

$$\begin{pmatrix} M_1 \\ \vdots \\ M_K \end{pmatrix} \quad \hat{y}_1 = a_1 x + b_1 + c_1 x^2$$

$$\hat{y}_K = a_K x + b_K + c_K x^2$$

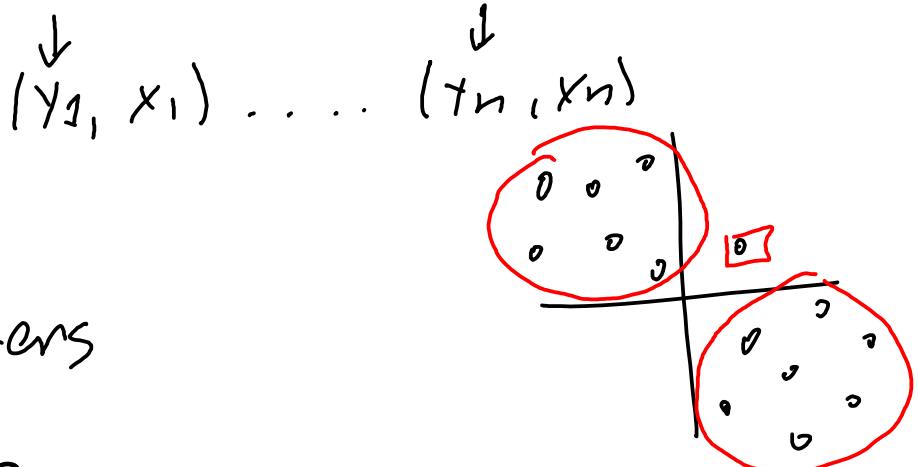
$$M(x) = \frac{\hat{y}_1 + \dots + \hat{y}_K}{K}$$

$$\neq \left(\frac{a_1 + \dots + a_K}{K} \right) x + \left(\frac{b_1 + \dots + b_K}{K} \right) + \dots$$

Clustering

Reference book: Bishop: "Pattern Recognition and Machine Learning" Chapter 9.1

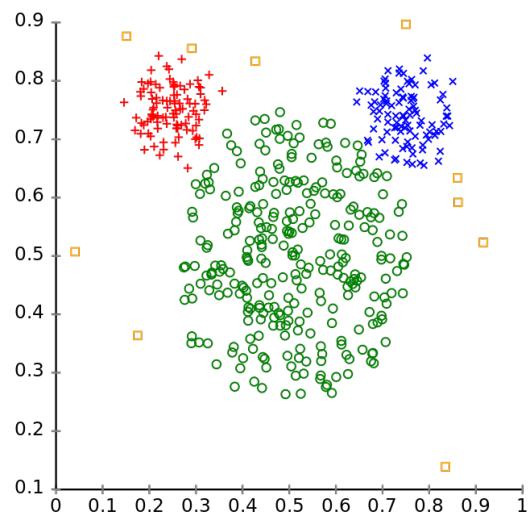
Clustering



- Unsupervised learning
- Generating “classes” *clusters*
- Distance/similarity measures
- Agglomerative methods
- Divisive methods

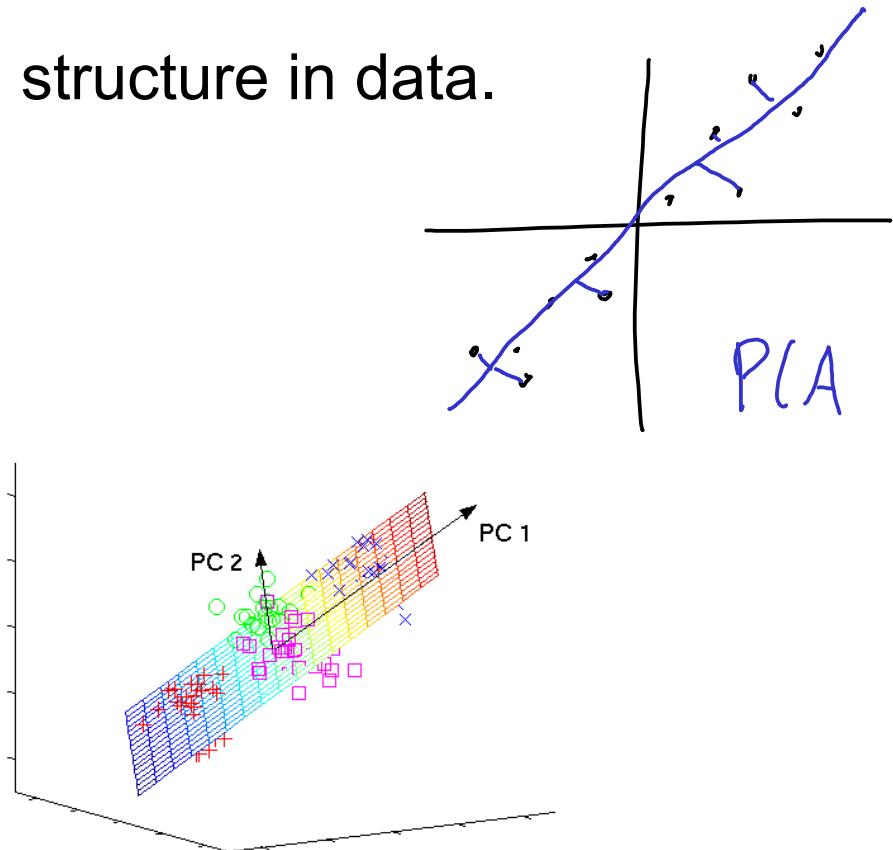
Unsupervised Learning

- No labels/responses. Finding structure in data.
- Dimensionality Reduction.



Clustering

$$T: \mathbb{R}^d \rightarrow \{1, 2, \dots, K\}$$

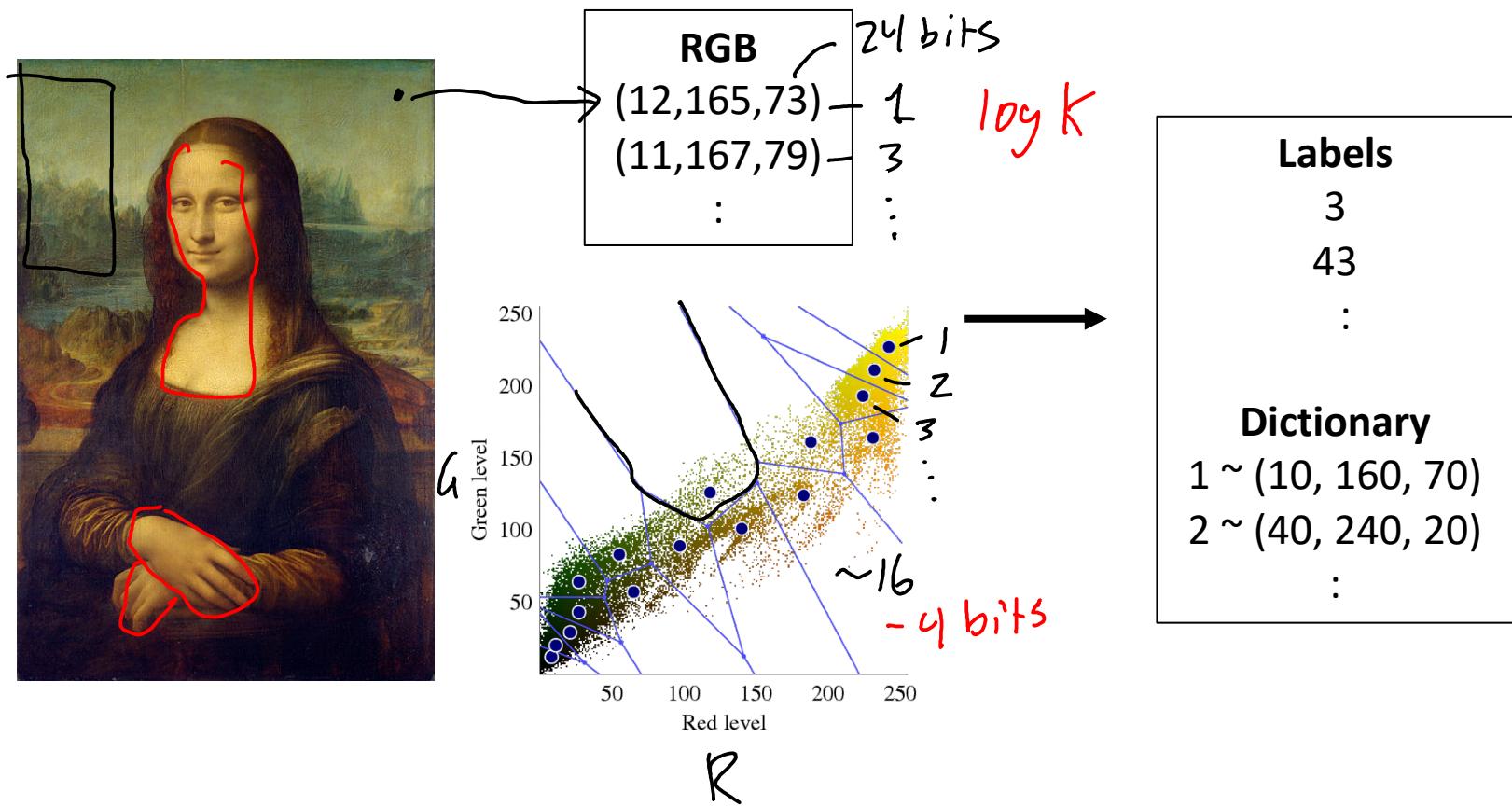


Subspace Learning

$$T: \mathbb{R}^d \rightarrow \mathbb{R}^m, m < D$$

Uses of Unsupervised Learning

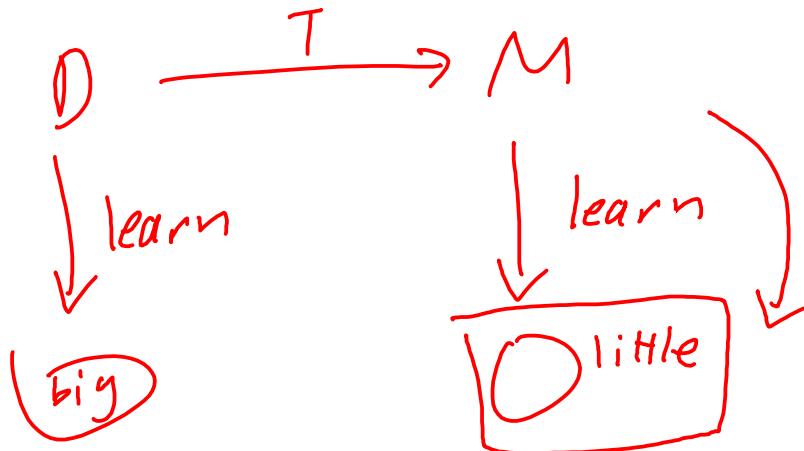
- Data compression



Uses of Unsupervised Learning

- To improve classification/regression (semi-supervised learning)

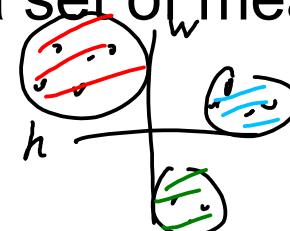
1. From **unlabeled data**, learn a good feature $T: \mathbb{R}^d \rightarrow \mathbb{R}^m$. $m \ll d$
2. To **labeled data**, apply transformation $T: \mathbb{R}^d \rightarrow \mathbb{R}^m$
$$(T(x_1), y_1), \dots, (T(x_N), y_N)$$
3. Perform classification/regression on transformed low-dimensional data.



What is Clustering?

no labels

- Unsupervised learning - no information from teacher
- The process of partitioning a set of data into a set of meaningful (hopefully) sub-classes, called **clusters**
- Cluster:
 - collection of data points that are “similar” to one another and collectively should be treated as group (*it is not the group we learned in linear algebra*)
 - as a collection, are sufficiently different from other groups



What is Clustering

Clustering Problem.

Input. Training data $S_N = \{x_1, x_2, \dots, x_N\}$, $x_n \in \mathbb{R}^d$.

Integer K

Partition

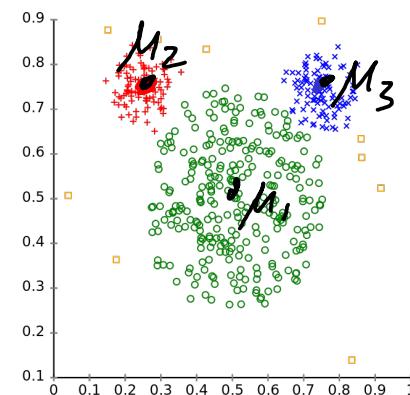
number of clusters
K-means

Output. Clusters $C_1, C_2, \dots, C_K \subset \{1, 2, \dots, N\}$ such that
every data point is in one and only one cluster.

Cluster representatives $\{\mu_1, \dots, \mu_K\}$

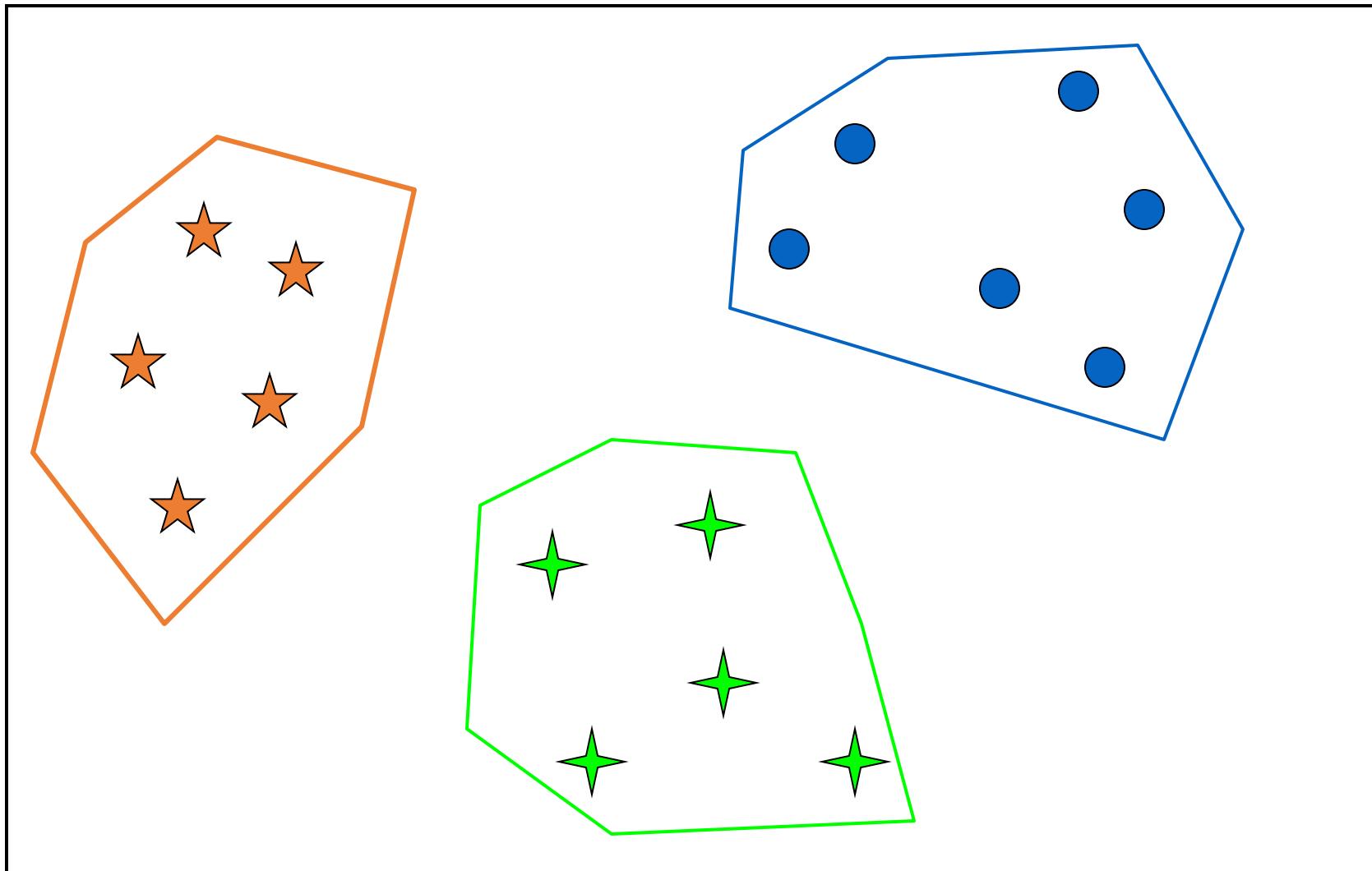
$$\{C_1 \cup C_2 \cup \dots \cup C_K = \{1, \dots, N\}$$

$$C_i \cap C_j = \emptyset \text{ if } i \neq j$$



Some clusters could
be empty!

Clusters

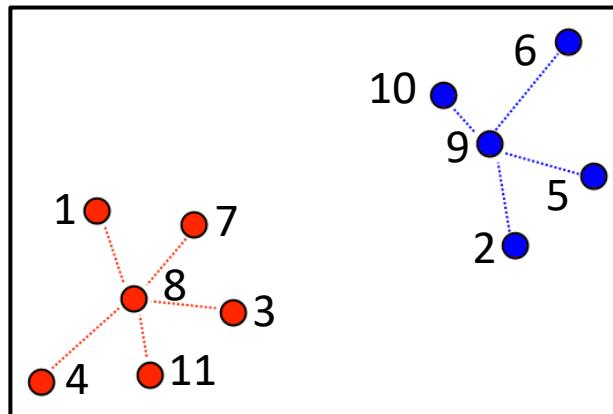


How to Specify a Cluster

- By listing all its elements

$$\mathcal{C}_1 = \{1, 3, 4, 7, 8, 11\}$$

$$\mathcal{C}_2 = \{2, 5, 6, 9, 10\}$$

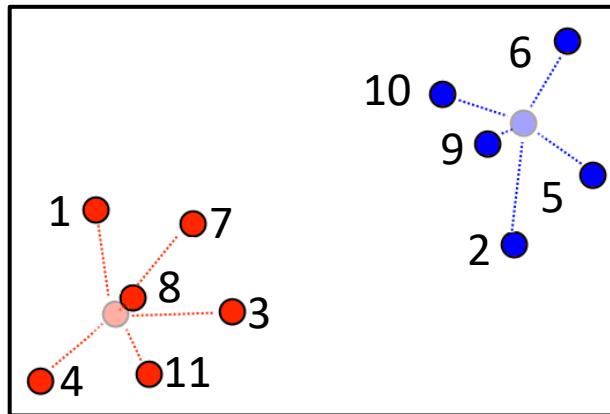


How to Specify a Cluster

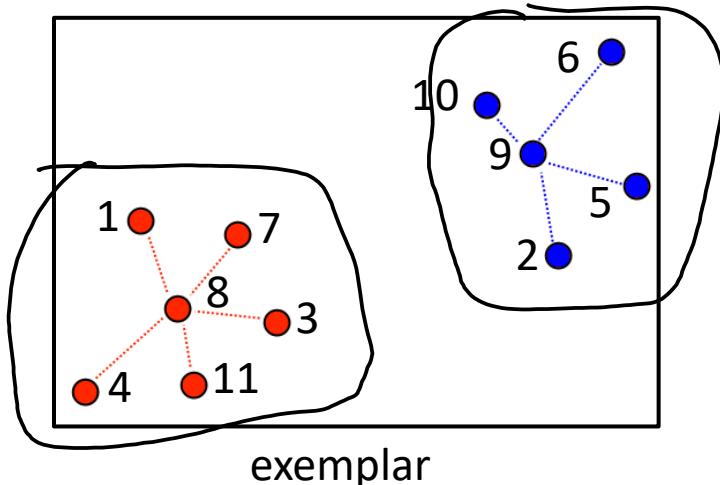
- Using a representative
average/mean of the cluster
 - a point in center of cluster (centroid)
 - b. A point in the training data (exemplar)

$$\sum_{i=1}^k \|\mu - x_i\|_2^2 \Rightarrow \text{choose } \mu = \frac{1}{N} \sum_i x_i$$

Each point x_i will be assigned the closest representative.



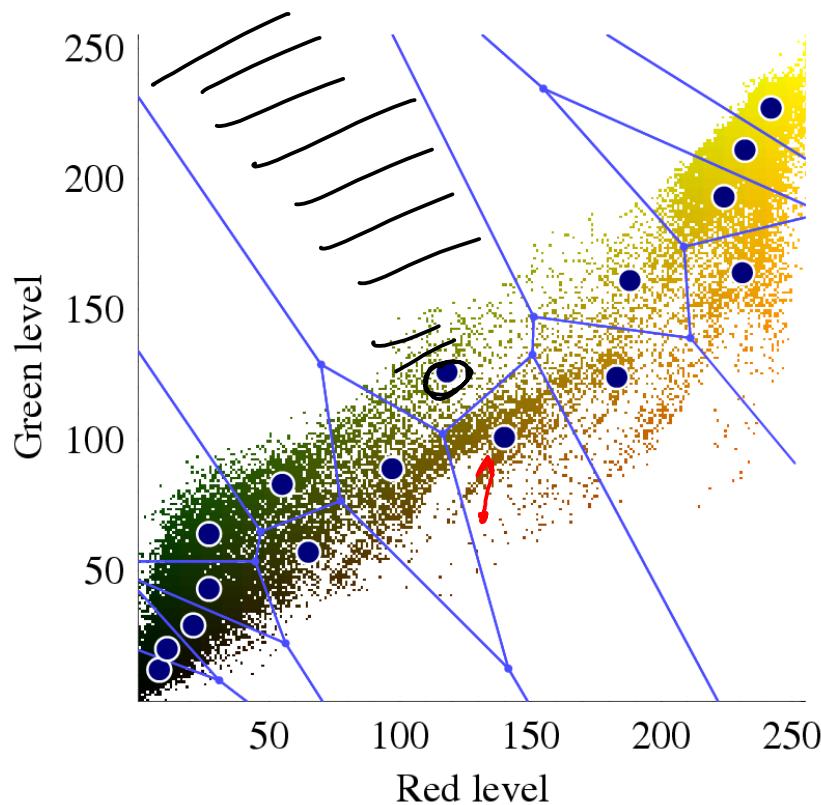
centroid



exemplar

Voronoi Diagram

We can partition all the points in the space into regions, according to their closest representative.



Distance/Similarity Measures

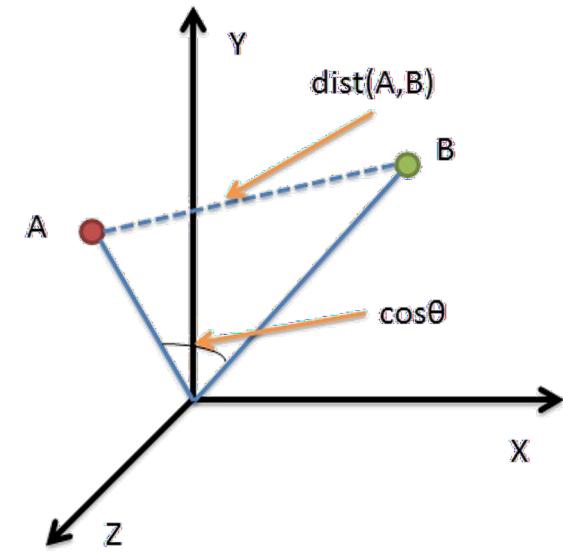
$$\begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{bmatrix} \quad \longleftrightarrow \quad \begin{bmatrix} x'_1 \\ \vdots \\ \vdots \\ x'_n \end{bmatrix}$$

(sometimes called loss functions)

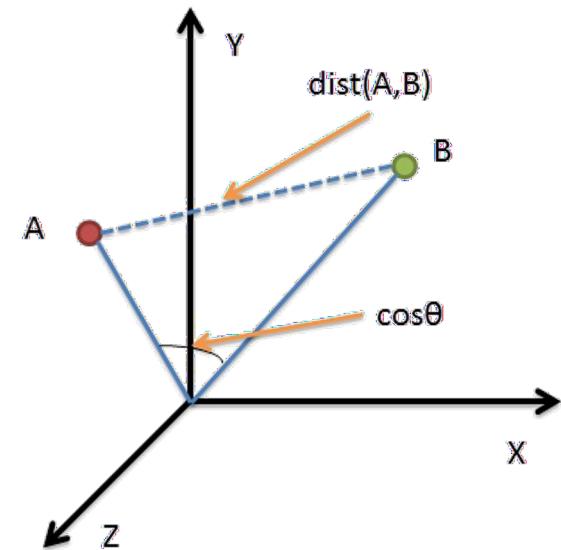
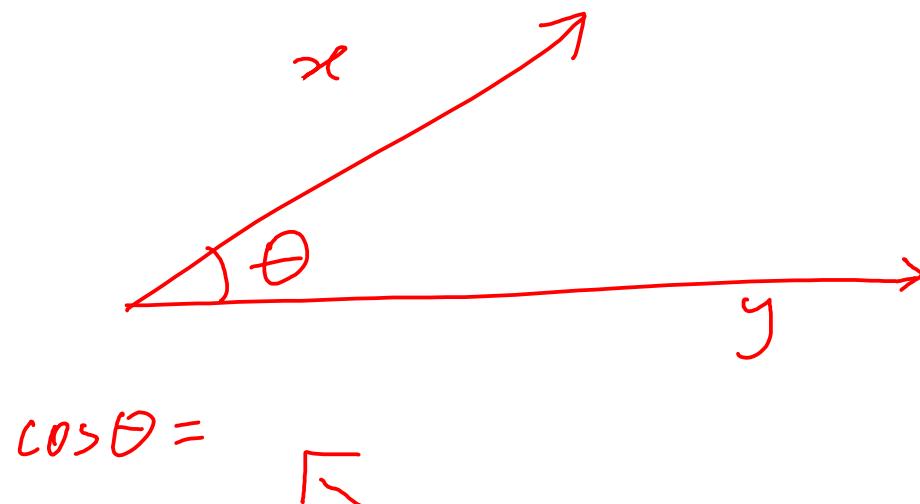
A measure of how close two data points are.
Nearby points (i.e. distance is **small**) are
more likely to belong to the same cluster.

- Euclidean Distance

$$\text{dist}(x, y) = \|x - y\|_2 := \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$



Distance/Similarity Measures



A measure of how alike two data points are.
Similar points (i.e. similarity is **large**) are more likely to belong to the same cluster.

- Cosine Similarity $\cos(x, y) = \frac{x^T y}{\|x\| \|y\|}$

Distance (Similarity) Matrix

- Similarity (Distance) Matrix

- based on the distance or similarity measure we can construct a symmetric matrix of distance (or similarity values)
- (i, j) th entry in the matrix is the distance (similarity) between items i and j

	I_1	I_2	\dots	I_n
I_1	•	d_{12}	\dots	d_{1n}
I_2	d_{21}	•	\dots	d_{2n}
\vdots	\vdots	\vdots	\vdots	\vdots
I_n	d_{n1}	d_{n2}	\dots	•

d_{ij} = similarity (or distance) of D_i to D_j

Note that $d_{ij} = d_{ji}$ (i.e., the matrix is symmetric). So, we only need the lower triangle part of the matrix.

The diagonal is all 1's (similarity) or all 0's (distance)

Example: Term Similarities in Documents

	T1	T2	T3	T4	T5	T6	T7	T8
Doc1	0	4	0	0	0	2	1	3
Doc2	3	1	4	3	1	2	0	1
Doc3	3	0	0	0	3	0	3	0
Doc4	0	1	0	3	0	0	2	0
Doc5	2	4	2	3	1	4	0	2

$$sim(T_i, T_j) = \sum_{k=1}^N (w_{ik} \cdot w_{jk})$$

相似度定义：对应元素乘积的和

Term-Term
Similarity Matrix

	T1	T2	T3	T4	T5	T6	T7
T2	7						
T3	16	8					
T4	15	12	18				
T5	14	3	6	6			
T6	14	18	16	18	6		
T7	9	6	0	6	9	2	
T8	7	17	8	9	3	16	3

Similarity (Distance) Thresholds

- A similarity (distance) threshold may be used to mark pairs that are “sufficiently” similar

	T1	T2	T3	T4	T5	T6	T7
T2	7						
T3	16	8					
T4	15	12	18				
T5	14	3	6	6			
T6	14	18	16	18	6		
T7	9	6	0	6	9	2	
T8	7	17	8	9	3	16	3

	T1	T2	T3	T4	T5	T6	T7
T1	0						
T2	0						
T3	1	0					
T4	1	1	1				
T5	1	0	0	0			
T6	1	1	1	1	0		
T7	0	0	0	0	0	0	
T8	0	1	0	0	0	1	0

Using a threshold value of 10 in the previous example

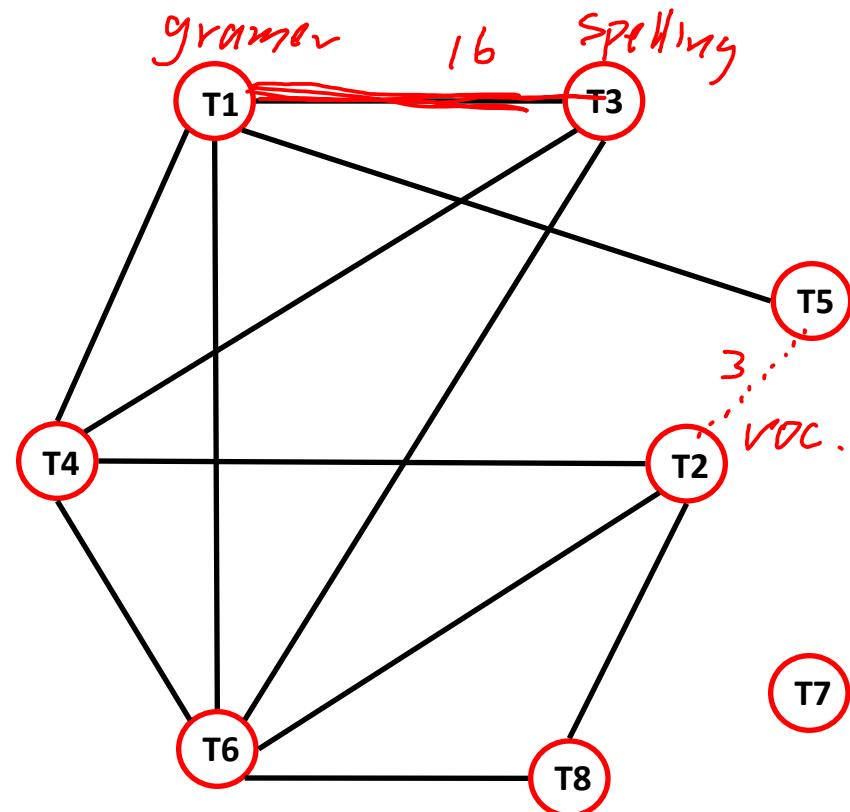
Similarity

Graph Representation

- The similarity matrix can be visualized as an undirected graph
 - each item is represented by a node, and edges represent the fact that two items are similar (a 1 in the similarity threshold matrix)

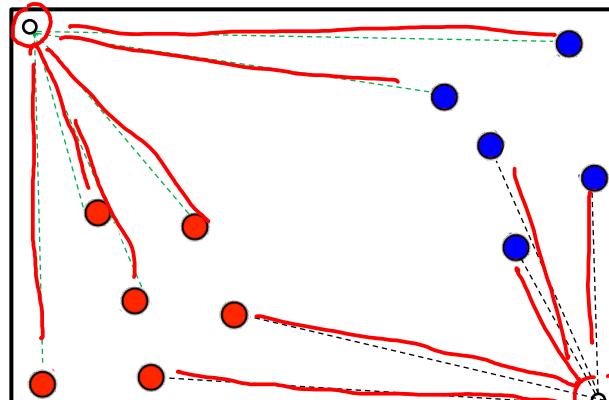
	T1	T2	T3	T4	T5	T6	T7
T2	0						
T3	1	0					
T4	1	1	1				
T5	1	0	0	0			
T6	1	1	1	1	0		
T7	0	0	0	0	0	0	
T8	0	1	0	0	0	1	0

If no threshold is used, then matrix can be represented as a weighted graph



Training Loss

Sum of squared distances to closest representative.

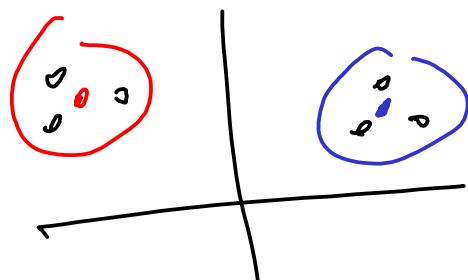


$$\text{loss} \approx 11 \times (1)^2 = 11$$

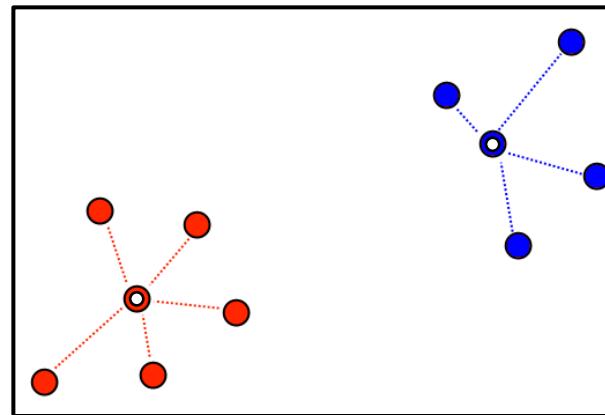
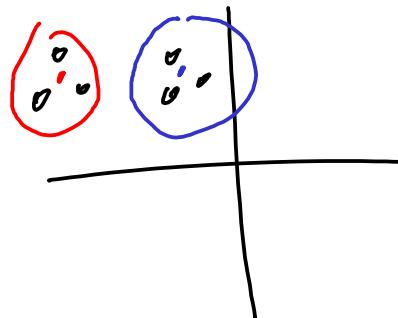
assume length of each
edge is about 1

Training Loss

Sum of squared distances to closest representative (cluster center).



// ✓



$$\text{loss} \approx 9 \times (0.1)^2 = 0.09 < //$$

assume length of each
edge is about 0.1

Training loss

Optimizing both clusters and representatives.

$$K=5$$

$$\begin{matrix} & x_1 & x_2 & \dots & x_5 \\ \downarrow & 3 & 4 & & \\ 3 & & & & \\ \downarrow & 0 & 1 & & \\ 0 & 0 & 0 & & \\ \downarrow & 0 & 0 & & \\ 0 & 0 & 0 & & \end{matrix}$$

$$\mathcal{L}(R, M; \mathcal{S}_n)$$

One hot encoding

$$\mathcal{L}(R, M; \mathcal{S}_n) = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - \mu_k\|_2^2$$

where

$$\mathcal{S}_n = \{x_1, x_2, \dots, x_N\}, x_i \in \mathbb{R}^d$$

r_{nk} = {1 if x_n assigned to cluster k , 0 else}

$$r_{nk} = \begin{cases} 1 & x_n \text{ to assigned cluster } k \\ 0 & \text{else} \end{cases}$$

$$R = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \vdots & & & & \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$r_{nk} \in \{0,1\}$ denotes which of the K clusters data point x_n is assigned to. If x_n is assigned to cluster k then $r_{nk} = 1$, and $r_{nj} = 0$ if $j \neq k$.

$R = \{r_{nk}\} \in \{0,1\}^{N \times K}, n = 1, \dots, N, k = 1, \dots, K$

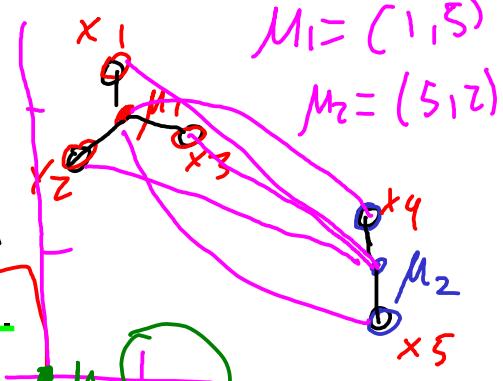
$M = [\mu_1, \dots, \mu_K]^T$
representatives

$$r_{32} = 0 \quad R = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Goal: find values for R and M so as to minimize \mathcal{L} .

$M = \begin{bmatrix} 1.5 \\ 5.2 \\ 5 \end{bmatrix}$

$M_1 = (1, 5)$
 $M_2 = (5, 2)$
 $M_3 = (5, 1)$



$$\mathcal{L}(R, M; s_n) = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - M_k\|_2^2 = \sum_{k=1}^K \sum_{x \in C_k} \|x - M_k\|_2^2$$

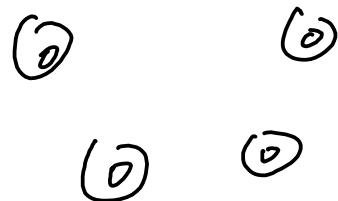
$\uparrow k^{\text{th}} \text{ Cluster}$

Basic Clustering Methodology

Two approaches:

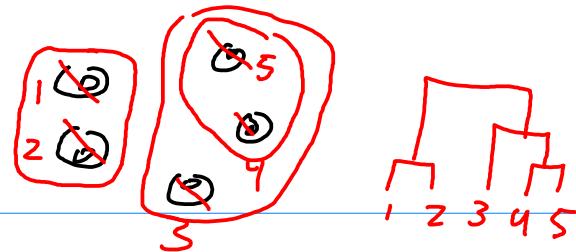
Agglomerative: pairs of items/clusters are successively linked to produce larger clusters

~~Divisive~~ (partitioning): items are initially placed in one cluster and then divided into separate groups



(hierarchical?)

Agglomerative Example:



Need to define an update rule for how to combine two clusters together.

Single-link Clustering: Distance between clusters to be shortest distance between any two points in each cluster:

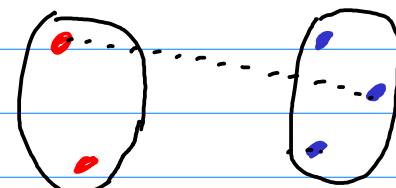
$$d(C_i, C_j) = \min_{x \in C_i} \min_{y \in C_j} d(x, y)$$

~~Hausdorff?~~



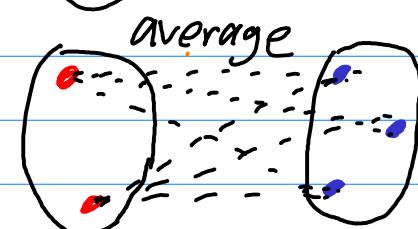
Complete-link Clustering: Distance between clusters to be longest distance between two points from each cluster:

$$d(C_i, C_j) = \max_{x \in C_i} \max_{y \in C_j} d(x, y)$$



Average link clustering: Distance is average distance between clusters

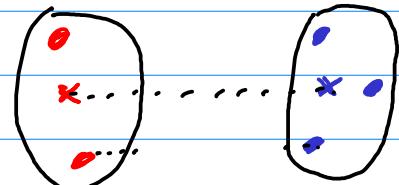
$$d(C_i, C_j) = \frac{1}{|C_i| \cdot |C_j|} \sum_{x \in C_i} \sum_{y \in C_j} d(x, y)$$



Centroid clustering: Distance is distance between cluster centroids

$$d(C_i, C_j) = \| \mu_i - \mu_j \|$$

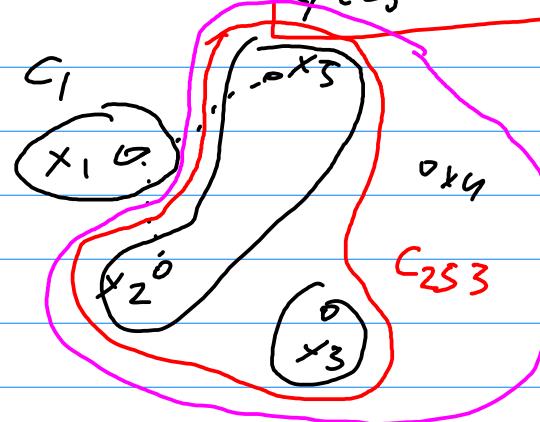
$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \quad \mu_j = \frac{\sum_{x \in C_j} x}{|C_j|}$$



Single link clustering

$$d(c_i, c_j) = \min_{x \in c_i} d(x, y) \quad \forall y \in c_j$$

	x_1	x_2	x_3	x_4
x_2	7			
x_3	16	8		
x_4	15	12	18	
x_5	19	3	6	6



Merge C_2 C_5

	C_1	C_{25}	C_3
C_{25}	7		
C_3	16	16	
C_4	15	6	18

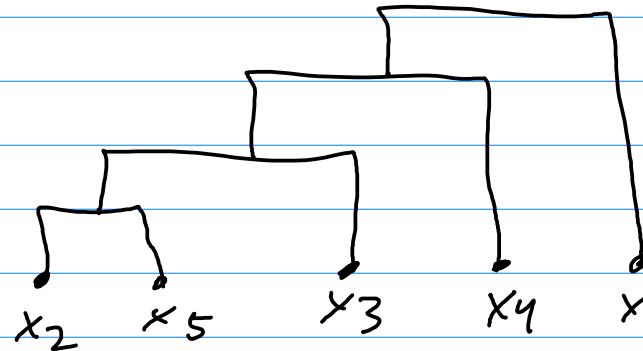
Merge C_{253} with 4

7, 14, 16, 15

C_1 7

Merge C_{25} C_3

	C_1	C_{253}
C_{253}	7	7
C_4	15	6



Complete Link

$$d(c_i, c_j) = \max_{\substack{x \in c_i \\ y \in c_j}} d(x, y)$$

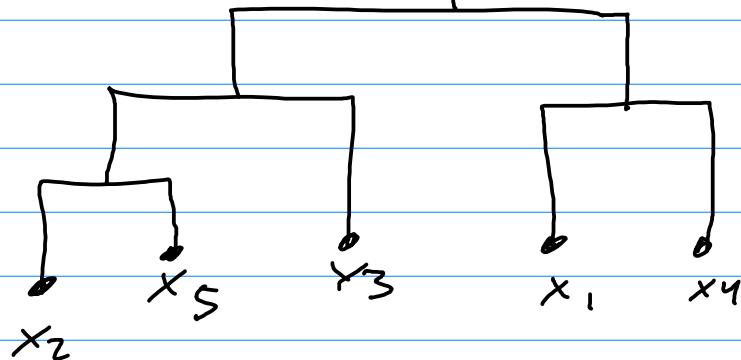
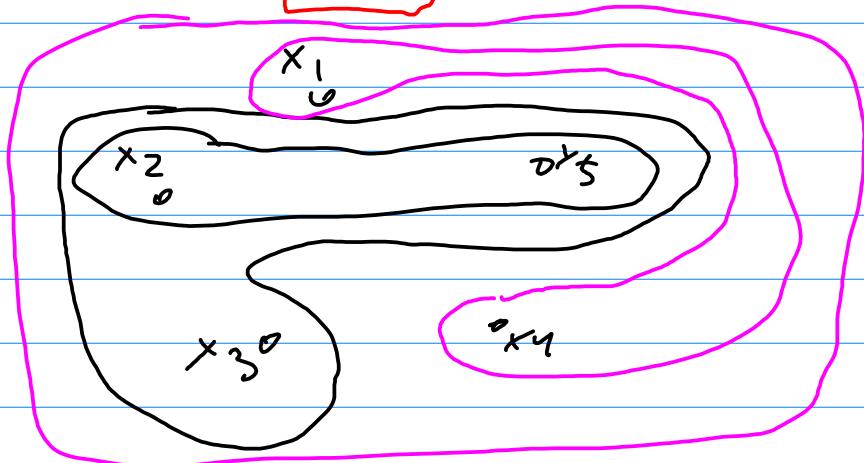
	x_1	x_2	x_3	x_4
x_2	7			
x_3	16	8		
x_4	15	12	18	
x_5	14	3	6	6

Merge $C_2 \oplus C_5$

	C_1	C_{25}	C_3
C_{25}	14	8	
C_3	16	16	
C_4	15	12	18

Merge $C_3 \oplus C_{25}$

	C_1	C_{253}
C_{253}	16	
C_4	15	18



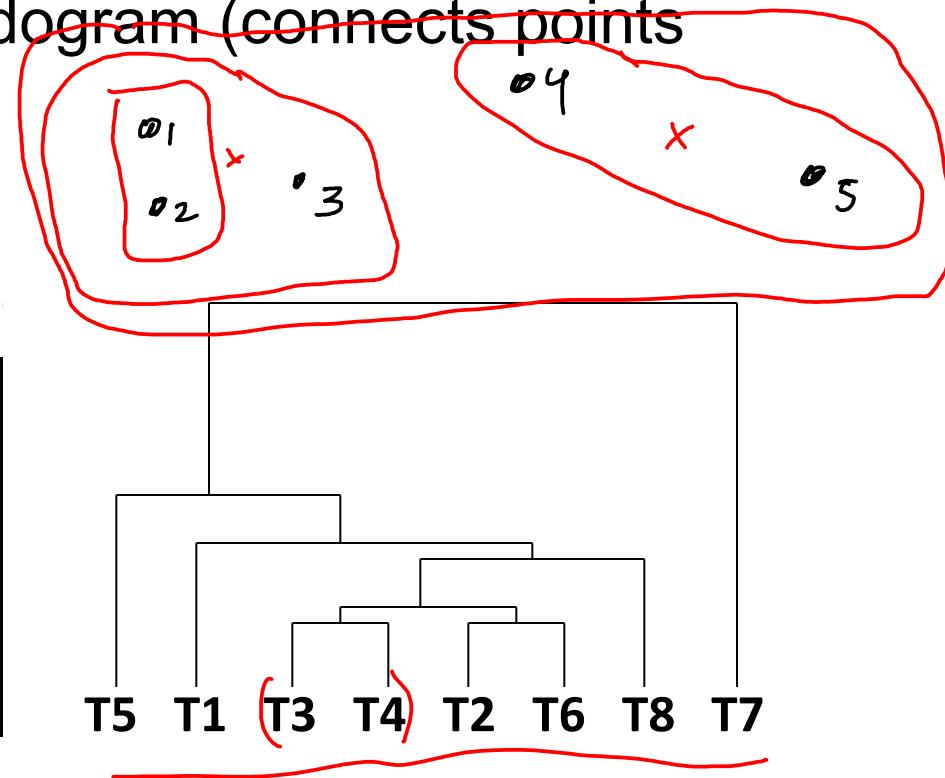
$$d(C_{253}, C_1) = \max \left\{ z^1, z^4, s^1, s^4, 3^1, 3^4 \right\} = \max \{ 7, 12, 14, 6, 16, 18 \}$$

Hierarchical/Agglomerative Methods

- Based on some methods of representing hierarchy of data points
- One idea: hierarchical dendrogram (connects points based on similarity)

$$S(t_2, \{t_3, t_4\}) = 10$$

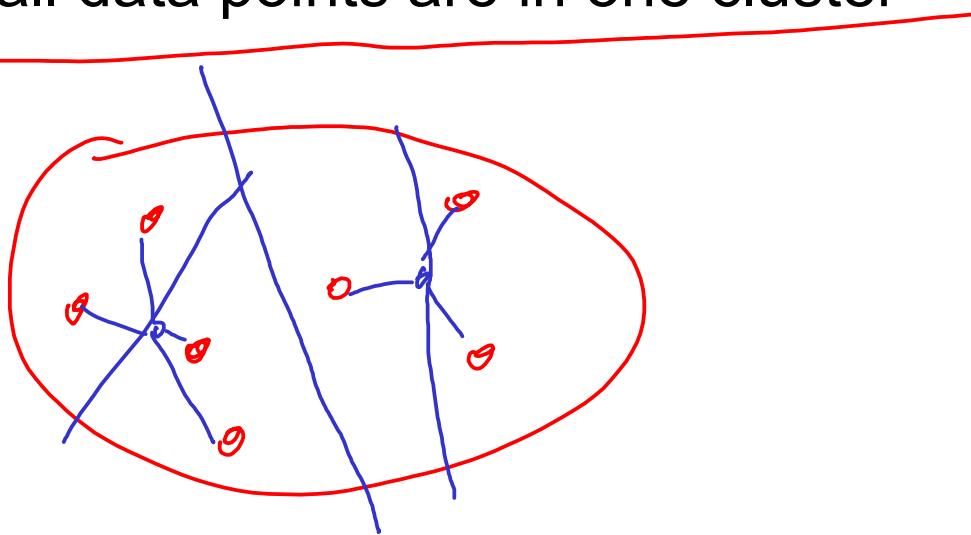
	T1	T2	T3	T4	T5	T6	T7
T2	7						
T3	16	8					
T4	15	12	18				
T5	14	3	6	6			
T6	14	18	16	18	6		
T7	9	6	0	6	9	2	
T8	7	17	8	9	3	16	3



Hierarchical Agglomerative

- Compute distance matrix
- Put each data point in its own cluster
- Find most similar pair of clusters
 - merge pairs of clusters (show merger in dendrogram)
 - update similarity matrix
 - repeat until all data points are in one cluster

Divisive .



K-Means

Optimization Algorithm

$$\mathcal{L}(R, M, S_n)$$

Goal. Minimize $\mathcal{L}(x, y)$.

$$\mathcal{L}(R, M; \mathcal{S}_n) = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - \mu_k\|^2$$

|
representative
responsibilities

Coordinate Descent (Optimization).

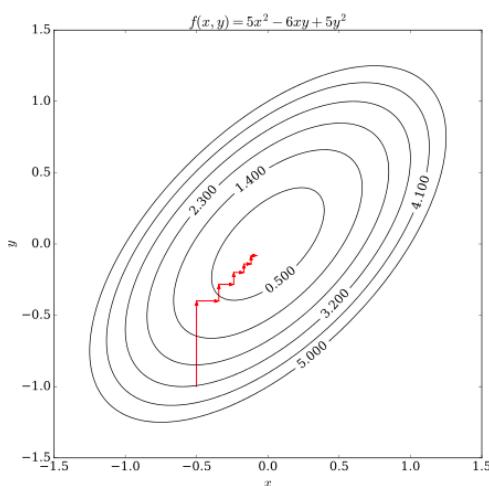
Repeat until convergence:

1. Find optimal x while holding y constant.
2. Find optimal y while holding x constant.

R

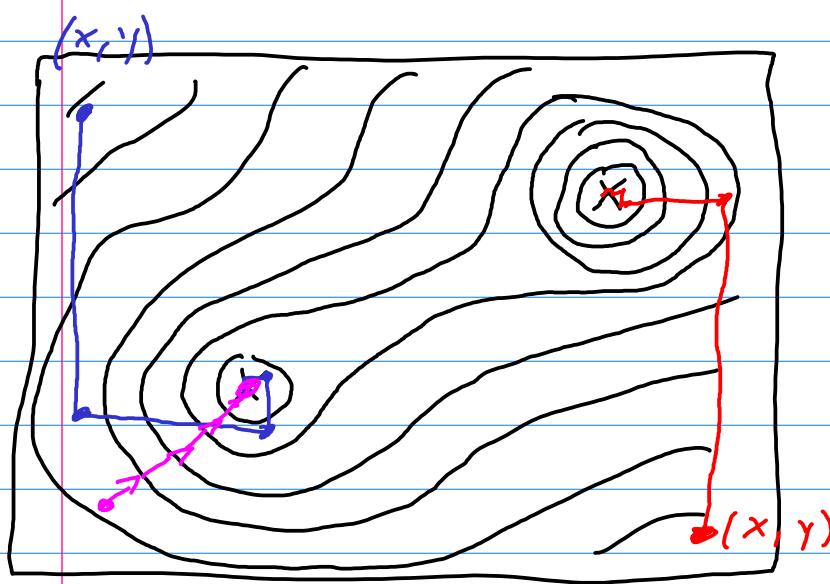
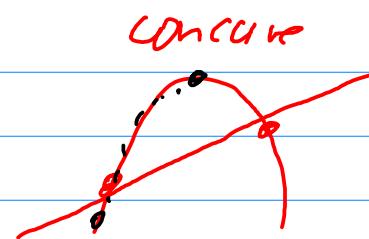
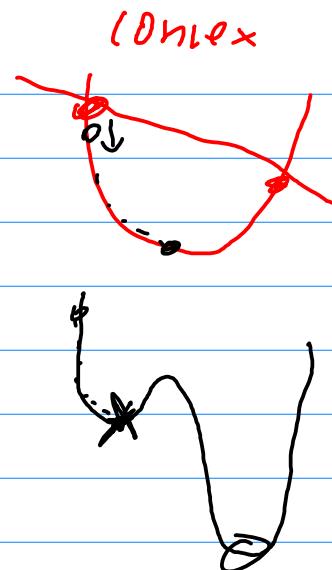
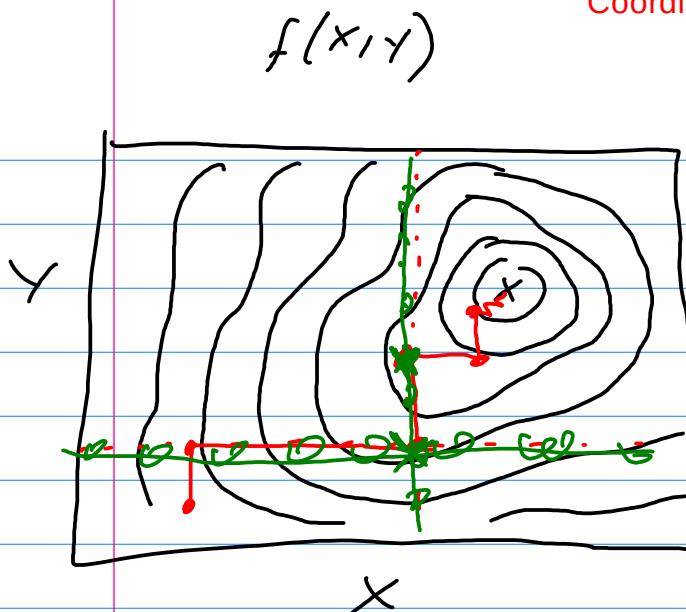
R

M



Coordinate descent is an optimization algorithm that successively minimizes along coordinate directions to find the minimum of a function.

Coordinate optimisation



Optimization Algorithm

Coordinate Descent (Optimization)

Repeat until convergence:

- Find best clusters given centres
- Find best centres given clusters

$$\mathcal{L}(\mathbf{R}, \underline{\mathbf{M}}; \mathcal{S}_n) = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

$$\mathbf{M} = [\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K]^T$$

Optimization Algorithm

NP-Hard

1. Initialize centers μ_1, \dots, μ_K from the data.
2. Repeat until no further change in training loss:

- a. For each $n \in \{1, \dots, N\}$,

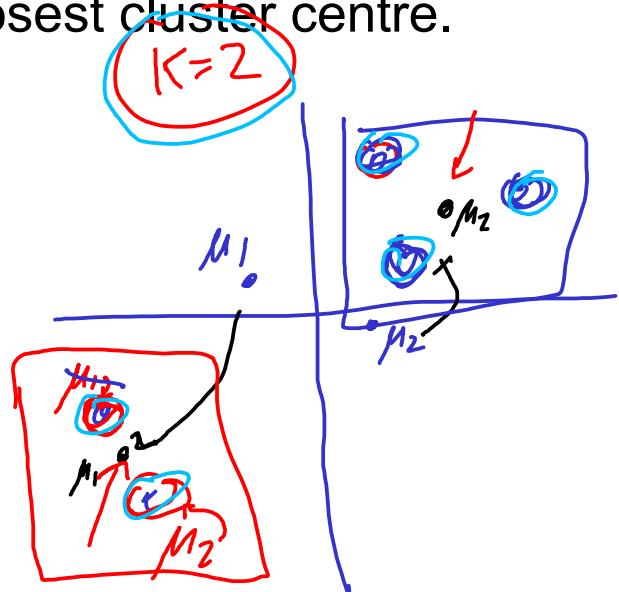
$$r_{nk} = \begin{cases} 1, & \text{if } k = \arg \min_j \|x_n - \mu_j\|^2 \\ 0, & \text{otherwise.} \end{cases}$$

We assign the n th data point to the closest cluster centre.

- b. For each $k \in \{1, \dots, K\}$,

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

We recompute cluster means.

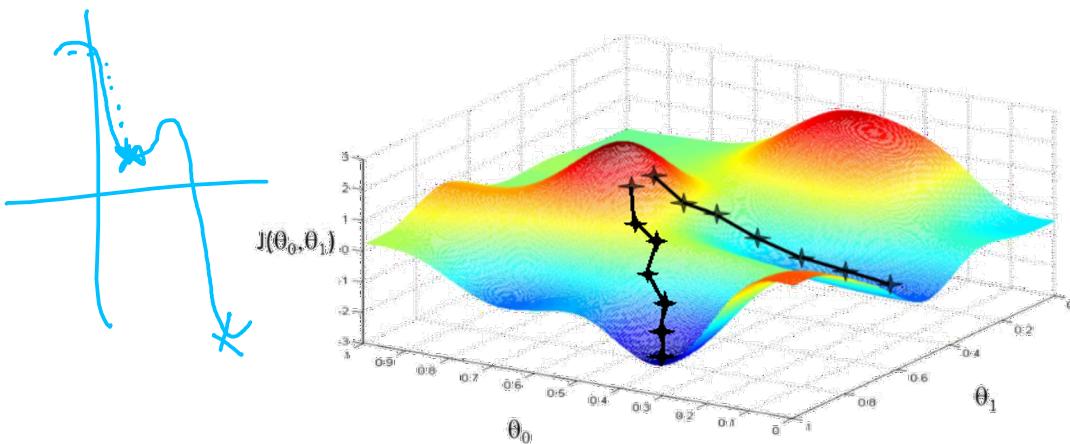


Convergence

$$\mathcal{L}(R, M, S_n)$$



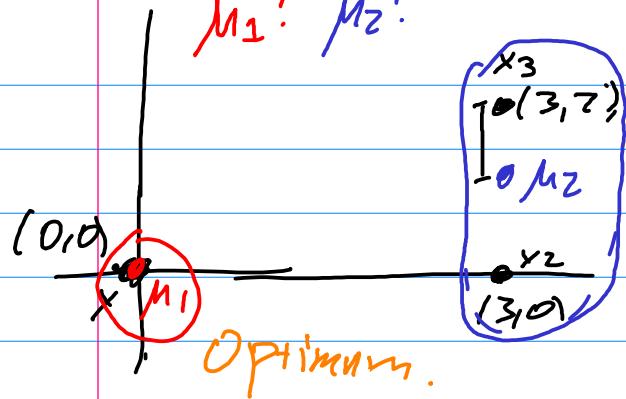
- Training loss always decreases in each step (coordinate descent).
- Converges to local minimum, not necessarily global minimum.



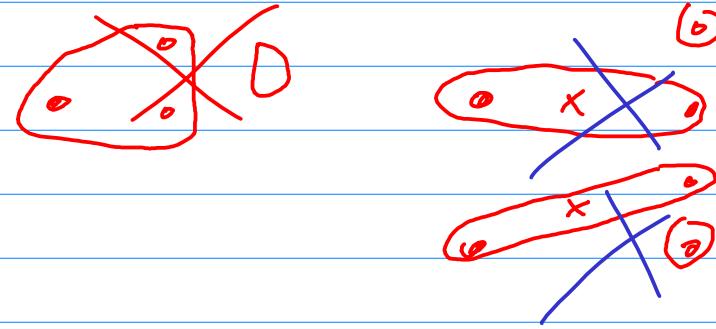
Repeat algorithm over many initial points, and pick the configuration with the smallest training loss.

$K=2$

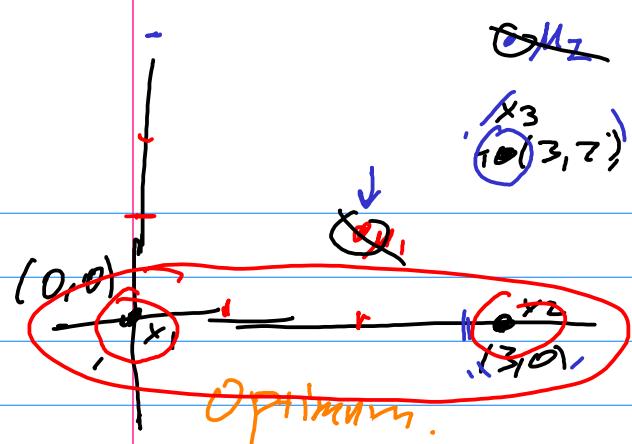
$\mu_1?$ $\mu_2?$



$$S_n = \left\{ \underline{(0,0)}, \underline{(3,0)}, \underline{(3,2)} \right\}$$



$$\begin{aligned} \mathcal{L}(R, M; S_n) &= \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - \mu_k\|_2^2 \\ &= \|x_1 - \mu_1\|_2^2 + \|x_2 - \mu_2\|_2^2 + \|x_3 - \mu_2\|_2^2 \\ &= 0 + 1^2 + 1^2 = 2 \end{aligned}$$



$$\boxed{\begin{aligned} \mu_1 &= (2, 1) \\ \mu_2 &= (3, 3) \end{aligned}}$$

$$r_{12} \quad r_{11}$$

$$r_{nh} = \begin{cases} 1 & x_n \text{ assigned } C_k \\ 0 & \text{else} \end{cases}$$

$$= \begin{cases} 1 & k = \arg \min_j \|x_n - \mu_j\| \\ 0 & \text{else.} \end{cases}$$

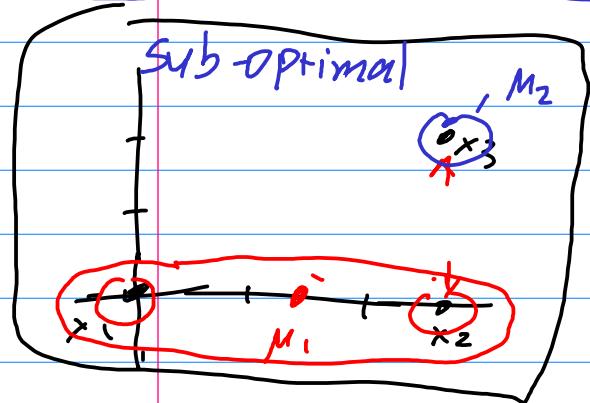
$$\|x_2 - \mu_1\|^2 = 2$$

$$\|x_2 - \mu_2\|^2 = 9$$

$$\|x_1 - \mu_1\|_2^2 = 2^2 + 1^2 = 5$$

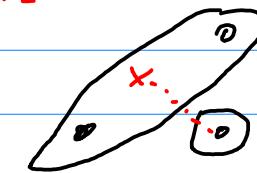
$$\|x_1 - \mu_2\|_2^2 = 1^2 + 3^2 = 10 > 5$$

$$x_1 \rightarrow \mu_1$$



$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}} \Leftarrow \mu_1 = \frac{\sum_{x \in C_1} x}{|C_1|} = \frac{x_1 + x_2}{2} = (1.5, 0)$$

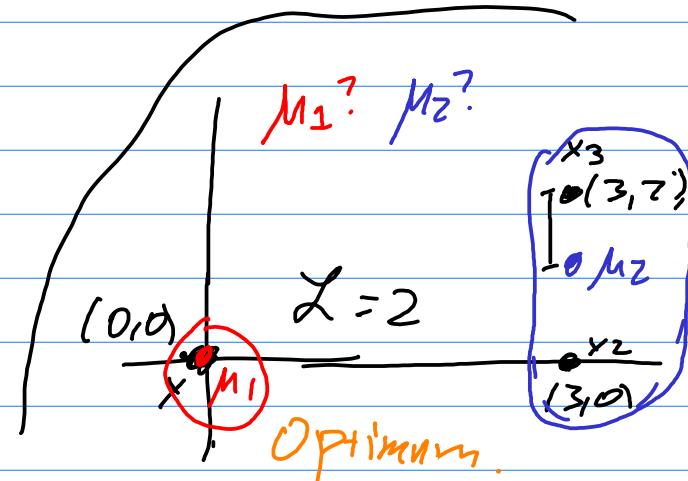
$$\mu_2 = x_3$$



$$\mathcal{L} = \|x_1 - \mu_1\|_2^2 + \|x_2 - \mu_1\|_2^2 + \|x_3 - \mu_2\|_2^2$$

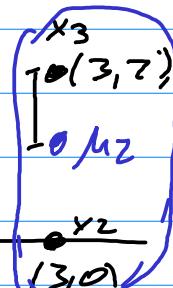
$$= 2.25 + 2.25 + 0$$

$$= \underline{4.5} > 2$$



$$\mathcal{L} = 2$$

Optimum.



Convergence of K-means.

- Training loss always decreases after every update step. Kmeans must terminate in finite time.

Holding M fixed

$$\hat{R}_{nh} = \begin{cases} 1 & \text{if } \\ 0 & \text{else.} \end{cases}$$

$$h = \underset{j}{\operatorname{argmin}} \left(\|x_n - \mu_j\|_2^2 \right)$$

$$R = \begin{bmatrix} i_2 & i_2 \\ 000 & 100 \\ 001 & 000 \\ \vdots & \vdots \end{bmatrix}$$

$$\mathcal{L}(R, M; S_n) \leq \mathcal{L}(R', M; S_n) \text{ for any } R' = \begin{bmatrix} i_2 & i_2 \\ -1 & 1 \\ -1 & -1 \\ \vdots & \vdots \end{bmatrix}$$

$$\mathcal{L}(R, M; S_n) = \sum_{n=1}^N \sum_{k=1}^K \underbrace{R_{nh}}_{\text{any sequence } i_1, i_2, \dots, i_N} \|x_n - \mu_k\|_2^2$$

$$= \sum_{n=1}^N \min_{j=1}^K \|x_n - \mu_j\|_2^2$$

$$\leq \sum_{n=1}^N \min_{j=1}^K \|x_n - \mu_j\|_2^2 \quad \text{any sequence } i_1, i_2, \dots, i_N$$

$$= \sum_{n=1}^N \sum_{k=1}^K R'_{nh} \|x_n - \mu_k\|_2^2 \text{ for some matrix } R'.$$

$$= \mathcal{L}(R', M; S_n)$$

Convergence of K-means.

- Training loss always decreases after every update step. Kmeans must terminate in finite time.

Hold R fixed. Update M

$$M_{Rk} = \frac{\sum_n v_{nk} x_n}{\sum_n v_{nk}}$$

Convex in M with fixed R .

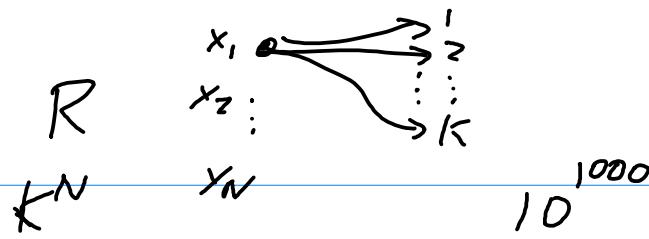
$$\mathcal{L}(R, M; s_n) = \sum_{n=1}^N \sum_{k=1}^K R_{nk} \|x_n - M_k\|_2^2$$

$\frac{\partial \mathcal{L}(R, M; s_n)}{\partial M_k} = 0$, solve for (M_k) .

$\|x_n - M_k\|_2^2$

best choice

$$\mathcal{L}(R, M, s_n) \leq \mathcal{L}(R, M', s_n) \text{ for any other } M'.$$



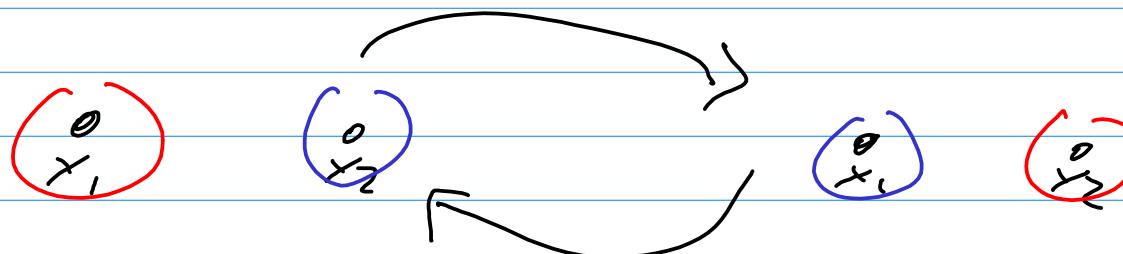
suppose k-means could run forever.

Finitely many configurations, (K^N) many.

Eventually, choose R now, and choose R again in the future, for some matrix R

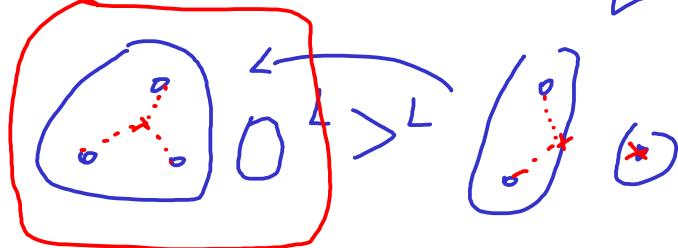
$$\underline{\mathcal{L}(R, M, s_n)} \overset{6}{\cancel{>}} \underline{\mathcal{L}(R, M, s_n)} \overset{6}{\cancel{<}} .$$

Contradiction.



Discussion

Initialization



$$\mu_k = \frac{\sum_{n} r_{nk} x_n}{\sum_{n} r_{nk}}$$

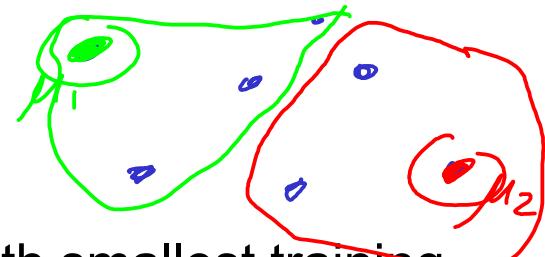
$\mu_1?$

$\mu_2?$

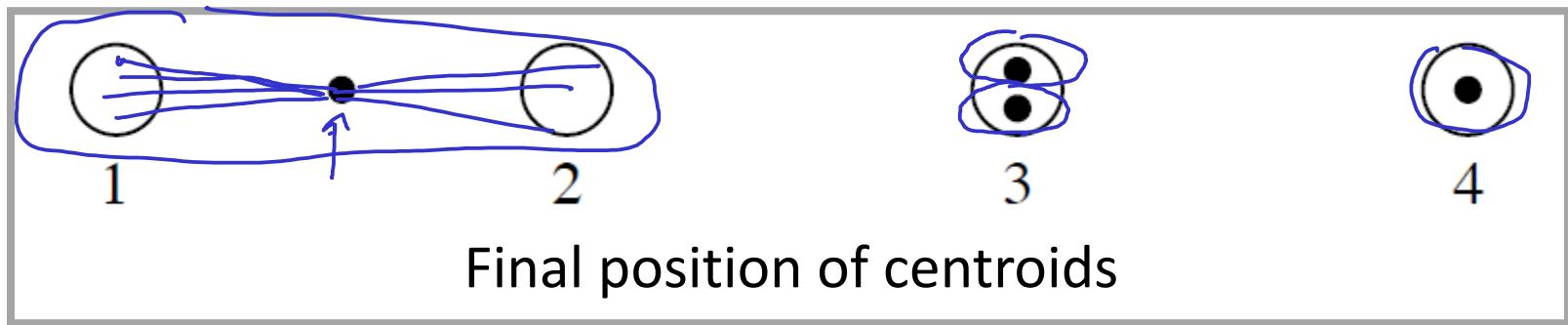
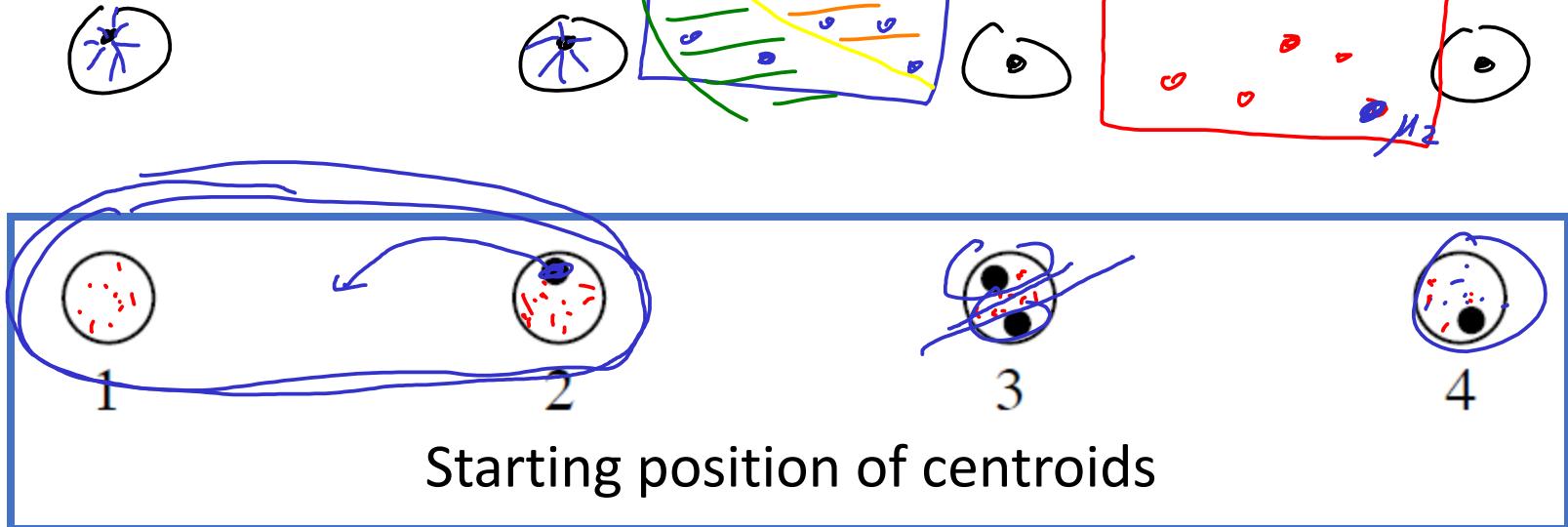
undefined.

- Empty clusters are bad.
 - Pick data points to initialize clusters
- Bad local minima
 - Initialize many times and pick solution with smallest training loss
 - Pick good starting positions *heuristically.*

Choose points in your dataset as the representative μ_k to begin with.



Initialization



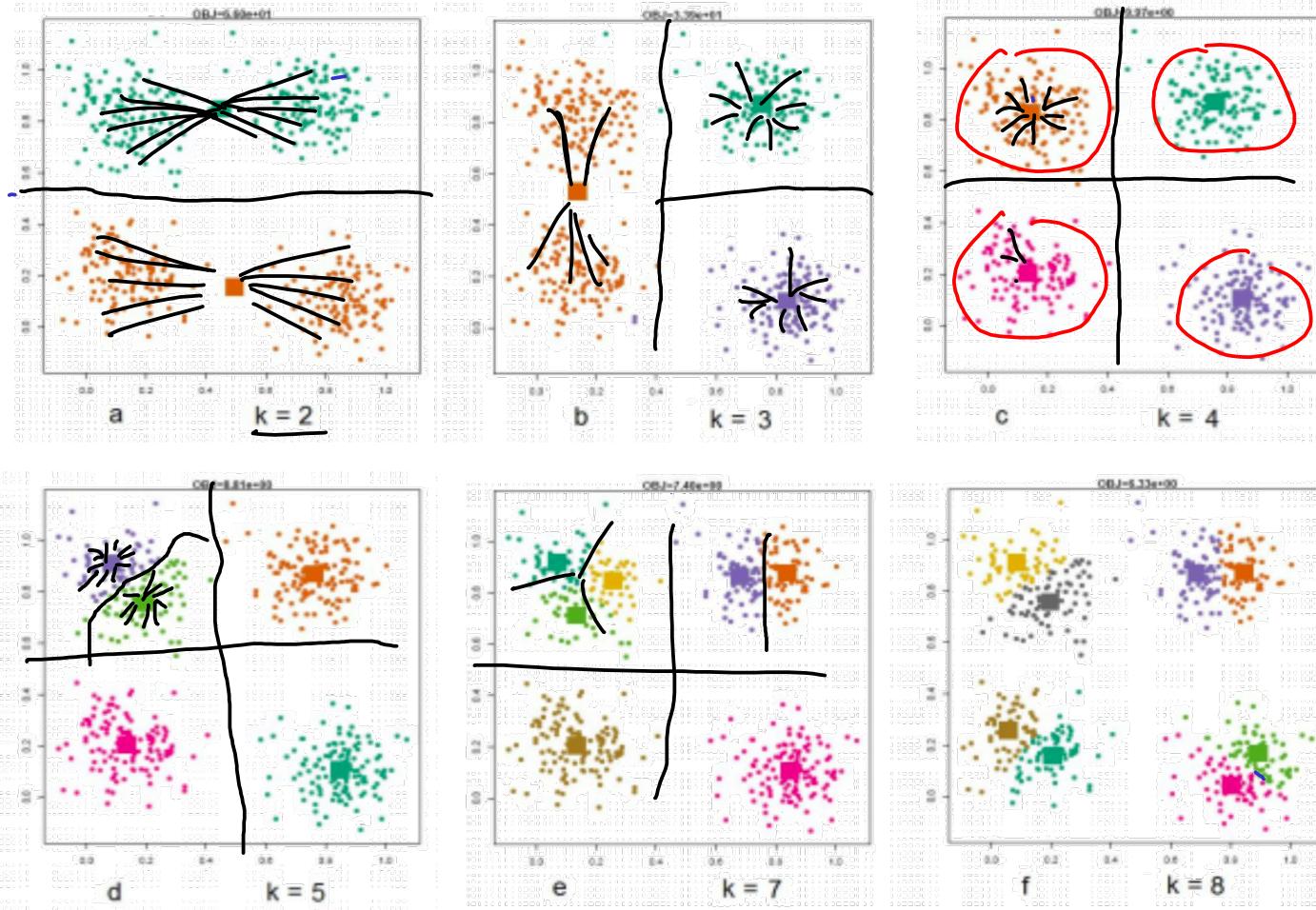
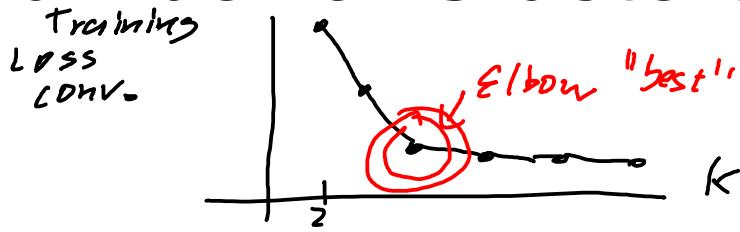
Problem. How to choose good starting positions?

Solution. Place them far apart with high probability.

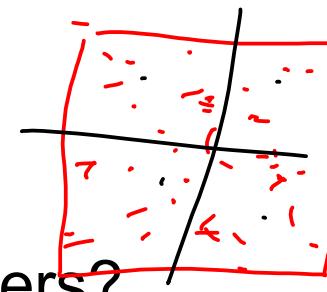
Number of Clusters

(K)

$K=N$

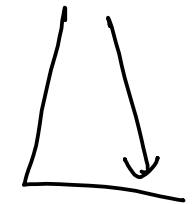
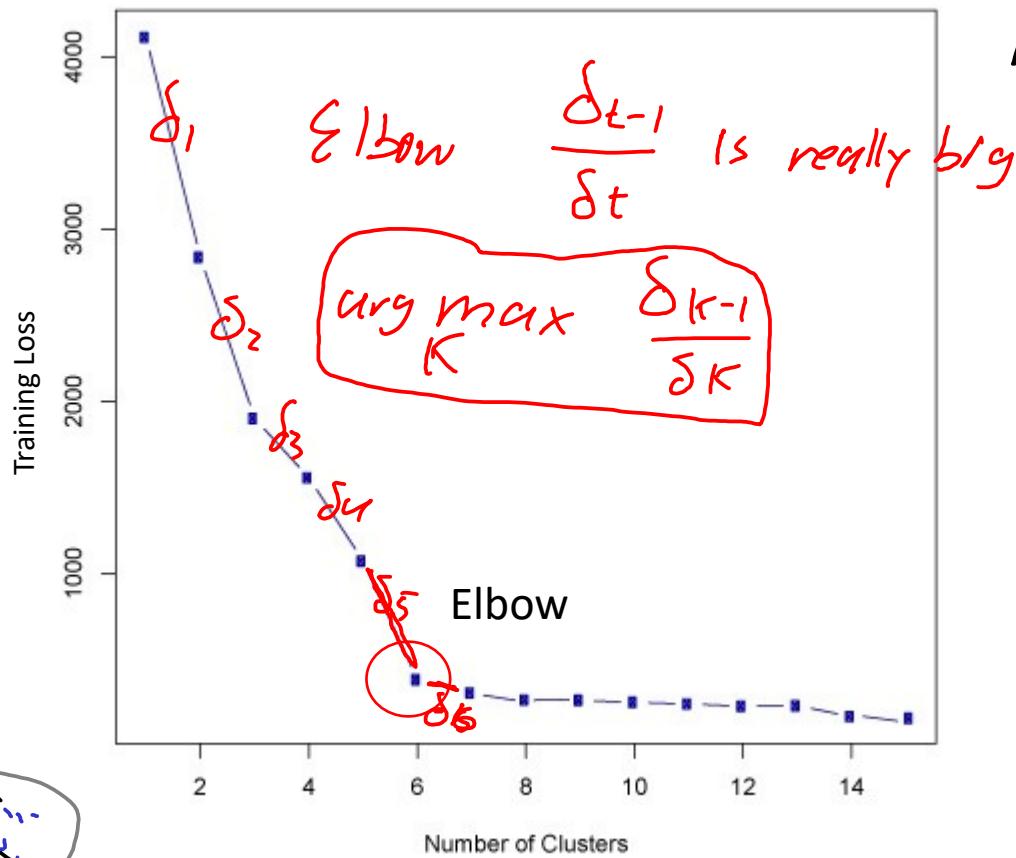
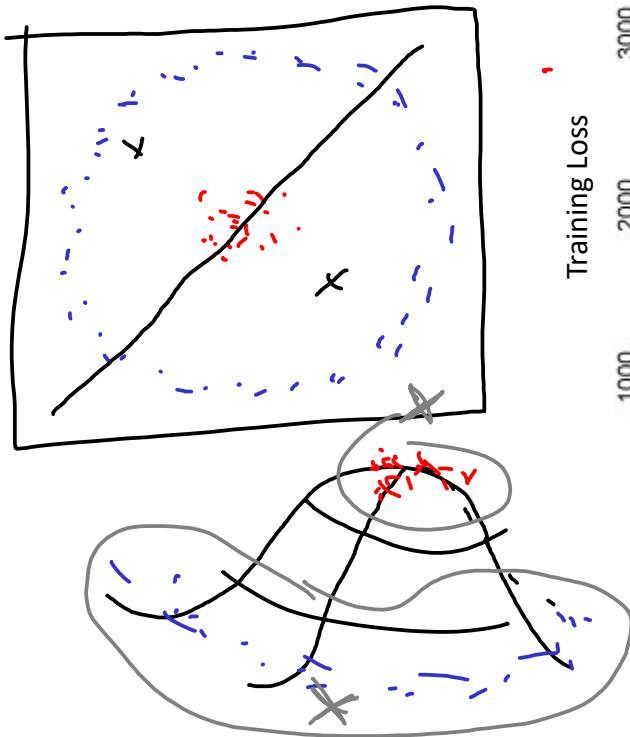


Number of Clusters

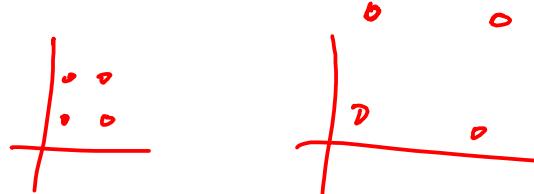


How do we choose ~~K~~, the optimal number of clusters?

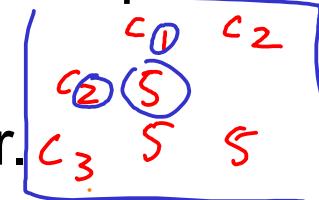
- Elbow method
 - Training Loss
 - Validation Loss



Check your understanding

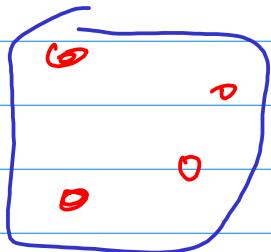


- Clustering gives you an idea of how the data is distributed.
- ✗ You have 1000 data vectors that are very similar to each other (e.g., Eu distance less than 0.0001). You can only divide them into a few clusters.
 10^{-20}
- ✗ When you use kmeans, you "usually" obtain a global minimum of the loss function
- ✗ When you use kmeans, you can never achieve global minimum of the loss function.
- ✗ The number of clusters is a parameter that can be optimized by kmeans. *K is a hyperparam.*
trick question.
- In K-fold cross validation, K is a hyperparameter.
- ✗ Agglomerative clustering as we introduced in this lecture, has a fixed clustering result given distance measurement and the number of clusters desired, and tie breaker.
strictly.

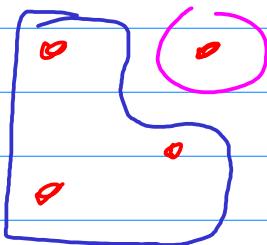


Increasing K for K-means always ensures the optimal solution for that K has smaller loss.

$K < N$



Optimal for $K=2$



At least as good for $K=3$

Optimal for $K=3$

