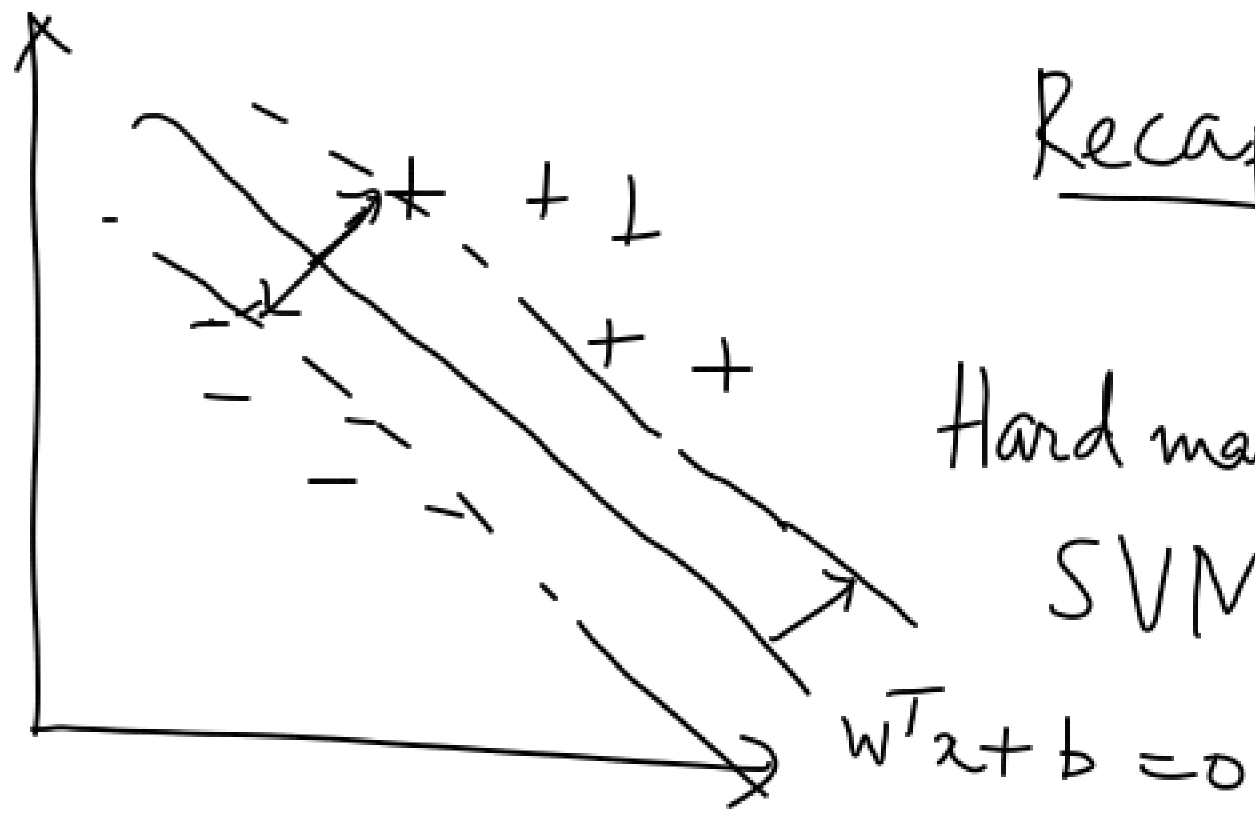
Lec 16: SVM (contd.)

Support vector machines (max margin classifier)



Recap: Hard margin SVM

$$\min \frac{1}{2}\|w\|^2 \qquad \text{PRIMAL}$$

$$\text{s.t. } y_i\left(w^T x_i + b\right) \geq 1$$

$$\forall\, i = 1, \dots, n$$

$$\boxed{x_i \in \mathbb{R}^d}$$

$d >> n$

$$w^* = \sum_{i=1}^{n} \lambda_i^* y_i x_i$$

DUAL: $\max\limits_{\lambda \geq 0} \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_j \sum_i \lambda_i \lambda_j\, y_i y_j\, \underbrace{x_i^T x_j}_{\phi(x_i)^T \phi(x_j)}$

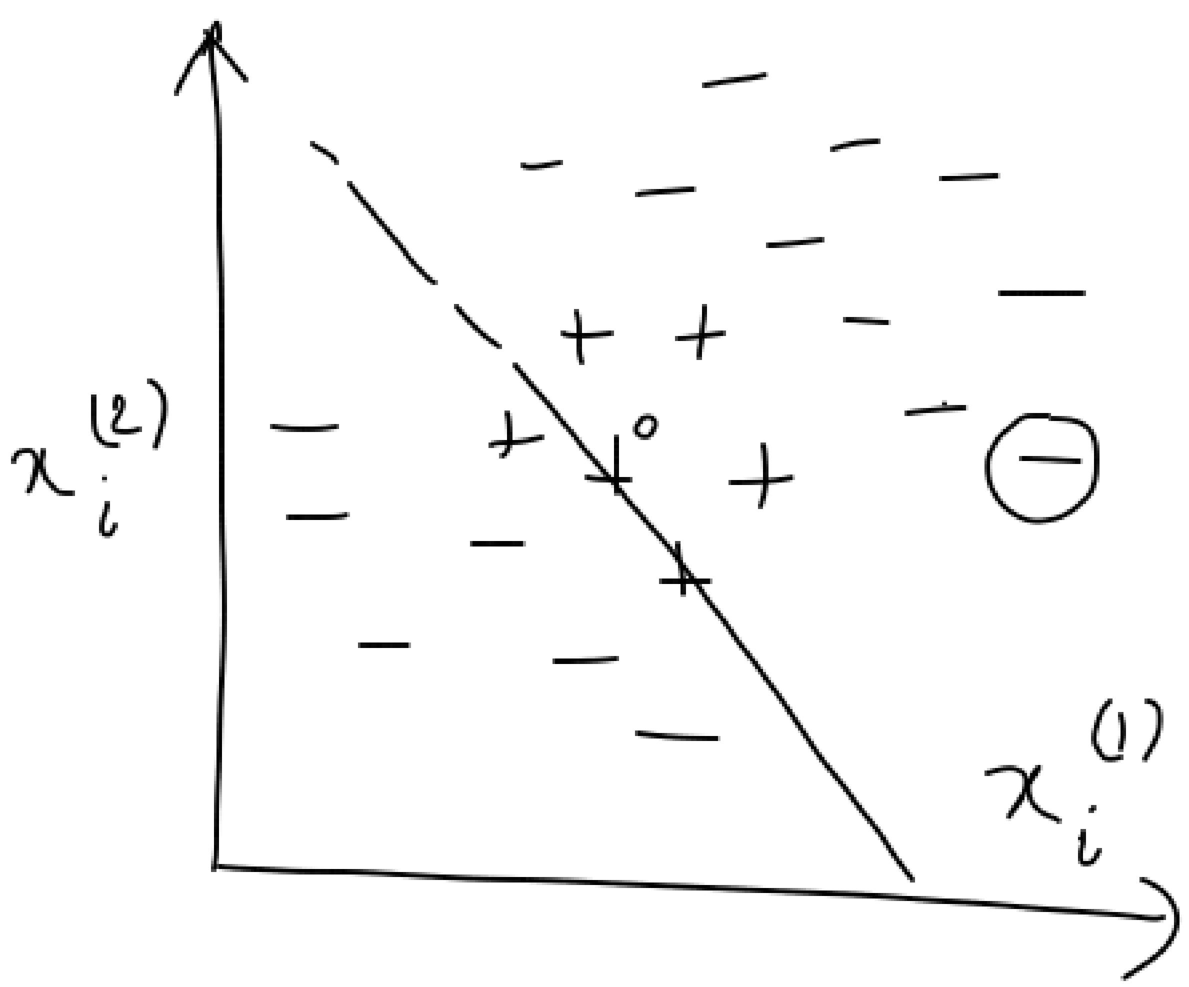
$\phi(x_i) \quad \phi(x_j)$

easier problem

$$\text{s.t. } \sum_{i=1}^{n} \lambda_i y_i = 0$$

Why dual?

1. less costly to solve.
2. Kernelization



Soft margin SVMs are also not very good.

$\rightarrow$ $\phi \rightarrow$ basis transform

$x_i \longmapsto \phi(x_i) \in \mathbb{R}^M$

x_i: high dimension

$M \gg d$

Given: linearly inseparable data

Goal: project the data to a high dimensional space

$\rightarrow$ solve SVM $\rightarrow$ find W, b in the new dimension.

E.g.

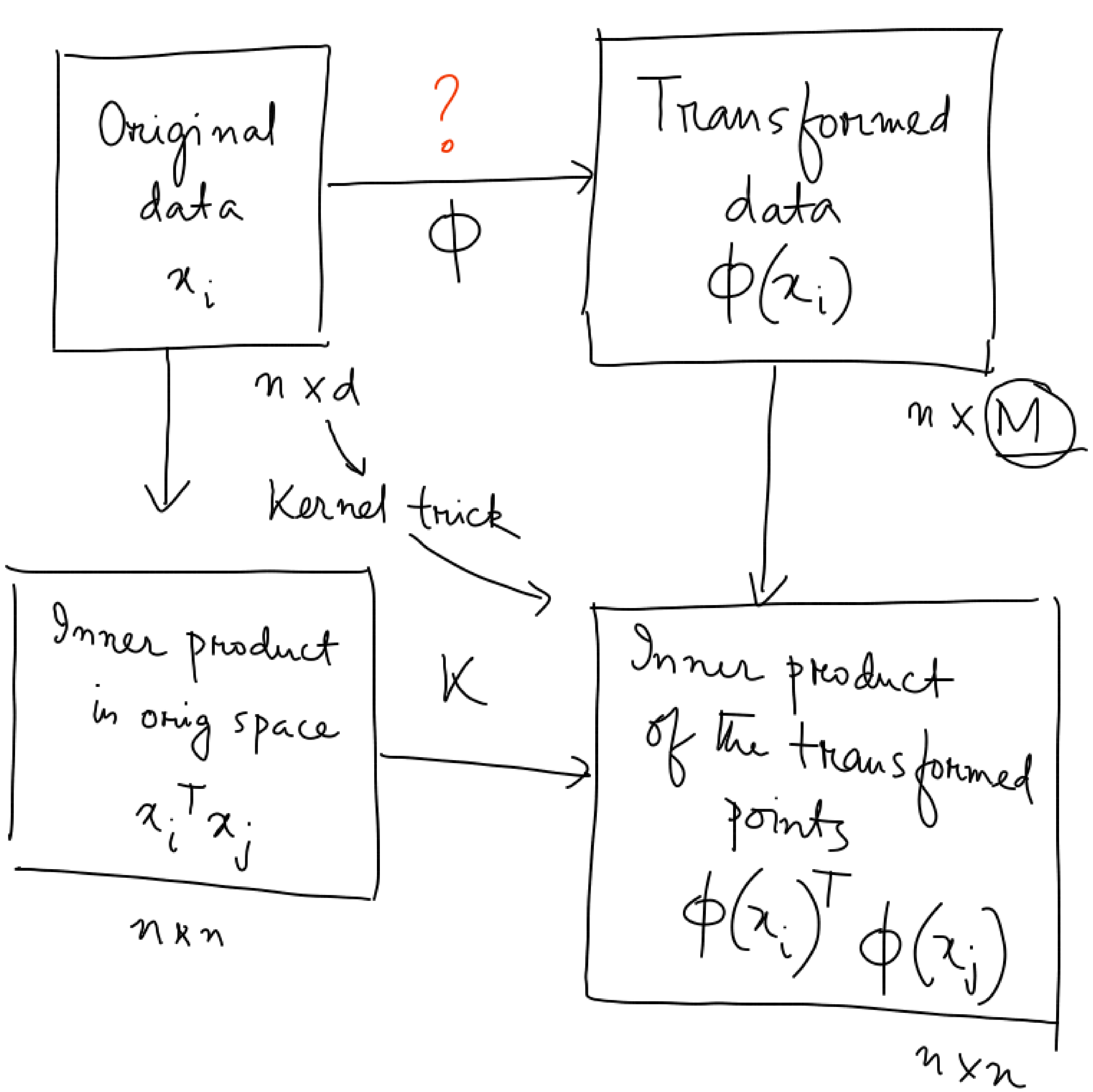
$$x_i = \begin{bmatrix} x_i^{(1)} \\ x_i^{(2)} \end{bmatrix} \xrightarrow{\phi} \begin{bmatrix} 1 \\ x_i^{(1)} \\ x_i^{(2)} \\ x_i^{(1)} x_i^{(2)} \\ x_i^{(1)\,2} \\ x_i^{(2)\,2} \end{bmatrix} = \phi(x_i)$$

Step 1:

$$\phi(x_i)^T \phi(x_j)$$

Step 2: Dual SVM

obj function has an inner product of the data in the higher dimension



Q: Do there exist functions that calculate the inner product in the transformed space <u>without explicitly computing the transformations</u>?

A: <u>Yes, via the kernel function</u>

E.g. previous example, terms

$$= \{1,\ x_i^{(1)} x_j^{(1)},\ x_i^{(2)} x_j^{(2)},\ x_i^{(1)} x_i^{(2)} x_j^{(1)} x_j^{(2)},\ {x_i^{(1)}}^2 {x_j^{(1)}}^2,\ {x_i^{(2)}}^2 {x_j^{(2)}}^2\}$$

$$K(x_i, x_j) = (1 + \underline{x_i^T x_j})^2 \rightarrow \text{Same terms}$$

$$x_i^{(1)} x_j^{(1)} + x_i^{(2)} x_j^{(2)}$$

Kernel trick

transformations are equivalent as long as we are calculating the dual of SVM.

Kernel Regression

Different Kernels

1. Linear: $K(x, z) = x^T z$
2. Polynomial: $K(x, z) = (1 + x^T z)^m$
3. Gaussian: $K(x, z) = e^{-\|x - z\|^2 / 2\sigma^2}$
4. Laplace/Radial: $K(x, z) = e^{-\|x - z\| / 2\sigma}$

Use cases of SVM: Handwriting recognition, protein structure, medical image classification.

A set of necessary and sufficient conditions govern the kernel functions

Which Kernel to pick?

grid seach →

Mercer's theorem

see note on webpage.

Limitations of SVM

1. Binary classification

   → One-vs-rest classifier

2. No probabilistic interpretation

3. Does not work very well when data is noisy.

Story so far:

Supervised learning

$$D = \{(x_i, y_i)\}_{i=1,\cdots n}$$

y_i: costly

Unsupervised learning

$$D = \{(x_i)\}_{i=1,\cdots,n}$$

Clustering : Unsupervised learning

$$D = \{x_1, x_2, \cdots, x_n\}, \quad x_i \in \mathbb{R}^d$$

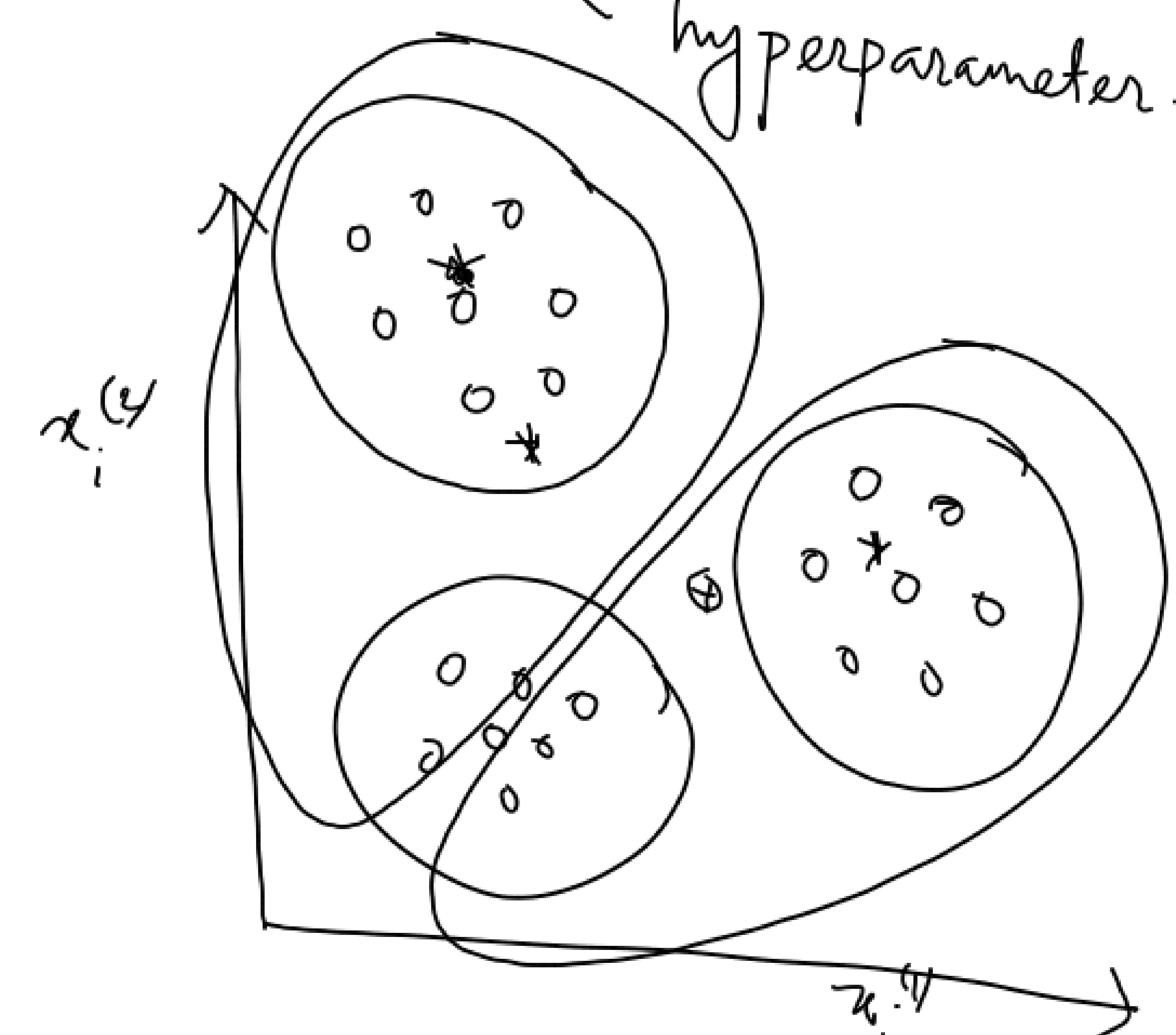
Goal: find a "well-separated" partition of the data

$$D = D_1 \cup D_2 \cup \cdots \cup D_k$$

k ← hyperparameter

hard cluster $D_i \cap D_j = \phi$

k-means clustering



$C(i) = \tilde{k}$ Clustering function

i^{th} data point

$C : [n] \to [k]$

$[n] = \{1, 2, \cdots, n\}$

$C^t(i) \neq k'$

$x_i \longleftrightarrow m^t_{k'}$

$m^t_{C^t(i)}$

k-means

Input = $D = \{x_1, \dots, x_n\}$

Initialize: k cluster means $m_1, \dots, m_k$

some arbitrary C^0

Repeat until convergence

(until assigments do not change)

for every $i = 1, \cdots, n$

if $\exists k' \neq C^t(i)$

$$\| m^t_{k'} - x_i \| < \| m^t_{C^t(i)} - x_i \|$$

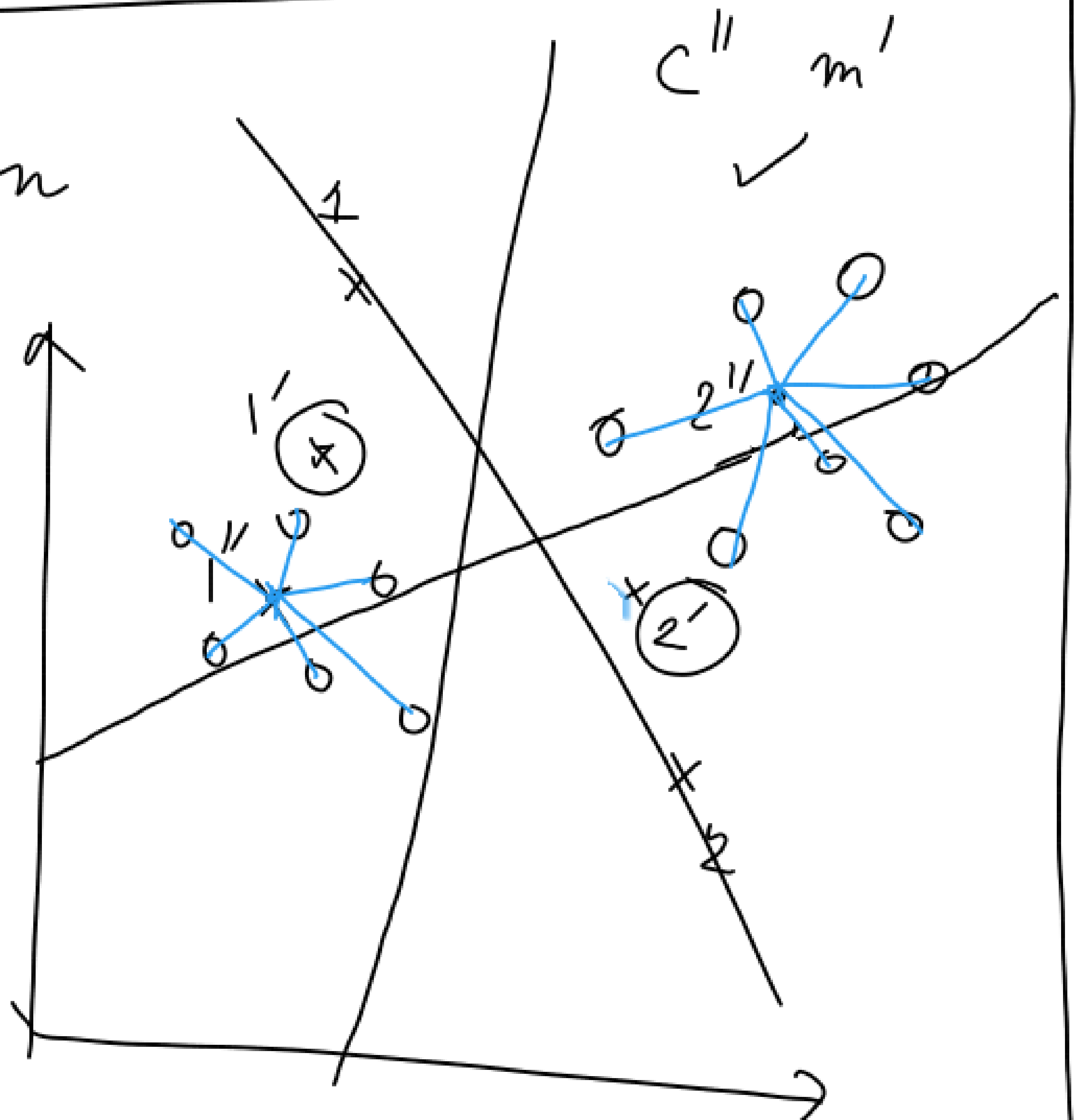
$$C^{t+1}(i) \leftarrow k'$$

Update cluster centers by average

of its current associated data points

$m_{k'}^{t+1}$

Illustration



non-convex

$$\min_{C \in \mathcal{C}} \sum_{i=1}^{n} \|x_i - m_{C(i)}\|^2$$

NP-hard

$$|\mathcal{C}| = k^n$$

set of all possible clusters

Squared error

SE

k values

$C(i) \rightarrow$ 1 2 … n $\rightarrow$ point

k means is a "reasonable" approach $\rightarrow$ converges. local optima.

Convergence

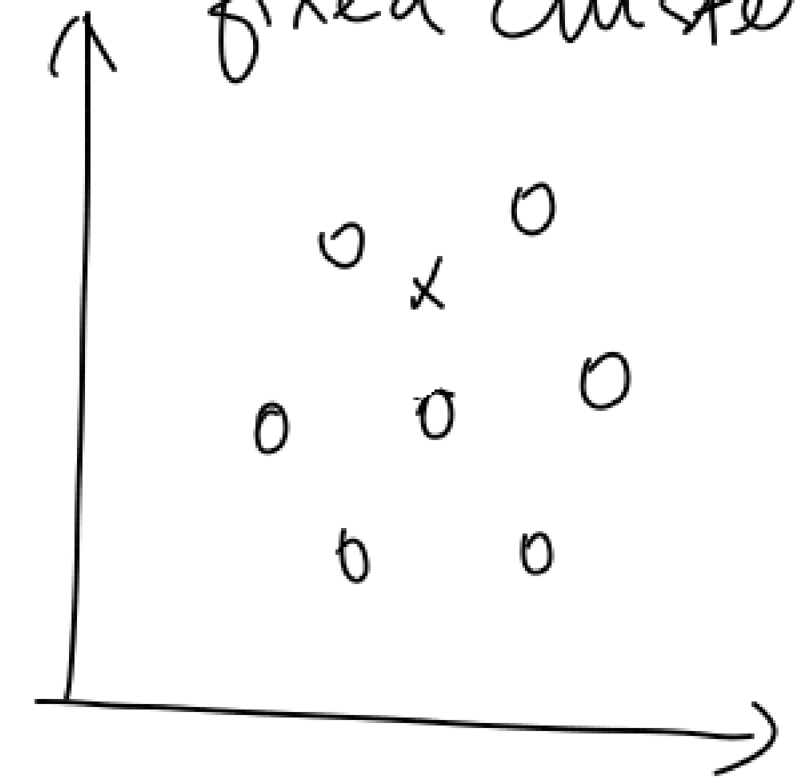
Lemma:

$$\underset{x}{\operatorname{argmin}} \sum_{i=1}^{n_k} \|x_i - x\|^2 = \bar{x} = \frac{1}{n_k}\sum_{i=1}^{n} x_i$$

Theorem: k-means converges to a local minima

- consider $t \to t+1$
- $SE(c^{t+1}, m^{t+1}) < SE(c^t, m^t)$

fixed clustering



C

Why is this sufficient?

- no clustering are repeated
- finite number of clusterings

Proof:

$m^t, c^t \to m^t, c^{t+1} \to m^{t+1}, c^{t+1}$

1. $SE(c^{t+1}, m^t) < SE(c^t, m^t)$ from the algorithm itself
2. $SE(c^{t+1}, m^{t+1}) \leq SE(c^{t+1}, m^t)$ Lemma.

Combining the two claims, get the theorem.

$C^k(i)$ = prob(i belongs to k)

Soft clustering