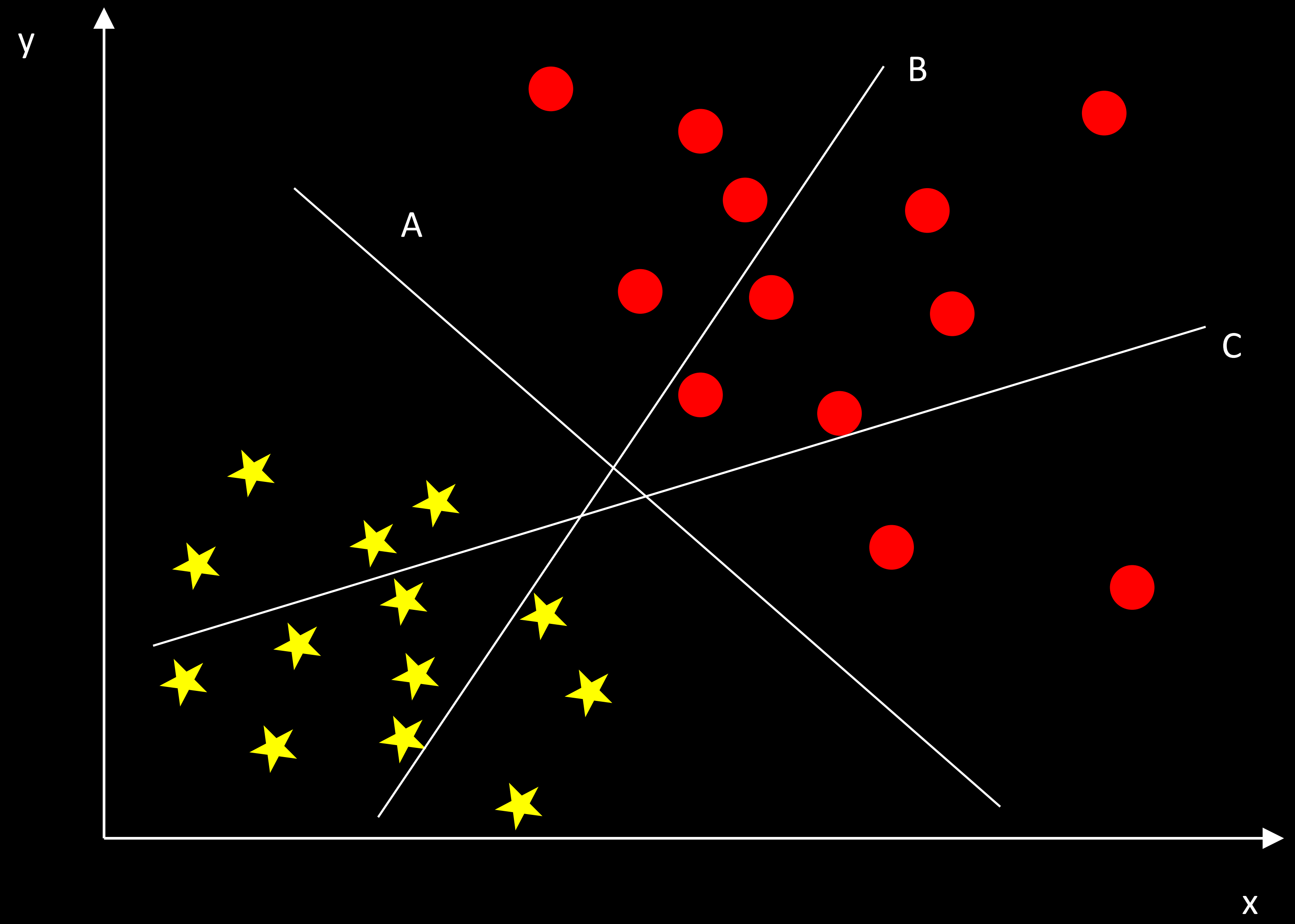
A Training Algorithm for Optimal Margin Classifiers

BE Boser, IM Guyon, VN Vapnik

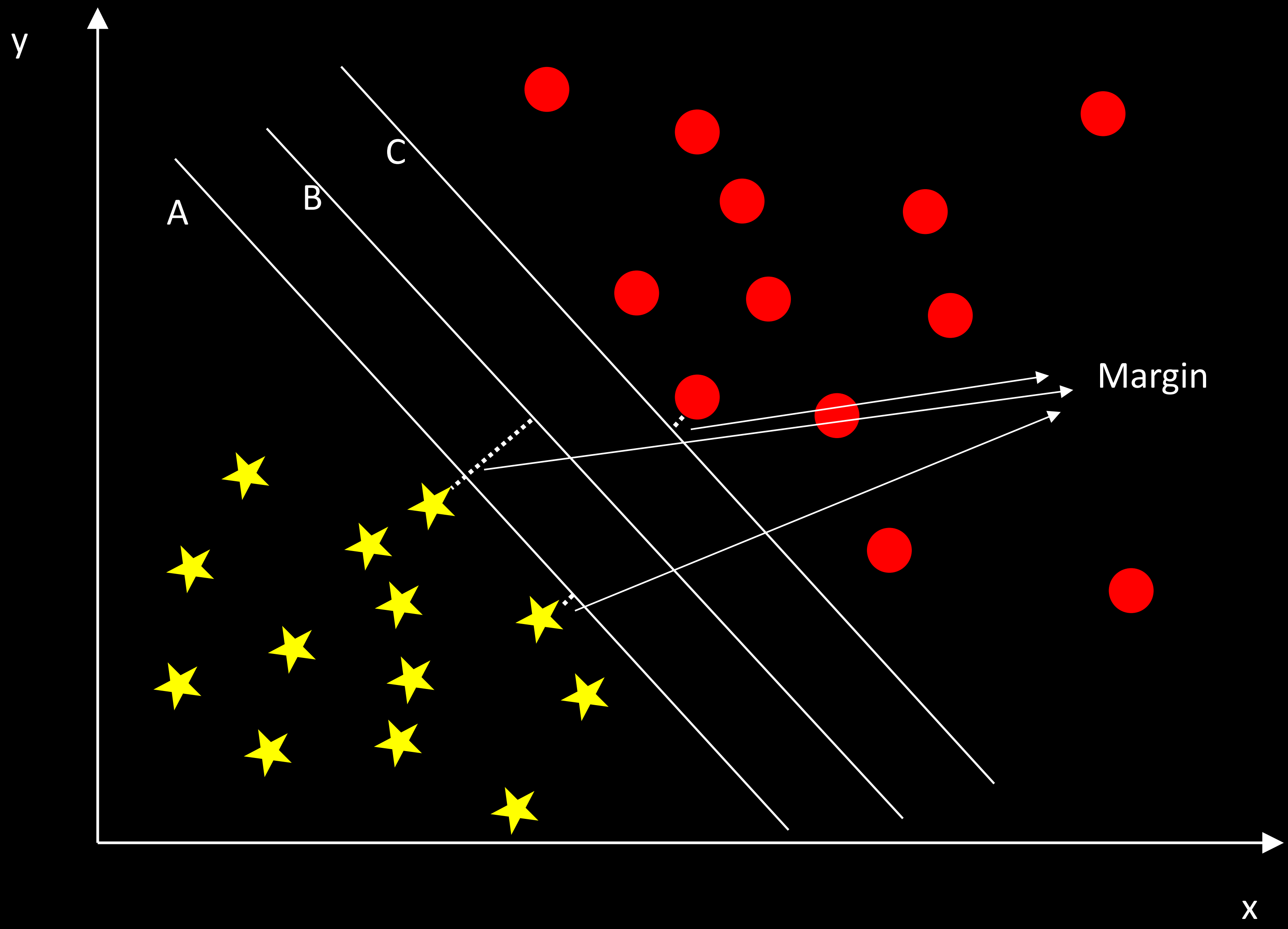
By Wenxiao Fu

What's a Margin

Scenario-1

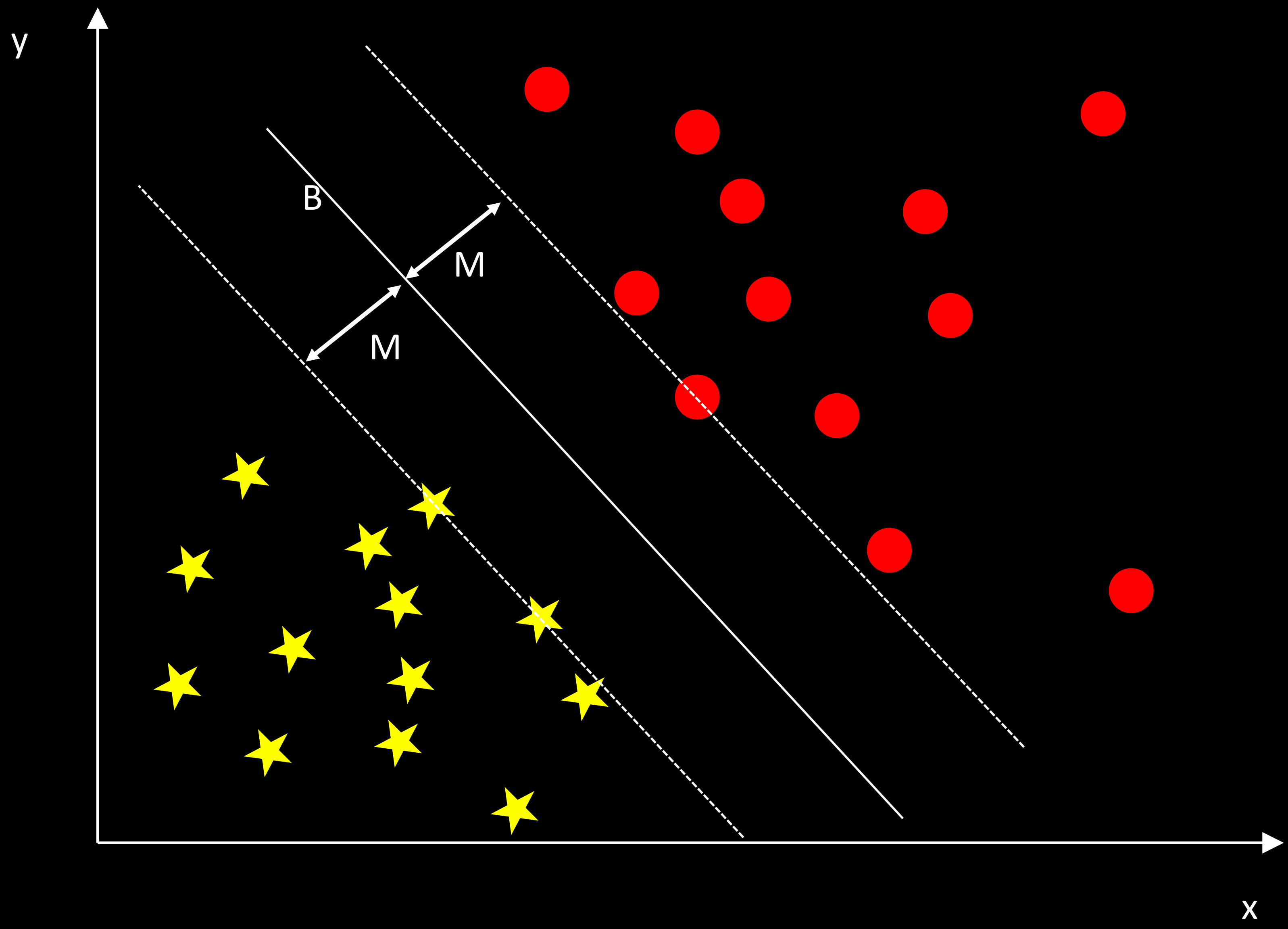


Scenario-2

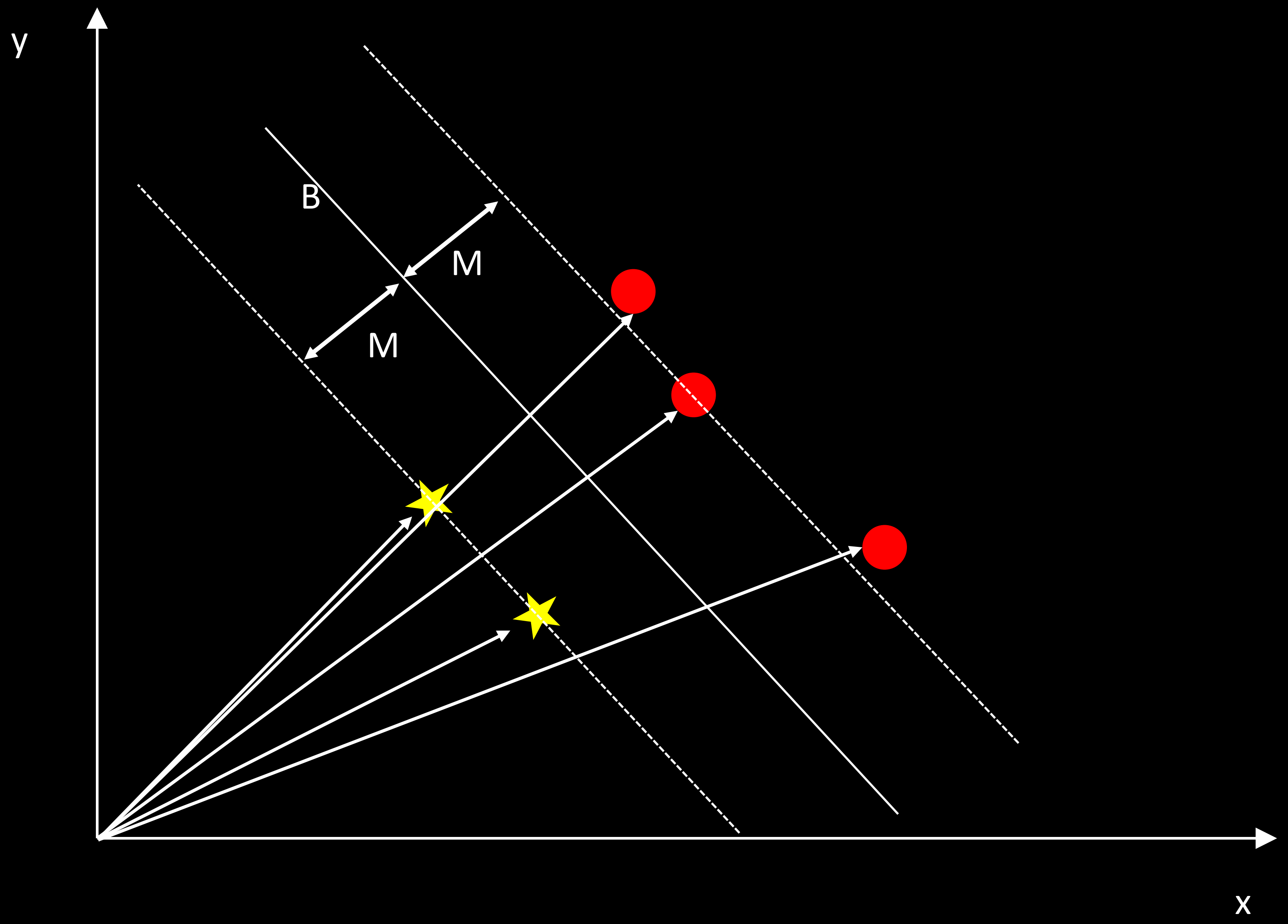


Our Task-Maximal Margin

Maximal Margin



Maximal Margin



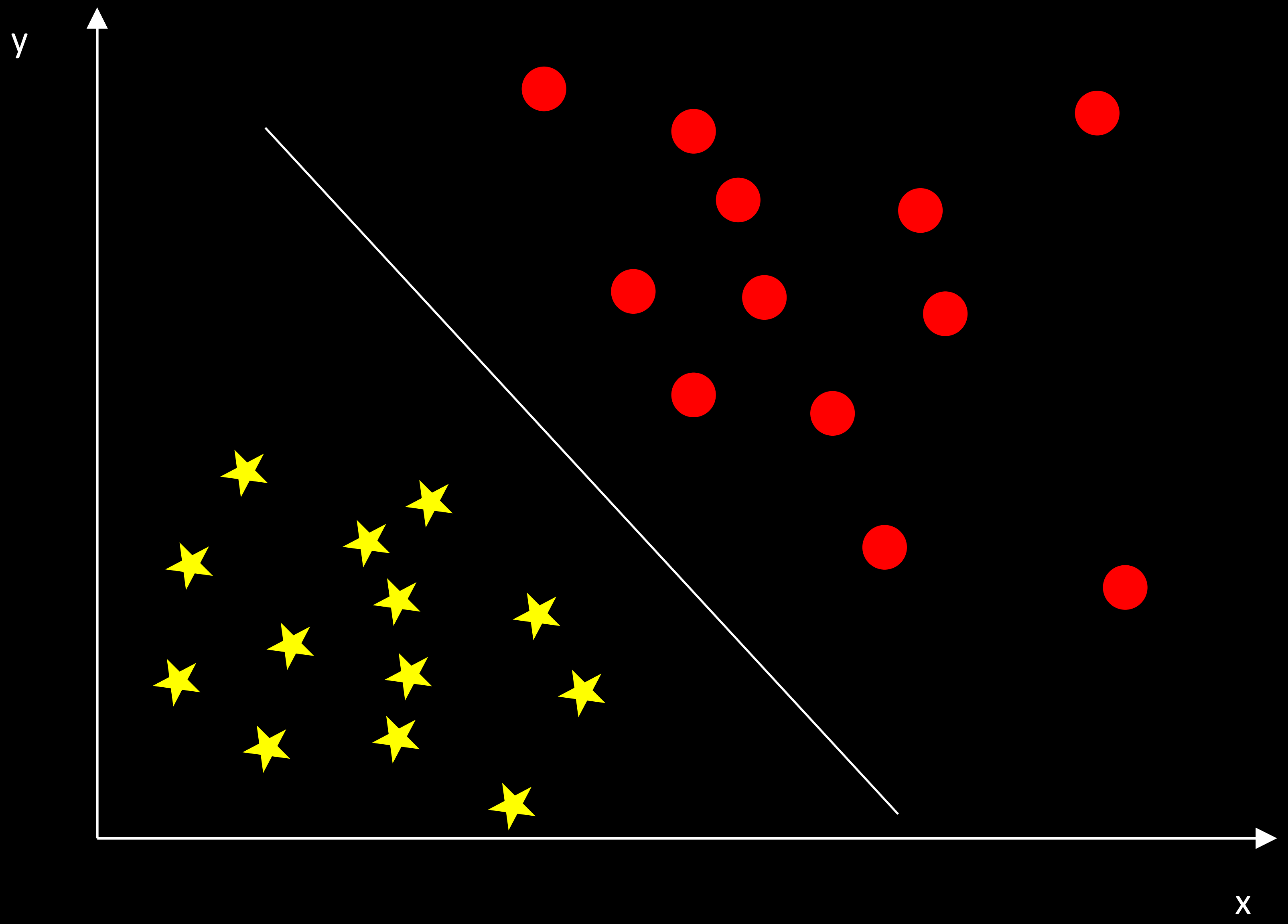
Support Vector Machine

SVM

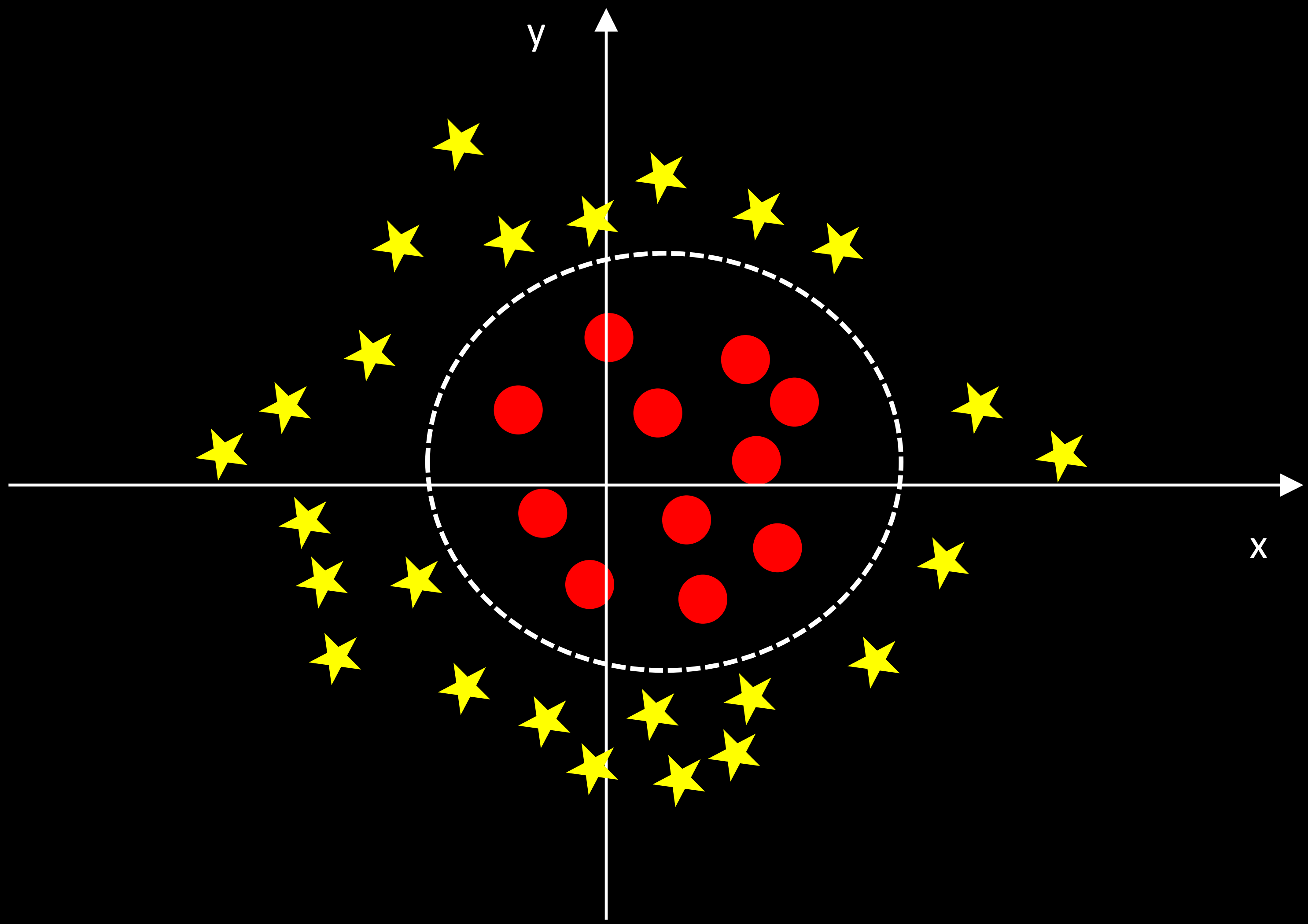
- A training algorithm that maximize the margin between the training patterns and the decision boundary.
- The solution is expressed as a linear combination of supporting patterns.

What SVM Can Do

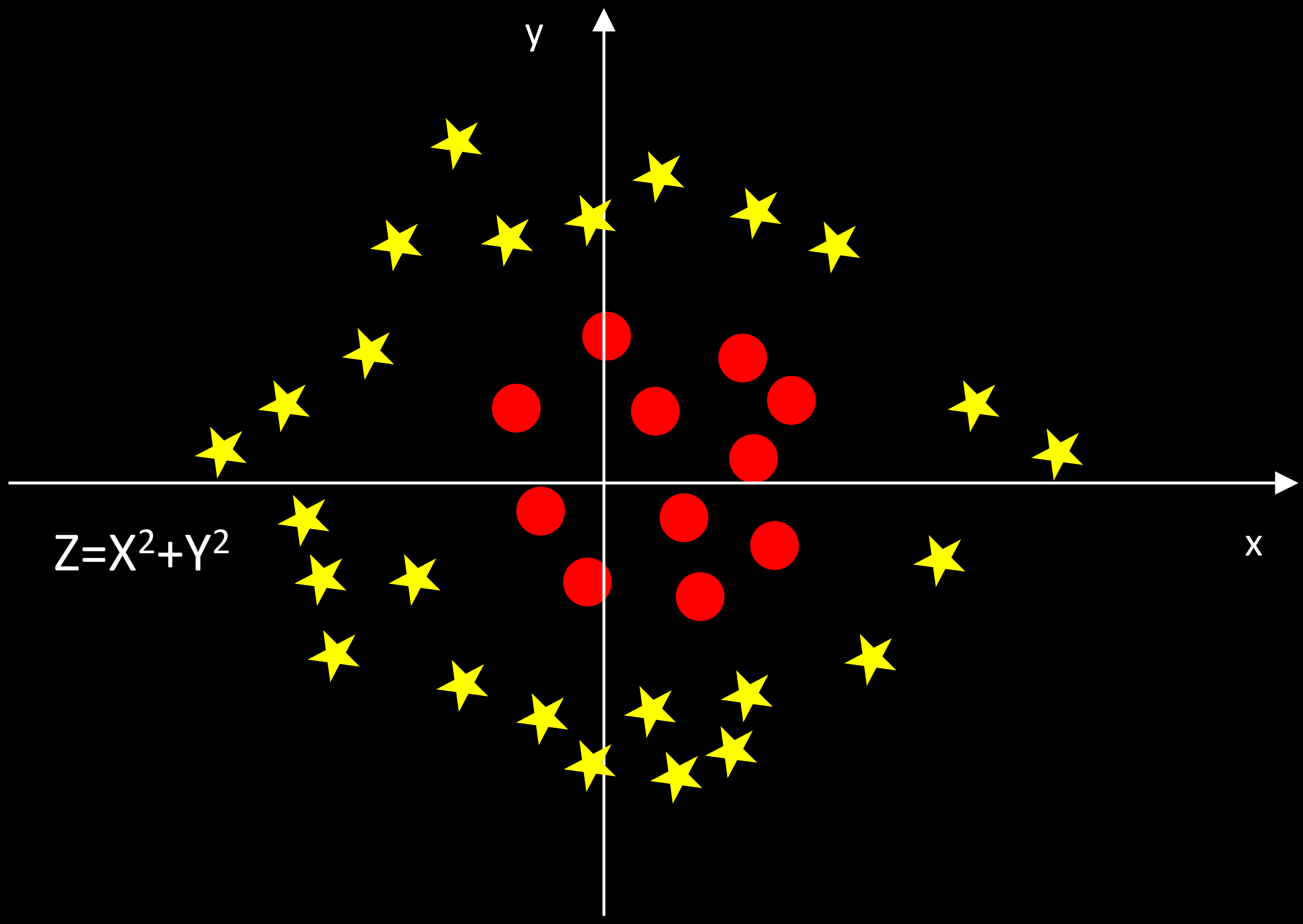
Scenario-2



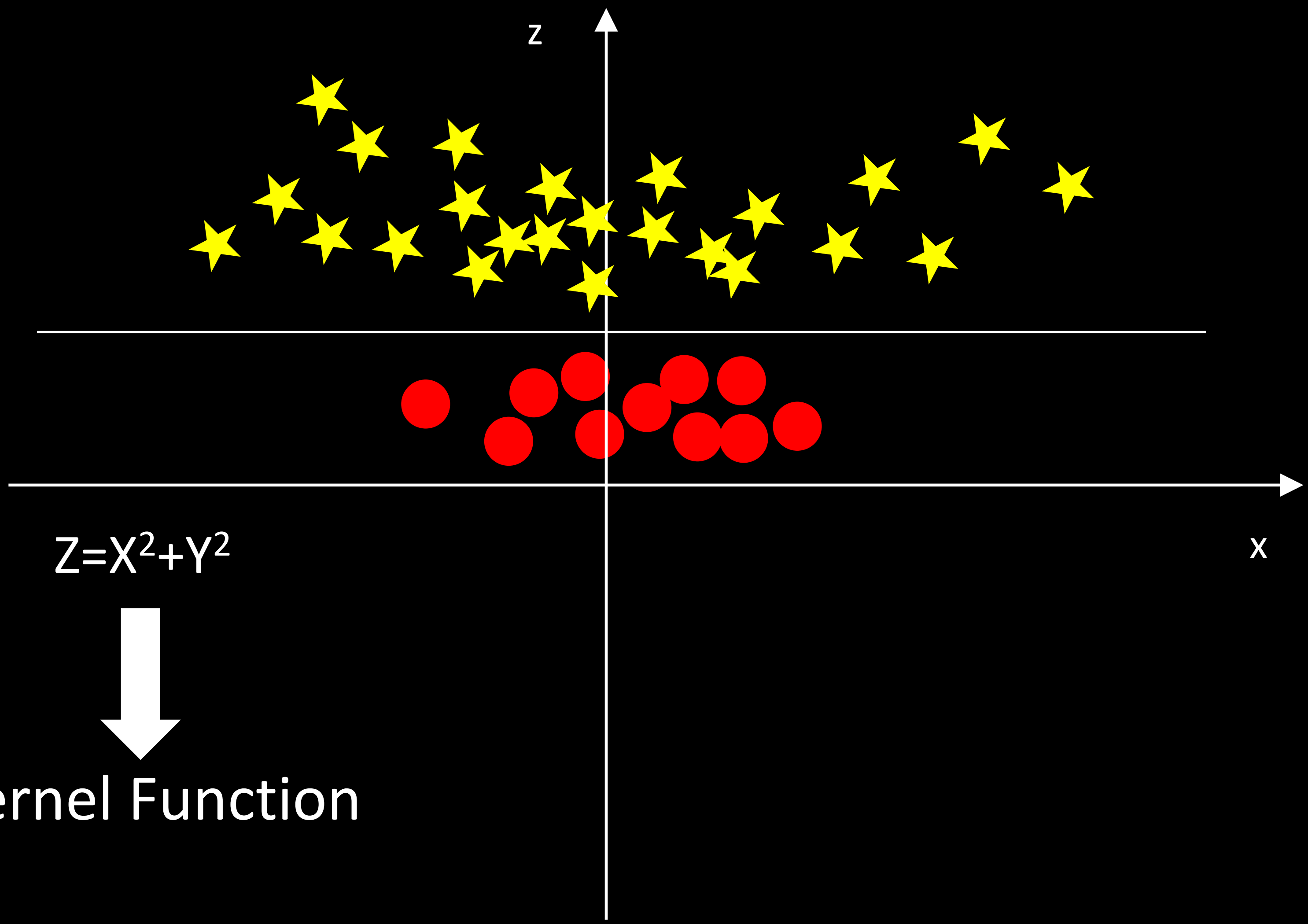
Scenario-3



Scenario-3



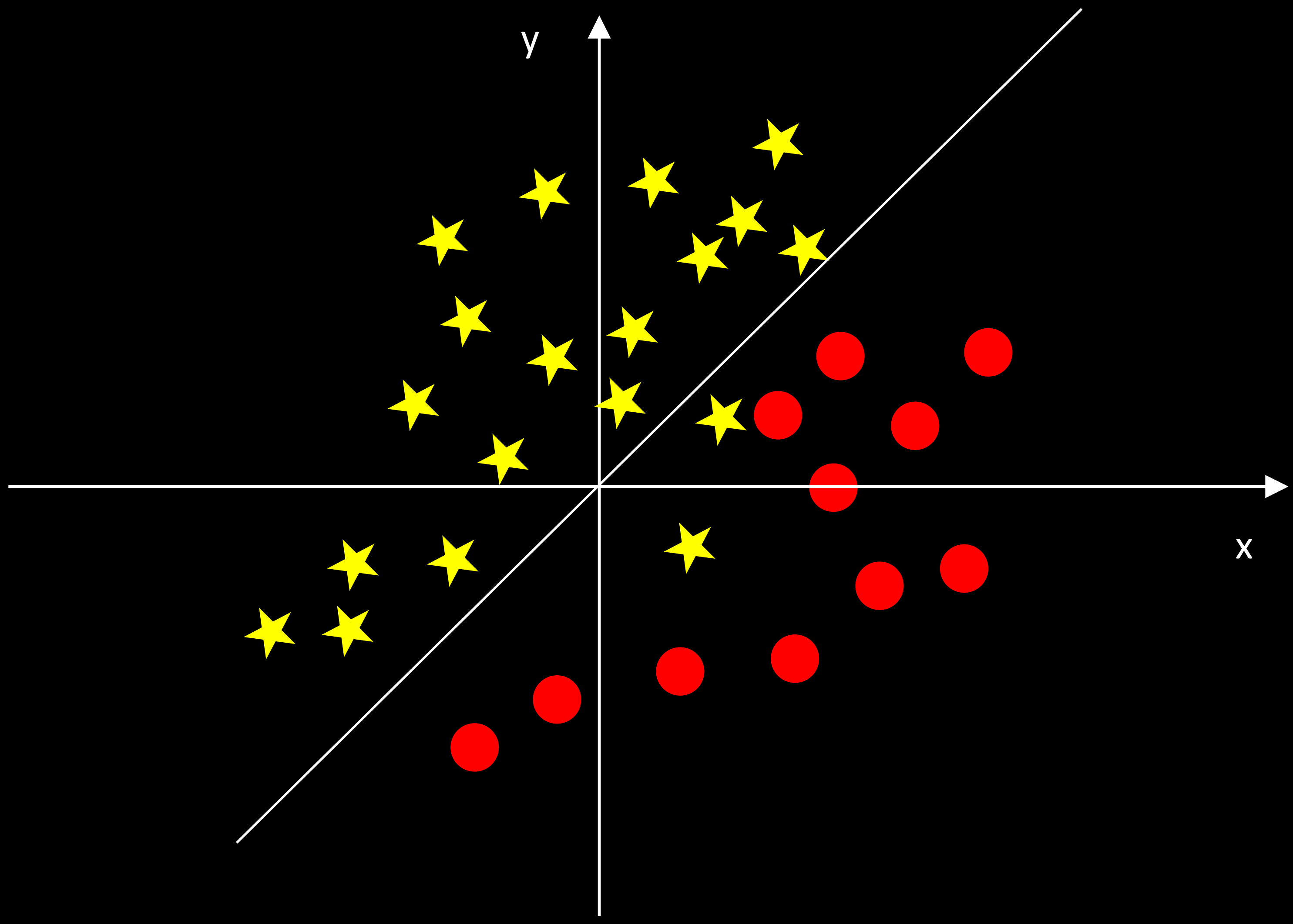
Scenario-3



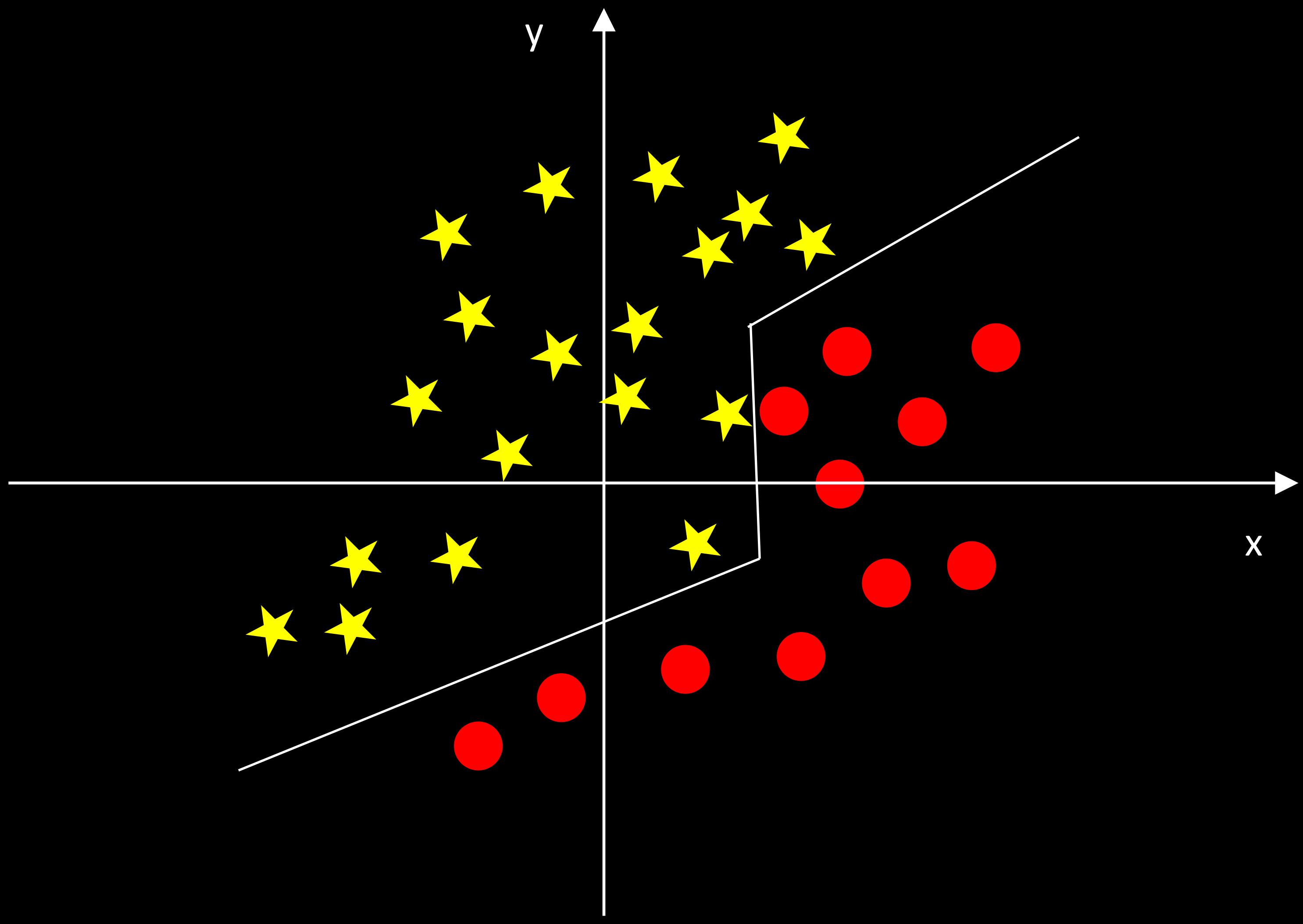
$Z = X^2 + Y^2$

Kernel Function

Scenario-4

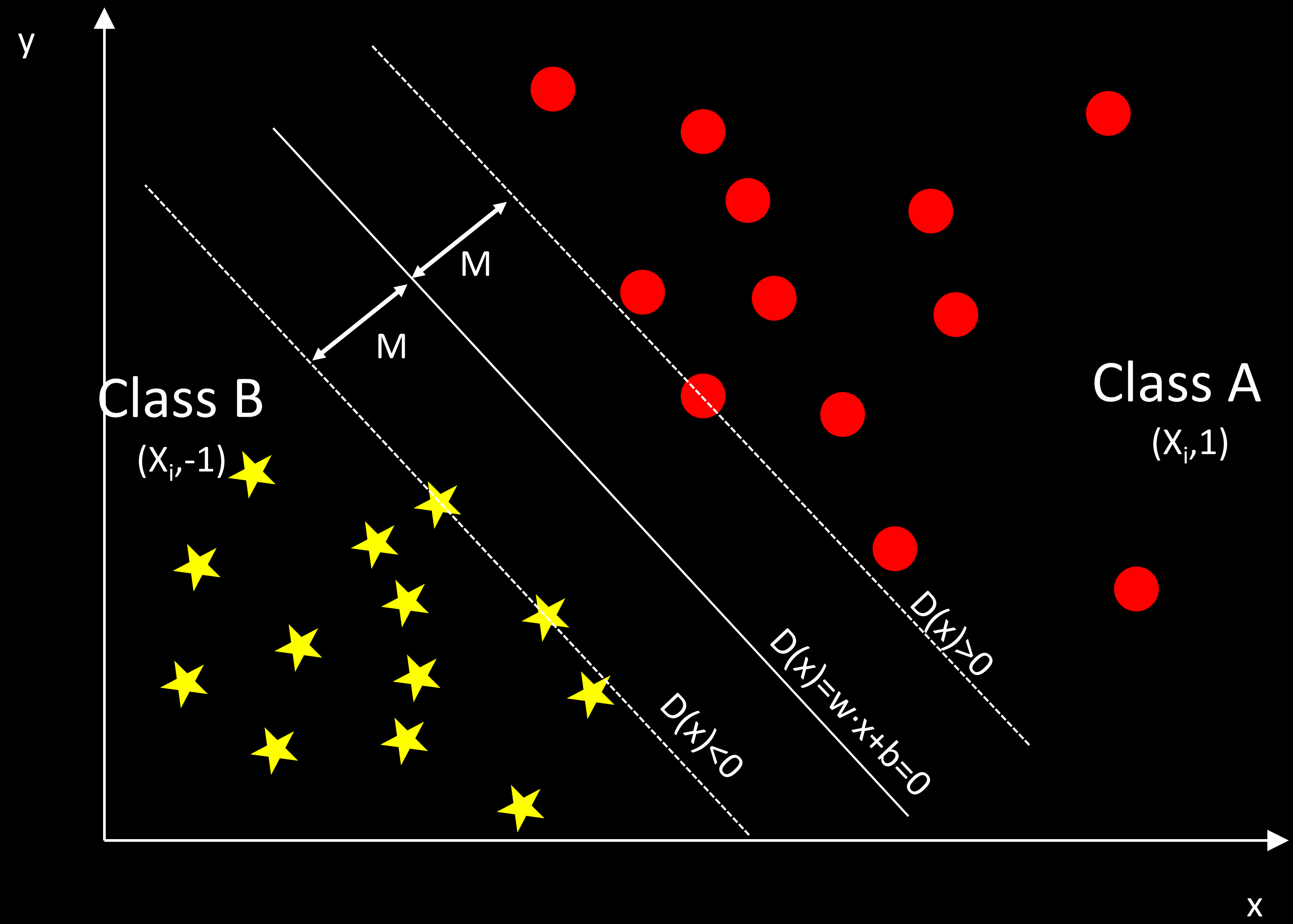


Scenario-4



How SVM Do

SVM-supervised learning



SVM-supervised learning

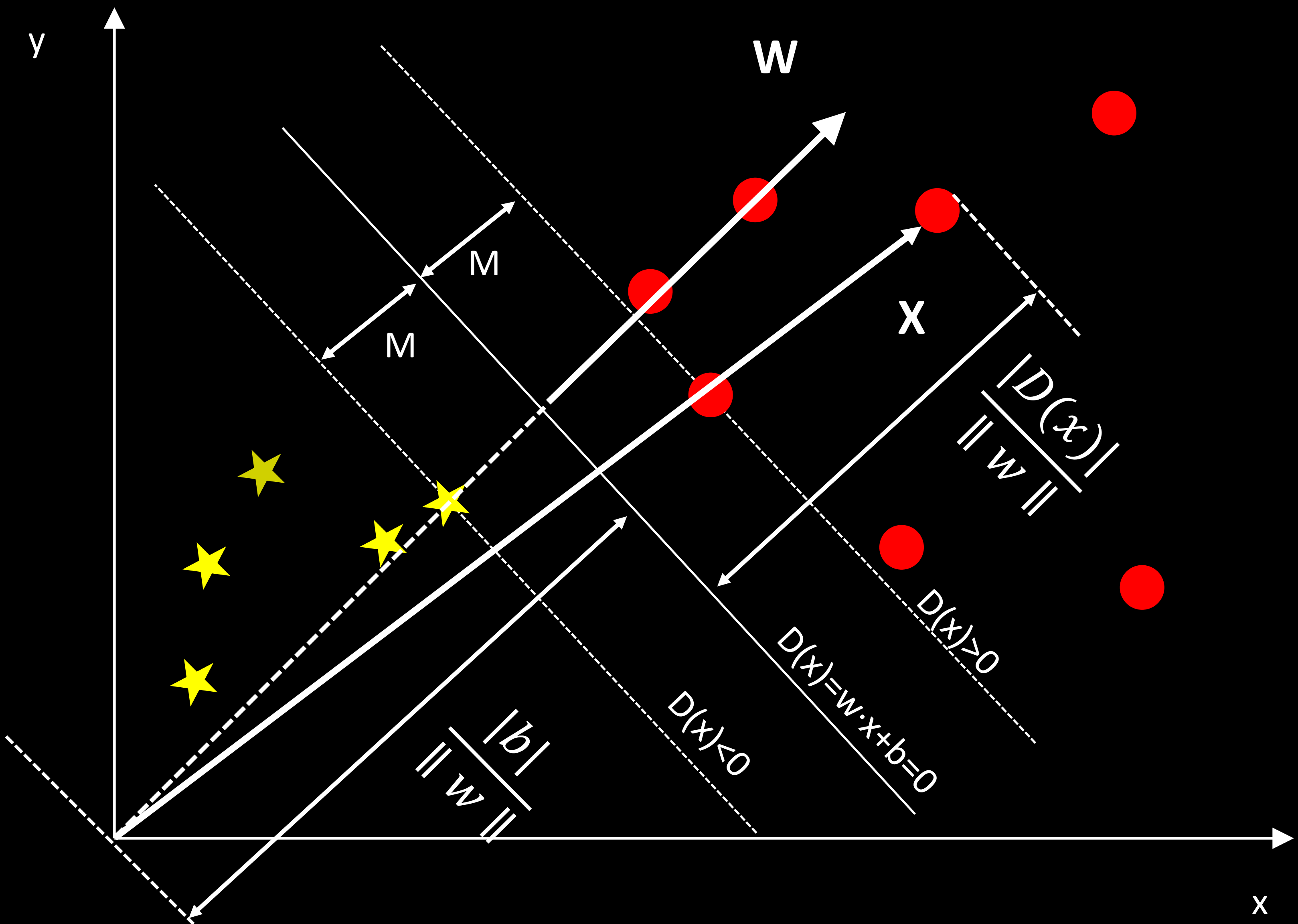
- Direct space

$$D\ (\vec{x}) = \vec{w} \cdot \varphi(\vec{x}) + b$$

- *Dual space*

$$D(\vec{x}) = \sum_{k=1}^{p} \alpha_k\, K(\vec{x}_k, \vec{x}) + b$$

SVM-direct space



SVM-direct space

$$M^* = \max_{w \| w \| = 1} M$$

$$subject\ to\ \frac{|D(x_k)|}{\|w\|} \geq M,\ k=1,2,\ldots p$$

$$\rightarrow\ w \| w \| = 1 \rightarrow M \| w \| = 1$$

$$\rightarrow\ \min_{w} \| w \|^2$$

$$subject\ to\ \ y_k D(x_k) \geq 1,\ k=1,2,\ldots p$$

Lagrangian Multiplier

find extrema of $f(x, y),\ subject\ to\ g(x, y) = c$

→ $\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda(g(x, y) - c)$

→ $\nabla_{x,y,\lambda}\mathcal{L}(x, y, \lambda) = 0$

SVM-Lagrangian Duality

$$L(w, b, \alpha) = \frac{1}{2} \parallel w \parallel^2 - \sum_{k=1}^{p} \alpha_k [y_k \, D(x_k) - 1]$$

subject to α_k>=0, k=1,2,...p.

condition $\alpha_k [y_k D(x_k) - 1] = 0$

SVM-Lagrangian Duality

$$L(w, b, \alpha) = \frac{1}{2} \parallel w \parallel^2 - \sum_{k=1}^{p} \alpha_k [y_k \, D(x_k) - 1]$$

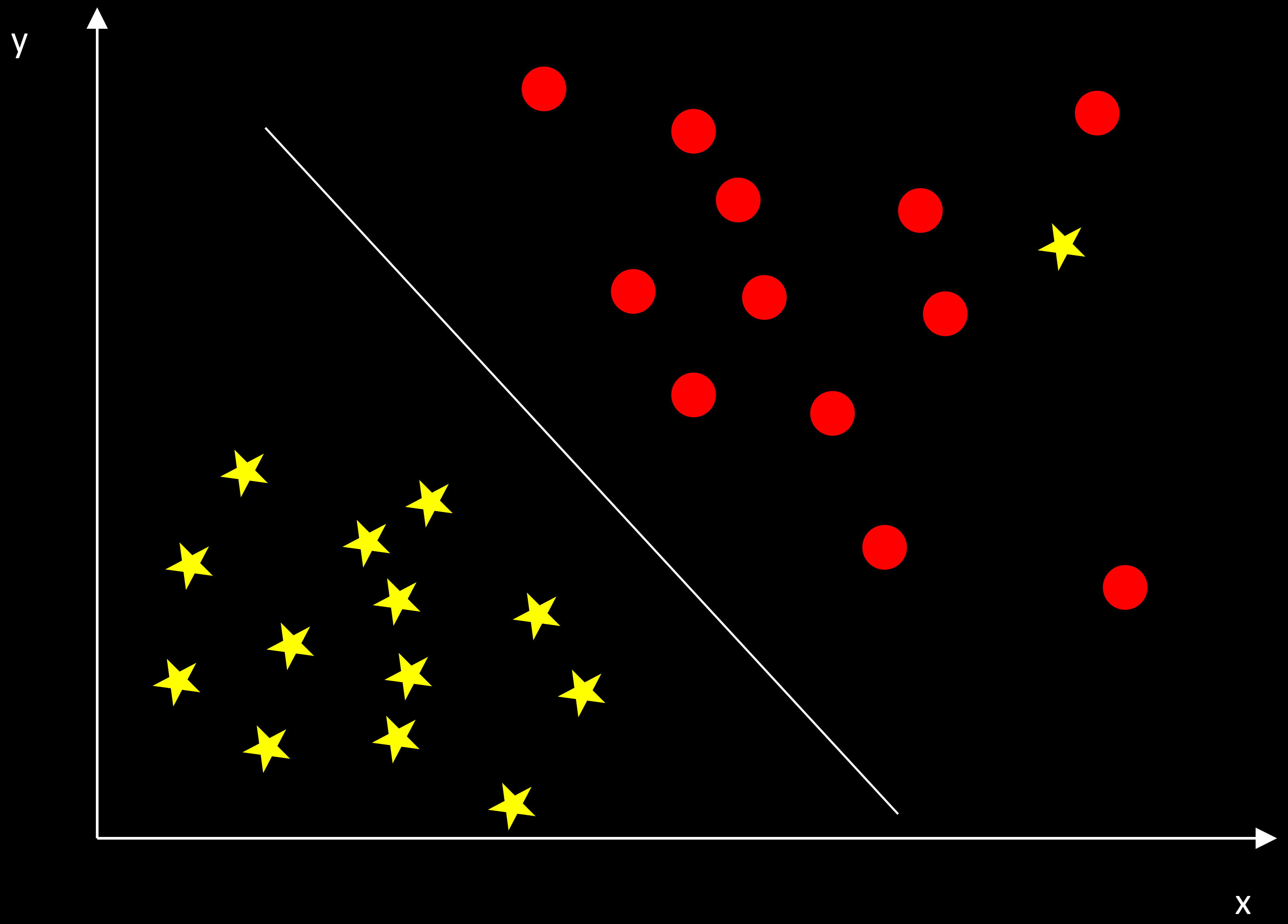
→ $D(\vec{x}) = \sum_{k=1}^{p} \alpha_k \, K(\vec{x}_k, \vec{x}) + b$

→ $K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \cdot \vec{x}_j + 1)^d$ *polynomial kernel*

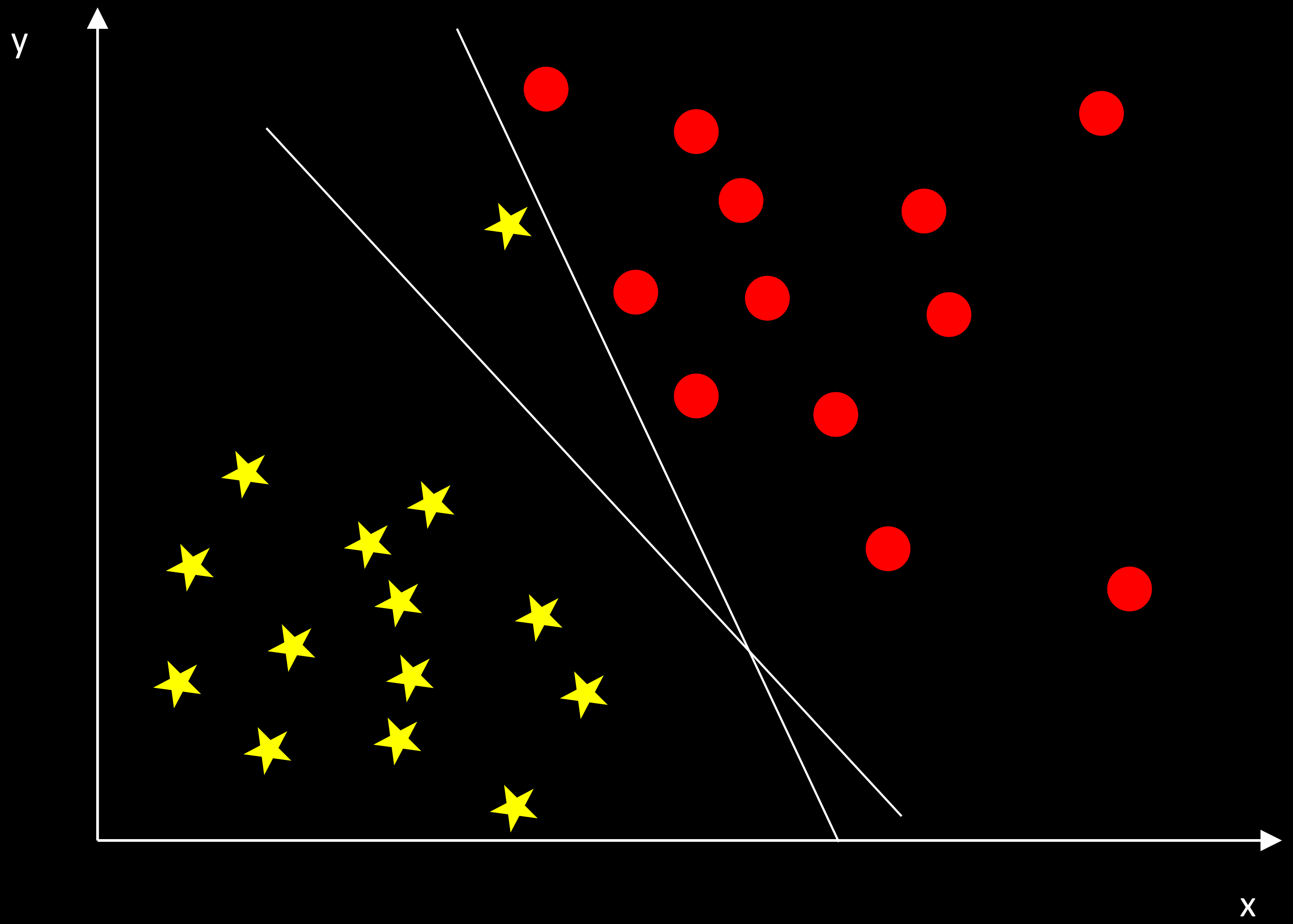
→ $K(\vec{x}_i, \vec{x}_j) = exp(-\gamma \parallel \vec{x}_i - \vec{x}_j \parallel^2)$ *gaussian kernel*

→ $K(\vec{x}_i, \vec{x}_j) = (k\vec{x}_i \cdot \vec{x}_j + c)^d$ hyperbolic tangent kernel

Properties of The Solution



Properties of The Solution



Properties of The Solution

- Outliers or meaningless patterns are identified
- Sensitive to atypical patterns closed to the decision boundary
- Insensitivity to limited computational accuracy

Issues

- Requires full labeling of input data
- 2-classes tasks
- Parameters of a solved model are difficult to interpret

Experimental-handwritten digits

DB1

- 1200 clean images
- 600 for training
- 600 for test
- 16*16 pixels

DB2

- 9300 actual mail pieces
- 7300 for training
- 2000 for test
- 16*16 pixels

Experimental Results

q	DB1 error	DB1 $<m>$	DB2 error	DB2 $<m>$	N
1 (linear)	3.2 %	36	10.5 %	97	256
2	1.5 %	44	5.8 %	89	$3 \cdot 10^4$
3	1.7 %	50	5.2 %	79	$8 \cdot 10^7$
4			4.9 %	72	$4 \cdot 10^9$
5			5.2 %	69	$1 \cdot 10^{12}$

- q: order of polynomial classifiers
- <m>: average number of support patterns per hypersurface
- N: dimension N of $\varphi - space$

Experimental-preprocessing

σ	DB1 error	DB1 $<m>$	DB2 error	DB2 $<m>$
no smoothing	1.5 %	44	4.9 %	72
0.5	1.3 %	41	4.6 %	73
0.8	0.8 %	36	5.0 %	79
1.0	0.3 %	31	6.0 %	83
1.2	0.8 %	31		

- q(DB1)=2, q(DB2)=4
- smoothing kernel is Gaussian with standard deviation

Experimental Results

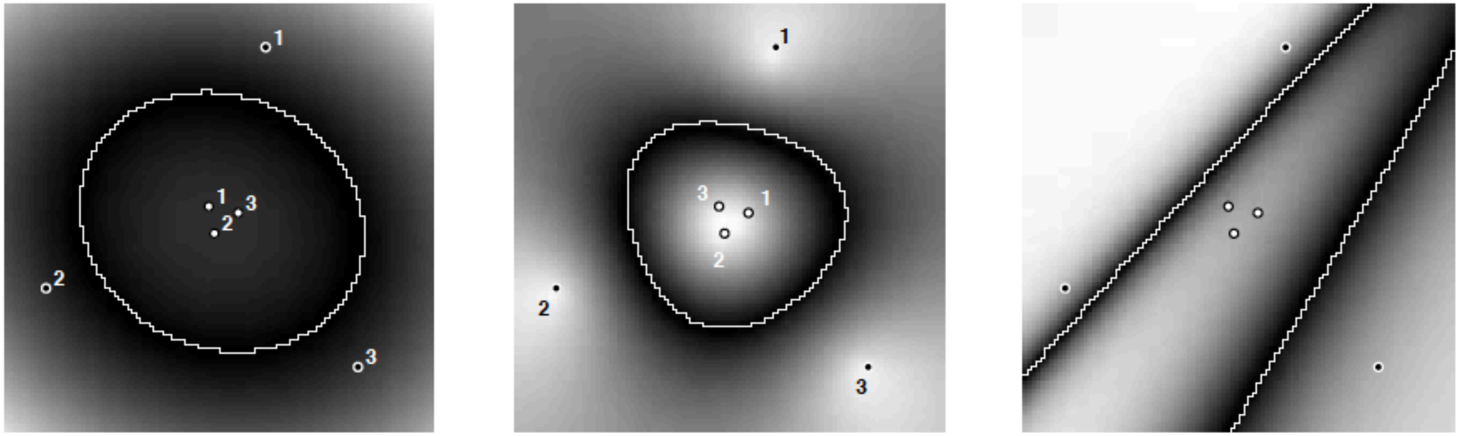


Figure 4: Decision boundaries for maximum margin classifiers with second order polynomial decision rule $K(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}' + 1)^2$ (left) and an exponential RBF $K(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|/2)$ (middle). The rightmost picture shows the decision boundary of a two layer neural network with two hidden units trained with backpropagation.

Thanks!