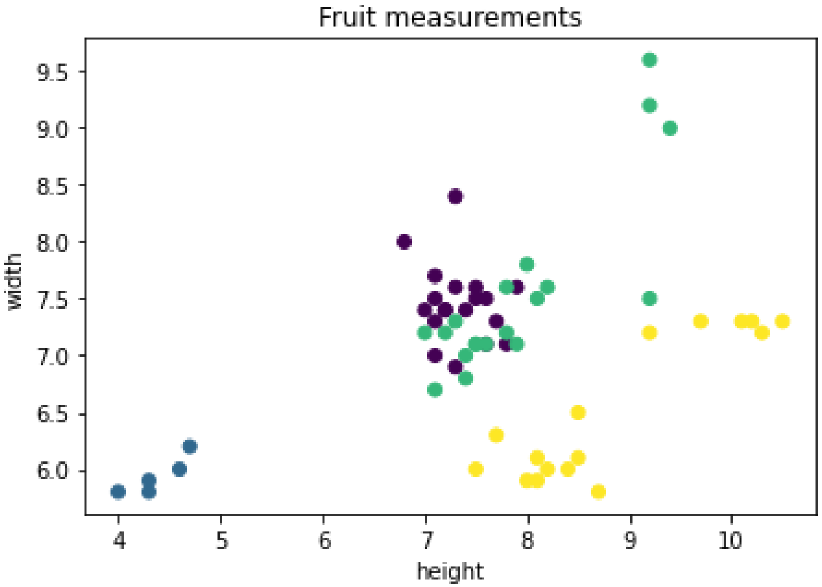


# What is clustering?

Given fruit measurement data $(\text{height}_i, \text{width}_i)_{i=1}^N$.

- Visually: There are multiple "categories" of data.
- How can we sort data into categories/clusters?

→ clustering problem



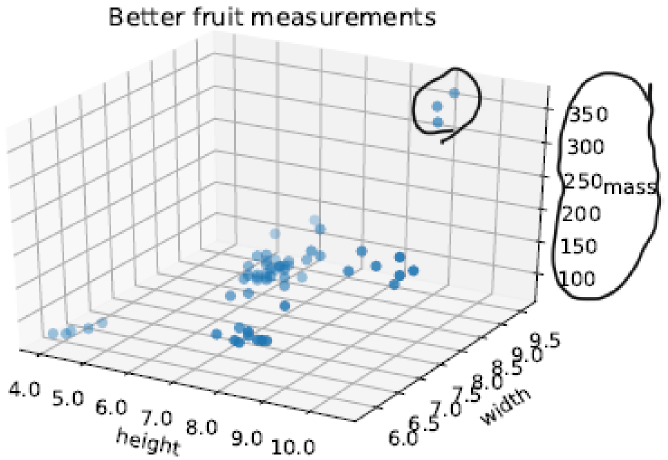
# What is clustering?

Given fruit measurement data $(\text{height}_i, \text{width}_i)_{i=1}^N$.

- Visually: There are multiple “categories” of data.
- How can we sort data into categories/clusters?

$\rightarrow$ clustering problem

- More measurements help.


Better fruit measurements

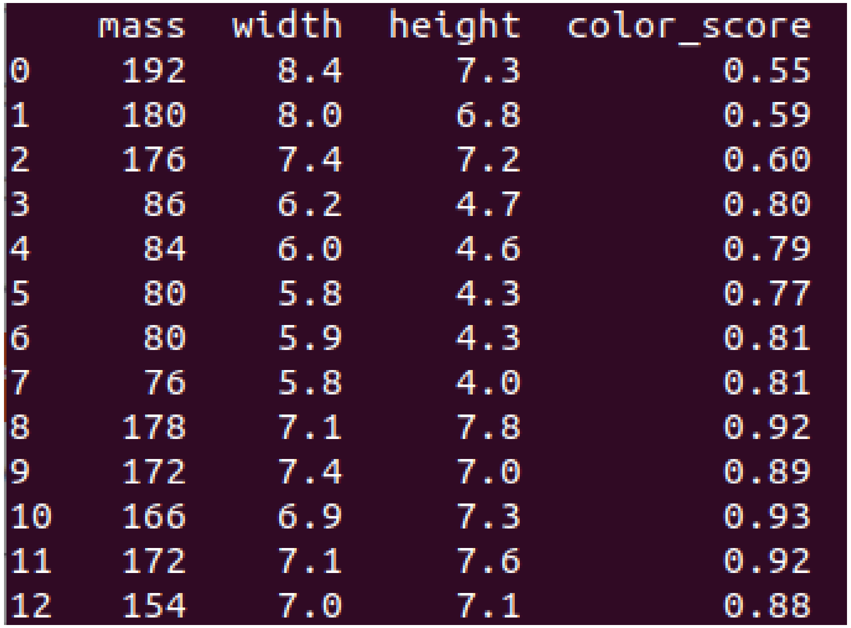
# What is clustering?

Given fruit measurement data $(\text{height}_i, \text{width}_i)_{i=1}^{N}$.

- Visually: There are multiple "categories" of data.
- How can we sort data into categories/clusters?

→ clustering problem

- More measurements help.
- Clustering the raw data by hand is cumbersome.



```
     mass  width  height  color_score
0     192    8.4     7.3         0.55
1     180    8.0     6.8         0.59
2     176    7.4     7.2         0.60
3      86    6.2     4.7         0.80
4      84    6.0     4.6         0.79
5      80    5.8     4.3         0.77
6      80    5.9     4.3         0.81
7      76    5.8     4.0         0.81
8     178    7.1     7.8         0.92
9     172    7.4     7.0         0.89
10    166    6.9     7.3         0.93
11    172    7.1     7.6         0.92
12    154    7.0     7.1         0.88
```

# Clustering Problem

Given

- $N$: number of data points
- $M$: number of variables (e.g. “mass”, “price”, “color”, ...)
- Data $X = \{x_1, \ldots, x_N\}$, where $x_n \in \mathbb{R}^M$ for all $n = 1, \ldots, N$
- $K$: number of *assumed* clusters

Want

# Clustering Problem

Given

- $N$: number of data points
- $M$: number of variables (e.g. “mass”, “price”, “color”, ...)
- Data $X = \{x_1, \ldots, x_N\}$, where $x_n \in \mathbb{R}^M$ for all $n = 1, \ldots, N$
- $K$: number of *assumed* clusters

Want

- *Assignment:* $x_n \mapsto k_n \in \{1, \ldots, K\}$ for all $n = 1, \ldots, N$

# Clustering Problem

Given

- $N$: number of data points
- $M$: number of variables (e.g. “mass”, “price”, “color”, ...)
- Data $X = \{x_1, \dots, x_N\}$, where $x_n \in \mathbb{R}^M$ for all $n = 1, \dots, N$
- $K$: number of *assumed* clusters

Want

- *Assignment:* $x_n \mapsto k_n \in \{1, \dots, K\}$ for all $n = 1, \dots, N$
- *Assignment rule:* $\mathbf{x} \mapsto k(\mathbf{x}) \in \{1, \dots, K\}$ for all $x \in \mathbb{R}^M$

# Clustering Problem

Given

- $N$: number of data points
- $M$: number of variables (e.g. “mass”, “price”, “color”, ...)
- Data $X = \{x_1, \dots, x_N\}$, where $x_n \in \mathbb{R}^M$ for all $n = 1, \dots, N$
- $K$: number of *assumed* clusters

Want

- *Assignment:* $x_n \mapsto k_n \in \{1, \dots, K\}$ for all $n = 1, \dots, N$
- *Assignment rule:* $\mathbf{x} \mapsto k(\mathbf{x}) \in \{1, \dots, K\}$ for all $x \in \mathbb{R}^M$
- *Reconstruction rule (’representative’):* $k \mapsto m_k \in \mathbb{R}^M$

On an abstract level:

- Determination of best possible clustering (w.r.t. some objective) is a classical combinatorial optimization problem
- K-means clustering: Determine $K$ points (“centres”, i.e. representatives) that minimize the sum of the squared Euclidean distance to its closest centre.

# Clustering Problems and Algorithms

- Already in simplified/restricted situations, the problem is difficult, i.e. *NP-hard*. We refrain from giving a formal definition of the meaning of this and refer to theoretical computer science for more details. However, we note it implies that we cannot expect to be able to determine an efficient algorithm that can efficiently determine the best clustering within polynomial time in the input size.
- more specifically: M. Mahajan, P. Nimbhorkar, K. Varadarajan: The planar k-means problem is NP-hard. Proceedings of WALCOM: Algorithms and Computation, S.274-285 (2009).
- Clustering is a very basic learning task. Depending on the application, different clustering algorithms work best. We will focus on well-known and often used algorithms, K-Means and Expectation Maximization. We will briefly touch hierarchical clustering and introduce principal component analysis for clustering and data reduction.

# K-means clustering as an optimization problem

Find clustering $\underline{C} = \{C_1, \ldots, C_K\}$ into subsets $C_k \subseteq X$,
i.e. 1.) $\cup_{k=1}^{n} C_k = X$, 2.) $C_k \cap C_l = \emptyset$ for $k \neq l$ and 3.) $C_k \neq \emptyset$ for all $k$,
and centres $\underline{m} = \{m_1, \ldots, m_K\}$, $m_k \in \mathbb{R}^M$, which minimize the clustering energy

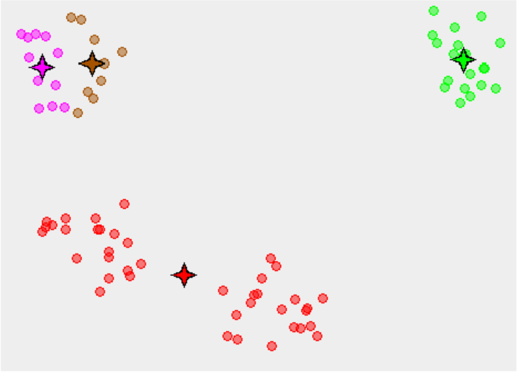
$$E(\underline{C}, \underline{m}) := \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2 .$$

# K-means clustering as an optimization problem

Find clustering $\underline{C} = \{C_1, \dots, C_K\}$ into subsets $C_k \subseteq X$,
i.e. 1.) $\cup_{k=1}^{n} C_k = X$, 2.) $C_k \cap C_l = \emptyset$ for $k \neq l$ and 3.) $C_k \neq \emptyset$ for all $k$,
and centres $\underline{m} = \{m_1, \dots, m_K\}$, $m_k \in \mathbb{R}^M$, which minimize the clustering energy

$$E(\underline{C}, \underline{m}) := \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2 .$$

Observation: The clustering energy has local minima



(picture from: `https://upload.wikimedia.org/wikipedia/commons/7/7c/K-means_convergence_to_a_local_minimum.png`, modified)

# Derivation of the K-means algorithm

Let us fix the clustering $\underline{C}$ in

$$E(\underline{C}, \underline{m}) := \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2 .$$

# Derivation of the K-means algorithm

Let us fix the clustering $\underline{C}$ in

$$E(\underline{C}, \underline{m}) := \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2 \, .$$

necessary first-order optimality condition: gradient with respect to $m_k$ is zero, i.e. we have a critical point.
Taking the gradient with respect to $m_k$,
we obtain the first-order optimality condition:

$$\nabla_{m_k} E(\underline{C}, \underline{m}) = \sum_{x \in C_k} (x - m_k) \overset{!}{=} 0 \quad \Rightarrow \quad \sum_{x \in C_k} x - |C_k| m_k \overset{!}{=} 0$$

# Derivation of the K-means algorithm

Let us fix the clustering $\underline{C}$ in

$$E(\underline{C}, \underline{m}) := \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2 .$$

necessary first-order optimality condition: gradient with respect to $m_k$ is zero, i.e. we have a critical point.
Taking the gradient with respect to $m_k$,
we obtain the first-order optimality condition:

$$\nabla_{m_k} E(\underline{C}, \underline{m}) = \sum_{x \in C_k} (x - m_k) \overset{!}{=} 0 \quad \Rightarrow \quad \sum_{x \in C_k} x - |C_k| m_k \overset{!}{=} 0$$

and hence

$$m_k = \frac{1}{|C_k|} \sum_{x \in C_k} x \mathrel{\widehat{=}} \text{mean of the cluster}$$

Problem: we do not know the clusters $C_k$.
Thus, we heuristically search for good means

# Derivation of the K-means algorithm

Conversely, let us fix the means $\underline{m}$ in

$$E(\underline{C}, \underline{m}) := \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2 .$$

# Derivation of the K-means algorithm

Conversely, let us fix the means $\underline{m}$ in

$$E(\underline{C}, \underline{m}) := \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2 .$$

perform the simple assignment step

$$C_k := \{x \in X \,:\, \|x - m_k\| \leq \left\|x - m_j\right\| \text{ for all } j = 1, \ldots, K\}$$

$\widehat{=}$ Voronoi cell of $m_k$

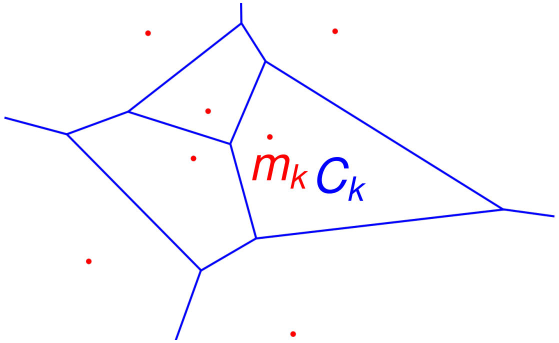
## Derivation of the K-means algorithm

Conversely, let us fix the means $\underline{m}$ in

$$E(\underline{C}, \underline{m}) := \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2 .$$

perform the simple assignment step

$$C_k := \{x \in X \,:\, \|x - m_k\| \leq \left\|x - m_j\right\| \text{ for all } j = 1, \ldots, K\}$$
$$\widehat{=} \text{ Voronoi cell of } m_k$$



# K-means clustering algorithm

Simple idea: alternating update of the means and the resulting clustering

```
Data: $X = \{x_1, \dots, x_N\}$ and number of clusters $K \in \mathbb{N}$
Result: cluster means $\underline{m} = (m_1, \dots, m_K)$
-----------------------------------------------------------------------
initialize $\underline{m}$ randomly;
repeat
    // assignment step:
    for $n \leftarrow 1$ to $N$ // assign n-th point to cluster with nearest mean do
        $k_n \leftarrow \operatorname{argmin}_k \|x_n - m_k\|$
    end
    // update step:
    for $k \leftarrow 1$ to $K$ do
        $C_k \leftarrow \{n \in \{1, \dots, N\} : k_n = k\}$ // cluster
        if $|C_k| > 0$ then
            $m_k \leftarrow \frac{1}{|C_k|} \sum_{n \in C_k} x_n$ // cluster mean
        end
    end
until assignment step does not do anything;
```

# K-means clustering algorithm

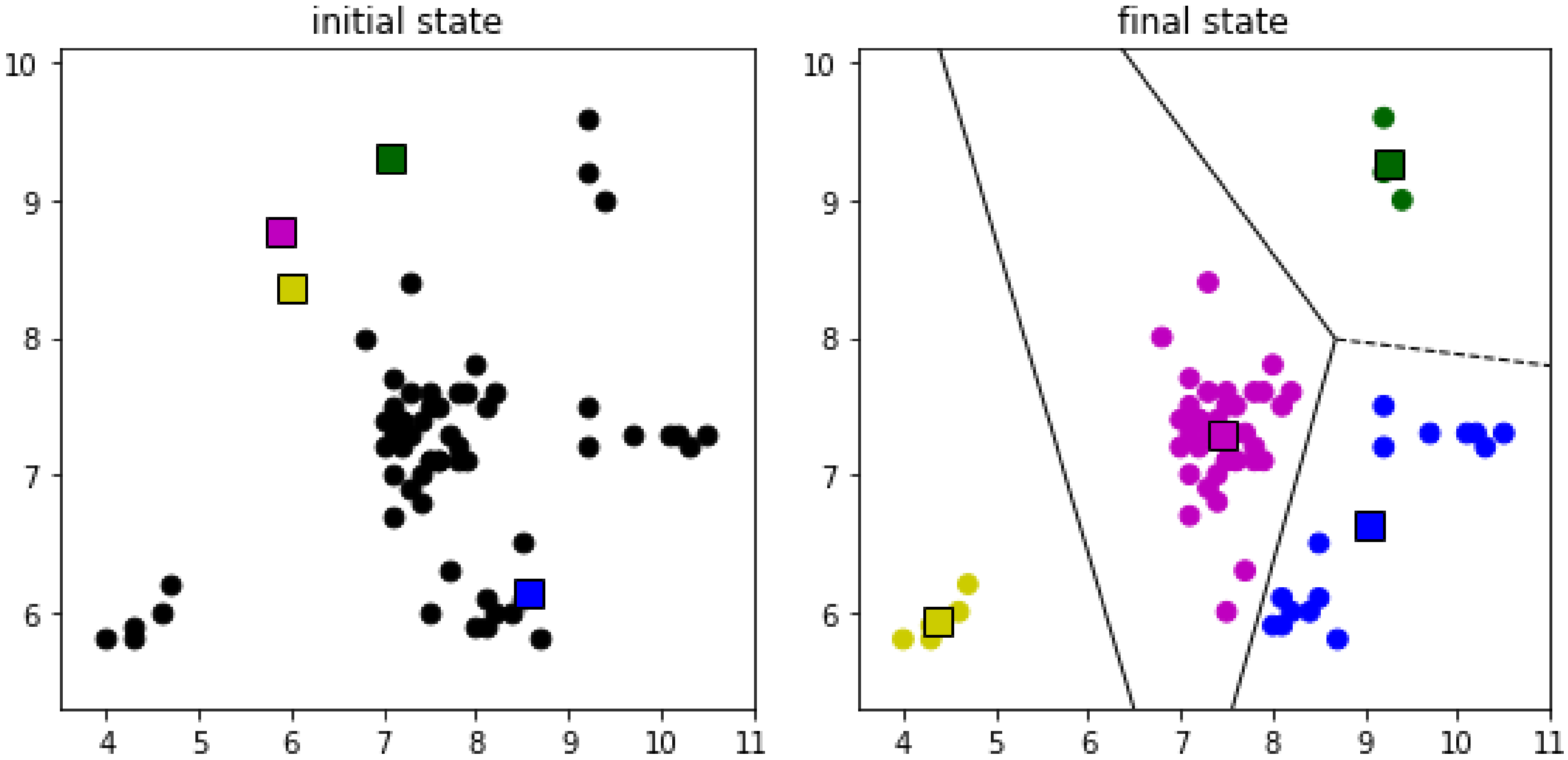
Simple idea: alternating update of the means and the resulting clustering

**Data:** $X = \{x_1, \dots, x_N\}$ and number of clusters $K \in \mathbb{N}$
**Result:** cluster means $\underline{m} = (m_1, \dots, m_K)$

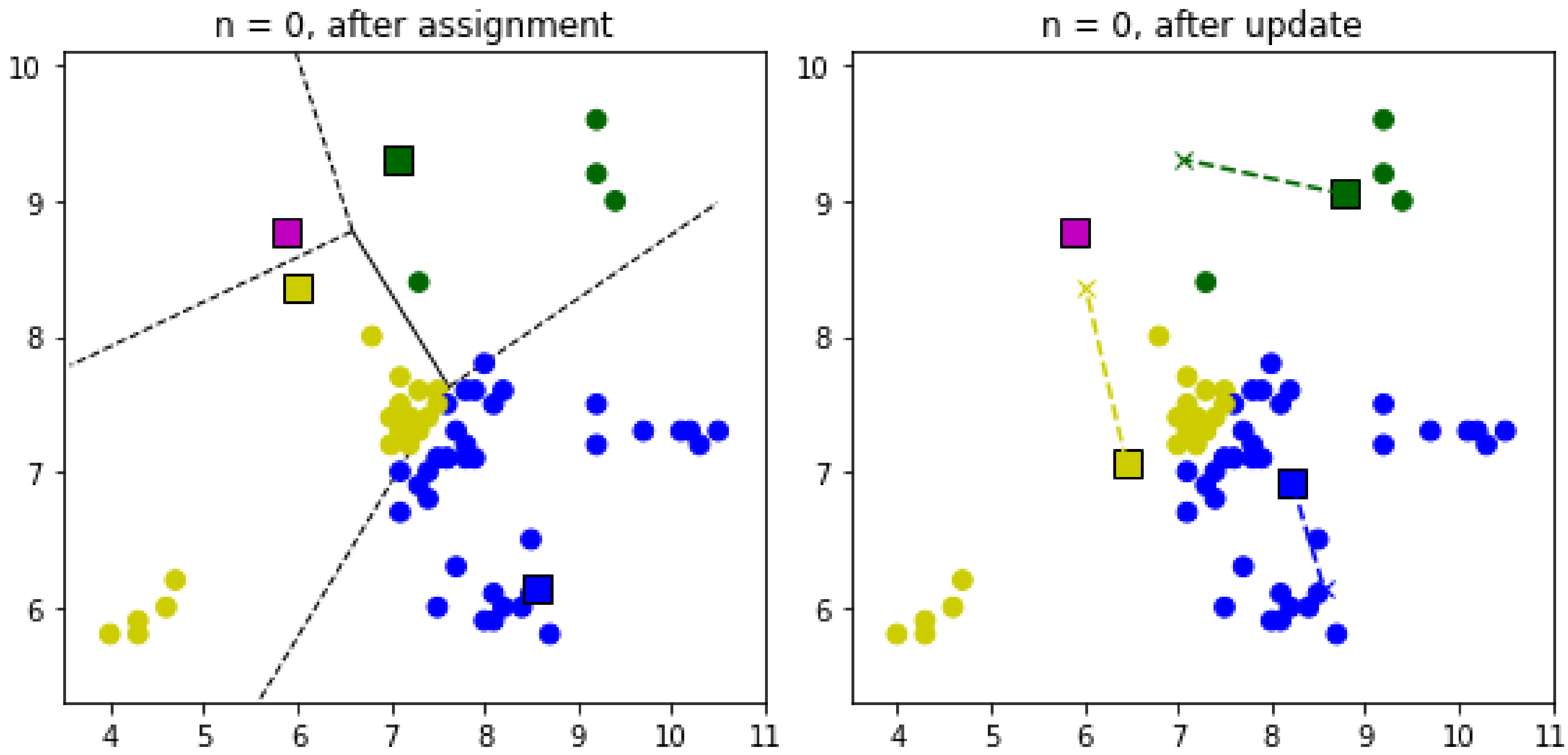
```
initialize $\underline{m}$ randomly;
repeat
    // assignment step:
    for $n \leftarrow 1$ to $N$ // assign n-th point to cluster with nearest mean do
        $k_n \leftarrow \operatorname{argmin}_k \|x_n - m_k\|$
    end
    // update step:
    for $k \leftarrow 1$ to $K$ do
        $C_k \leftarrow \{n \in \{1, \dots, N\} : k_n = k\}$ // cluster
        if $|C_k| > 0$ then
            $m_k \leftarrow \frac{1}{|C_k|} \sum_{n \in C_k} x_n$ // cluster mean
        end
    end
until assignment step does not do anything;
```

- Assignment rule: $\mathbf{x} \mapsto \operatorname{argmin}_k \|x - m_k\|$.
- Reconstruction rule: $k \mapsto m_k$

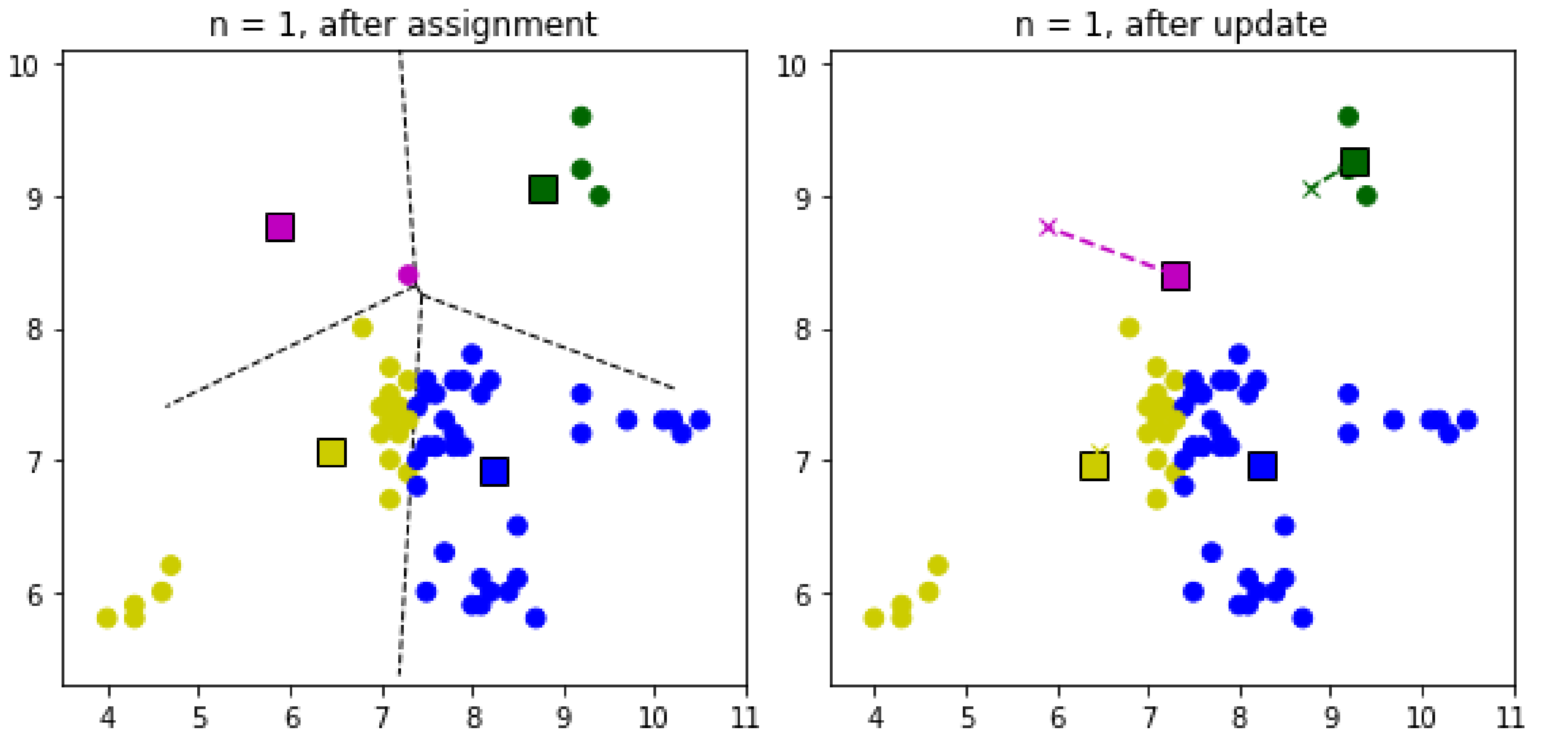
# Back to our fruits data set



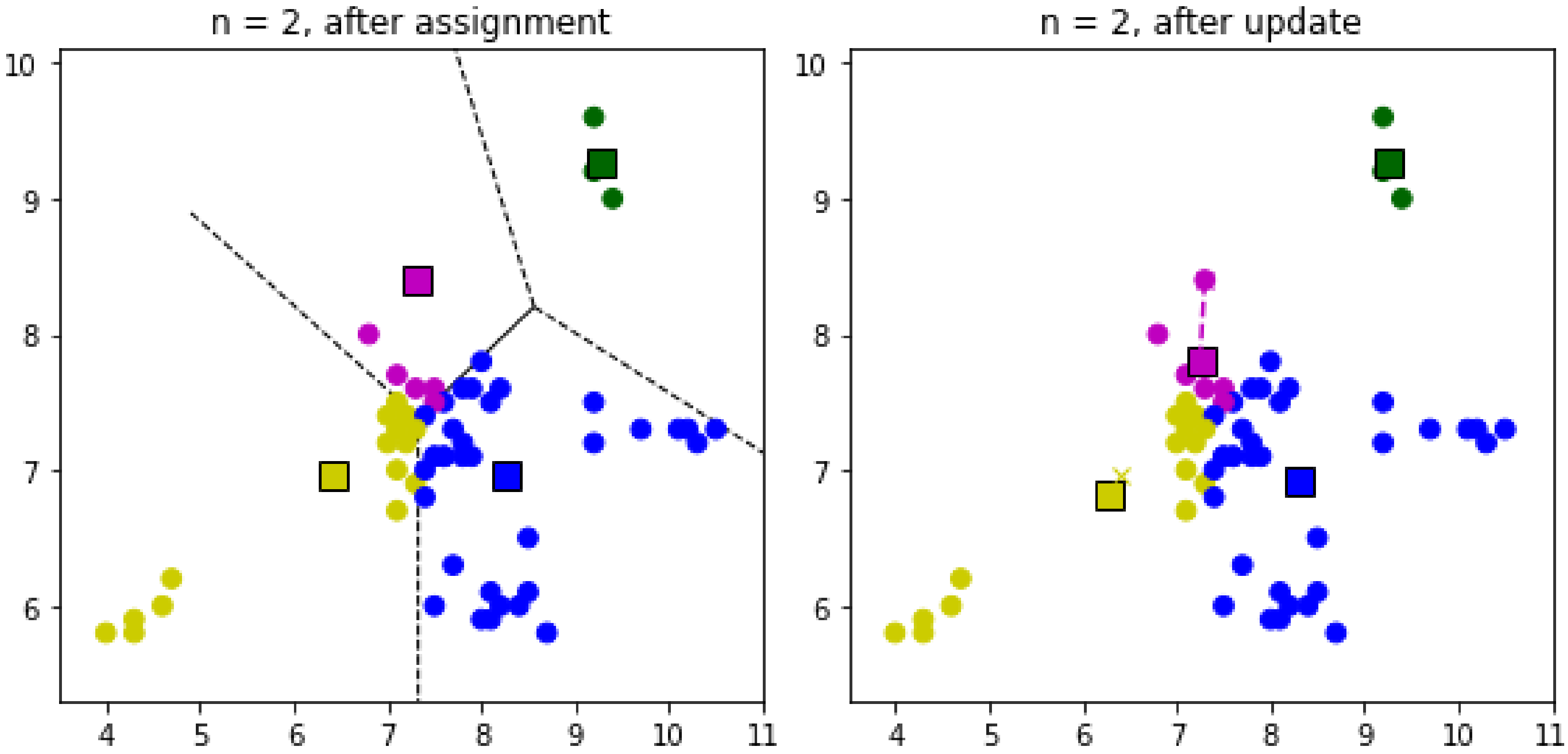
# Back to our fruits data set



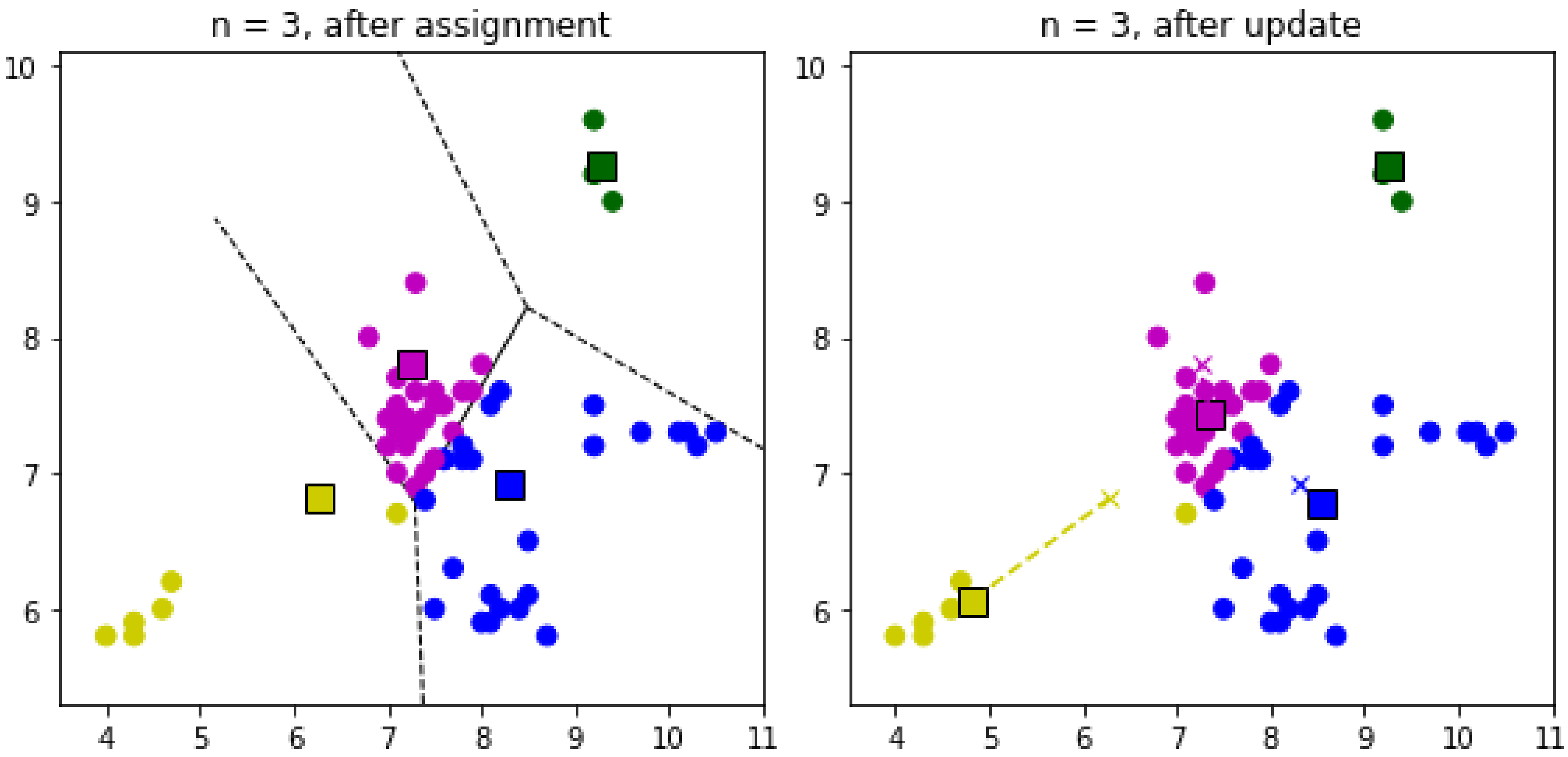
# Back to our fruits data set



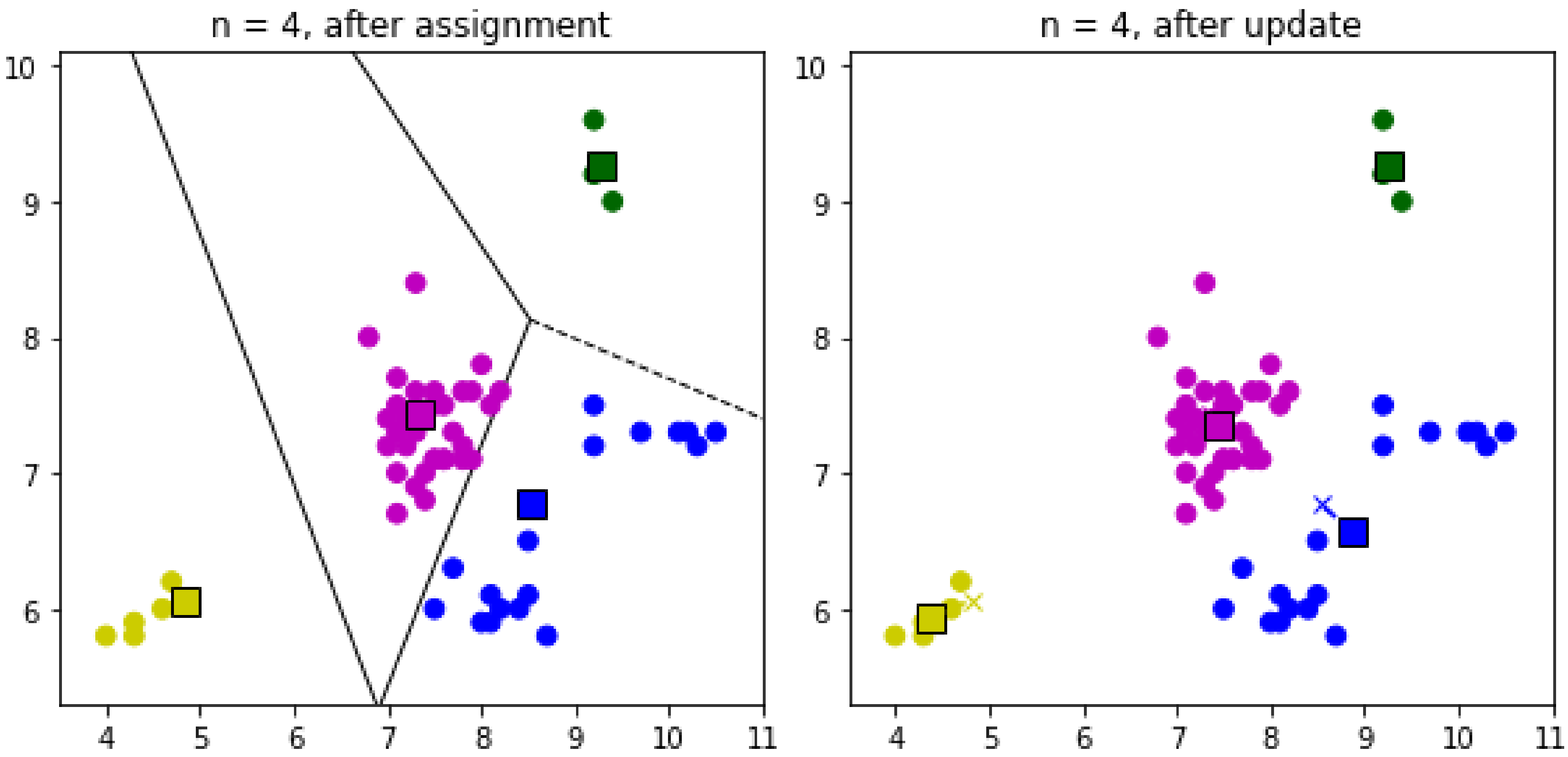
# Back to our fruits data set



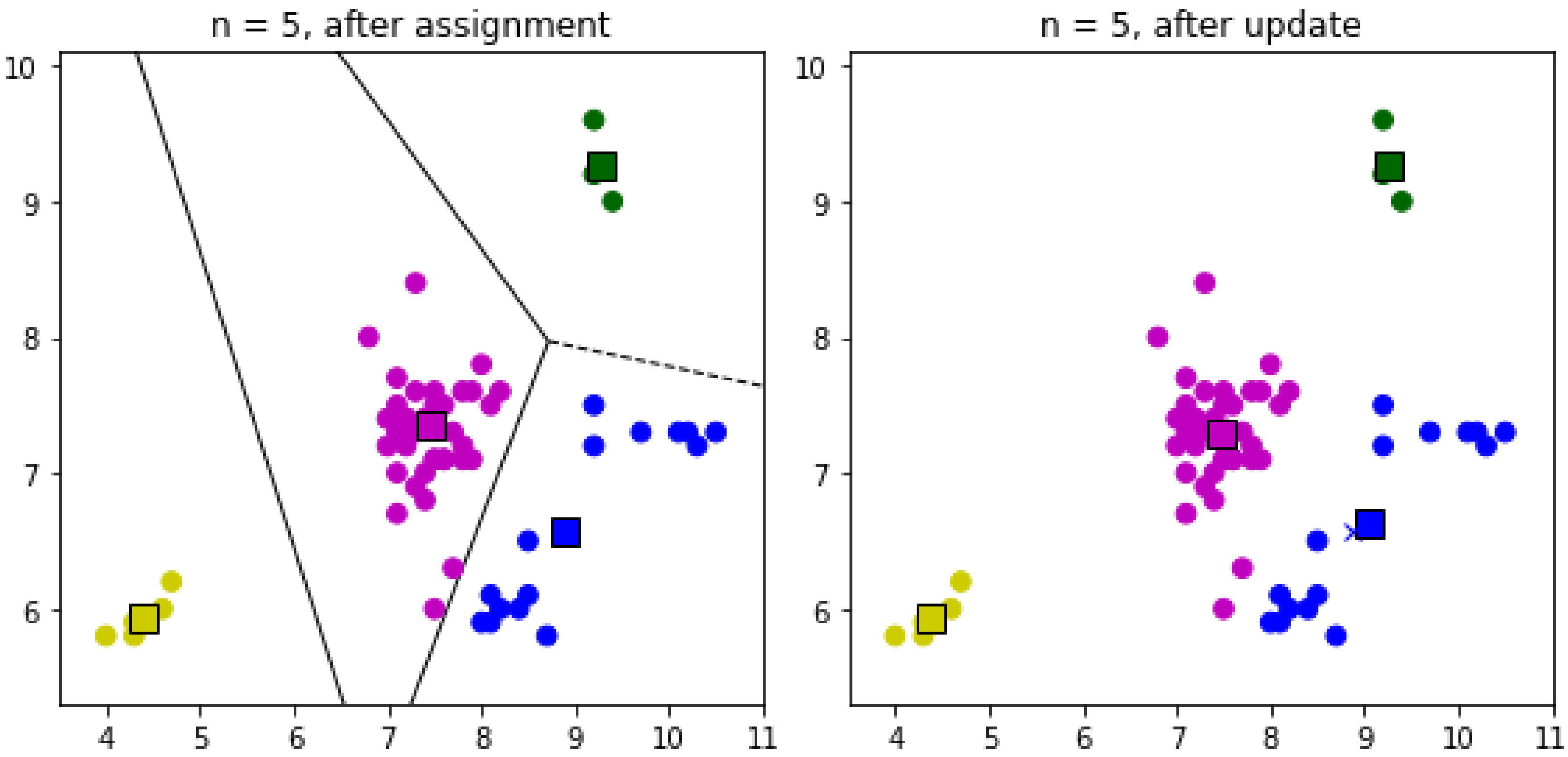
# Back to our fruits data set



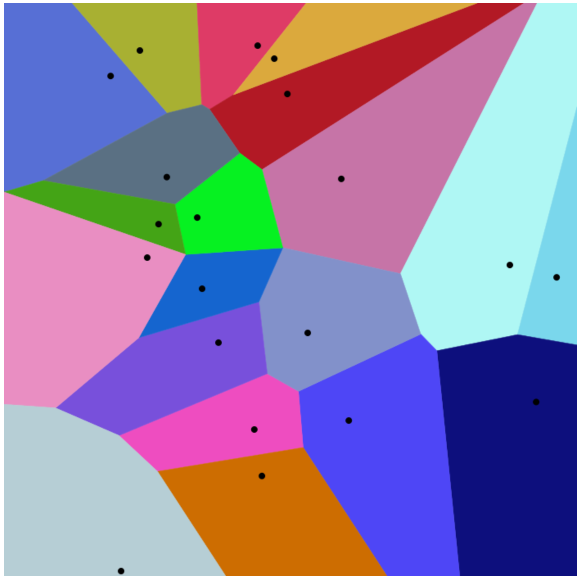
# Back to our fruits data set



# Back to our fruits data set



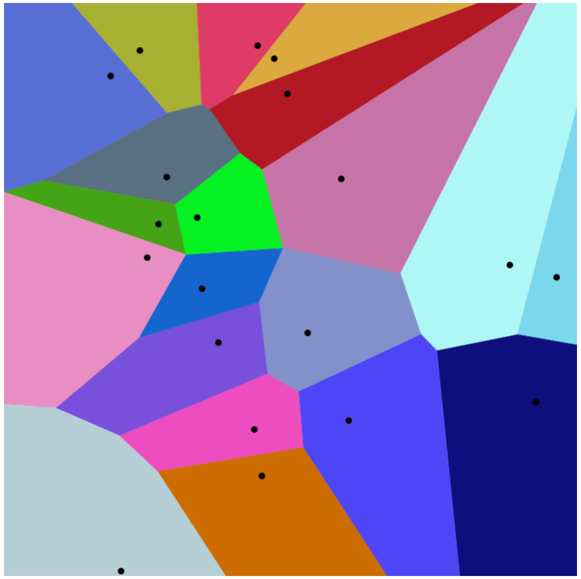
# Different metrics lead to different (convex) clusters



using squared Euclidean distance ($L_2$-norm):

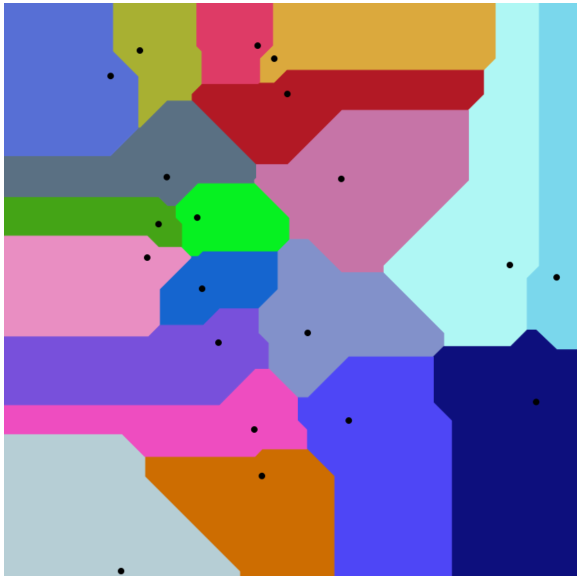
$$E(\underline{C}, \underline{m}) = \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2$$

# Different metrics lead to different (convex) clusters



using squared Euclidean distance ($L_2$-norm):

$$E(\underline{C}, \underline{m}) = \frac{1}{2} \sum_{k=1}^{K} \sum_{x \in C_k} \|x - m_k\|^2$$



using Manhattan distance ($L_1$-norm):

$$E(\underline{C}, \underline{m}) = \sum_{k=1}^{K} \sum_{x \in C_k} |x - m_k|$$

# Advantages and Disadvantages of K-Means

It is a very well known clustering algorithm that is used often.

*Advantages:*

- easy to implement
- can run with only a set of real data vectors and a given value of $K$
- A feasible clustering is always available.

# Advantages and Disadvantages of K-Means

It is a very well known clustering algorithm that is used often.

*Advantages:*

- easy to implement
- can run with only a set of real data vectors and a given value of $K$
- A feasible clustering is always available.

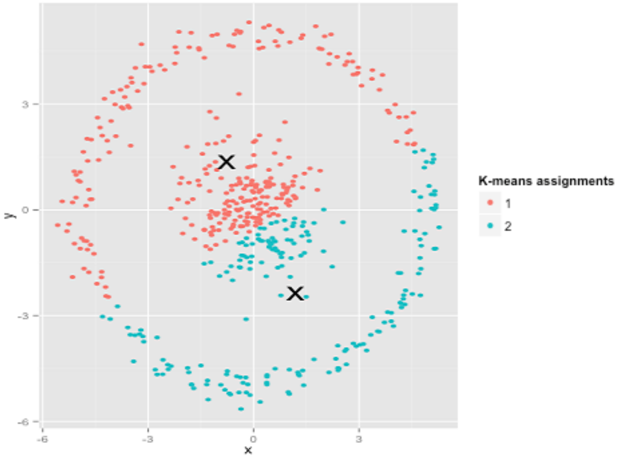
*Disadvantages:*

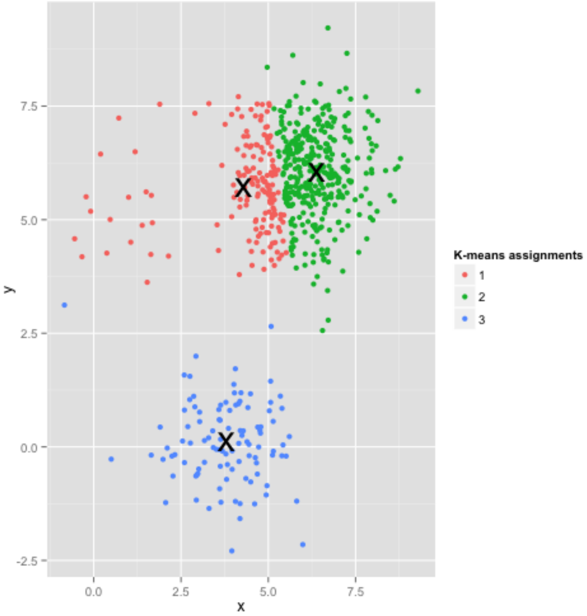
- Choosing a good value of $K$ can be difficult.
  Potential way out: test several values.
- Assumes numerical data, not categorical data such as ’car’, ’truck’, etc.
  We will see a clustering approach for categorical data later.
- K-means aims at minimizing the Euclidean distances.
  This is not always the right objective.
- The result strongly depends on initialization due to local optima,
  but some improvements are known (such as repeated random initialization).
- Assumes that clusters are *convex*.

# Advantages and Disadvantages of K-Means

*Disadvantages:*

- K-means sometimes does not work well,
  in particular for non-spherical/non-convex data or for unevenly sized clusters.
  That is, it has some implicit assumptions:





from varianceexplained.org

next: some improvements.

# Expectation-Maximization Clustering Algorithm (EM)

- Further reading: The Elements of Statistical Learning, Chapter 14
- Recall K-means has implicit assumptions (clusters are convex & roughly equally sized) that may not be satisfied.
- Alternative approach: Decide for each data point the probability with which it belongs to a certain cluster, i.e. a ’soft’ clustering. Allows clusters of different size, can detect correlations
- Problem: This probability distribution is unknown.
- Task: Estimate probability distribution, improve estimate iteratively.

# Mixture of Gaussian Distributions

- Make a quite general assumption: This unknown distribution is a mixture, i.e. a superposition, of $K$ (multi-dimensional) Gaussian distributions. This means that the probability function is of the form

$$f(x; \underline{p}, \underline{\mu}, \underline{\Sigma}) = \sum_{k=1}^{K} p_k \cdot \mathcal{N}(x; \mu_k, \Sigma_k)$$

- with

$$\begin{aligned} p &= (p_1, \dots, p_K), & p_k &\in \mathbb{R} \text{ probability vector} \\ \mu &= (\mu_1, \dots, \mu_K), & \mu_k &\in \mathbb{R}^n \text{ vector of means} \\ \Sigma &= (\Sigma_1, \dots, \Sigma_K), & \Sigma_k &\in \mathbb{R}^{n \times n} \text{ covariance matrix} \end{aligned}$$

where $M$ is the dimension of a data point $x$, i.e. $x \in \mathbb{R}^M$.
Recall Gaussian distribution in dimension $M$ with mean vector $\mu \in \mathbb{R}^M$, variance $\Sigma \in \mathbb{R}^{M \times M}$ : $\mathcal{N}(x; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^M \det(\Sigma)}} \exp(-\frac{1}{2}(x - \mu)^\top \Sigma^{-1}(x - \mu))$

# Recall Covariance Matrix

Let $x = (x_1, \dots, x_M)^\top$ be a random vector with finite variance and mean. The covariance matrix $\Sigma = (\Sigma_{x_i,x_j}) \in \mathbb{R}^{M \times M}$ is defined as

$$\Sigma_{x_i,x_j} = E((x_i - E(x_i))(x_j - E(x_j))),$$

where $E$ denotes the expected value.

The covariance matrix

- represents important statistical information, in particular correlation between data
- is a real, quadratic, symmetric matrix
- is positive-semidefinite matrix

# More Details on Mixture of Gaussians

$$f(x; \underline{p}, \underline{\mu}, \underline{\Sigma}) = \sum_{k=1}^{K} p_k \cdot \mathcal{N}(x; \mu_k, \Sigma_k)$$

Clustering task: Given data points, estimate $\underline{p}, \underline{\mu}, \underline{\Sigma}$.

Output: Yields for each data point *n* and for each cluster *k* an estimated probability that *n* was sampled from *k*.

Assign each data point the mean with highest probability.

Advantage: also enables clusters that are overlapping:

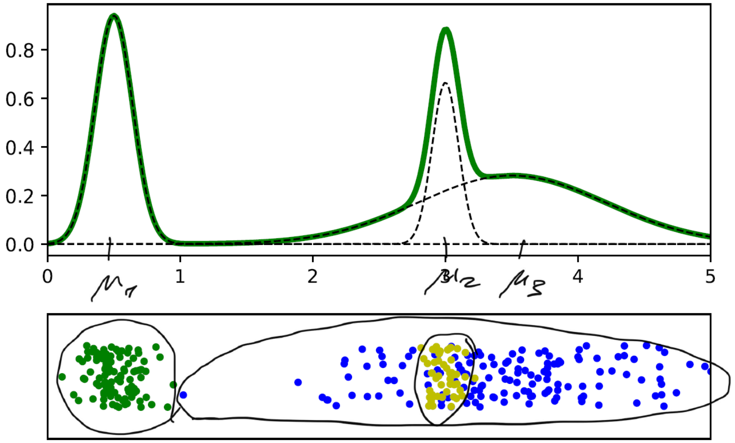


Figure: $K = 3$, Gaussian Mixture in 1d and generated data.

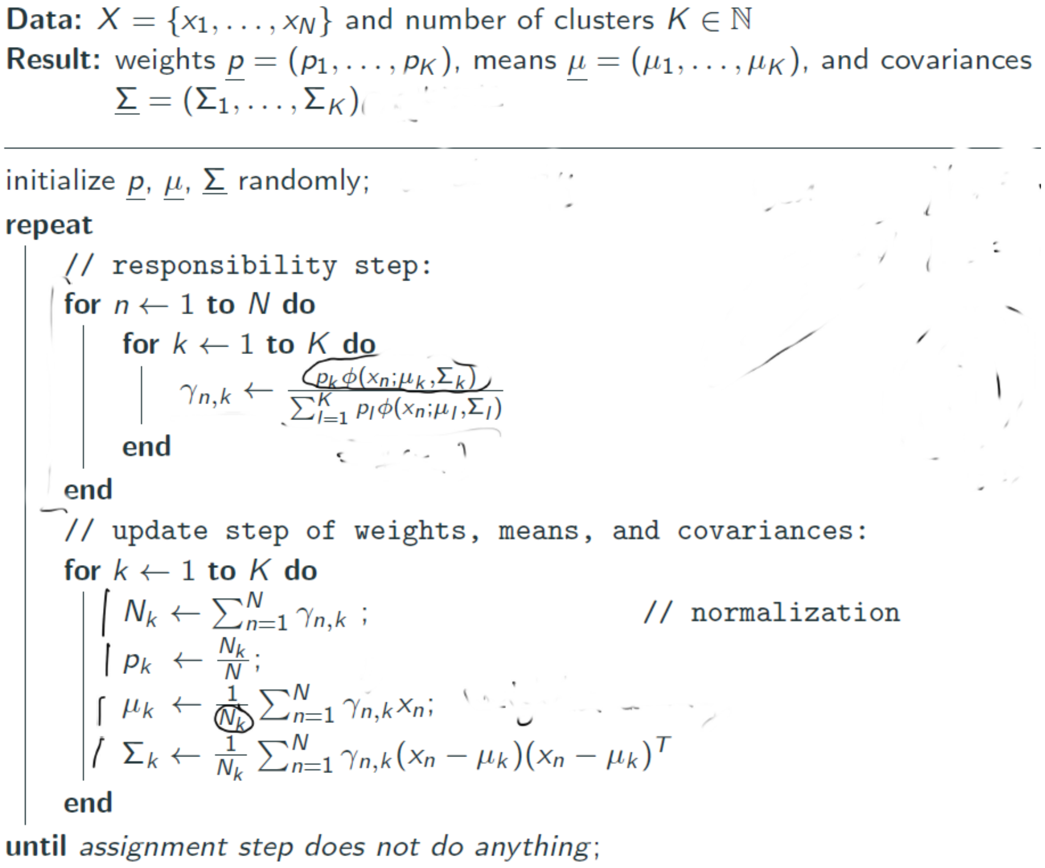
# Responsibility Calculation

Define the *responsibility of cluster $k$ for $x$* by observed relative frequencies, i.e. the probability that $x$ was sampled by Gaussian $k$.

$$\gamma(x;k) = \frac{p_k \mathcal{N}(x;\mu_k,\Sigma_k)}{\sum_{j=1}^{K} p_j \mathcal{N}(x;\mu_j,\Sigma_j)}$$

Let the density function of the (standard) normal distribution be denoted by $\phi$.

# Outline of Expectation-Maximization Clustering

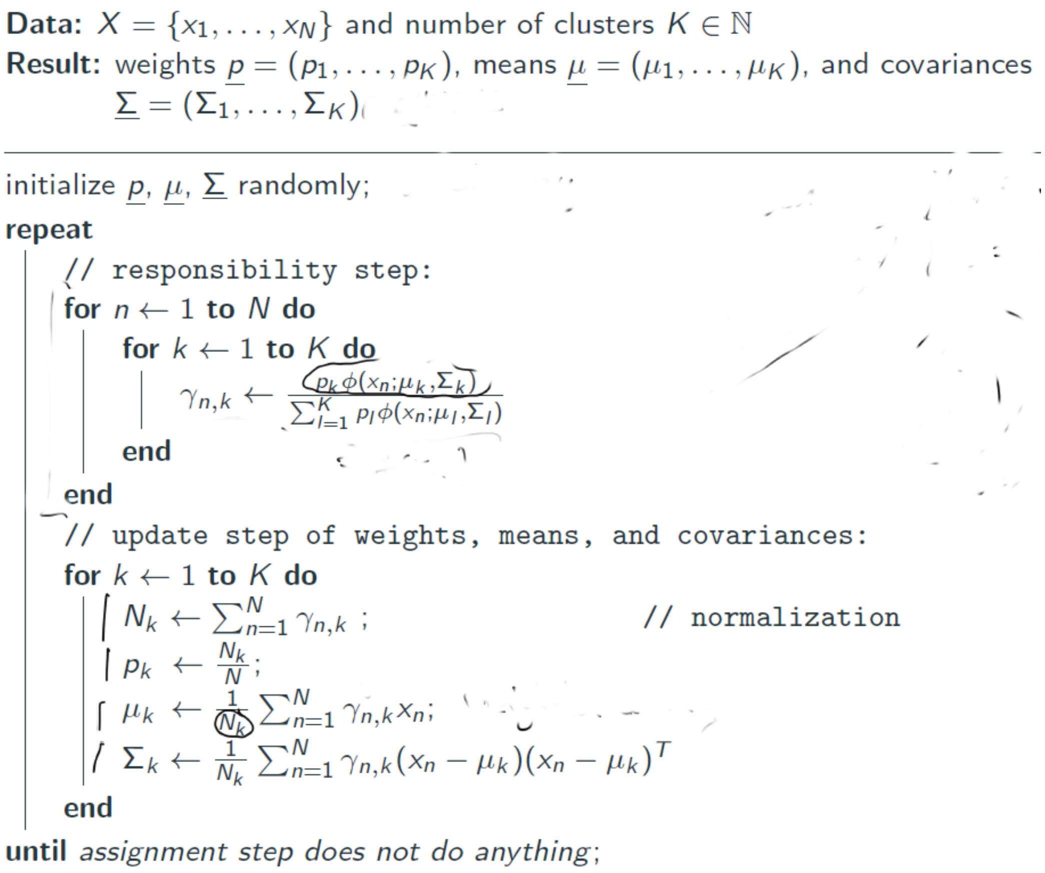


**Data:** $X = \{x_1, \ldots, x_N\}$ and number of clusters $K \in \mathbb{N}$

**Result:** weights $\underline{p} = (p_1, \ldots, p_K)$, means $\underline{\mu} = (\mu_1, \ldots, \mu_K)$, and covariances $\underline{\Sigma} = (\Sigma_1, \ldots, \Sigma_K)$

```
initialize $\underline{p}$, $\underline{\mu}$, $\underline{\Sigma}$ randomly;
repeat
    // responsibility step:
    for $n \leftarrow 1$ to $N$ do
        for $k \leftarrow 1$ to $K$ do
            $\gamma_{n,k} \leftarrow \frac{p_k \phi(x_n; \mu_k, \Sigma_k)}{\sum_{l=1}^{K} p_l \phi(x_n; \mu_l, \Sigma_l)}$
        end
    end
    // update step of weights, means, and covariances:
    for $k \leftarrow 1$ to $K$ do
        $N_k \leftarrow \sum_{n=1}^{N} \gamma_{n,k}$ ;                // normalization
        $p_k \leftarrow \frac{N_k}{N}$;
        $\mu_k \leftarrow \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{n,k} x_n$;
        $\Sigma_k \leftarrow \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{n,k} (x_n - \mu_k)(x_n - \mu_k)^T$
    end
until assignment step does not do anything;
```

# Outline of Expectation-Maximization Clustering

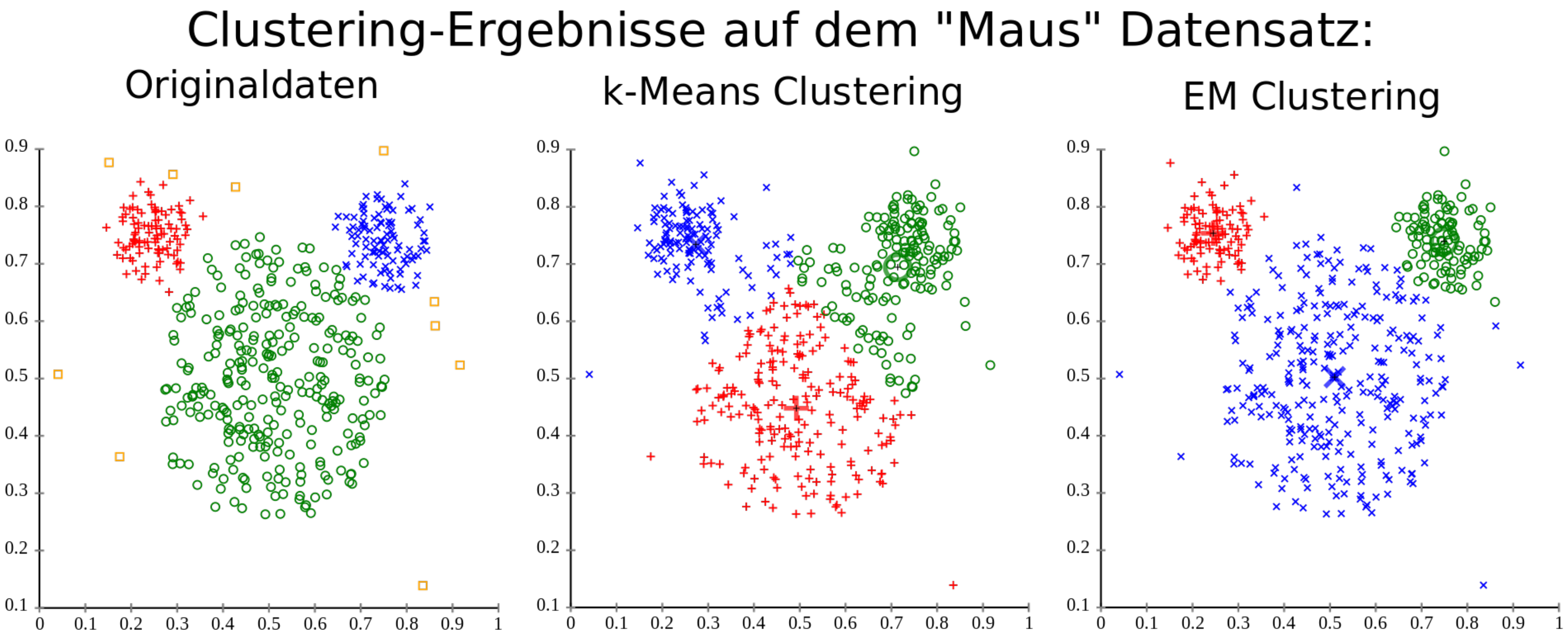


**Data:** $X = \{x_1, \dots, x_N\}$ and number of clusters $K \in \mathbb{N}$
**Result:** weights $\underline{p} = (p_1, \dots, p_K)$, means $\underline{\mu} = (\mu_1, \dots, \mu_K)$, and covariances $\underline{\Sigma} = (\Sigma_1, \dots, \Sigma_K)$

```
initialize $\underline{p}$, $\underline{\mu}$, $\underline{\Sigma}$ randomly;
repeat
    // responsibility step:
    for $n \leftarrow 1$ to $N$ do
        for $k \leftarrow 1$ to $K$ do
            $\gamma_{n,k} \leftarrow \frac{\boxed{p_k \phi(x_n; \mu_k, \Sigma_k)}}{\sum_{l=1}^{K} p_l \phi(x_n; \mu_l, \Sigma_l)}$
        end
    end
    // update step of weights, means, and covariances:
    for $k \leftarrow 1$ to $K$ do
        $N_k \leftarrow \sum_{n=1}^{N} \gamma_{n,k}$ ;                // normalization
        $p_k \leftarrow \frac{N_k}{N}$;
        $\mu_k \leftarrow \frac{1}{\boxed{N_k}} \sum_{n=1}^{N} \gamma_{n,k} x_n$;
        $\Sigma_k \leftarrow \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{n,k} (x_n - \mu_k)(x_n - \mu_k)^T$
    end
until assignment step does not do anything;
```

Update step takes data points into account via relative frequencies.
Indeed, K-means can be seen as a special case of EM.

# A visual comparison



Drawback: **K-means and EM only find local optima.**
(recall already K-means problem is NP-hard...)
play around with skikit-learn (machine learning library in python)

# Hierarchical Clustering Methods

- Sometimes, there is some hierarchical structure in the data that we want to disclose in a clustering.
- In hierarchical methods, we do not need to specify the number of clusters $K$ as input
- We need a measure of dissimilarity, e.g., ’distance’ between (disjoint) groups of observations. This is typically based on pairwise dissimilarities.
- Clusters at each levels of hierarchy are created by merging clusters at next lower level.
- The lowest level contains one point each, highest level contains all points.
- Both agglomerative (bottom-up) or divisive (top-down) methods exist.

# Hierarchical Clustering Methods

- Idea of agglomerative approach: start at bottom. At each level, recursively merge a selected pair of clusters into a single cluster. Next higher level contains one cluster less. Merge those two clusters with the smallest dissimilarity. $\Rightarrow$ This leads to $N - 1$ levels in hierarchy.
- Clusterings are often drawn by a rooted binary tree. It contains a node as root. Each node has not more than two children in the tree.

# Visualization as a Dendrogram

from statistical learning book:

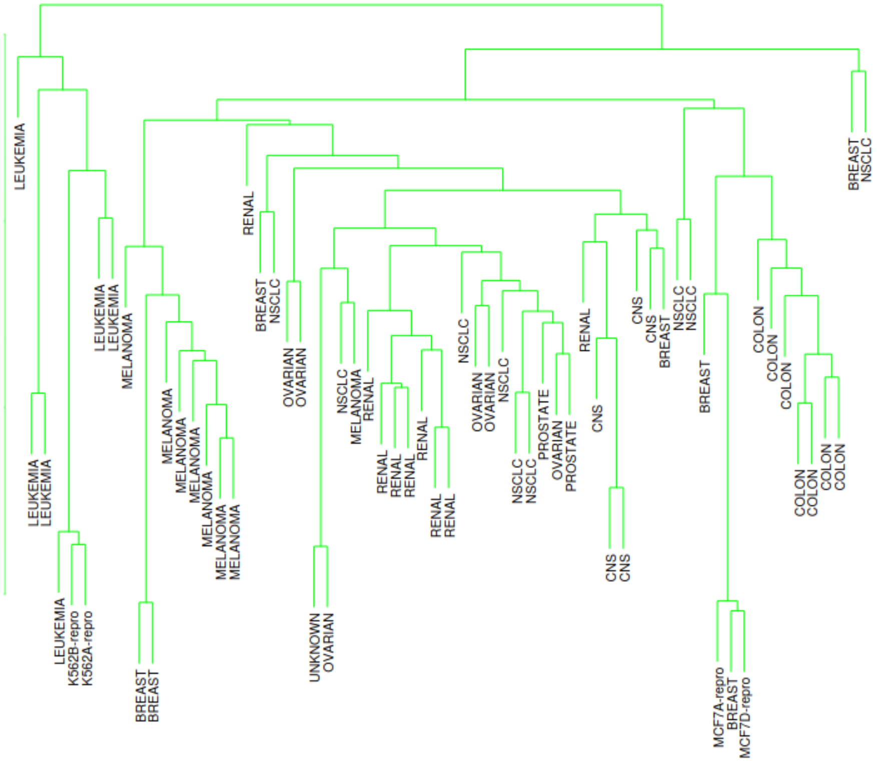


**FIGURE 14.12.** *Dendrogram from agglomerative hierarchical clustering with average linkage to the human tumor microarray data.*

# Dendrogram

- A word of caution: small changes in the data can lead to quite different dendrograms
- For a clustering application, we need to decide whether the hierarchical structure is actually intrinsic to the data or not.

# Clustering in More General Contexts

How can we compute (pairwise) distances in a clustering algorithm? ...for example in ordinal or in categorial contexts? E.g. Suppose we have measurements or averages over personal judgements by participants who are asked to judge differences between objects.
Often, one uses *dissimilarities* based on attributes for the distance calculation in a clustering algorithm.
Dissimilarities are calculated as follows:

- Suppose we have for objects $i = 1, \dots, N$ measurements $x_{ij}$ for variables (attributes) $j = 1, \dots, p$.
- We define dissimilarity between objects $i$ and $i'$ as

$$D(x_i, x_{i'}) = \sum_{j=1}^{p} d_j(x_{ij}, x_{ij'})$$

- In this definition, different attributes could also be weighted differently.

# Alternatives for Calculating Distances

Depending on the variable types, distances need to be calculated differently.

- Ordinal variables: e.g. contiguous integers, ordered sets such as academic grades (A, B, C, D, F), degree of preference (can't stand, dislike, OK, like, terrific). Dissimilarities for ordinal variables are typically defined by replacing their $M$ original values with

$$\frac{i - \frac{1}{2}}{M}, i = 1, \dots, M$$

  in the prescribed order of their original values.
- (unordered) categorical variables: Dissimilarity between pairs of values must be delineated explicitly.
  If a variable can assume $M$ distinct values, these can be arranged in a symmetric $M \times M$ matrix with $L_{rr'} = L_{r'r}, L_{rr} = 0, L_{rr'} \geq 0$. Often: $L_{rr'} = 1$ for all $r \neq r'$. Unequal losses can be used to emphasize some distances more than others.

How can we cluster data in such contexts?

# K-medoids Algorithm

As discussed before, the $K$-means clustering algorithm is designed for *numerical and quantitative* values.
Look at $K$-means again: iteratively,

1. Each data point is assigned to the cluster to which the distance *(or dissimilarity)* is minimal.
2. Then the new cluster mean is calculated.

The optimization step 1) can easily be generalized to dissimilarities stored in matrices, as studied on the last slides.

# $K$-Medoids Algorithm

1. For a given cluster assignment $C$, find the observation or attribute in the cluster that minimizes the total distance to the other points in that cluster:

   $$i_k^\star = \operatorname*{argmin}_{\{i:C(i)=k\}} \sum_{C(i')=k} D(x_i, x_{i'}).$$

   Then

   $$m_k = x_{i_k^\star}, k = 1, 2, \ldots K$$

   are the current estimates of the cluster centres.
2. Given a current set of cluster centres $\{m_1, \ldots, m_K\}$, minimize the total dissimilarity by assigning each observation to the closest (current) cluster centre:

   $$C(i) = \operatorname*{argmin}_{1 \le k \le K} D(x_i, m_k).$$
3. Iterate steps 1 and 2 until the assignments do not change.

# How to choose the number of clusters $K$?

For a clustering problem, best number of clusters depends on the goal and on the knowledge on the application.

- Sometimes, the best value of $K$ is given as input (e.g. $K$ salespeople are employed, the task is to cluster a database in $K$ many segments)
- However, if ’natural’ clusters need to be determined, the best value of $K$ is unknown and needs to be estimated from data as well.

# (Heuristic) estimate of good $K$ values

- Examine the within-cluster dissimilarity $D_{k'}$ as a function of the number of clusters $k'$.
- Separate solutions are obtained for $k' \in \{1, 2, \ldots, K_{\text{max}}\}$.
- Values $\{D_1, D_2, \ldots, D_{K_{\text{max}}}\}$ decrease with increasing $k'$.
- Intuition: if there are $K$ natural groupings, then for $k' < K$, clusters will each contain a *subset* of the true underlying groups, i.e. dissimilarity $D_{k'}$ will (noticeably) decrease for increasing $k'$.
- For $k' > K$ instead, at least one natural group will be assigned separate clusters. That is, $D_{k'}$ will decrease only mildly.
- Estimate best number of clusters $K$ by identifying a 'kink' in the plot of $D_{k'}$ as a function of $k'$.

# (Heuristic) estimate of good $K$ values

- Examine the within-cluster dissimilarity $D_{k'}$ as a function of the number of clusters $k'$.
- Separate solutions are obtained for $k' \in \{1, 2, \ldots, K_{\max}\}$.
- Values $\{D_1, D_2, \ldots, D_{K_{\max}}\}$ decrease with increasing $k'$.
- Intuition: if there are $K$ natural groupings, then for $k' < K$, clusters will each contain a *subset* of the true underlying groups, i.e. dissimilarity $D_{k'}$ will (noticeably) decrease for increasing $k'$.
- For $k' > K$ instead, at least one natural group will be assigned separate clusters. That is, $D_{k'}$ will decrease only mildly.
- Estimate best number of clusters $K$ by identifying a 'kink' in the plot of $D_{k'}$ as a function of $k'$.

Next, we will turn to a fundamental method in learning that can be used for clustering as well, and also more generally for reducing data dimensions, called *Principal Component Analysis*.