K-Means

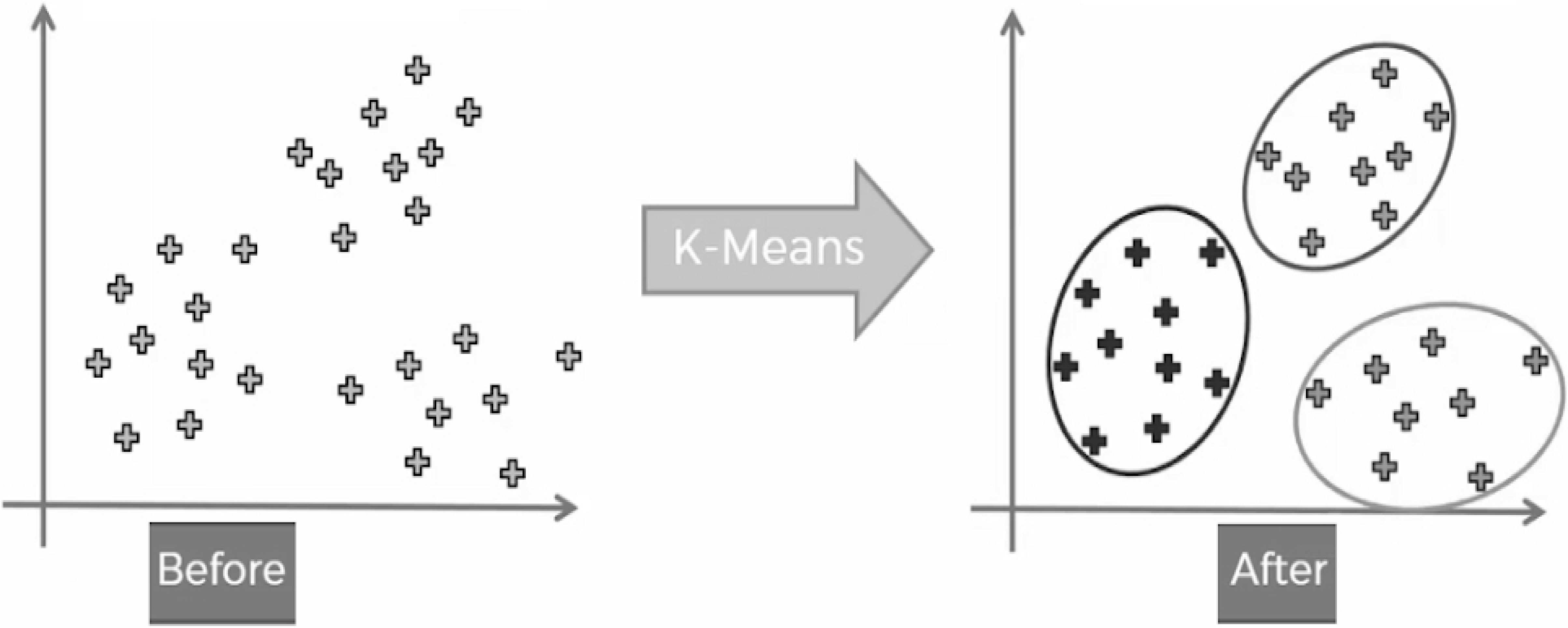
Daniel Ledda and Shanshan Bai

Vertiefung der Grundlagen der Computerlinguistik WiSe 2019/20

Overview: Clustering

- Clustering algorithms group data units in a dataset into clusters
    - The objects in one cluster are ideally more similar to each that to objects in other clusters -> similar inside, dissimilar outside
    - These are represented as mathematical partitions of the dataset

- Clustering may be supervised or unsupervised
    - Supervised learning starts with hypotheses and creates groups that new data is assigned to
    - Unsupervised learning “uncovers” them in the data that is already present


Before
K-Means
After

Purpose and Application

- Clustering algorithms help provide insight into data
    - Marketing: Who are my customers?
    - Downsampling pictures (-> “downsampling” text?)
    - Clustering of gene expression data
    - Web: who uses what? What’s out there?
    - NLP:
        - What document categories exist?
        - What kind of person wrote this text?
        - What sentiment is expressed by this text?
        - etc.

- Sometimes datasets need to be clustered for preprocessing
    - Moving from unsupervised to supervised learning

Clustering Approaches

- Partitioning algorithms create a mathematical partition between sets of data in order to classify new data
  - K-Means is one such algorithm
  - Seek to minimise an objective function during execution to find the best partition of the data

- Others;
  - EM: Expectation Maximisation
  - Hierarchical Methods
  - Machine learning
  - etc.

Basic Idea

- Choose **k** $\in \mathbb{N}$ representatives (e.g. randomly) of the **k** different clusters.
  - These are “fake” datapoints -> not real data
- **Assign data** to nearest cluster $\mathbf{C_i}$
- We need to **improve** these representative points **C** to reduce the value of our objective function
- **Reassign the data** to the updated points **C**
- Is our objective function small enough?
  - Yes -> Stop
  - No -> Keep modifying our cluster representatives

K-Means Implementation

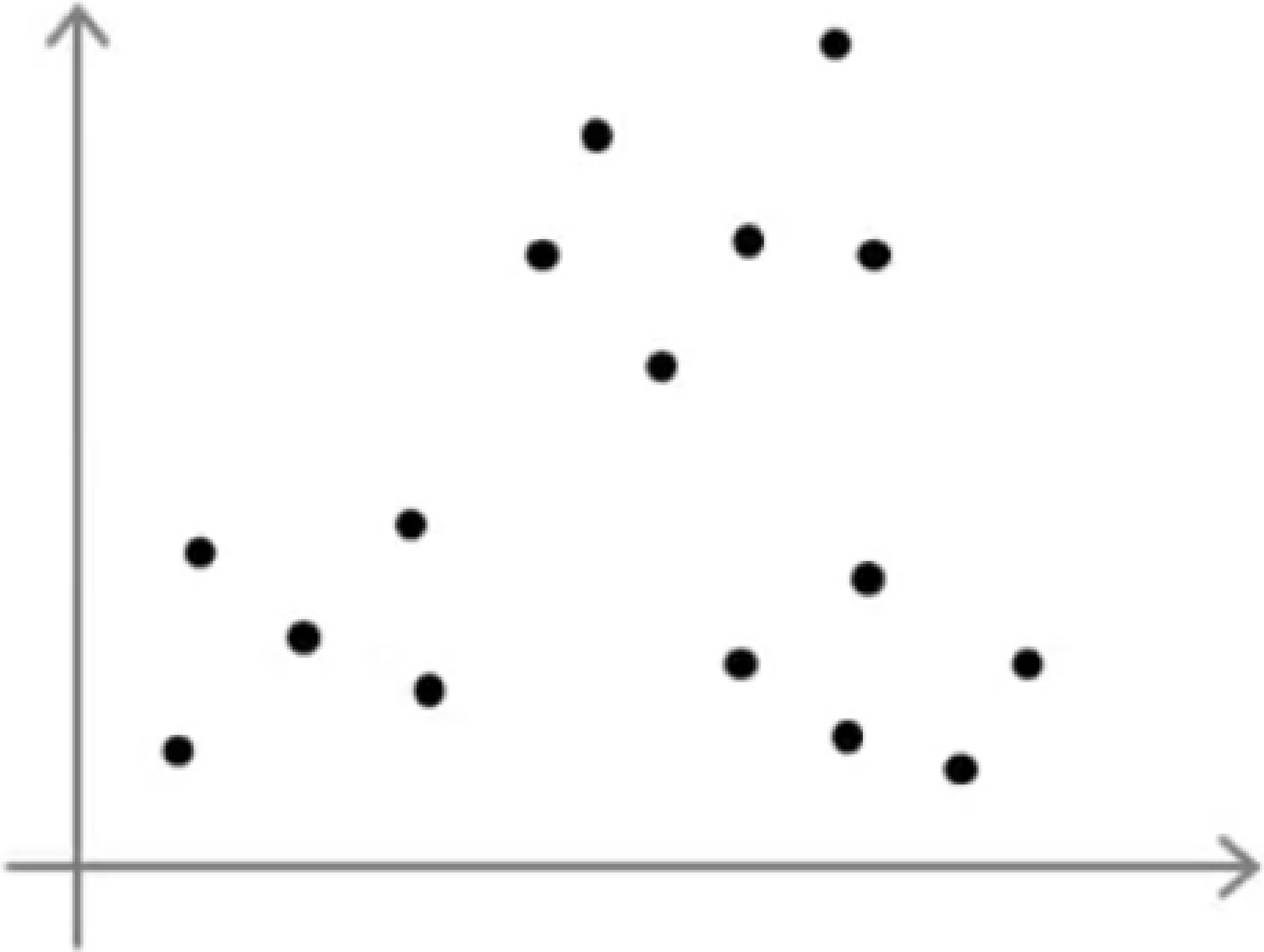
- The objective function for K-Means is defined as follows:

$$SSE(C_j) = \sum_{p \in C_j} ||p - \mu_{C_j}||_2^2$$

  - With $\boldsymbol{C_j}$ = one cluster group and μ the representative datapoint (“mean” of the cluster)

- Once all points have been reassigned, the means are recalculated.
- Points are then reassigned to the new nearest mean $\boldsymbol{C_j}$

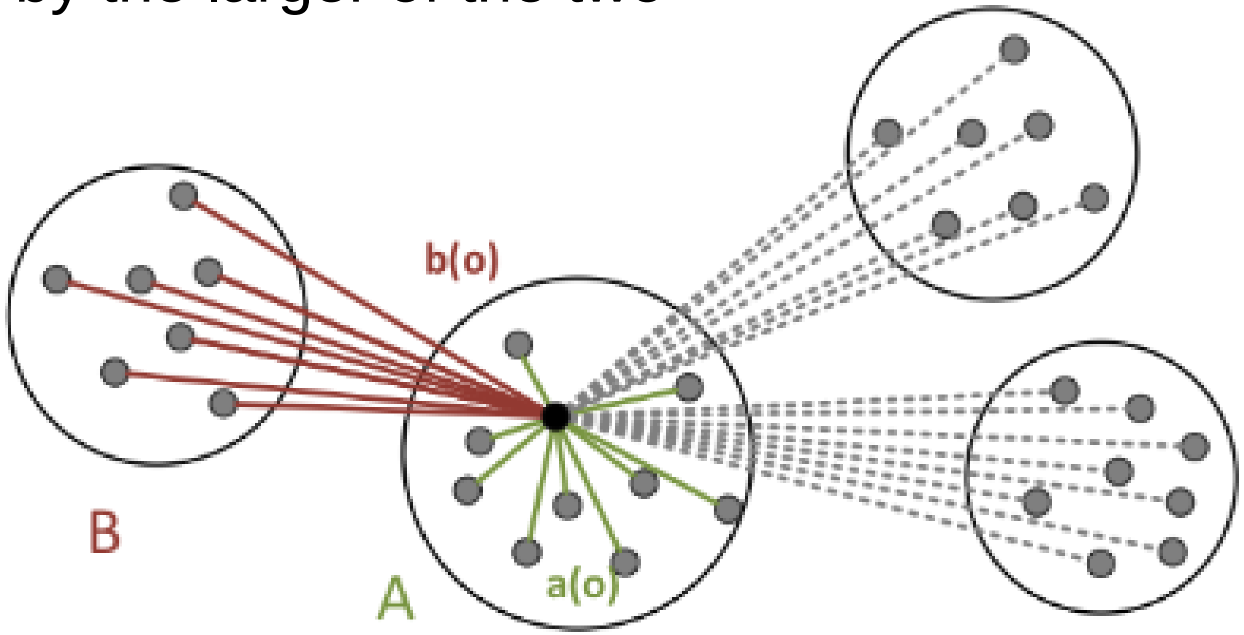
Toy Example



How do I know if my clustering is any good?

- One technique involves determining how well each data point as been assigned:
  - Sillhouette Coefficient
- The average of the lines in b(o) is ideally much larger than the average of the green ones, a(o)
  - The difference is then normalised by the larger of the two

$$s(o) = \begin{cases} 0 & \text{if } a(o) = 0 \\ \frac{b(o)-a(o)}{max(a(o),b(o))} & \text{else} \end{cases}$$



dataset with 10 clusters

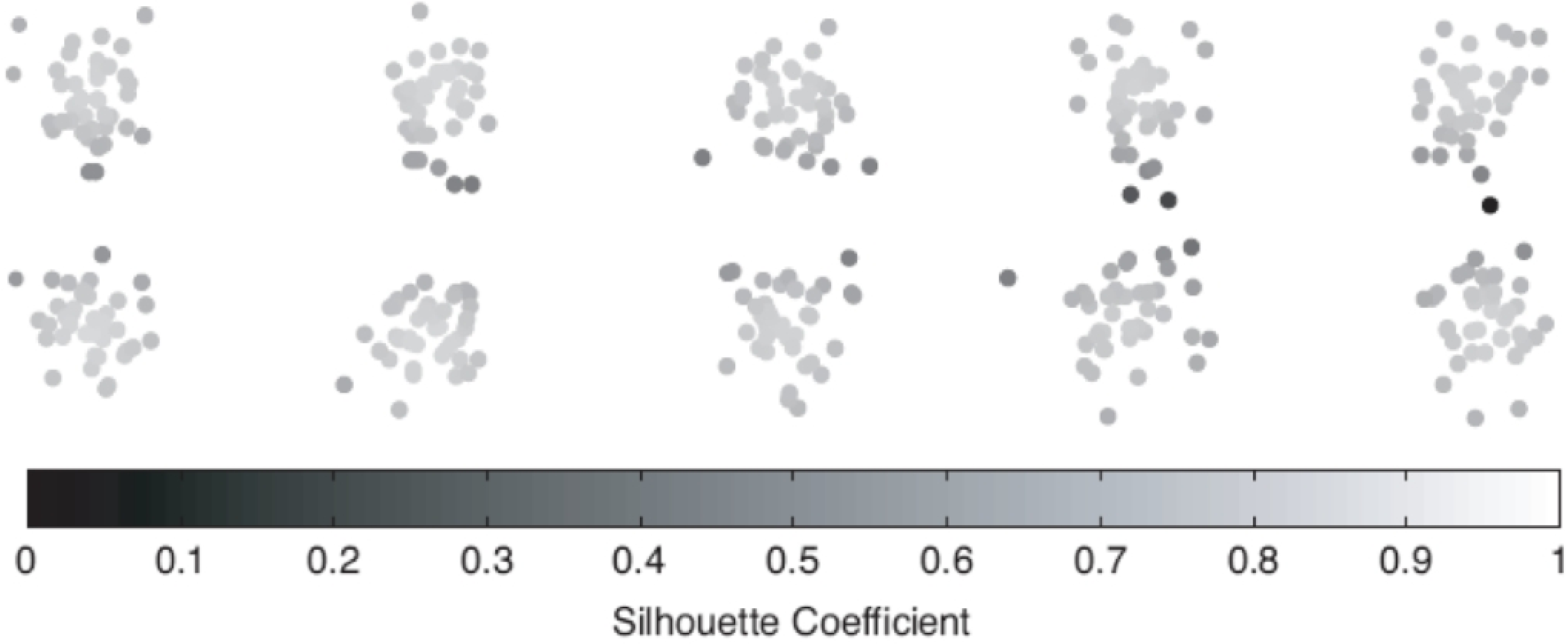


Image from Tan, Steinbach, Kumar: Introduction to Data Mining (Pearson, 2006)

K-Means Algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ $\quad x^{(i)} \in \mathbb{R}^n$

K-Means Algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ $x^{(i)} \in \mathbb{R}^n$

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

for i = 1 to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

for k = 1 to K

μ_k := average (mean) of points assigned to cluster k

}

K-Means Algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ $\quad x^{(i)} \in \mathbb{R}^n$

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

cluster assignment step

for i = 1 to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

for k = 1 to K

μ_k := average (mean) of points assigned to cluster k

}

K-Means Algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ $\quad x^{(i)} \in \mathbb{R}^n$

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

cluster assignment step

for i = 1 to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

move centroid step

for k = 1 to K

μ_k := average (mean) of points assigned to cluster k

}

K-Means Algorithm

Optimization objective:

$$J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K) = \frac{1}{m} \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2$$

$$\min_{\substack{c^{(1)}, \dots, c^{(m)}, \\ \mu_1, \dots, \mu_K}} J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$$

The square distance between each example $x^{(i)}$ and the location of the cluster centroid to which $x^{(i)}$ has been assigned

K-Means Algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ $\quad x^{(i)} \in \mathbb{R}^n$

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

cluster assignment step

for i = 1 to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

move centroid step

for k = 1 to K

μ_k := average (mean) of points assigned to cluster k

}

K-Means Algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ $\quad x^{(i)} \in \mathbb{R}^n$

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

cluster assignment step

for i = 1 to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

minimizing J(···) w.r.t $c^{(1)}, c^{(2)}, \ldots c^{(m)}$

move centroid step

for k = 1 to K

μ_k := average (mean) of points assigned to cluster k

}

K-Means Algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ $\quad x^{(i)} \in \mathbb{R}^n$

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

cluster assignment step

for i = 1 to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

minimizing J(⋯) w.r.t $c^{(1)}, c^{(2)}, \ldots c^{(m)}$

move centroid step

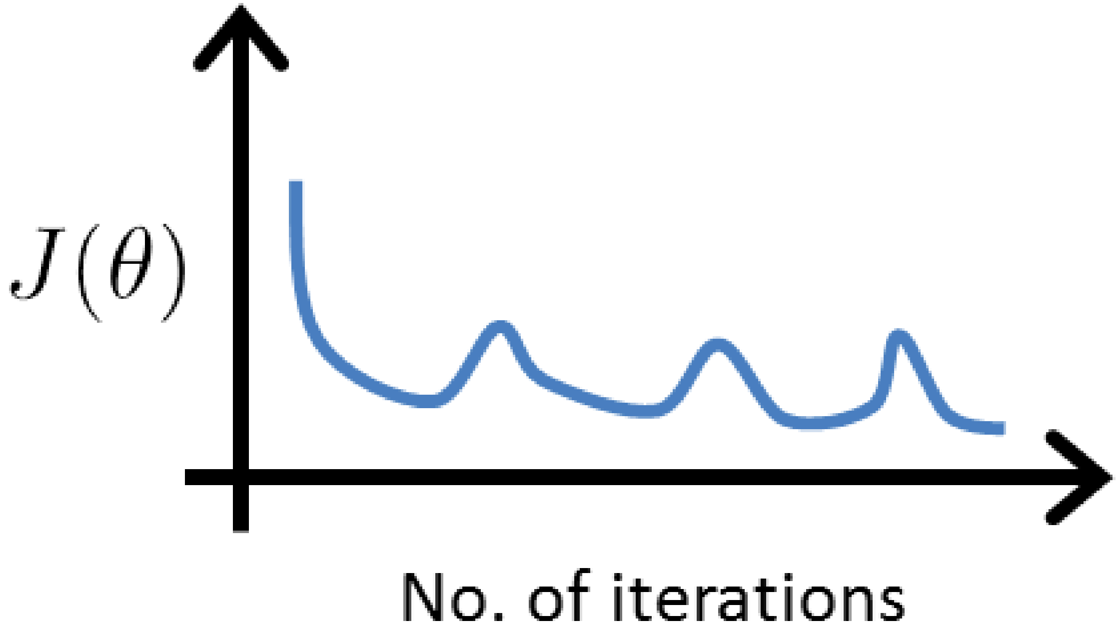
for k = 1 to K

μ_k := average (mean) of points assigned to cluster k

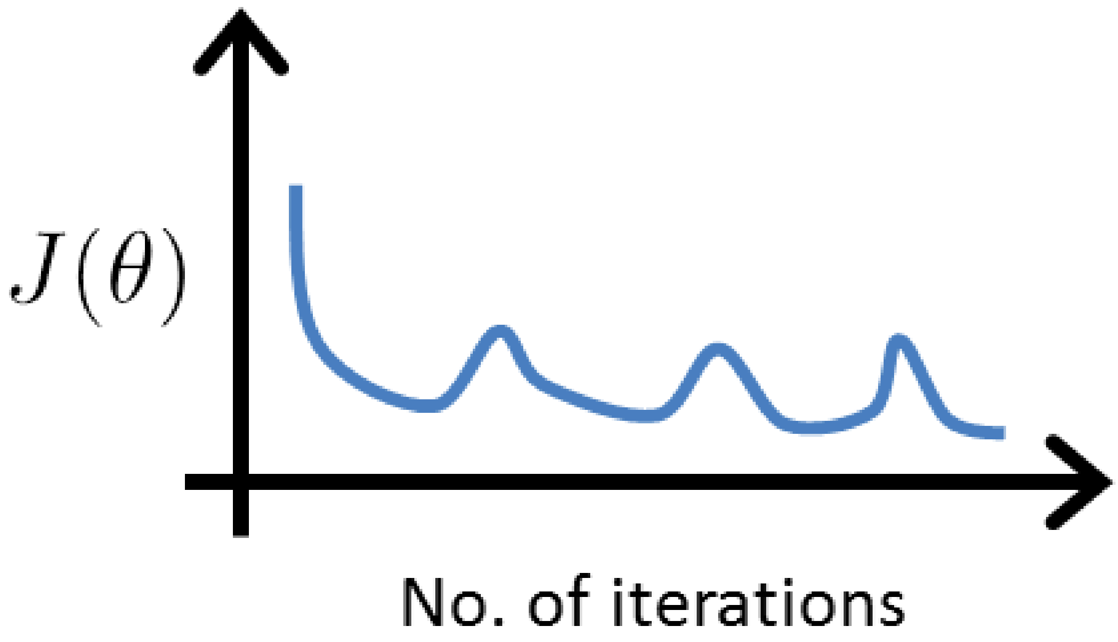
minimizing J(⋯) w.r.t $\mu_1, \mu_2, \ldots \mu_k$

}

Debug



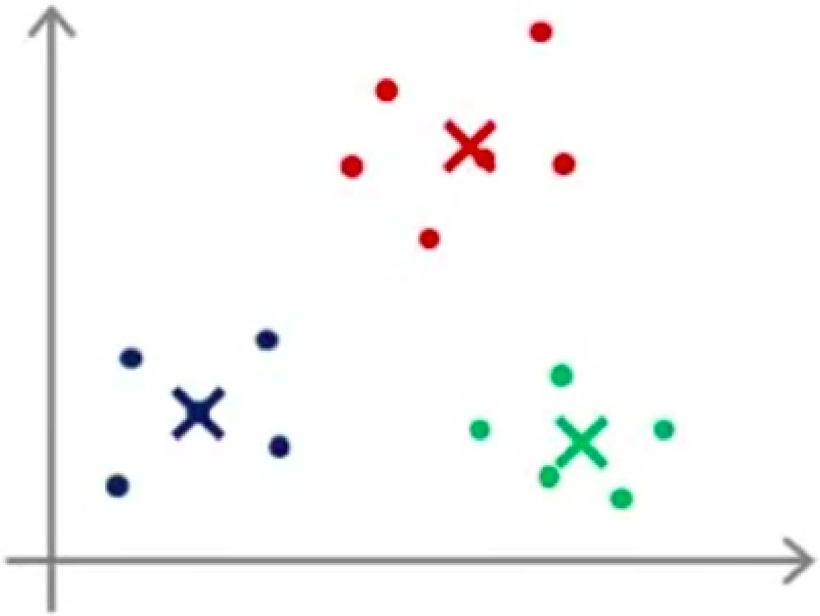
Debug



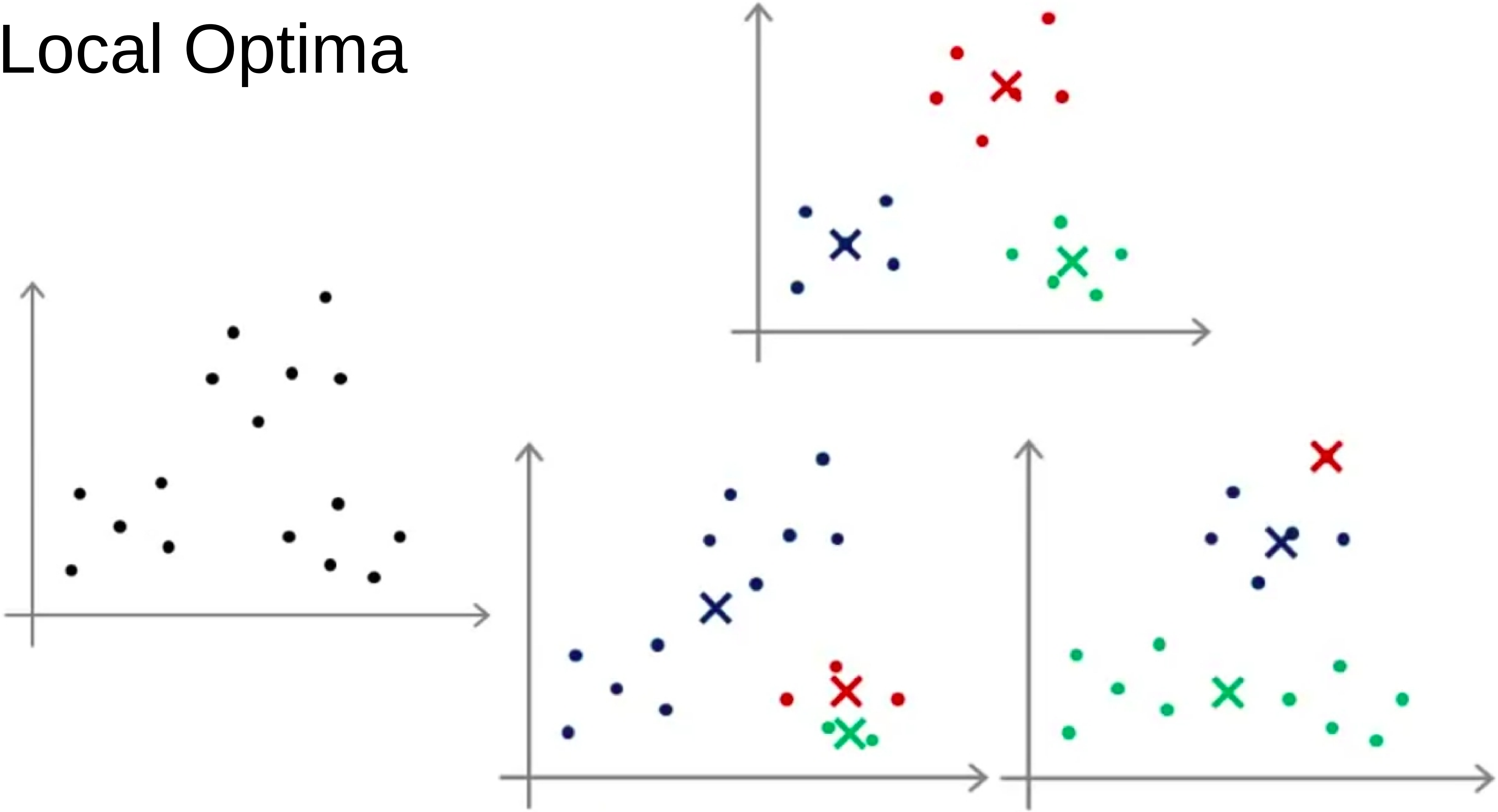
It is not possible for the cost function to sometimes increase. There must be a bug in the code.

Local Optima

Local Optima



Local Optima



Local Optima

For i = 1 to 100 {

Randomly initialize K-means.

Run K-means. Get $c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K$.

Compute cost function (distortion)

$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$$

}

Local Optima

For i = 1 to 100 {

Randomly initialize K-means.

Run K-means. Get $c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K$.

Compute cost function (distortion)

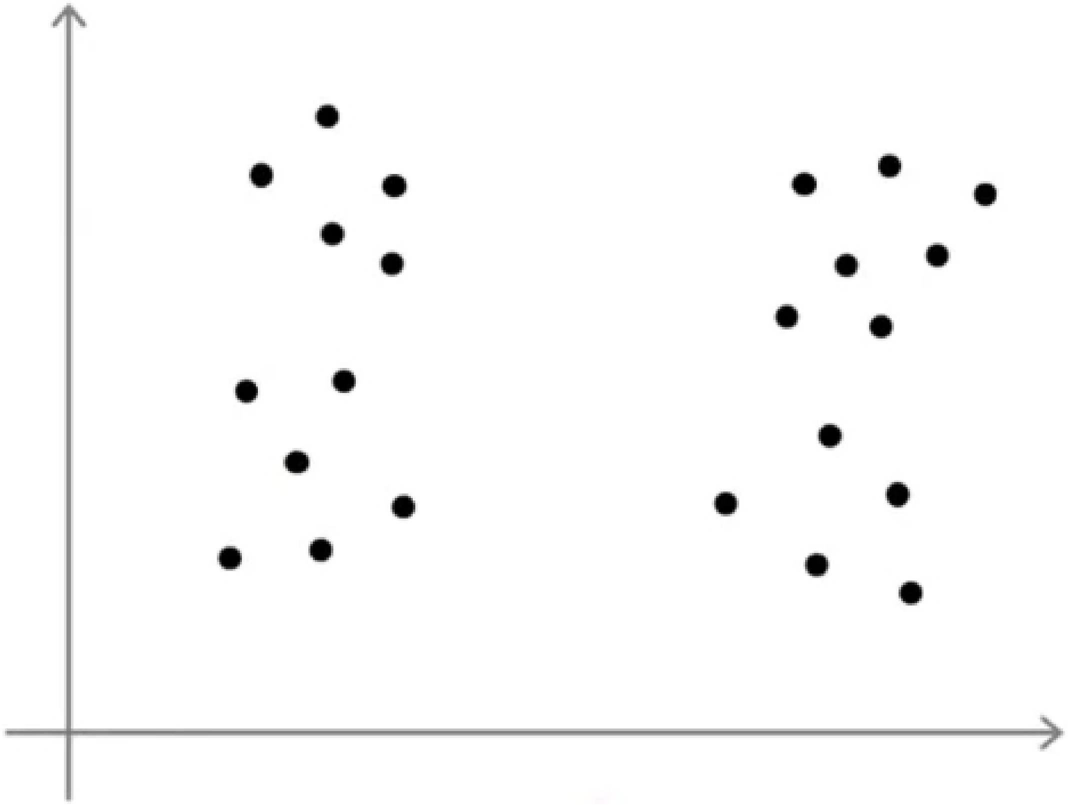
$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$$

}

Pick clustering that gave lowest cost $J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$

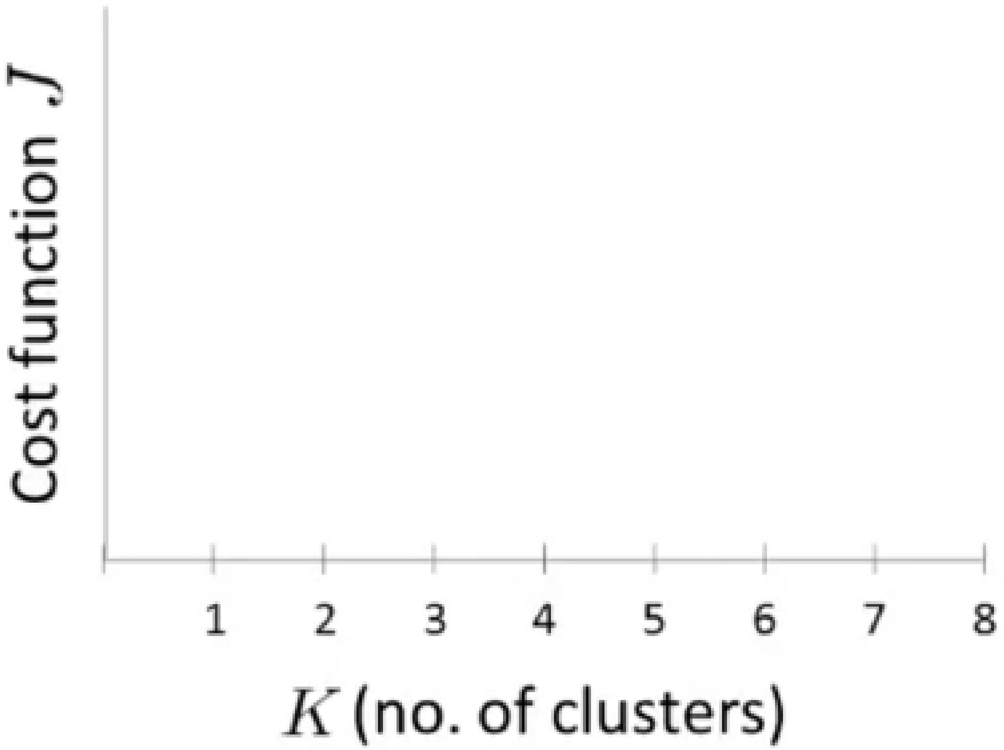
Choosing K

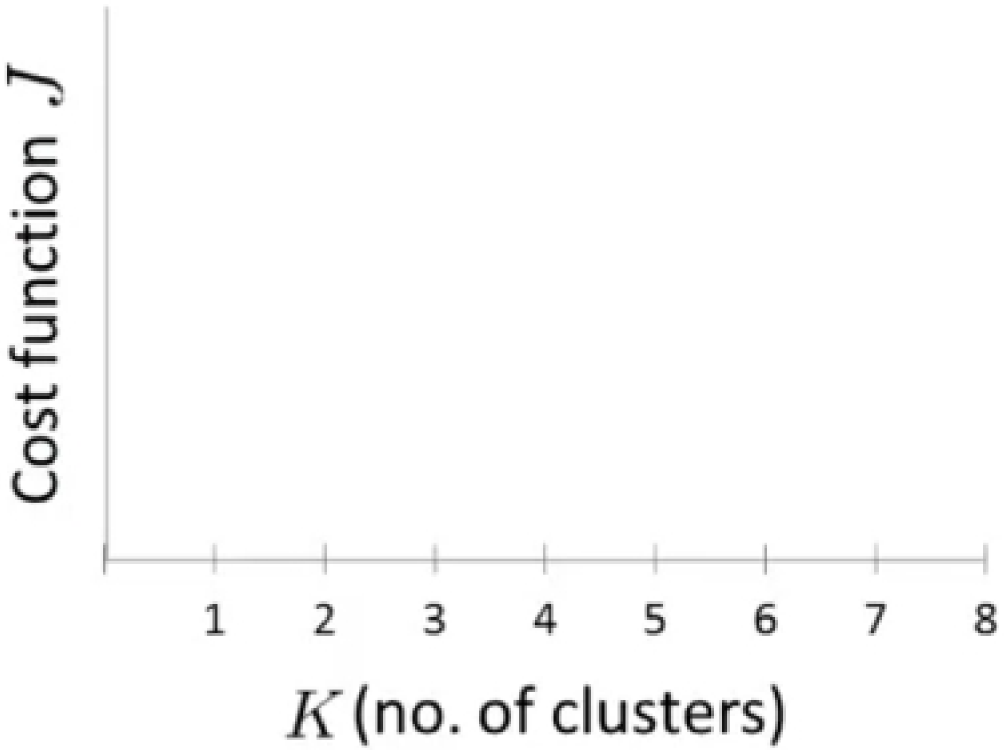
What is the right calculation of K ???



Choosing K

Elbow method:

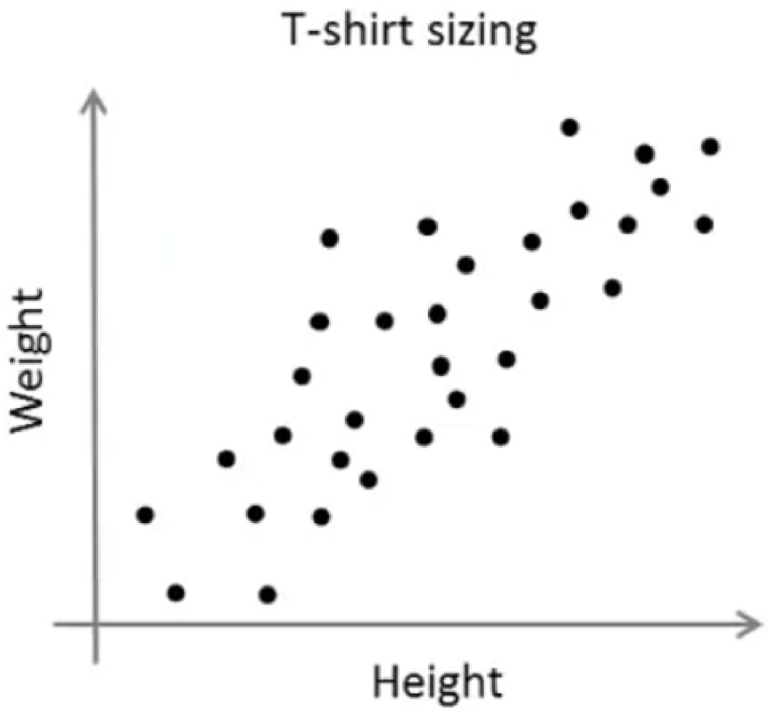


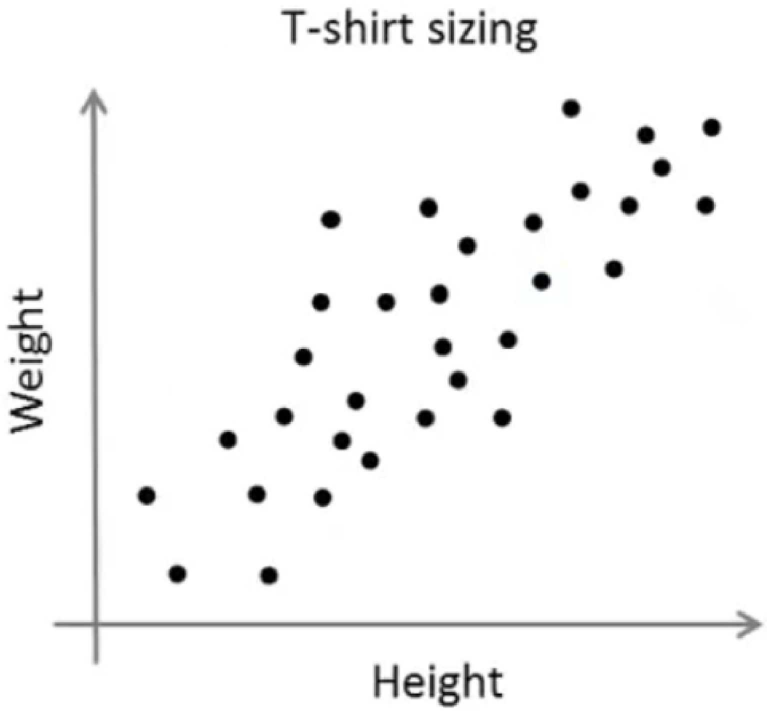


Choosing K

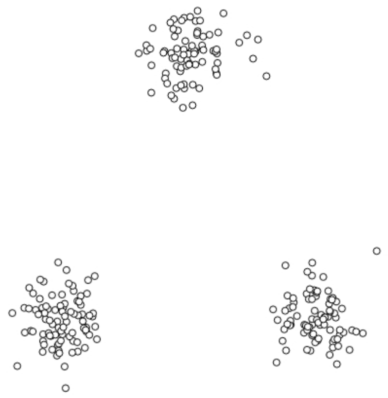
Sometimes, you're running K-means to get clusters to use for some later/downstream purpose. Evaluate K-means based on a metric for how well it performs for that later purpose.

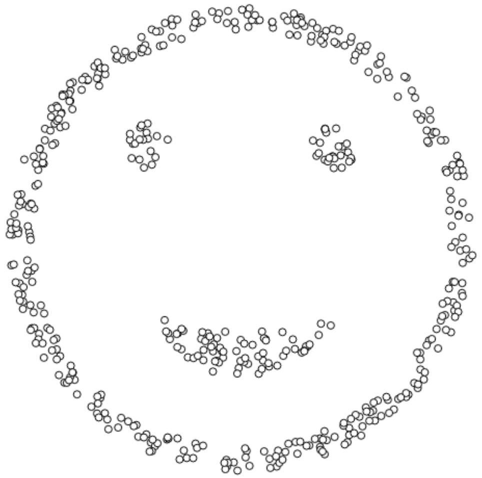
E.g.





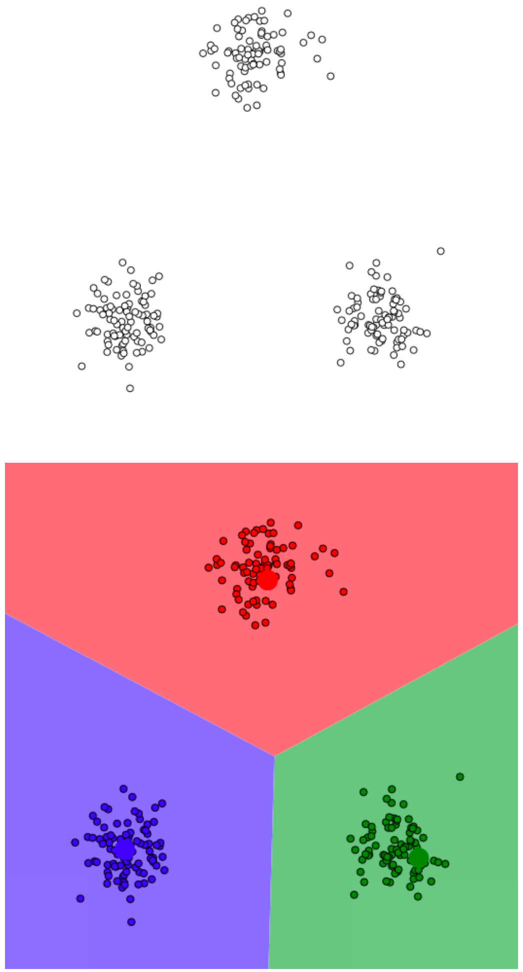
K-Means for non-convex set

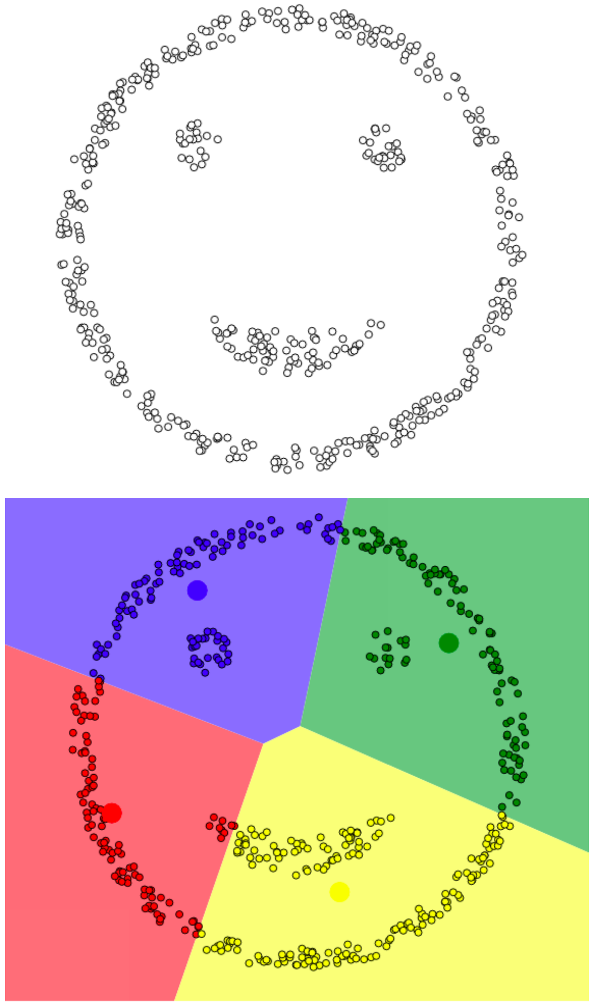




[Visualizing K-Means Clustering](https://www.naftaliharris.com/blog/visualizing-k-means-clustering) https://www.naftaliharris.com › blog › visualizing-k-means-clustering

K-Means for non-convex set





[Visualizing K-Means Clustering](https://www.naftaliharris.com/blog/visualizing-k-means-clustering) https://www.naftaliharris.com › blog › visualizing-k-means-clustering

Recap

Strength

- Relatively efficient
- Easy implementation

Weakness

- Need to specify k in advance
- Sensitive to noisy data and outliers
- Clusters are forced to convex space partitions
- Result and runtime strongly depend on the initial partition; often terminates at a local optimum – however: methods for a good initialization exist

References

Introduction to K-Means, Standford NLP:
https://nlp.stanford.edu/IR-book/html/htmledition/k-means-1.html

Knowledge Discovery in Databases (WiSe 2019/20):
https://www.dbs.ifi.lmu.de/Lehre/KDD/WS1920/lecture_notes/KDD1_IV.pdf

Machine Learning — Andrew Ng, Stanford University:

https://www.youtube.com/playlist?list=PLLssT5z_DsK-h9vYZkQkYNWcItqhlRJLN

Visualizing K-Means Clustering:
https://www.naftaliharris.com › blog › visualizing-k-means-clustering