# Happy Wednesday!

- **Quiz 2,** mean is 76% and average completion time 6min53s
- **Assignment 1 due tonight Sep 9th by 11:59pm → NO EXTENSIONS**
- **Third round of project seminars**, available Thursday, Sep 10th
- **Open office hours on Thursday, 7pm to 8pm**
  - https://primetime.bluejeans.com/a2m/live-event/qfsqxjec
- **Quiz 3, Friday, Sep 11th 6am until Sep 12th 6am**
  - K-means clustering
- **Quizzes on Fridays – a discussion**

# Highlights of foundations

- Linear algebra
  - Covariance and correlation
  - Eigendecomposition
  - SVD
- Probability theory
  - Sum rule
  - Product rule
  - Bayes theorem
- Information theory
  - Information
  - Entropy
  - Mutual information
  - KL Divergence
- Optimization
  - Objective function
  - Constraints
  - Lagrangian

# Unsupervised and supervised learning

$$
\begin{array}{r}
\\
\text{Point 1} \\
\text{Point 2} \\
\text{Point 3} \\
\text{Point 4} \\
\text{Point 5}
\end{array}
\begin{array}{c}
\begin{array}{cccc}
\text{Weight(lb)} & \text{Height(cm)} & \text{Fur color} & \text{Eye color}
\end{array} \\
\begin{bmatrix}
10 & 20 & w & g \\
50 & 100 & br & bl \\
8 & 15 & bl & bl \\
12 & 25 & w & bl \\
14 & 10 & bl & g
\end{bmatrix}_{X_{n\times d}}
\end{array}
=
\begin{array}{c}
\text{Label} \\
\begin{bmatrix}
cat \\ dog \\ dog \\ cat \\ dog
\end{bmatrix}_{Y_{n\times 1}}
\end{array}
$$

WE DON'T HAVE THIS IN UNSUPERVISED

UNSUPERVISED

Unsupervised just focuses on $X_{n\times d}$

Supervised just focuses on $X_{n\times d}$ and $Y_{n\times 1}$

CS4641B Machine Learning

# Lecture 07: Clustering Analysis and K-Means

Rodrigo Borela ▸ rborelav@gatech.edu

These slides are based on slides from Chao Zhang, Le Song and Mahdi Roozbahani

# Outline

- Clustering
- Distance functions
- K-Means algorithm
- Analysis of K-Means

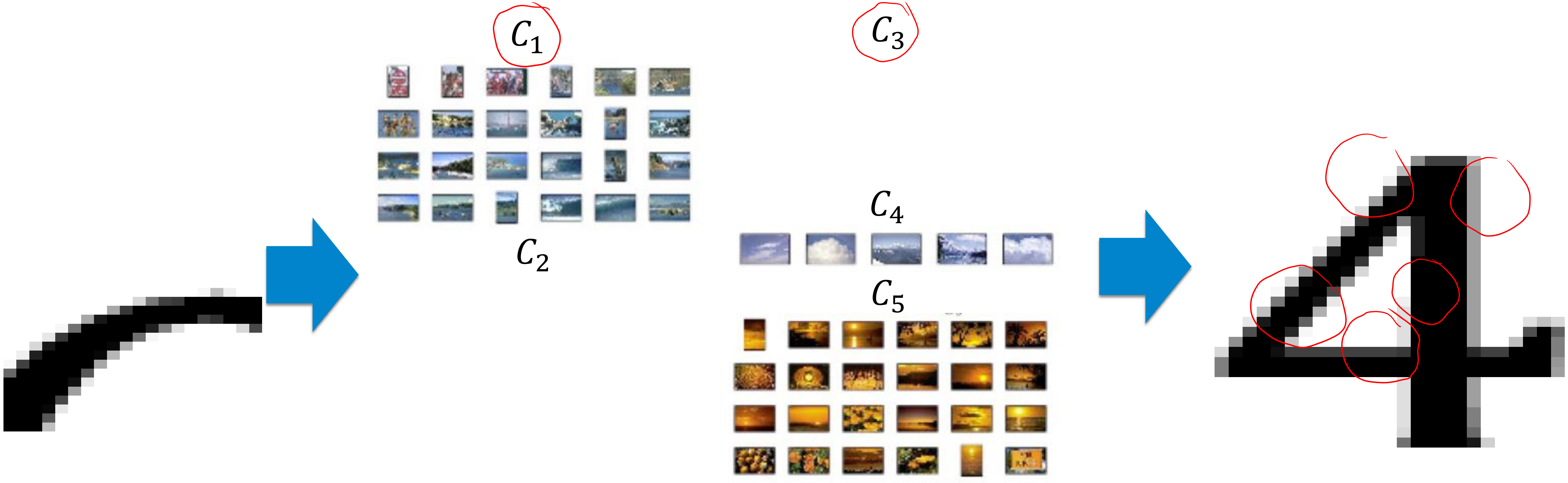
*Complementary reading: Bishop PRML – Chapter 9, Sections 9.1 through 9.1.1*

# Outline

- Clustering
- Distance functions
- K-Means algorithm
- Analysis of K-Means

# Clustering images

- **Goal of clustering:** Divide objects into groups such that objects within a group are more similar than those outside the group

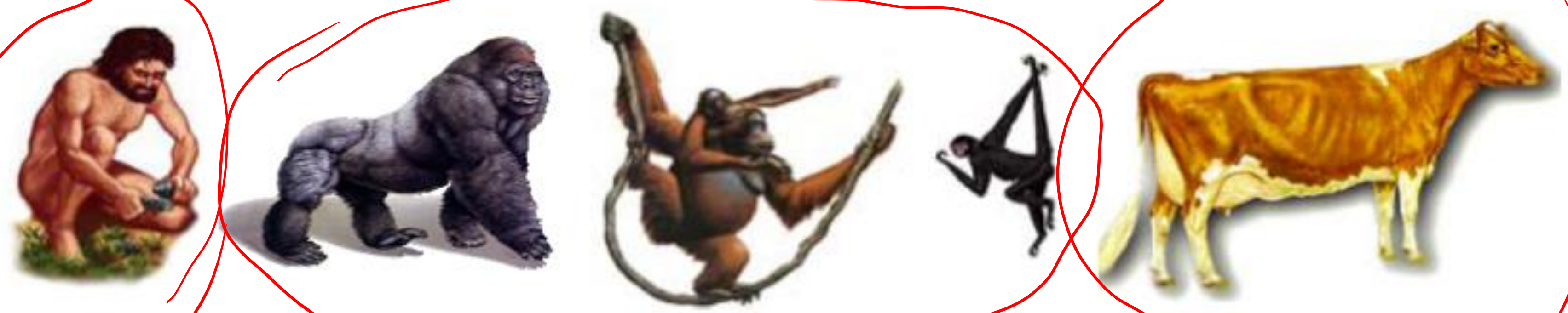


# Clustering other objects


Flags

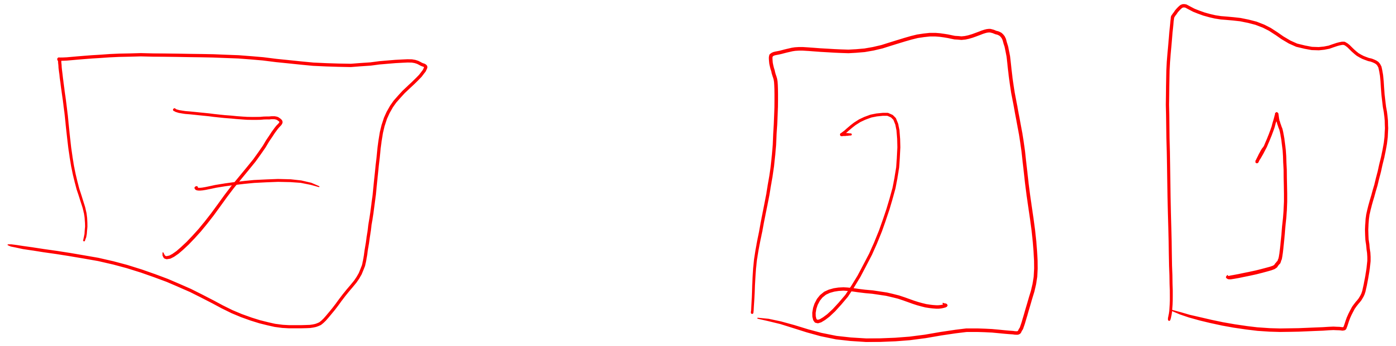
Piotr Belarusian
Pyotr Azerbaijani
Petros Greek
Pietro Italian
Pedro Portuguese
Pierre French
Piero Italian
Peter Dutch
Peder Danish
Peka Finish
Peadar Irish

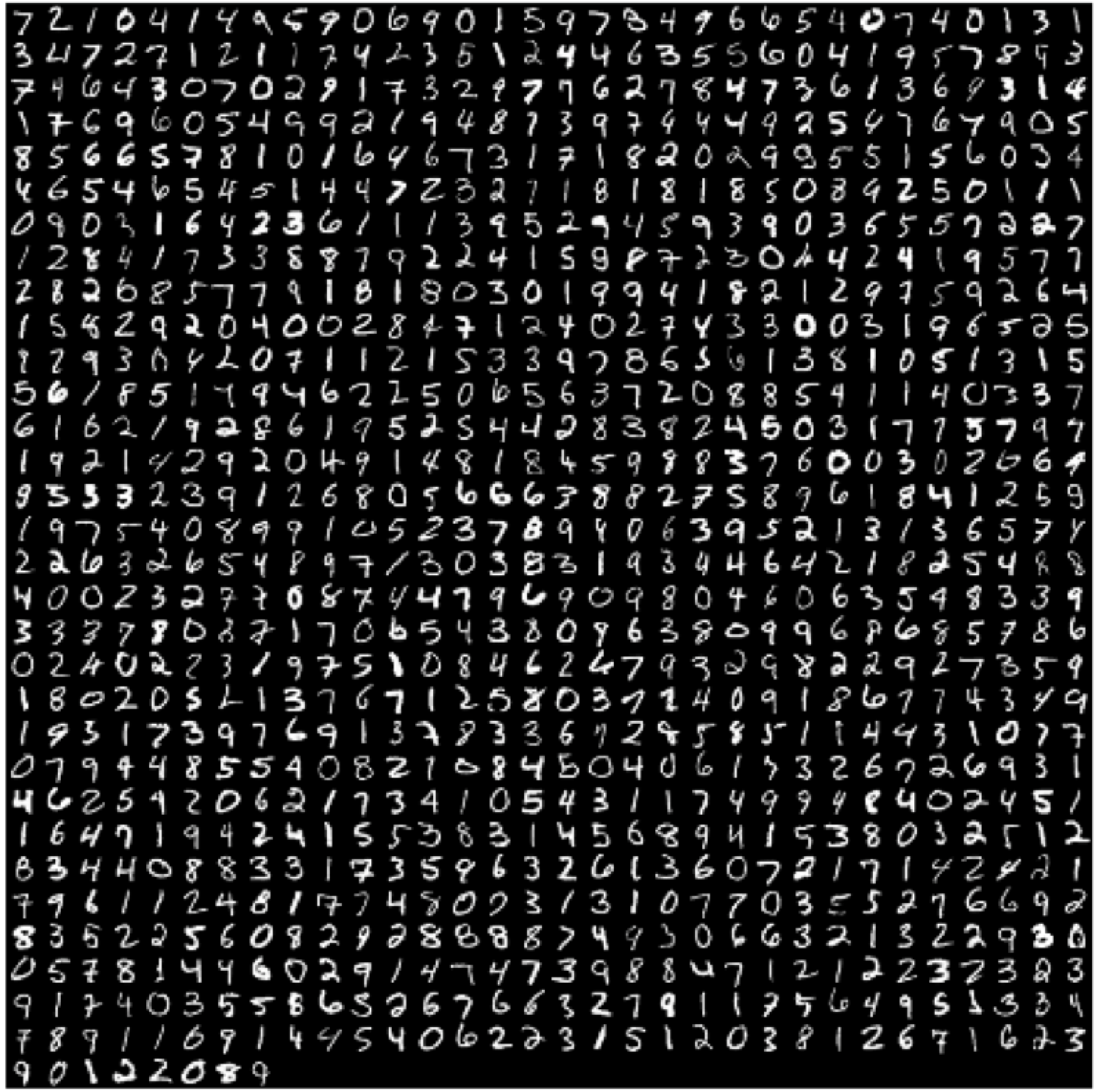
Linguistic Similarity



Species

# Clustering hand digits





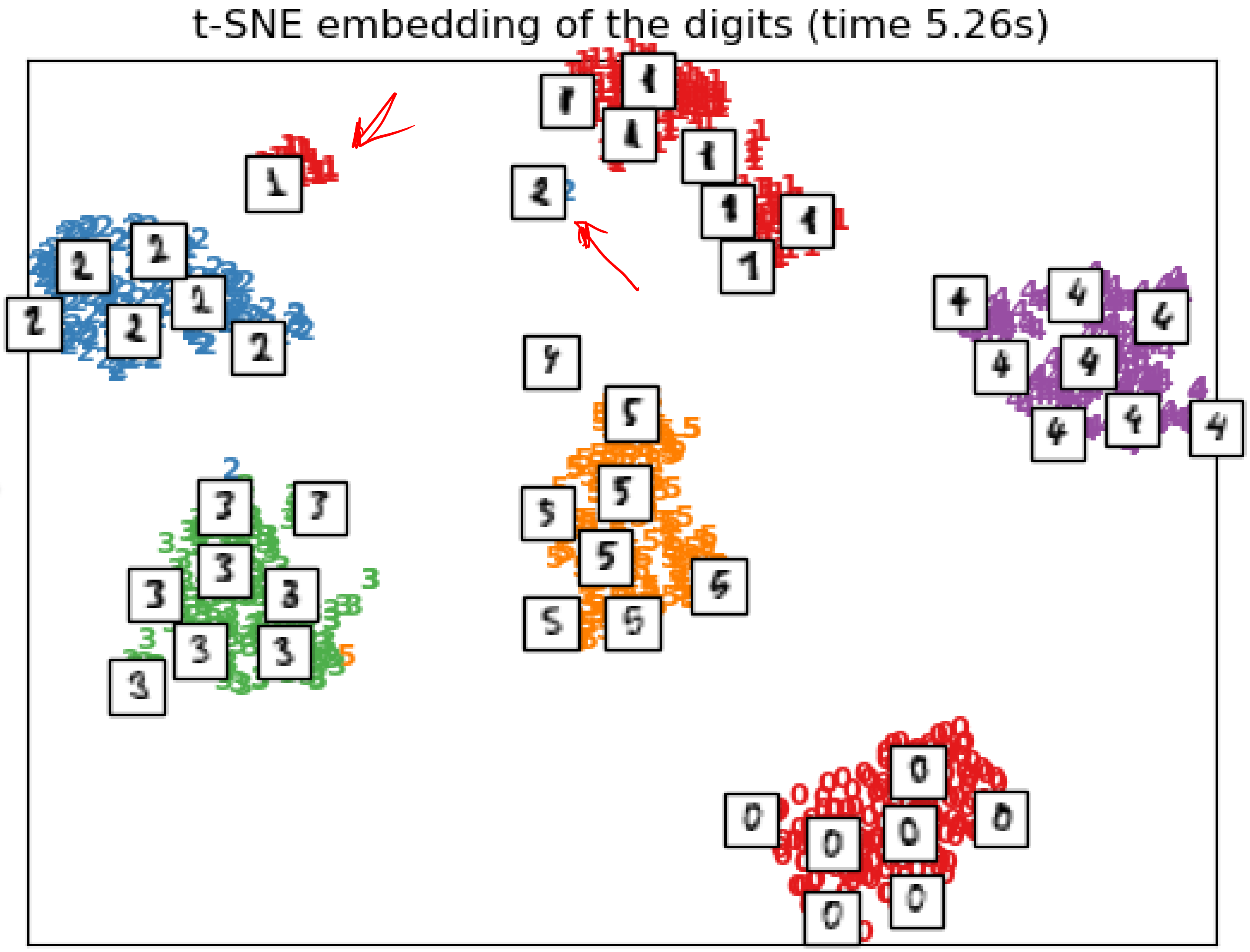
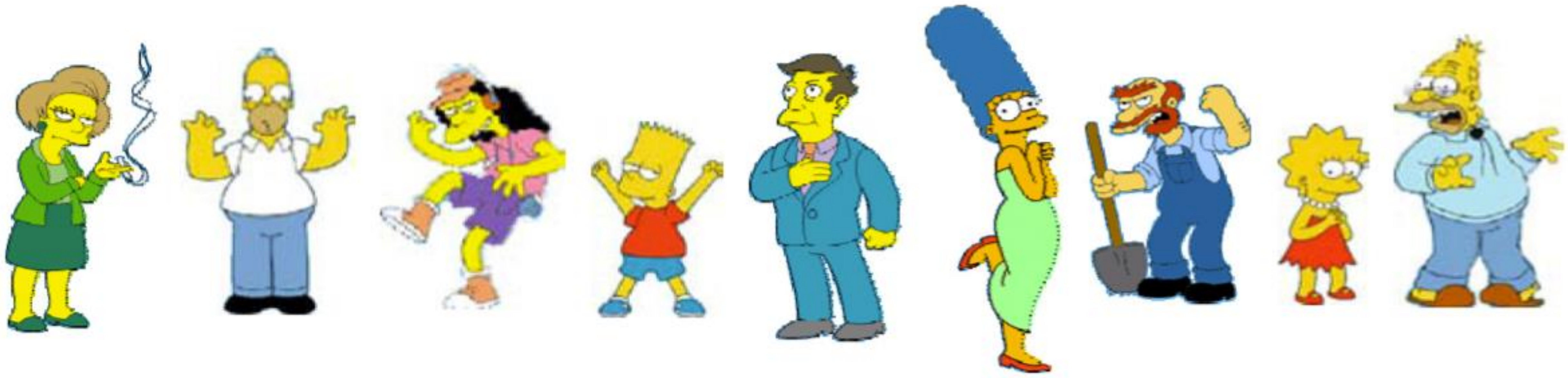
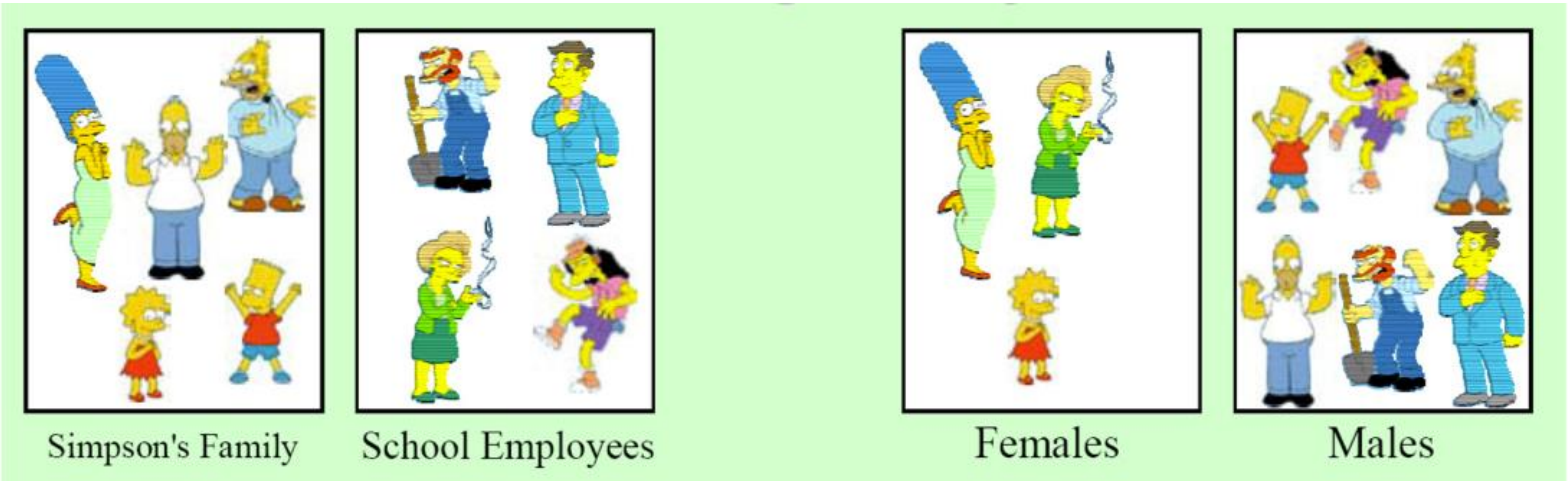



Image credit: Scikit learn

# Clustering is subjective



What is consider similar/dissimilar?



# What is clustering in general?

- You pick your similarity/dissimilarity function
- The algorithm figures out the grouping of objects based on chosen similarity/dissimilarity function
  - Points within a cluster are similar
  - Points across clusters are not so similar
- Issues for clustering
  - How to represent objects? (Vector space? Normalization?)
  - What is a similarity/dissimilarity function for your data?
  - What are the algorithm steps?

# Outline

- Clustering
- Distance functions
- K-Means algorithm
- Analysis of K-Means

# Properties of distance functions

- Desired properties of distance functions
- Symmetry: $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$
  - Otherwise you could claim "Alex looks like Bob, but Bob looks nothing like Alex"
- Positive separability: $d(\mathbf{x}, \mathbf{y}) = 0$, if and only if $\mathbf{x} = \mathbf{y}$
  - Otherwise there are objects that are different, but you cannot tell them apart
- Triangular inequality: $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{z}, \mathbf{y})$
  - Otherwise you could claim "Alex is very like Bob, and Alex is very like Carl, but bob is very unlikely Carl"

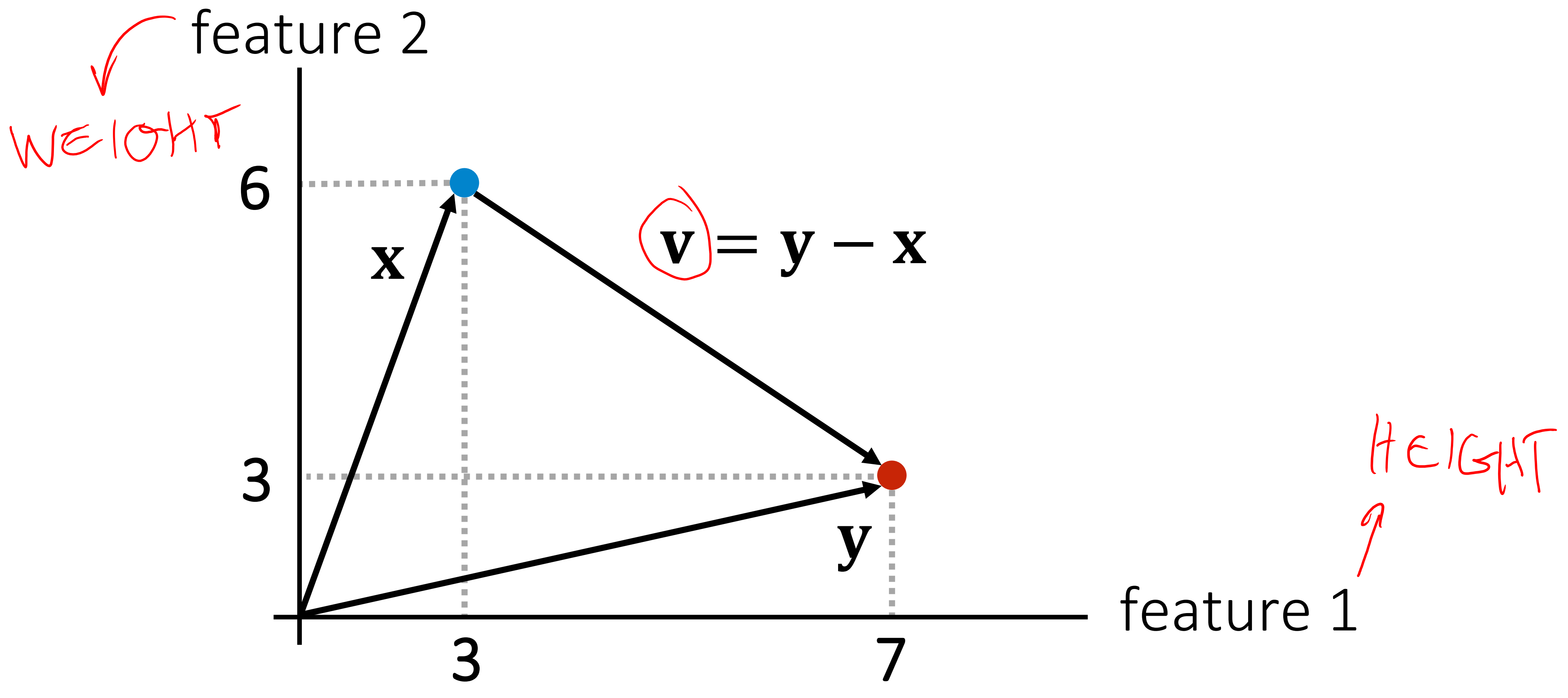
# Distance functions for vectors

- Suspose two data points, both in $\mathbb{R}^{\boldsymbol{D}}$ → D DIMENSIONAL VECTOR
- $\mathbf{x} = (x_1, x_2, \dots, x_D)^T$
- $\mathbf{y} = (y_1, y_2, \dots, y_D)^T$

- Euclidean distance: $d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2}$ → L2-NORM
- Minkowski distance: $d(\mathbf{x}, \mathbf{y}) = \sqrt[p]{\sum_{i=1}^{D}(x_i - y_i)^p}$ → P-NORM
  - Euclidean distance: $p = 2$
  - Manhattan distance: $p = 1, d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{D} |x_i - y_i|$
  - "Inf"-distance: $p = \infty, d(\mathbf{x}, \mathbf{y}) = \max_i |x_i - y_i|$

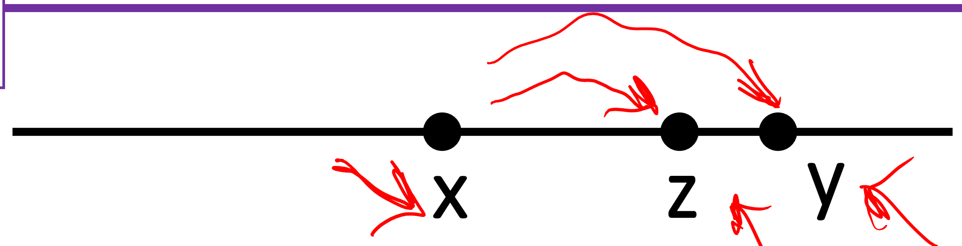
# Example

- Euclidean distance: $d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2} = \sqrt{(7-3)^2 + (3-6)^2} = 5$
- Manhattan distance: $d(\mathbf{x}, \mathbf{y}) = |7-3| + |3-6| = 7$
- "Inf"-distance: $d(\mathbf{x}, \mathbf{y}) = \max(|7-3|, |3-6|) = 4$



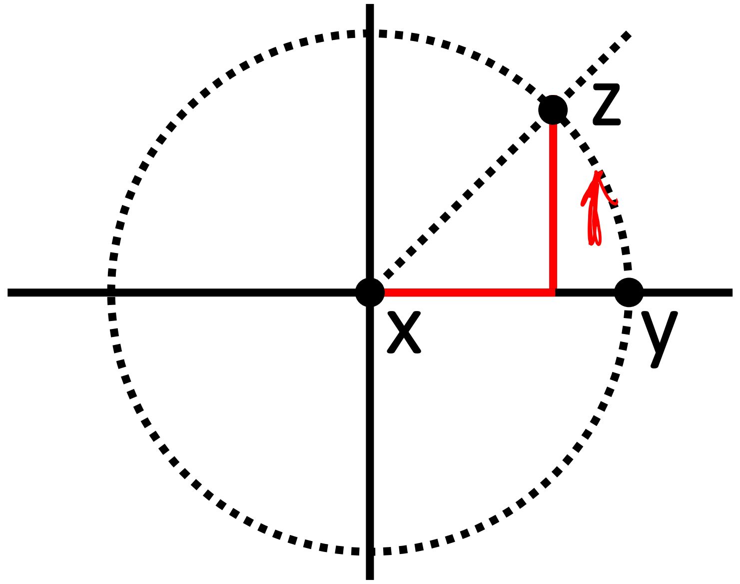
# Problems with Euclidean distance

**1D**

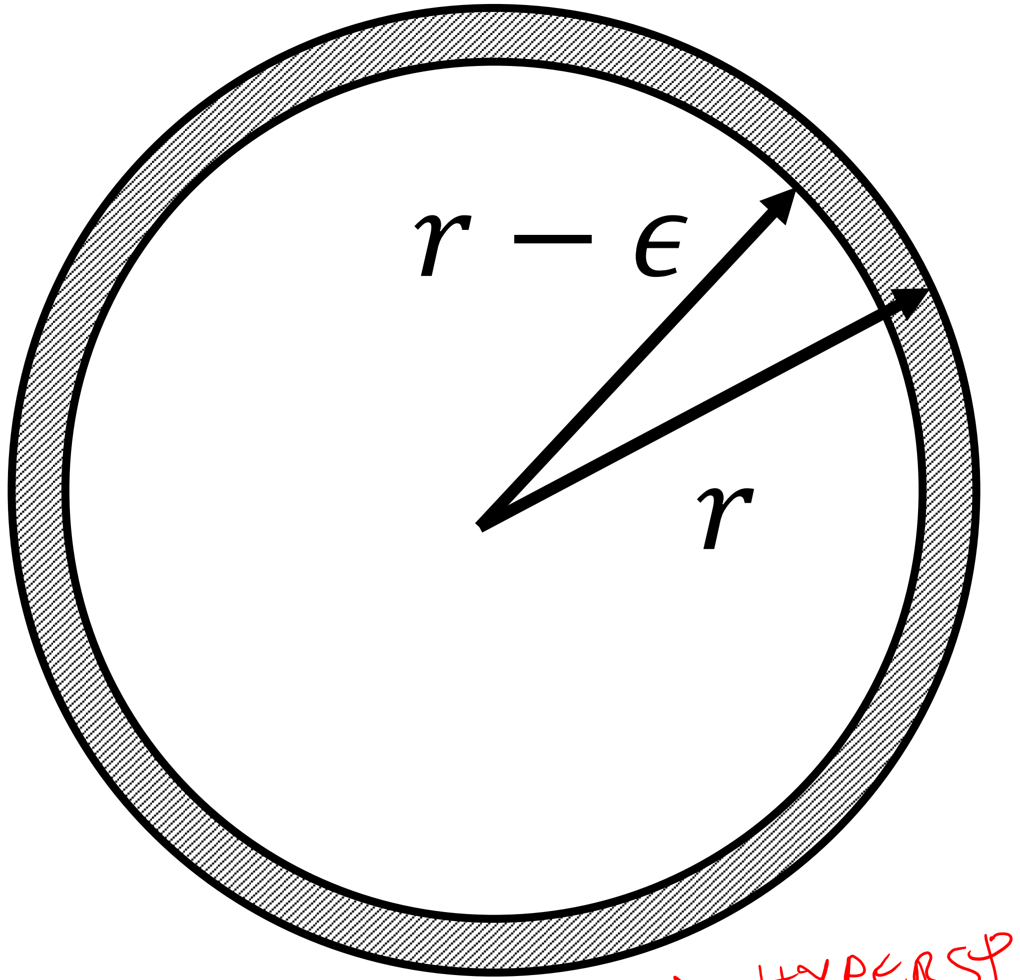



Euclidean: $d(x,y) > d(x,z)$
Manhattan: $d(x,y) > d(x,z)$

**2D**




Euclidean: $d(x,y) = d(x,z)$
Manhattan: $d(x,y) < d(x,z)$




$$V_{sphere} = V_3 = \frac{4}{3}\pi r^3$$

$$V_{hypersphere} = V_D = K_D r^D$$

DEPENDS ON D

HYPERSPHERE

$$\frac{V_{shell}}{V_{sphere}} = \frac{V_D(r) - V_D(r-\epsilon)}{V_D(r)} = 1 - (1-\epsilon)^D$$

[Curse of dimensionality]

# Hamming distance

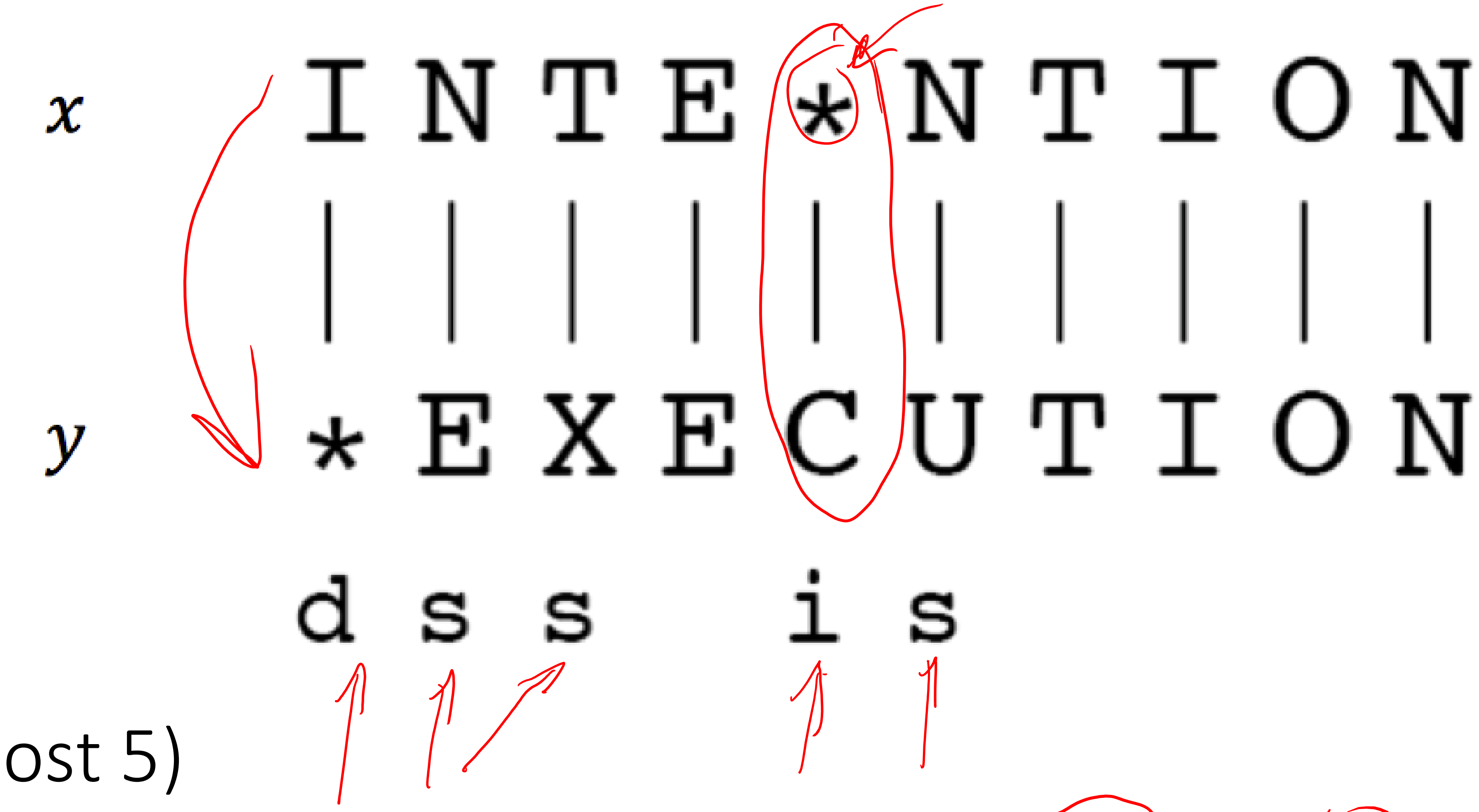
- Manhattan distance is also called **Hamming distance when all features are binary**
  - Count the number of difference between two binary vectors
  - Example, $\mathbf{x}, \mathbf{y} \in \{0,1\}^{17}$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $x$ | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| $y$ | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

$$d(\mathbf{x}, \mathbf{y}) = 5$$

# Edit distance

- Transform one of the objects into the other, and measure how much effort it takes




- d: deletion (cost 5)
- s: substitution (cost 1)
- i: insertion (cost 2)

*(These costs are arbitrary)*

$$d(\mathbf{x}, \mathbf{y}) = 5 \times 1 + 3 \times 1 + 1 \times 2 = 10$$

# Edit distance

- Transform one of the objects into the other, and measure how much effort it takes

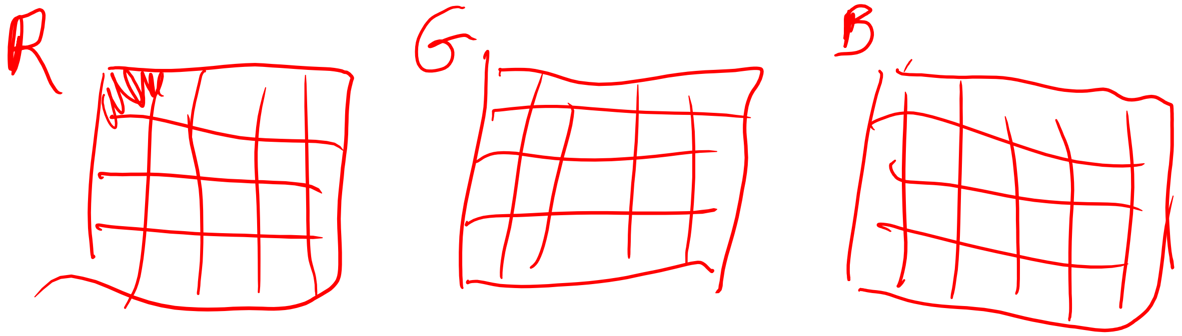
- d: deletion (cost 5)
- s: substitution (cost 1)
- i: insertion (cost 2)

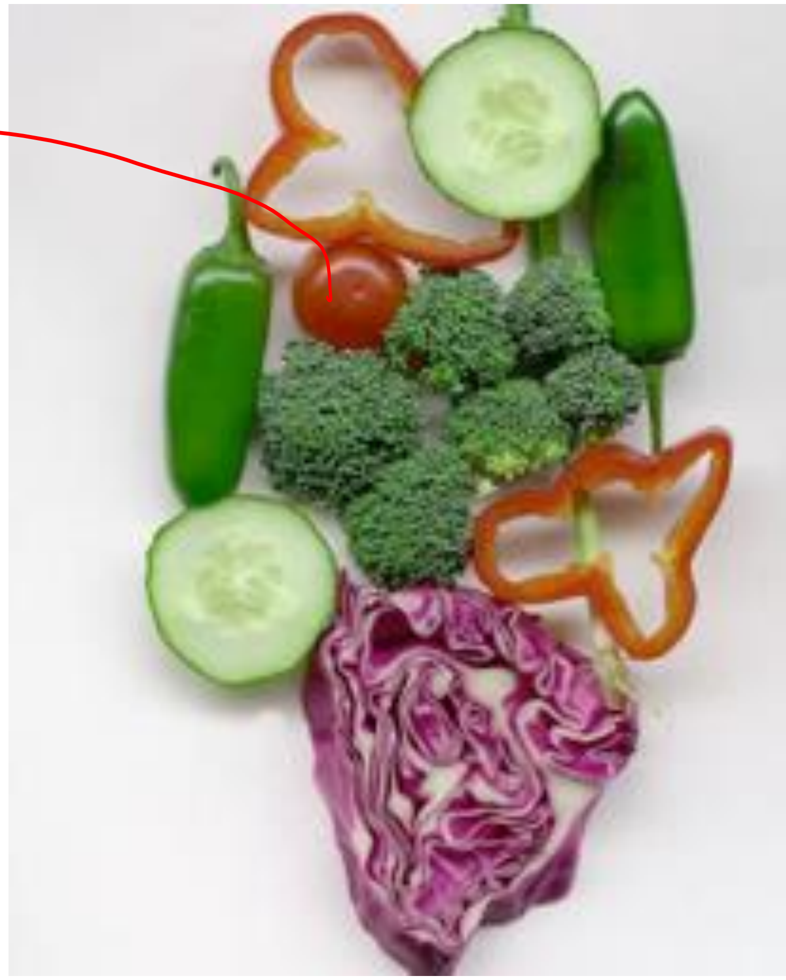
*(These costs are arbitrary)*

# Outline

- Clustering
- Distance functions
- K-Means algorithm
- Analysis of K-Means

# Results of K-means clustering



Image

Clusters on intensity

Clusters on color

K-means clustering using intensity alone and color alone

Image

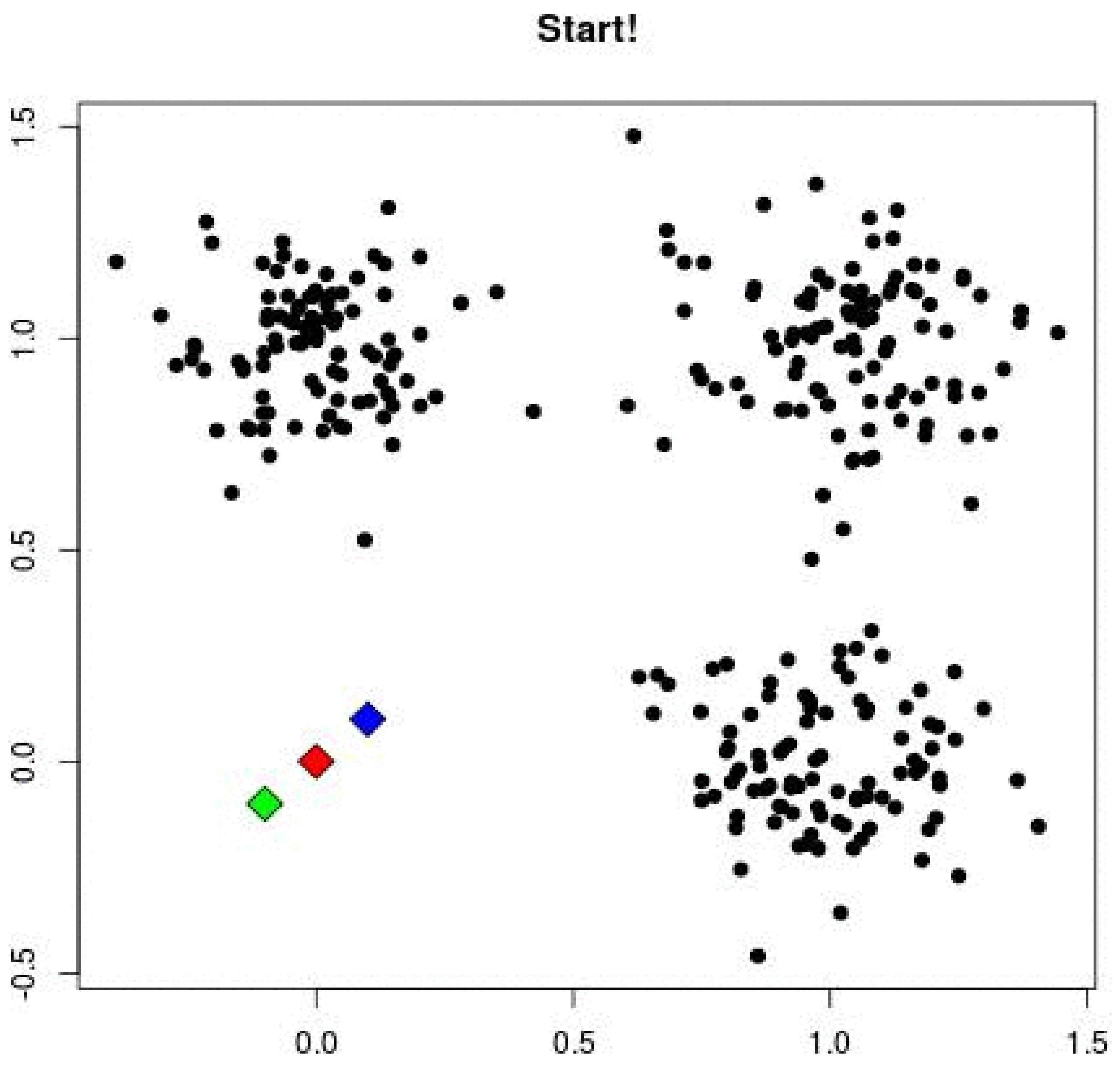
Clusters on color

BETTER

K-means using color alone, 11 segments (clusters)

* Pictures from Mean Shift: A Robust Approach toward Feature Space Analysis, by D. Comaniciu and P. Meer http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html

# K-means algorithm

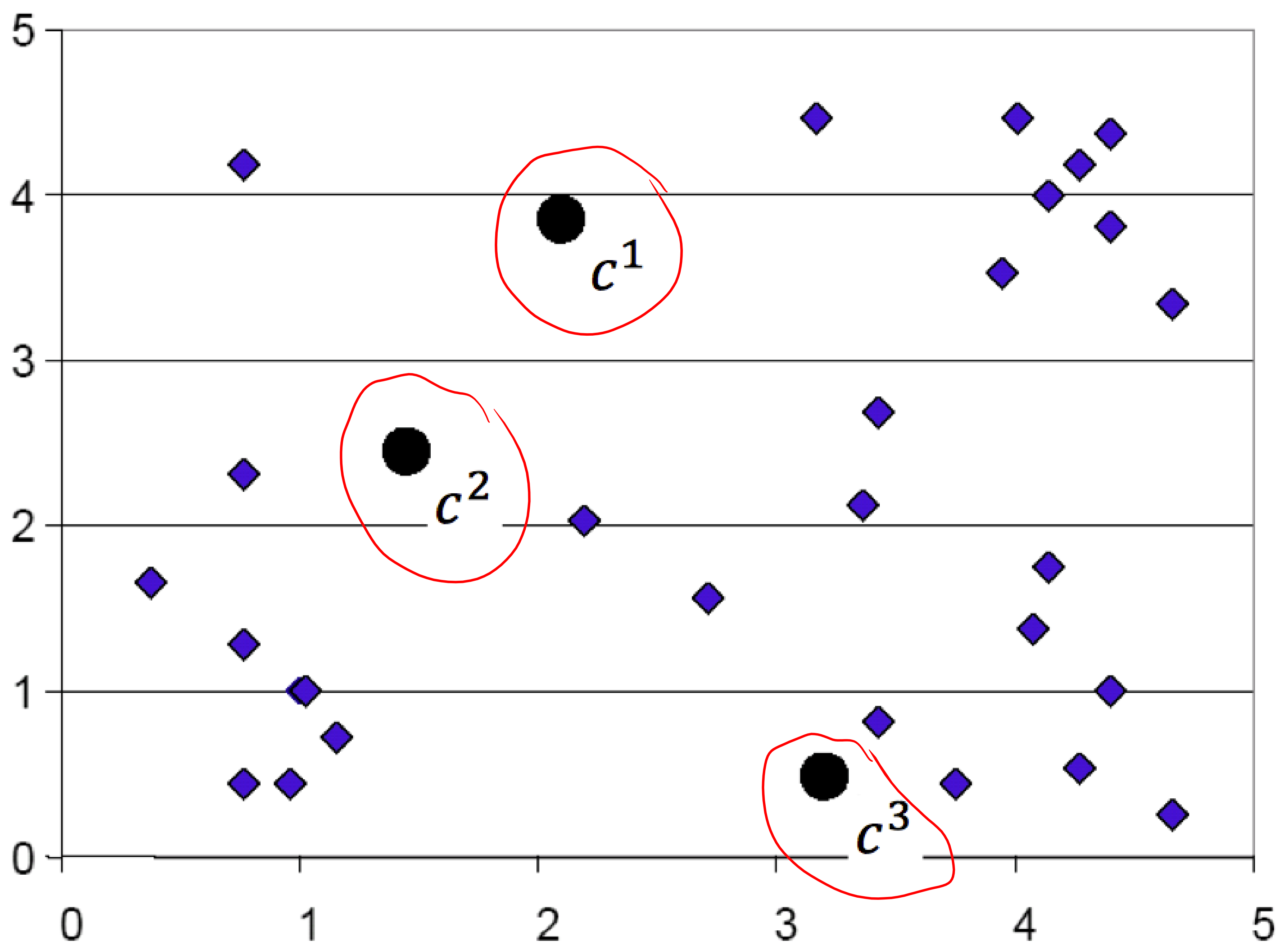



[Visualizing K-Means Clustering]

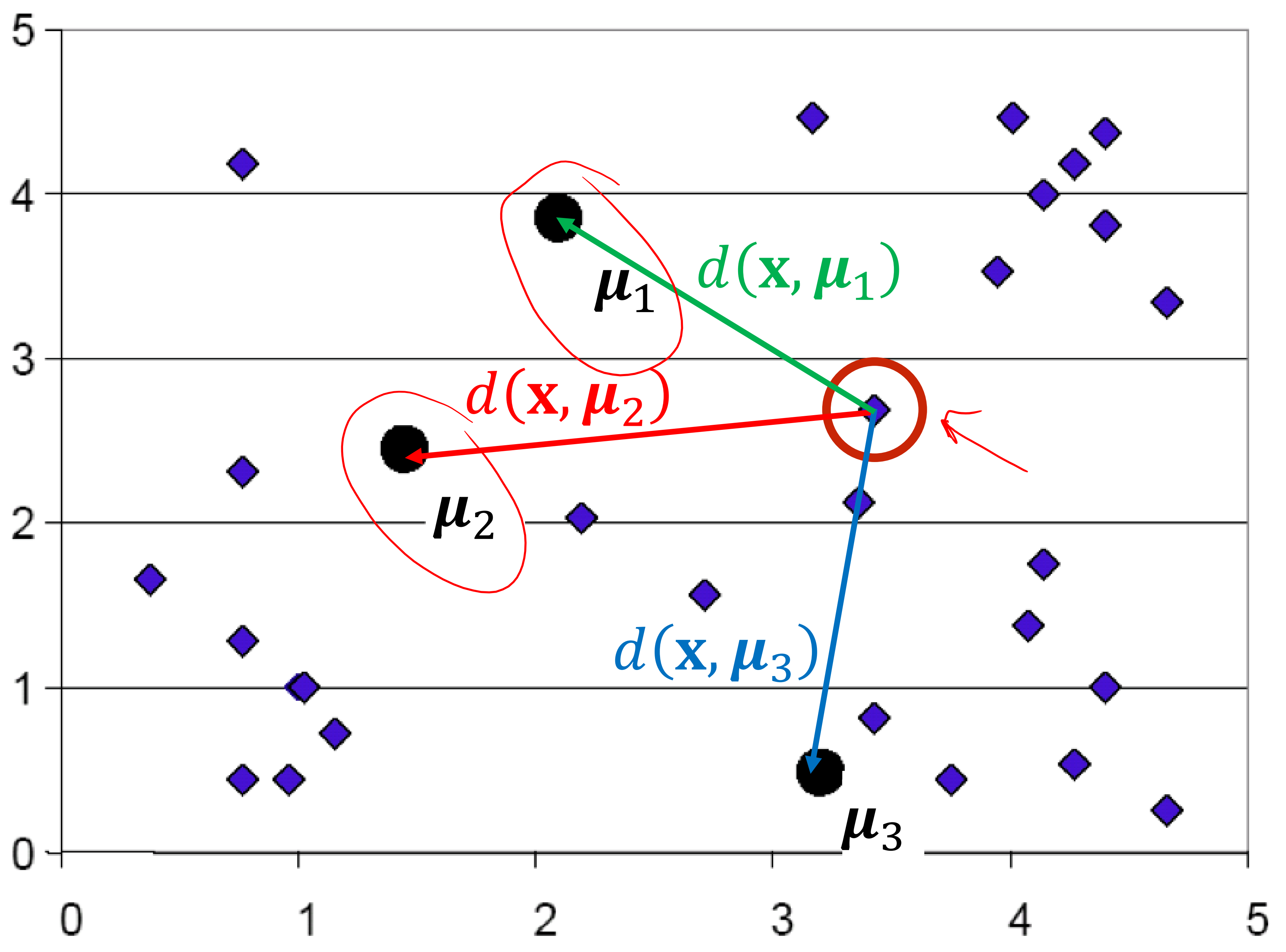
◇ CLUSTER CENTER

1. Initialize the number of clusters and their centers
2. Compute the distance between each point and each cluster center.
3. Assign each point the cluster id of the nearest cluster center
4. Recompute the cluster centers based on the cluster assignment to each point
5. Repeat steps 2 and 3 until convergence

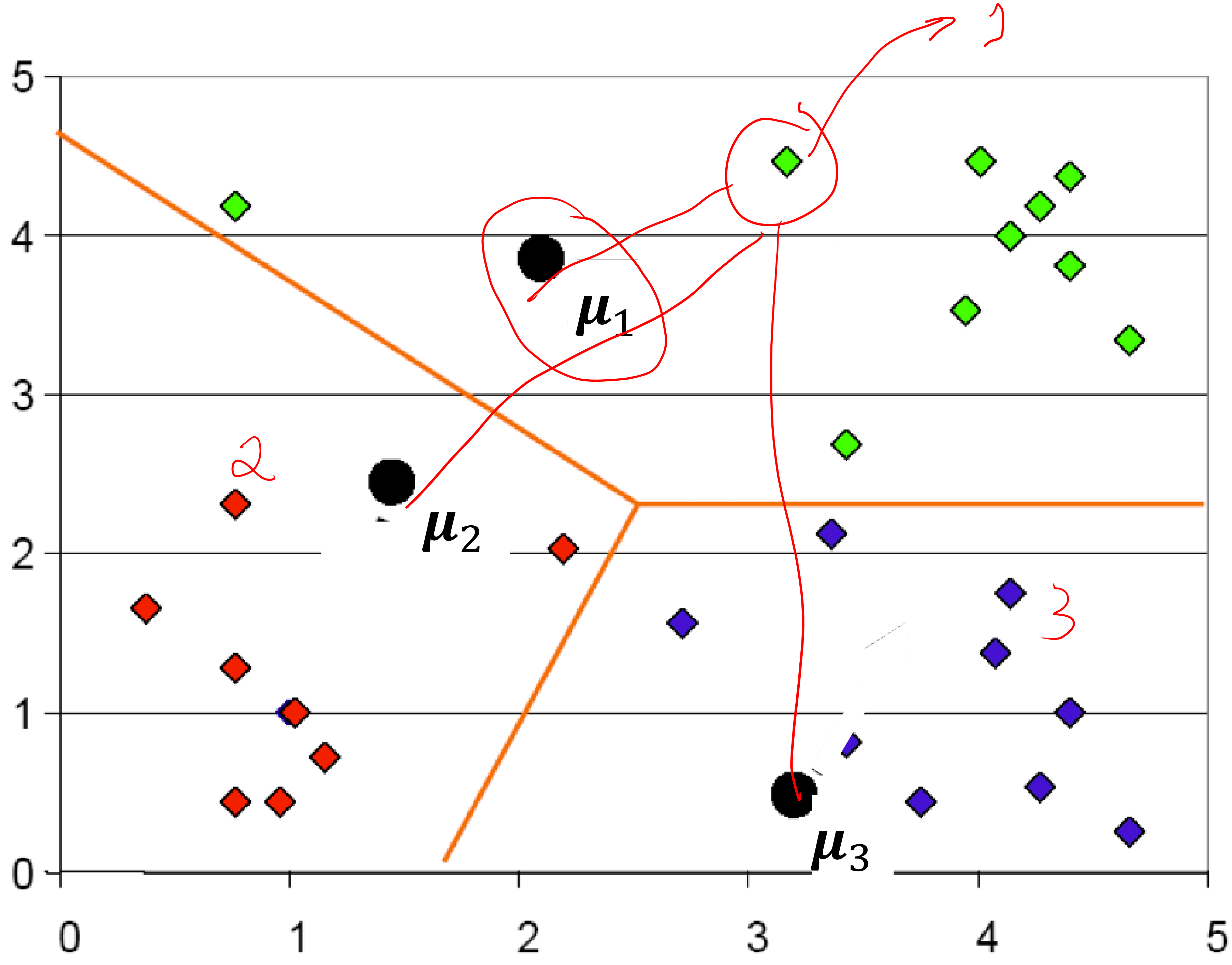
# K-means step 1: Initialization



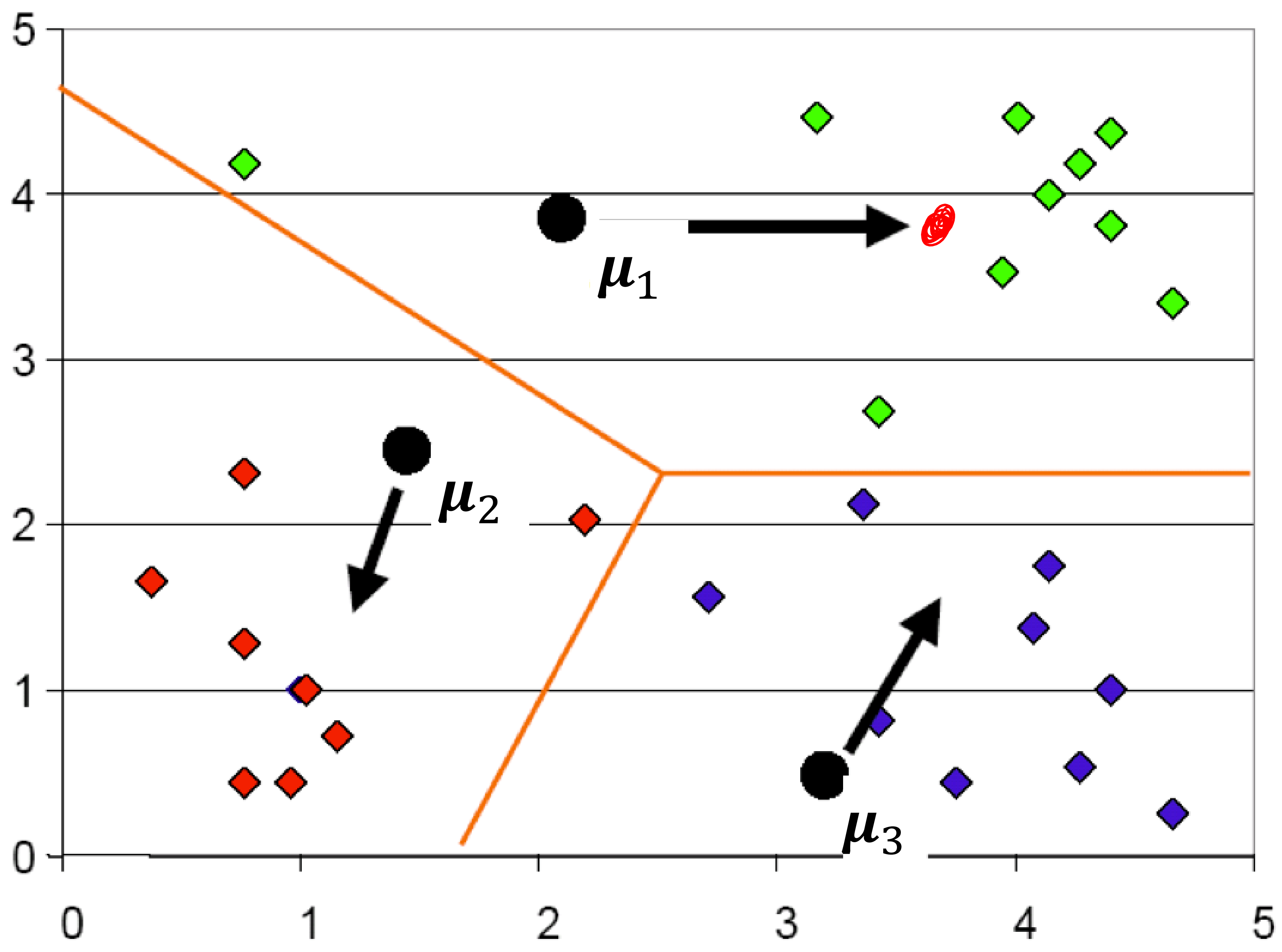
# K-means step 2: Compute dissimilarity



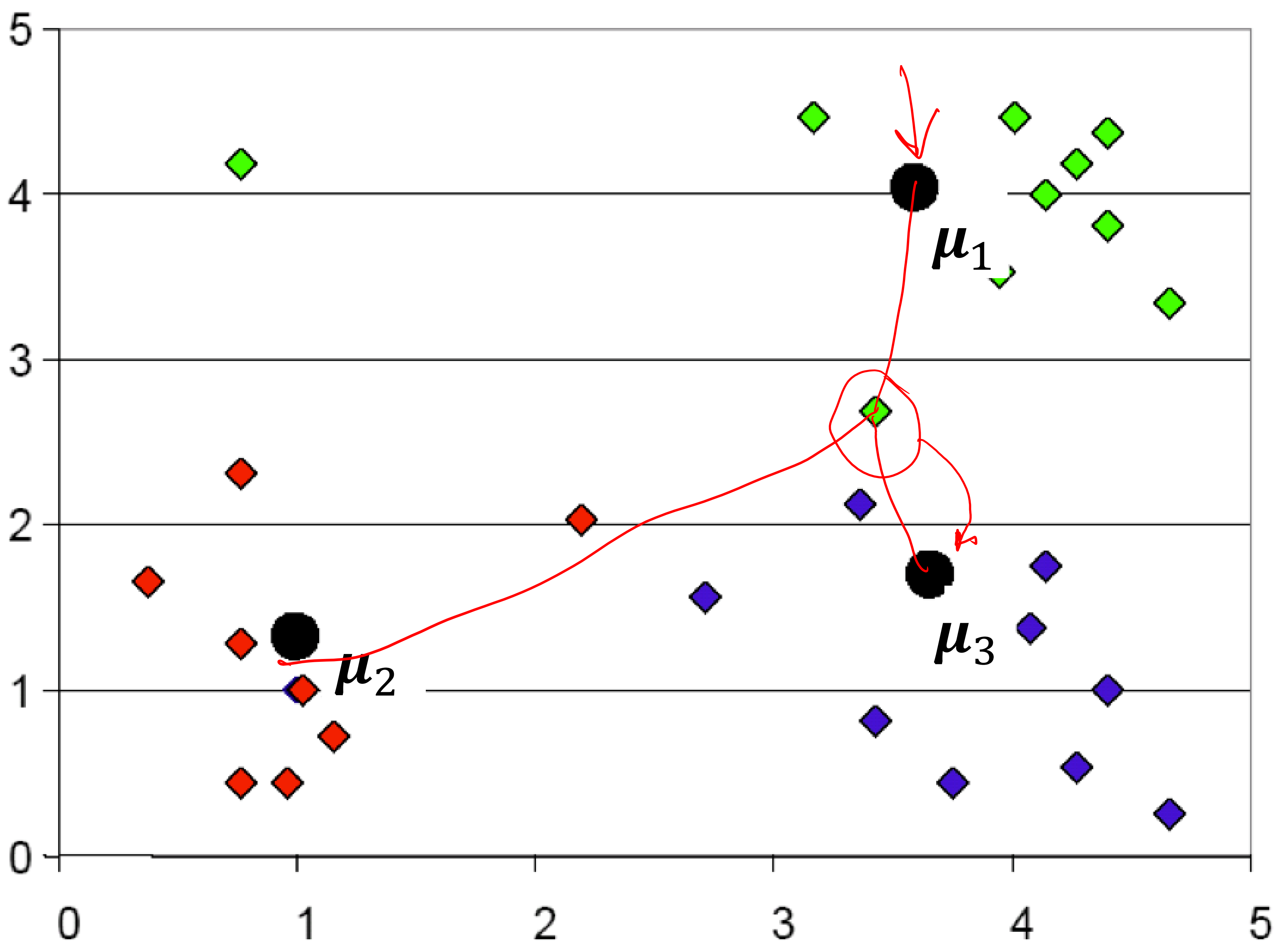
# K-means step 3: Define cluster assignment



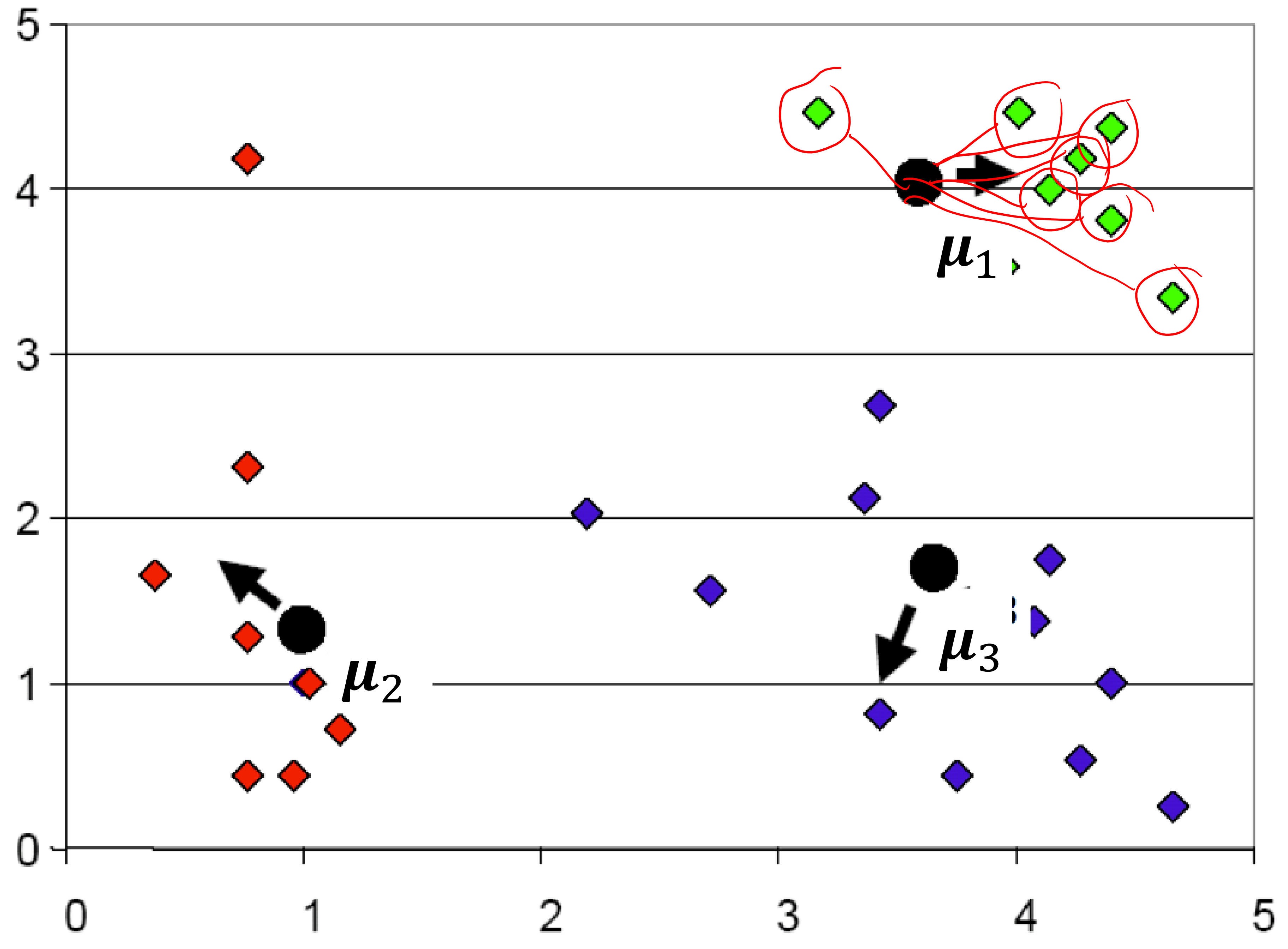
# K-means step 4: Recompute cluster centers



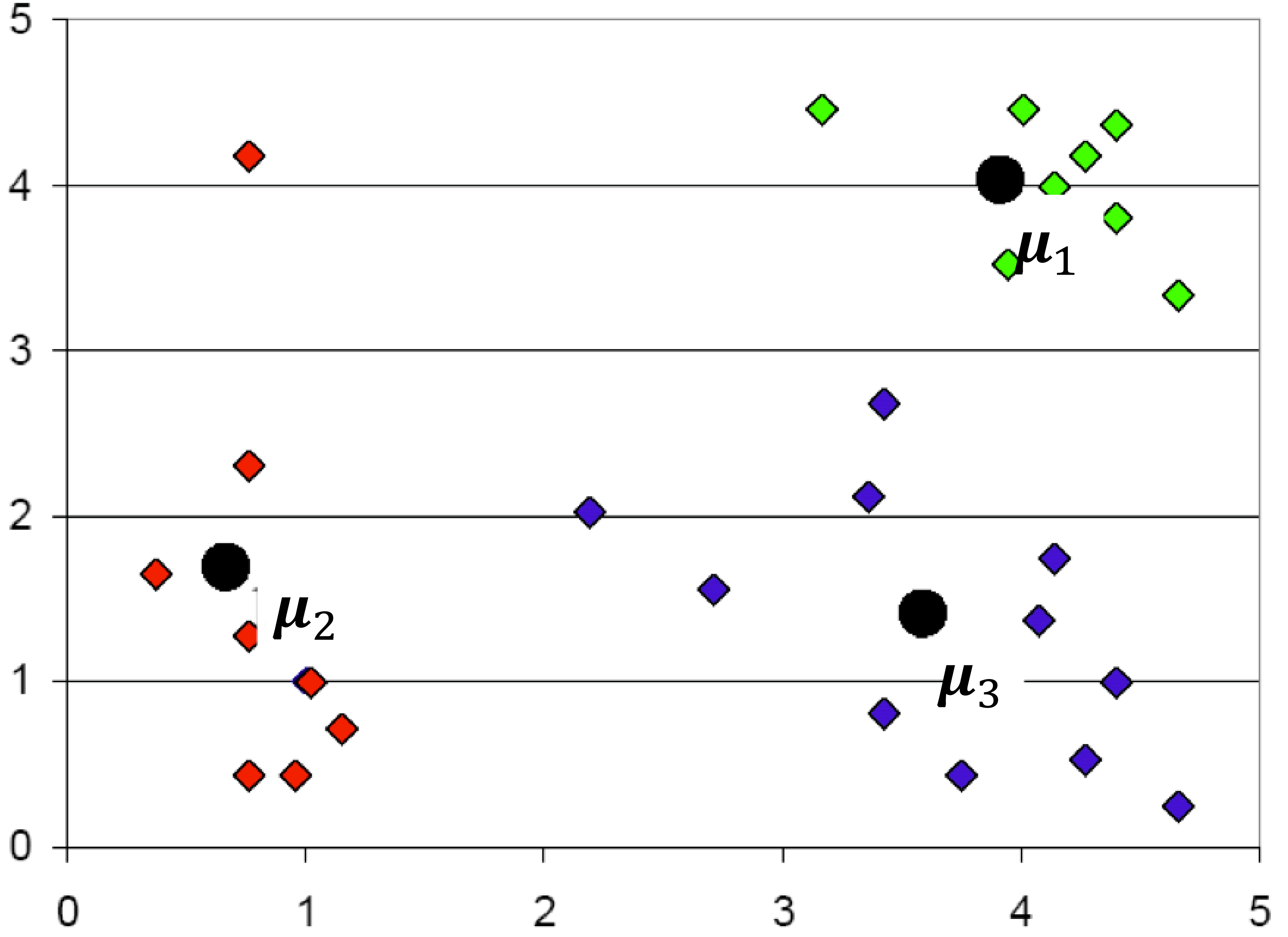
# K-means step 4: Recompute cluster centers



# K-means: Repeat until convergence



# K-means: Repeat until convergence



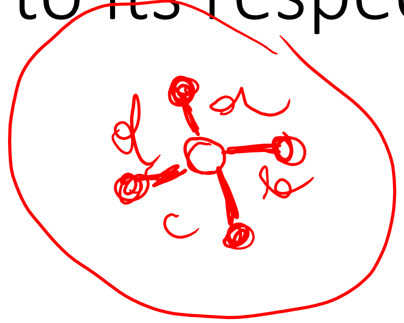
# Outline

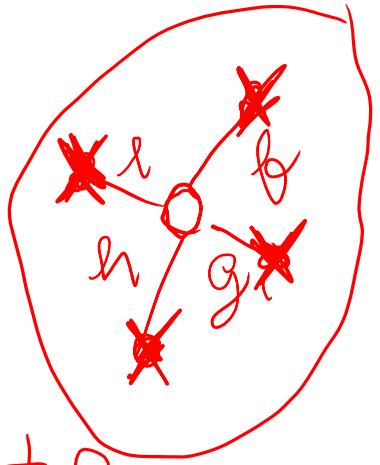
- Clustering
- Distance functions
- K-Means algorithm
- Analysis of K-Means

# Formal statement of the clustering problem

- Given $N$ data points, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \in \mathbb{R}^{N \times D}$
- Find $k$ cluster centers $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_K\} \in \mathbb{R}^{K \times D}$
- And assign each data point $\mathbf{x}_n$ to one cluster $k$ such that $r_{nk} = 1$ and $r_{nj} = 0$ for $j \neq k$ (1-of-K encoding)
- Such that the average square distances from each data point to its respective cluster center (distortion measure) is small:

$$\min_{\boldsymbol{\mu}_k, r_{nk}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2$$





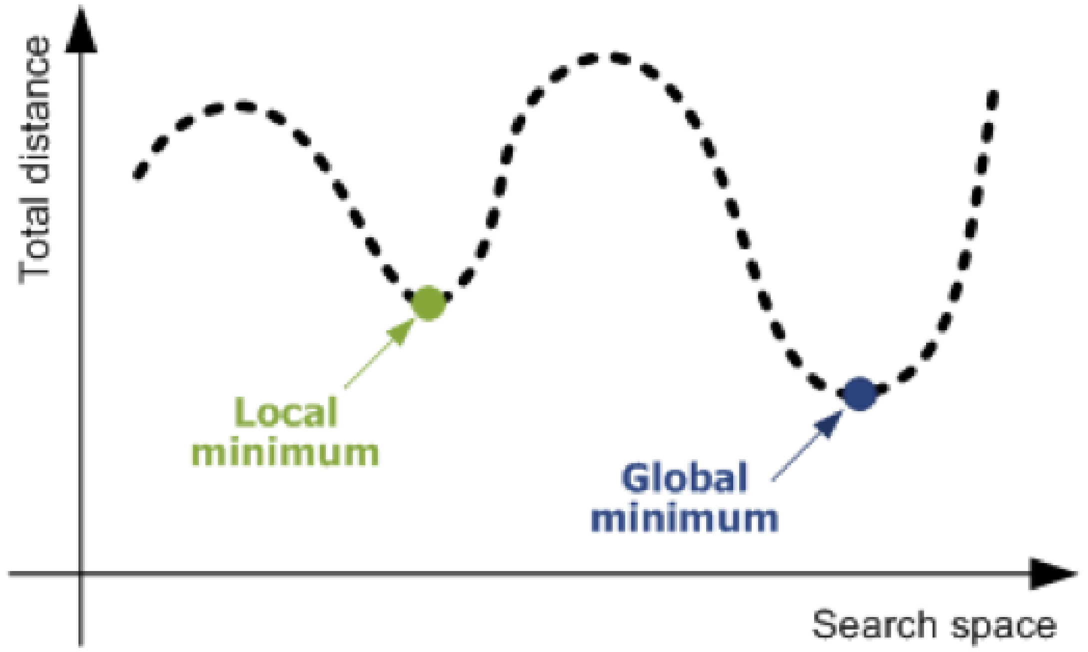
# Clustering is NP-Hard

- Given $N$ data points, $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \in \mathbb{R}^{N \times D}$ and assign each data point $\mathbf{x}_n$ to one cluster $k$ such that $r_{nk} = 1$ and $r_{nj} = 0$ for $j \neq k$ to minimize

$$\min_{\boldsymbol{\mu}_k, r_{nk}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2$$

NP-Hard

- **A search problem over the space of discrete assignments**
  - For all $N$ data point together, there are $K^N$ possibilities
  - The cluster assignment determines cluster centers and vice versa



# Clustering is NP-Hard: example

- Consider the problem of assigning a set of $N = 3$ datapoints $X = \{A, B, C\}$, to $k = 2$ clusters.

| Cluster 1 | Cluster 2 |
|---|---|
| $A, B, C$ | $\{\ \}$ |
| $A, B$ | $C$ |
| $A, C$ | $B$ |
| $B, C$ | $A$ |
| $A$ | $B, C$ |
| $B$ | $A, C$ |
| $C$ | $A, B$ |
| $\{\ \}$ | $A, B, C$ |

- For all N data point together, there are $8 = 2^3 = K^N$ possibilities

# K-means algorithm revisited

- Perform the minimization iteratively in **two steps** where we first minimize our objective wrt $r_{nk}$ keeping $\boldsymbol{\mu}_k$ fixed, and then we minimize the objective wrt $\boldsymbol{\mu}_k$ keeping $r_{nk}$ fixed.
  - **Step 1**: Keeping $\boldsymbol{\mu}_k$ and computing the squared distances between $\mathbf{x}_n$ and $\boldsymbol{\mu}_k$, we can optimize the objective simply by assigning $\mathbf{x}_n$ to the nearest cluster center

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

→ CLOSEST CLUSTER TO THAT POINT

  - **Step 2:** Keeping $r_{nk}$ fixed we can optimize the objective with respect to $\boldsymbol{\mu}_k$ by setting the derivative wrt to $\boldsymbol{\mu}_k$ to zero

$$\frac{\partial}{\partial \boldsymbol{\mu}_k} \sum_{n=1}^{N} r_{nk}(\mathbf{x}_n - \boldsymbol{\mu}_k)^T(\mathbf{x}_n - \boldsymbol{\mu}_k) = 2\sum_{n=1}^{N} r_{nk}(\mathbf{x}_n - \boldsymbol{\mu}_k) = 0 \rightarrow \boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\mathbf{x}_n}{\sum_n r_{nk}}$$

[1 0 0]
K = 3

AVERAGE OF POINTS ASSIGNED TO CLUSTER K

# K-means algorithm data structure: example

Dataset: $\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{11} \\ x_{21} & x_{22} & \cdots & x_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{ND} \end{bmatrix}_{N \times D}$ → $\mathbf{x}_n^T = [5.0 \quad 7.8 \quad \cdots \quad 0.5]$

SPEED, SUCCESSFUL THROWS, BENCH

Cluster assignment: $\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1K} \\ r_{21} & r_{22} & \cdots & r_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ r_{N1} & r_{N2} & \cdots & r_{NK} \end{bmatrix}_{N \times K}$ → $\boldsymbol{r}_n^T = [0 \quad 1 \quad \cdots \quad 0]$

K = 2

Cluster centers: $\mathbf{M} = \begin{bmatrix} \mu_{11} & \mu_{12} & \cdots & \mu_{1D} \\ \mu_{21} & \mu_{22} & \cdots & \mu_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{K1} & \mu_{K2} & \cdots & \mu_{KD} \end{bmatrix}_{K \times D}$ → $\boldsymbol{\mu}_j^T = [2.0 \quad 4.5 \quad \cdots \quad 1.3]$

SPEED, SUCCESSFUL, BENCH

# K-means algorithm revisited

- Initialize $k$ cluster centers $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_K\}$ randomly
- Do
  - Compute dissimilarity between the data points and the cluster centers and decide cluster membership or each point $\mathbf{x}_n$, by assigning it to the nearest cluster center

STEP 1

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2 \\ 0 & otherwise \end{cases}$$

  - Update the cluster center position

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\mathbf{x}_n}{\sum_n r_{nk}}$$

STEP 2

  - While any cluster center has changed

# Let's ask ourselves some questions:

- Will different initializations lead to different results?
  a. Yes
  b. No
  c. Sometimes

- Will the algorithm always stop after some iteration?
  a. Yes
  b. No (we have to set a maximum number of iterations)
  c. Sometimes

# Convergence of K-means

- Will the K-means objective oscillate?

$$\min_{\boldsymbol{\mu}_k, r_{nk}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2$$

L2 - NORM

SQUARED

- The minimum value of the objective is finite
- Each iteration of the K-means algorithm decreases the objective
  - Cluster assignment step decreases the objective

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2 \\ 0 & \quad otherwise \end{cases}$$

CLOSEST CENTER

  - Center update step decreases the objective, because for each cluster we are only summing over the closest points

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \mathbf{x}_n}{\sum_n r_{nk}}$$

AVERAGE OF CLOSEST POINTS

SEARCH NP-HARD $O(K^N)$

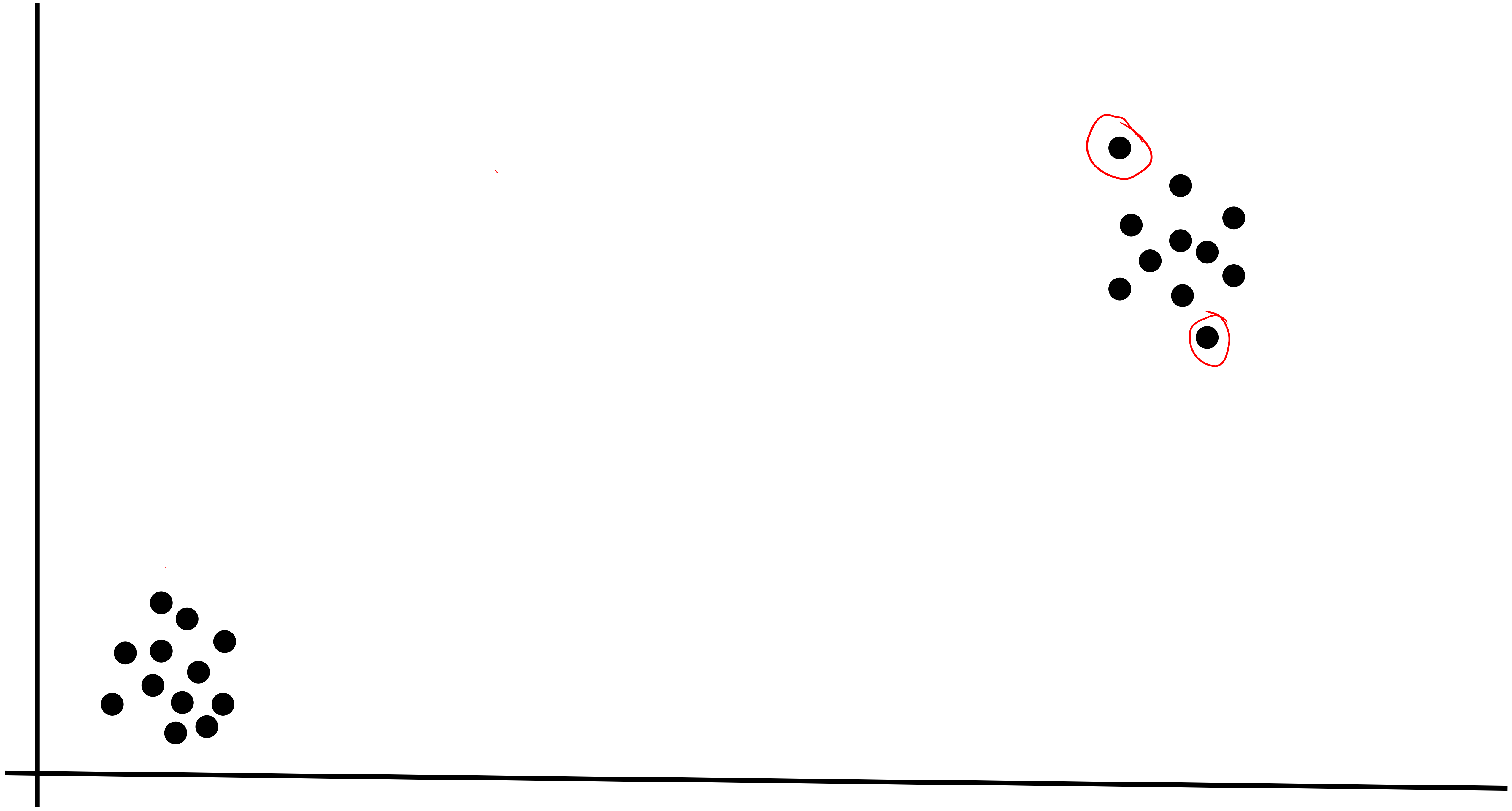
# Time complexity

- Assume computing distance between two instances is $O(D)$ where $D$ is the dimensionality of the vectors.
- Reassigning clusters for all datapoints:
  - $O(KN)$ distance computations (when there is one feature)
  - $O(KND)$ (when there is D features)
- Computing centroids: Each instance vector gets added once to some centroid (finding centroid for each feature): $O(ND)$
- Assume these two steps are each done once for $I$ iterations: $O(IKND)$.



Slide credit: Ray Mooney

# How to initialize the K-means?



# How to choose K?

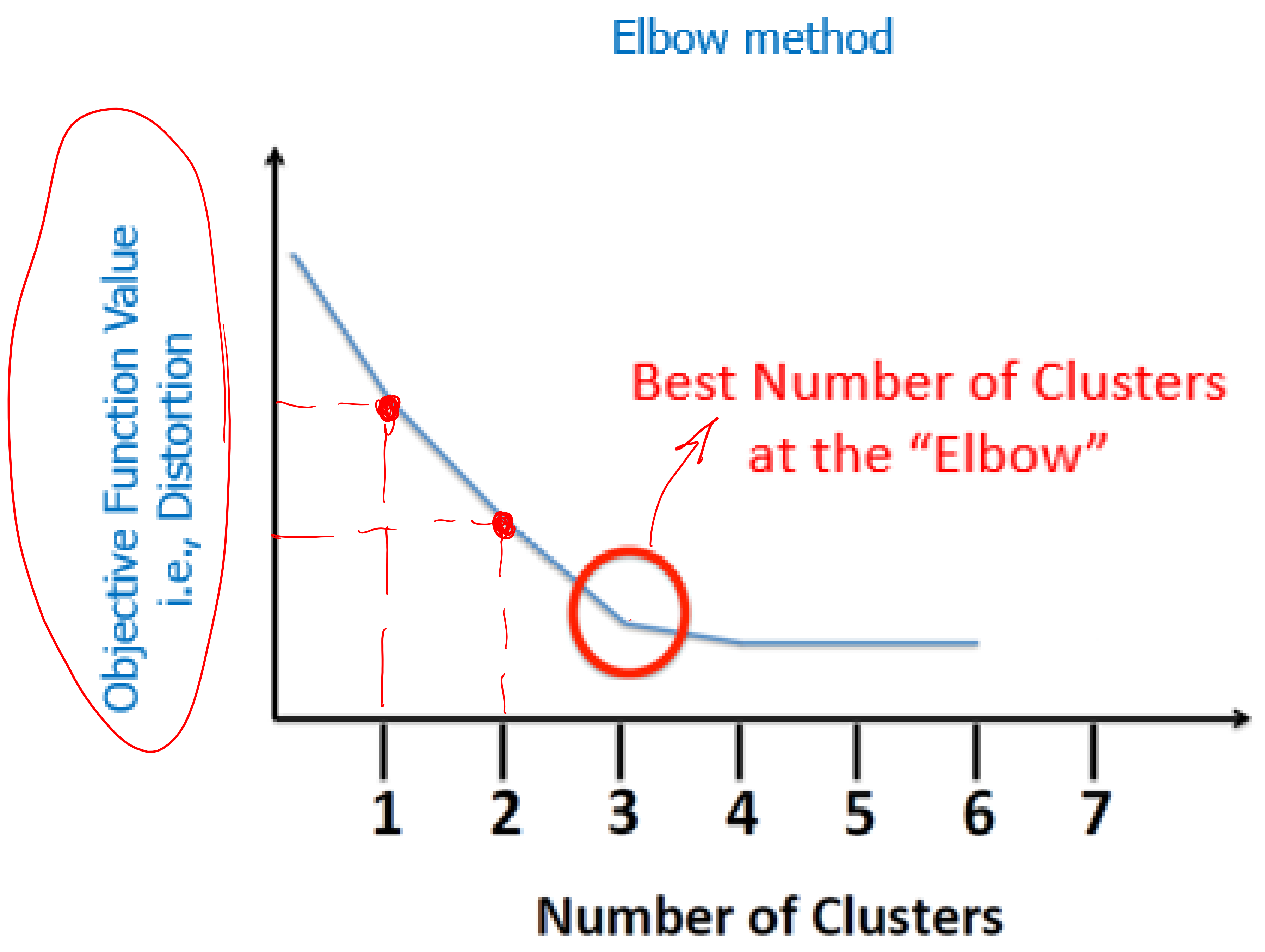



Image credit: Dileka Madushan