Machine Learning

By Ghazal Lalooha

Support Vector Machines

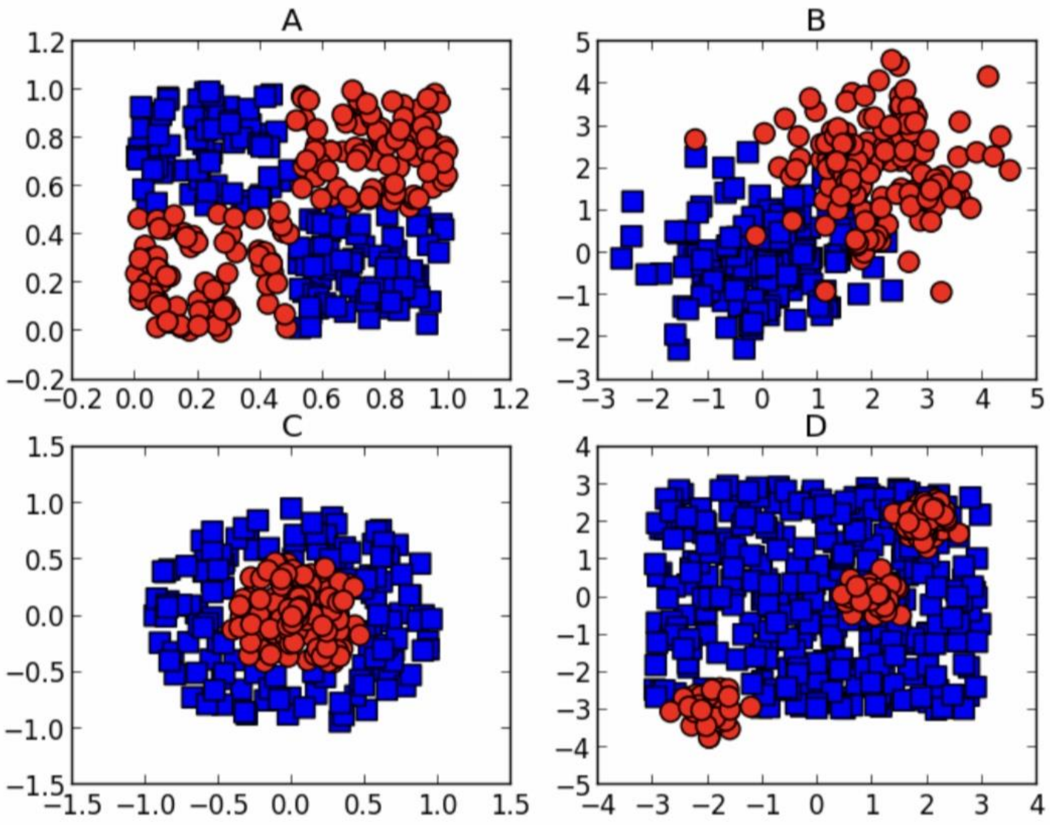
Table of Contents

- Basic concepts
- SVM problems: original problem and dual problem
- Training of linear non-linear SVMs
- Selection of kernel parameters and functions
- Multi-class classification
- Discussion

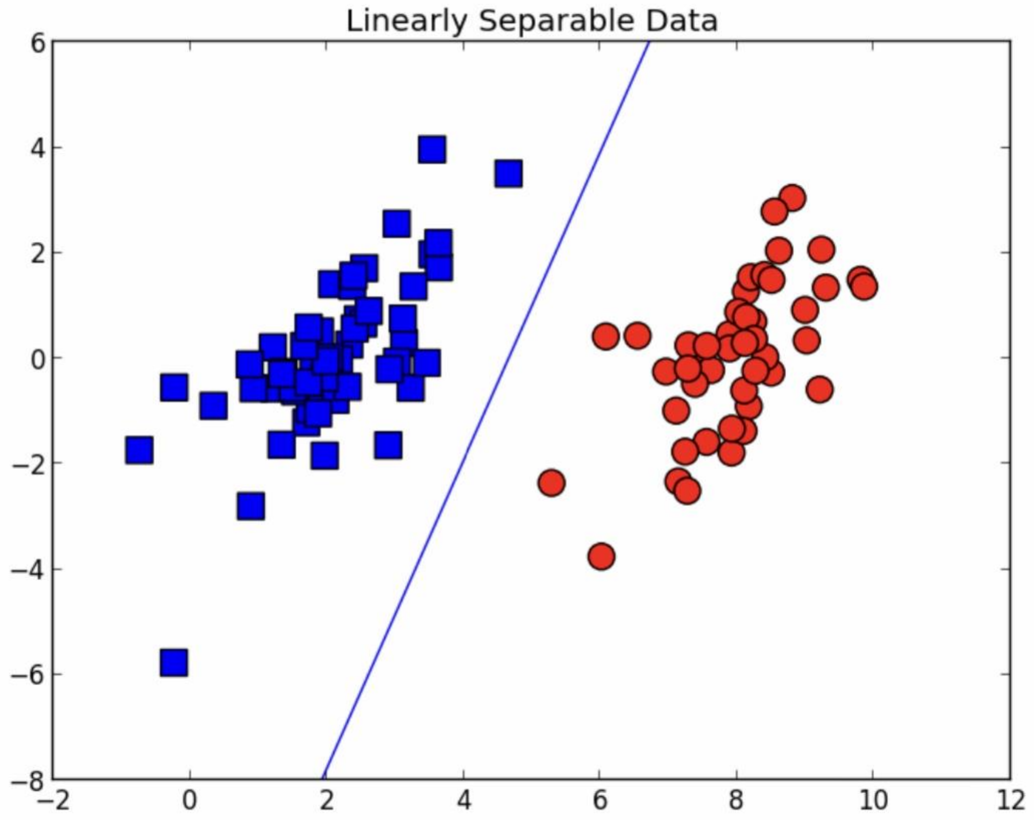
Introduction

- Support vector machines:
- One of the most popular machine learning algorithms!
- better separation of data than other machine learning methods (classification problems)
- It is relatively easy to use!
- Using the kernel trick:
- Classification, regression, estimation, distribution, single class classification and...

Linearly inseparable data

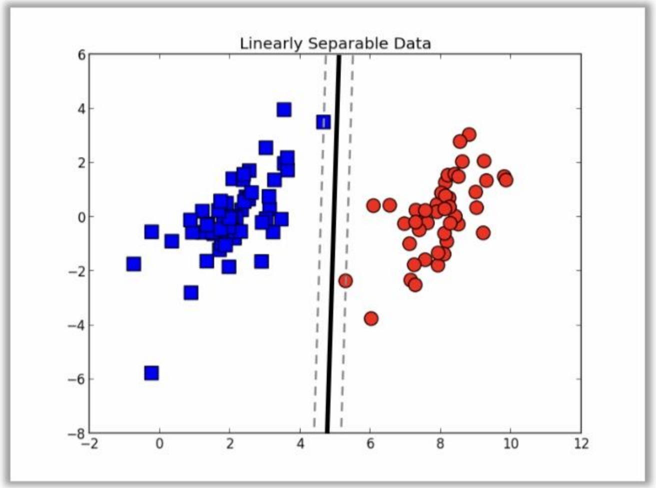


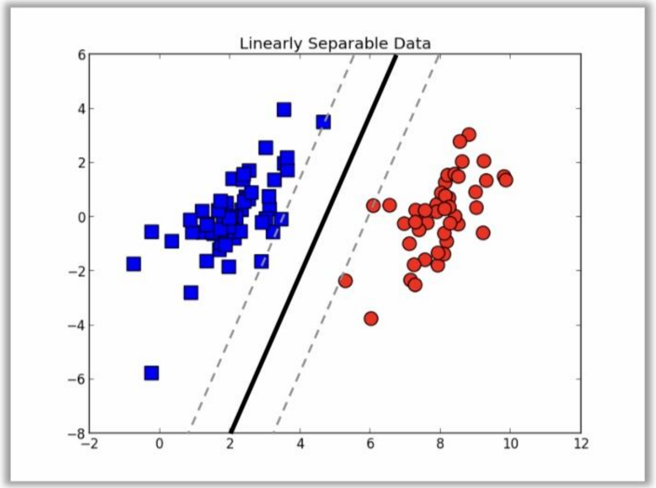
Linearly separable data



Optimum decision boundary

- Question: which decision boundary is better?



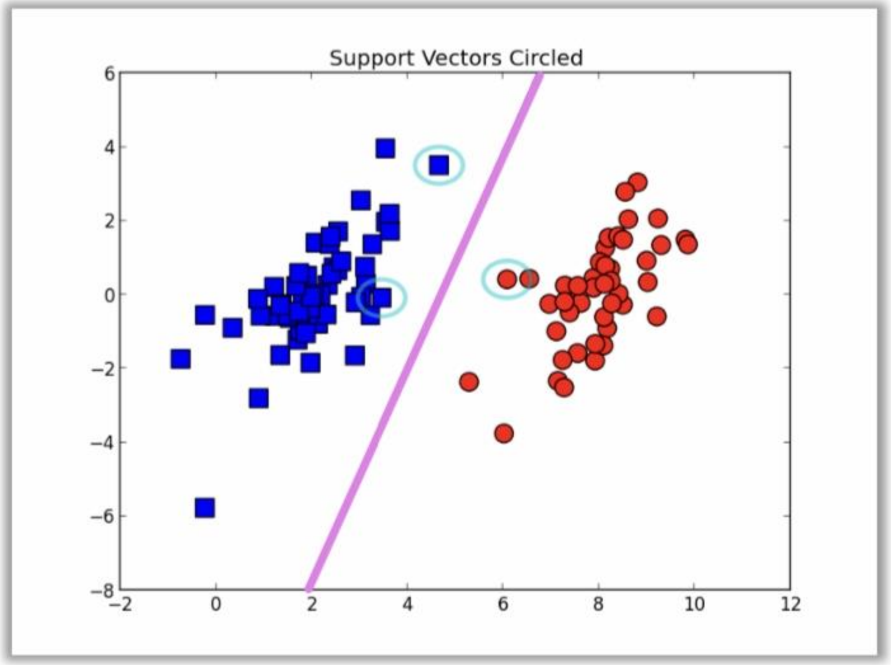


- Maximum margin solution:

  Maximum stability against data destruction. (increasing generalizability)

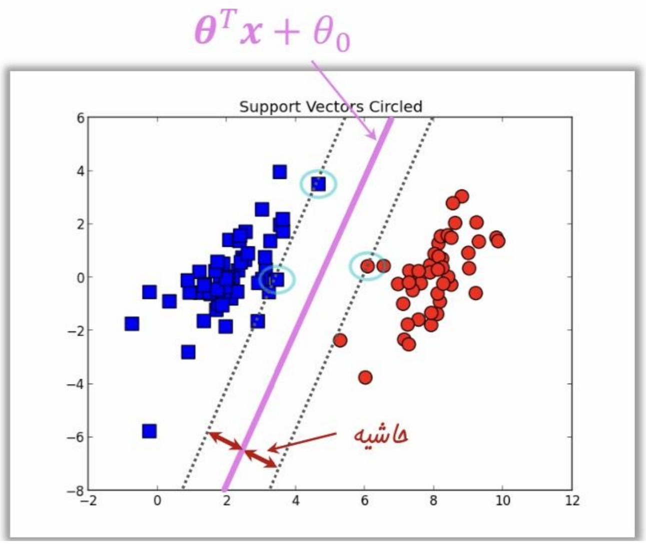
Support Vectors

- Support vector: the closest points to the decision boundary.
- Objective: to maximize the distance of the support vectors from the decision boundary.



Support Vector Machines: classifier with maximum margin

- Margin: the distance between support vectors and decision boundary.
- Goal: maximizing the distance between support vectors and decision boundary.



$$\frac{|\boldsymbol{\theta}^T \boldsymbol{x} + \theta_0|}{||\boldsymbol{\theta}||} \geq \rho$$

Optimum decision boundary: Symbols

- Training samples

$$X = (\boldsymbol{x}^t, y^t), \qquad y^t = \begin{cases} +1 & if\ x^t \in C_1 \\ -1 & if\ x^t \in C_2 \end{cases}$$

Goal: finding θ vector and θ_0 value so that:

$$\begin{aligned} \boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0 &\geq +1 \quad \text{for} \quad y^t = +1 \\ \boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0 &\leq -1 \quad \text{for} \quad y^t = -1 \end{aligned} \;\Rightarrow\; y^t(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) \geq +1$$

The objective function

- Objective: to maximize the distance of the support vectors from the decision boundary.
  - The distance between data x and decision boundary:

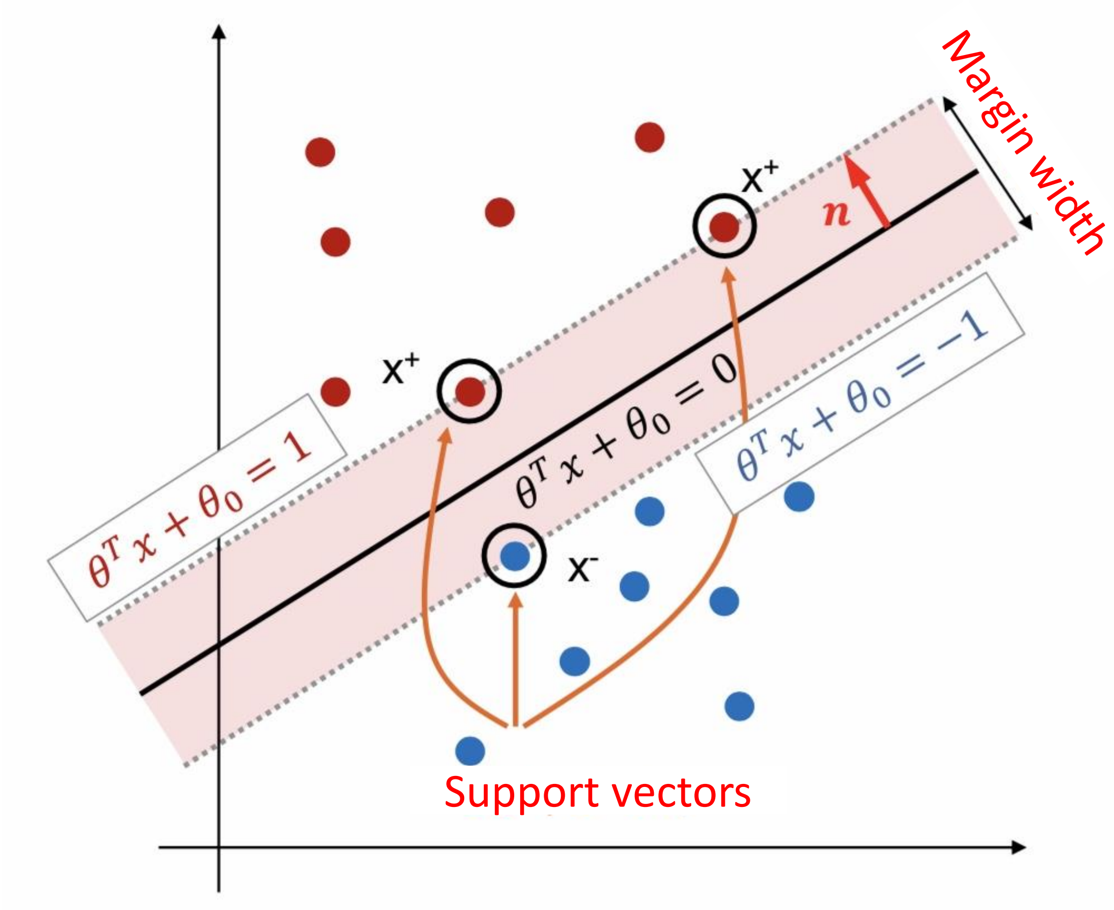
$$\frac{|\boldsymbol{\theta}^T \boldsymbol{x} + \theta_0|}{||\boldsymbol{\theta}||} \geq \rho \quad \Rightarrow \quad |\boldsymbol{\theta}^T \boldsymbol{x} + \theta_0| \geq \rho ||\boldsymbol{\theta}||$$

  - This equation has infinite answers. By considering $\rho||\boldsymbol{\theta}|| = 1$ we will have:

$$\rho||\boldsymbol{\theta}|| = 1 \Rightarrow \rho = \frac{1}{||\boldsymbol{\theta}||}$$

- Goal: to maximize the margin, we can minimize the size of θ vector.
- Constraints: The decision boundary must correctly separate the data of two classes from each other.

The objective function



We know:

$$\theta^T x^+ + \theta_0 = +1$$

$$\theta^T x^- + \theta_0 = -1$$

So:

$$M = (x^+ - x^-) \cdot n$$

$$= (x^+ - x^-) \cdot \frac{\theta}{\|\theta\|} = \frac{2}{\|\theta\|}$$

Objective function: formal expression

- Objective function:

$$\min \frac{1}{2} \|\boldsymbol{\theta}\|^2$$

$$\text{s.t.} \quad (\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) \geq +1 \quad \textit{if } y^t = +1$$

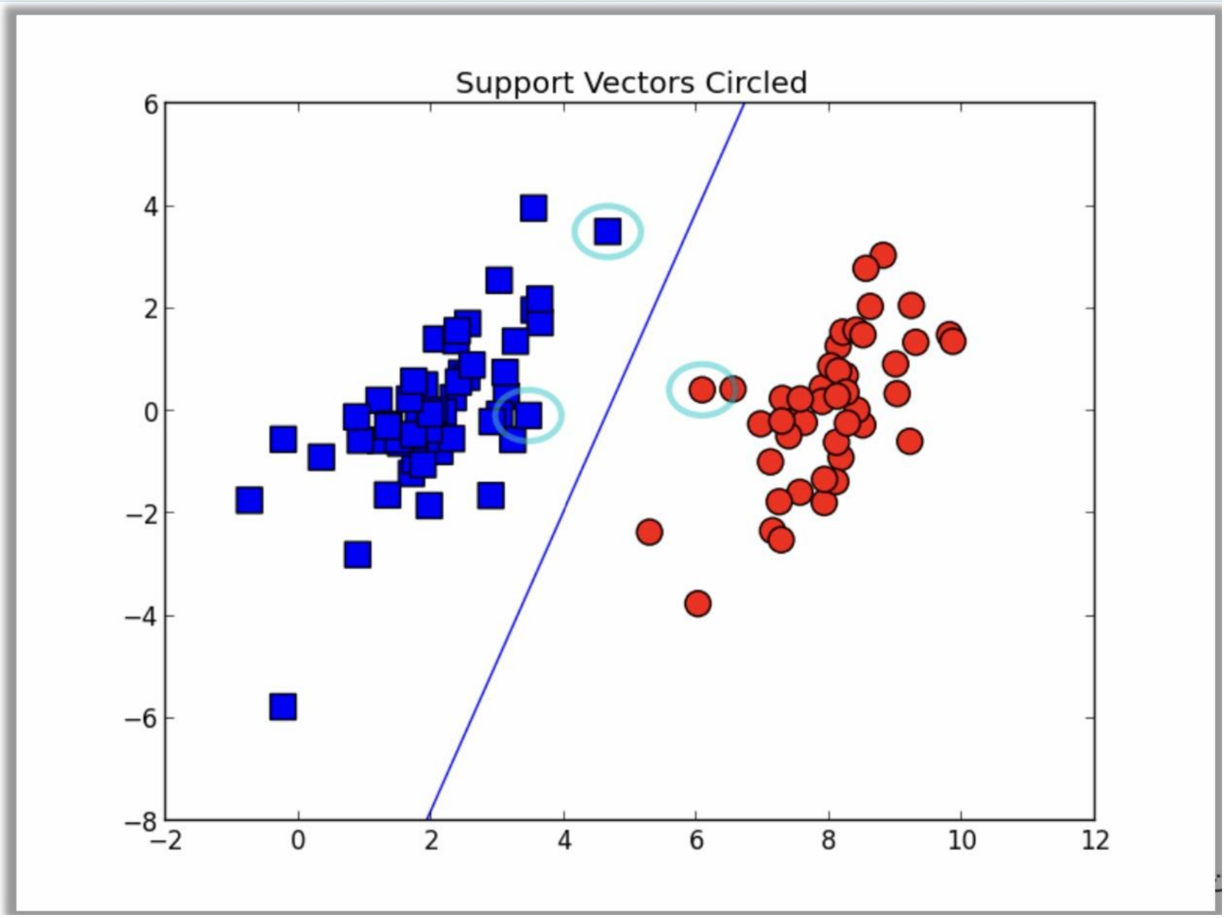
$$(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) \leq -1 \quad \textit{if } y^t = -1$$

- Simplification:

$$\min \frac{1}{2} \|\boldsymbol{\theta}\|^2$$

$$\text{s.t.} \quad y^t(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) \geq +1$$

Objective function



Objective function: formal expression

- Objective function:

$$\min \frac{1}{2}\|\boldsymbol{\theta}\|^2$$
$$\text{s.t.} \quad y^t(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) \geq +1$$

Convex optimization

- Problem solving using lagrange factor:

$$L_p = \frac{1}{2}\|\boldsymbol{\theta}\|^2 - \sum_{t=1}^{m} \alpha^t [y^t(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) - 1]$$

$$= \frac{1}{2}\|\boldsymbol{\theta}\|^2 - \sum_{t=1}^{m} \alpha^t y^t(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) + \sum_{t=1}^{m} \alpha^t$$

Joseph Louis Lagrange
1813-1736

Objective function: formal expression

$$L_p = \frac{1}{2}\|\boldsymbol{\theta}\|^2 - \sum_{t=1}^{m} \alpha^t[y^t(\boldsymbol{\theta}^T\boldsymbol{x}^t + \theta_0) - 1]$$

$$= \frac{1}{2}\|\boldsymbol{\theta}\|^2 - \sum_{t=1}^{m} \alpha^t y^t(\boldsymbol{\theta}^T\boldsymbol{x}^t + \theta_0) + \sum_{t=1}^{m} \alpha^t$$

Decision boundary of a linear combination of training data

$$\frac{\partial L_p}{\partial \boldsymbol{\theta}} = 0 \Rightarrow \boldsymbol{\theta} = \sum_{t=1}^{m} \alpha^t y^t \boldsymbol{x}^t$$

$$\frac{\partial L_p}{\partial \theta_0} = 0 \Rightarrow \sum_{t=1}^{m} \alpha^t y^t = 0$$

Objective function: formal expression

Minimal sequential optimization algorithm
Pellet (1999)

$$
\begin{aligned}
L_d &= \frac{1}{2}(\boldsymbol{\theta}^T\boldsymbol{\theta}) - \boldsymbol{\theta}^T \sum_{t=1}^{m} \alpha^t y^t \boldsymbol{x}^t - \theta_0 \sum_{t=1}^{m} \alpha^t y^t + \sum_{t=1}^{m} \alpha^t \\
&= -\frac{1}{2}(\boldsymbol{\theta}^T\boldsymbol{\theta}) + \sum_{t=1}^{m} \alpha^t \\
&= -\frac{1}{2}\sum_{t=1}^{m}\sum_{s=1}^{m} \alpha^t \alpha^s y^t y^s (\boldsymbol{x}^t)^T \boldsymbol{x}^s + \sum_{t=1}^{m} \alpha^t
\end{aligned}
$$

subject to $\sum_{t=1}^{m} \alpha^t y^t = 0$ and $\alpha^t \geq 0 \ \forall t$

- Most alphas are zero, and only a few are greater than zero.
- The x's for which the alpha value is greater than zero are the support vectors.

Objective Function: Simplified Format

Minimal sequential optimization algorithm ⟶
Pellet (1999)

$$L_d = -\frac{1}{2}\sum_{t=1}^{m}\sum_{s=1}^{m}\alpha^t\alpha^s y^t y^s (\boldsymbol{x}^t)^T \boldsymbol{x}^s + \sum_{t=1}^{m}\alpha^t$$

$$= -\frac{1}{2}\alpha^T Q\alpha + e^T\alpha$$

$Q_{ts} = y^t y^s (x^t)^T x^s,$ $\quad e = [1 \quad 1 \quad \dots \quad 1]^T \in \mathbb{R}^m$

subject to $\sum_{t=1}^{m}\alpha^t y^t = 0$ and $\alpha^t \geq 0\ \forall t$

- Most alphas are zero, and only a few are greater than zero.
- The x's for which the alpha value is greater than zero are the support vectors.

Linearly inseparable data: soft margin

- Question: What if the data is not linearly separable?
- Soft Margin: Allowing a slight margin of

error in separation

$$y^t(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) \geq 1 - \varepsilon^t$$

- Soft error:

$$soft\ error = \sum_{t=1}^{m} \varepsilon^t$$

- New objective function:

Penalty factor

$$\min \frac{1}{2} \|\boldsymbol{\theta}\|^2 + C \sum_{t=1}^{m} \varepsilon^t$$

$$\text{s.t.} \quad y^t(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) \geq 1 - \varepsilon^t$$

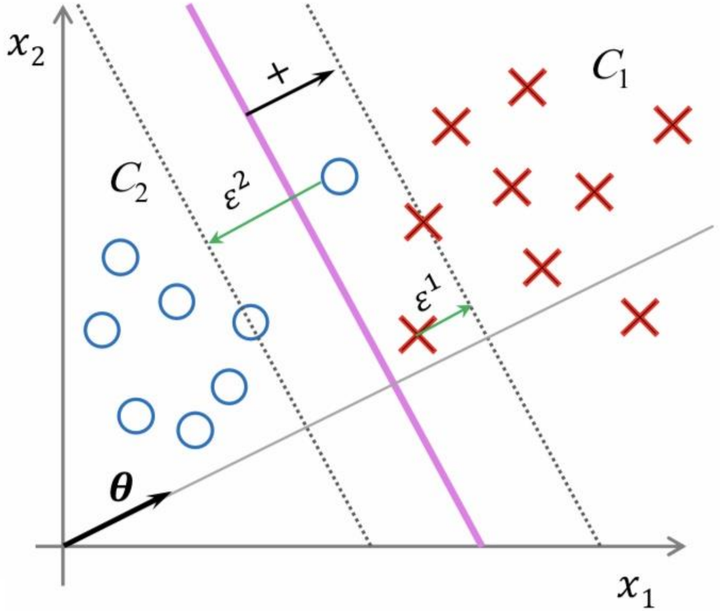
$$\varepsilon^t \geq 0$$

Linearly inseparable data: soft margin

$$\min \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{t=1}^{m} \varepsilon^t$$

$$\text{s.t.} \quad y^t(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) \geq 1 - \varepsilon^t$$

$$\varepsilon^t \geq 0$$



Lagrange factors

$$L_p = \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{t=1}^{m} \varepsilon^t - \sum_{t=1}^{m} \alpha^t[y^t(\boldsymbol{\theta}^T \boldsymbol{x}^t + \theta_0) - 1 + \varepsilon^t] - \sum_{t=1}^{m} \mu^t \varepsilon^t$$

Linearly inseparable data: soft margin

$$L_p = \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{t=1}^{m}\varepsilon^t - \sum_{t=1}^{m}\alpha^t[y^t(\boldsymbol{\theta}^T\boldsymbol{x}^t + \theta_0) - 1 + \varepsilon^t] - \sum_{t=1}^{m}\mu^t\varepsilon^t$$

$$\frac{\partial L_p}{\partial \boldsymbol{\theta}} = 0 \Rightarrow \boldsymbol{\theta} = \sum_{t=1}^{m}\alpha^t y^t \boldsymbol{x}^t$$

$$\frac{\partial L_p}{\partial \theta_0} = 0 \Rightarrow \sum_{t=1}^{m}\alpha^t y^t = 0$$

$$\frac{\partial L_p}{\partial \varepsilon^t} = 0 \Rightarrow C - \alpha^t - \mu^t = 0 \Rightarrow 0 \le \alpha^t \le C$$

Objective function: Dugan

$$L_d = -\frac{1}{2}\sum_{t=1}^{m}\sum_{s=1}^{m}\alpha^t\alpha^s y^t y^s (\boldsymbol{x}^t)^T\boldsymbol{x}^s + \sum_{t=1}^{m}\alpha^t$$
$$= -\frac{1}{2}\alpha^T Q\alpha + e^T\alpha$$

subject to $\sum_{t=1}^{m}\alpha^t y^t = 0$ and $0 \le \alpha^t \le C\ \forall t$

Minimal sequential optimization algorithm
Pellet (1999)

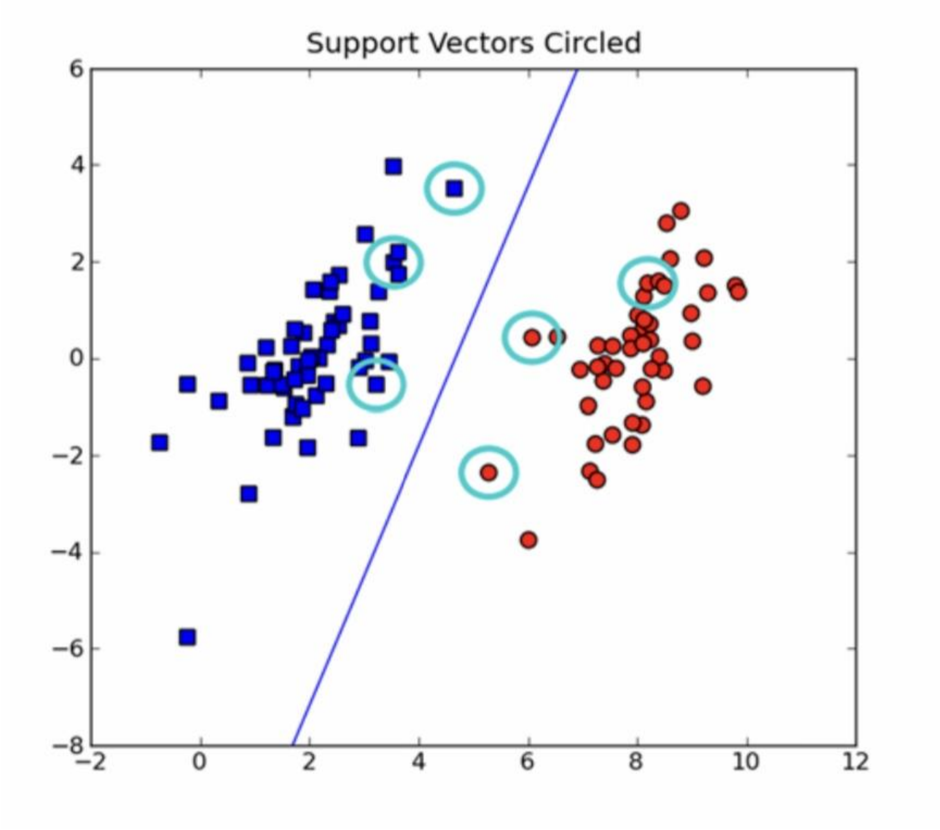
Error estimation based on the number of support vectors:

- Most alphas are zero, and only a few are greater than zero.
- The x's for which the alpha value is greater than zero are the support vectors.

$$E_m[P(error)] \le \frac{E_m[\,\#of\ support\ vectors]}{m}$$

Soft Margin Solution

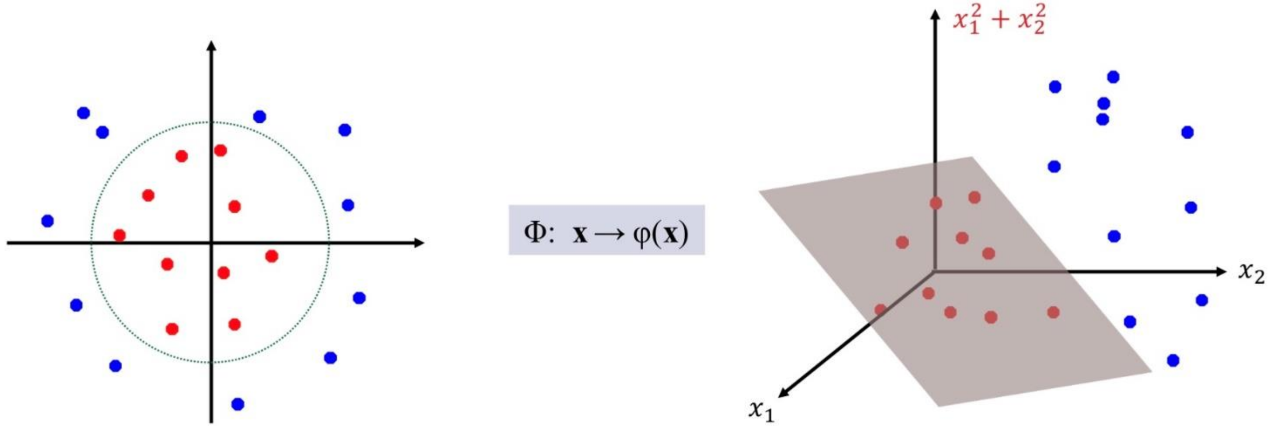
- Support Vectors:



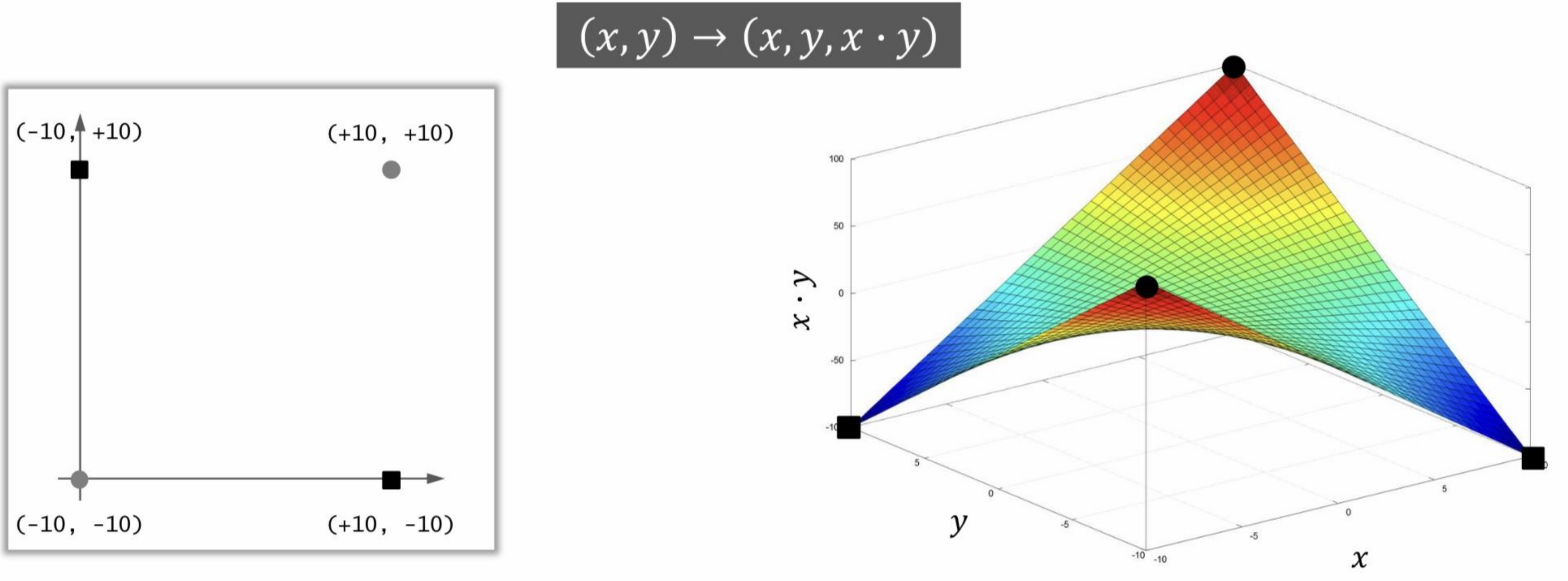
Kernel Trick

Kernel Function

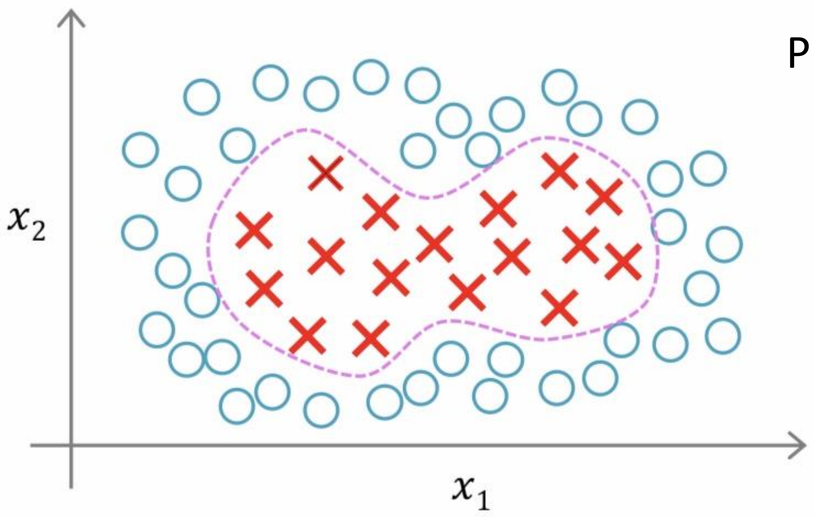
- Idea: Mapping the problem to a new feature space using non-linear transformations.
  - Using a linear model in the new space to classify data.
  - A linear model in the new space corresponds to a non-linear model in the original space.



Example: XOR problem



Non-linear Decision Boundaries



Prediction: y = 1 if:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2 + \cdots \geq 0$$

Features:

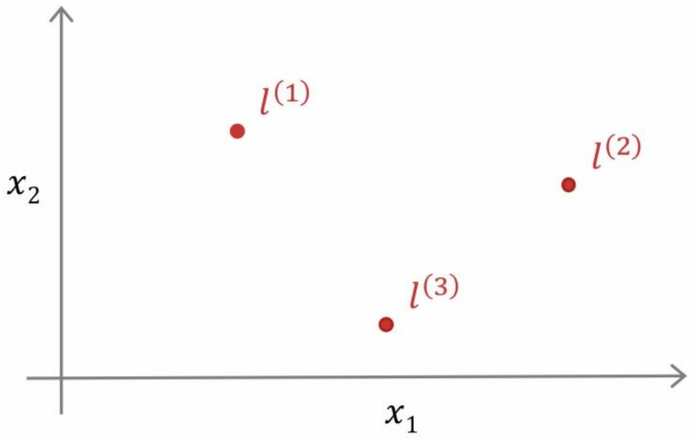
$$f_1 = x_1, \quad f_2 = x_2, \quad f_3 = x_1^2, \quad f_4 = x_2^2, \quad f_5 = x_1 x_2, \quad \cdots$$

$$h_\theta(f) = \theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \theta_4 f_4 + \theta_5 f_5 + \cdots$$ ⟵ Linear Decision Boundaries

- Question: Is there a better way to select new features f_1, f_2, ...?

Kernel

- Idea: Given x, select a new set of features based on its similarity to guide points l_1, l_2, l_3.



$$f_1 = sim(x, l^{(1)}) = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

$$f_2 = sim(x, l^{(2)}) = \exp\left(-\frac{\|x - l^{(2)}\|^2}{2\sigma^2}\right)$$

$$f_3 = sim(x, l^{(3)}) = \exp\left(-\frac{\|x - l^{(3)}\|^2}{2\sigma^2}\right)$$

Kernel (Gaussian kernel)

- Kernel function: a measure to calculate the similarity between x and y data

Kernels as a measure of similarity

Kernel function:

$$f_i = sim(x, l^{(i)}) = \exp\left(-\frac{\left\|x - l^{(i)}\right\|^2}{2\sigma^2}\right)$$

The first situation: $x \approx l^{(i)}$

$$f_i \approx \exp\left(-\frac{0}{2\sigma^2}\right) = \exp(0) = 1$$

The second situation: x is too far from $l^{(i)}$

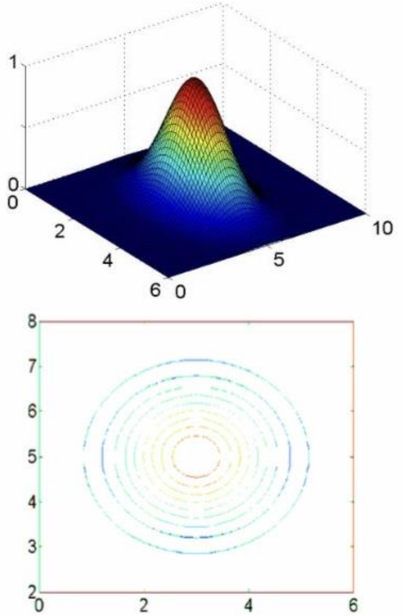
$$f_i \approx \exp\left(-\frac{\infty}{2\sigma^2}\right) = \exp(-\infty) = 0$$

Kernels as a measure of similarity

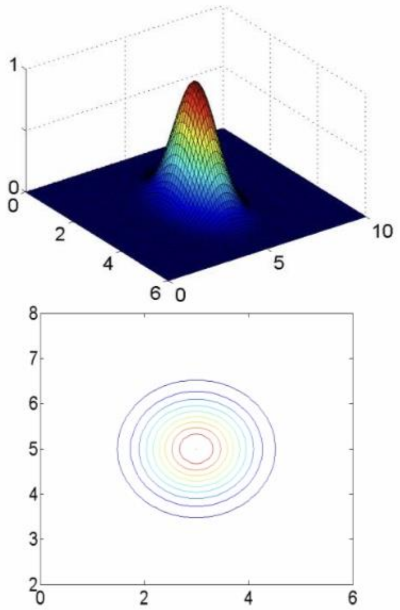
Example:

$$l^{(1)} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$$

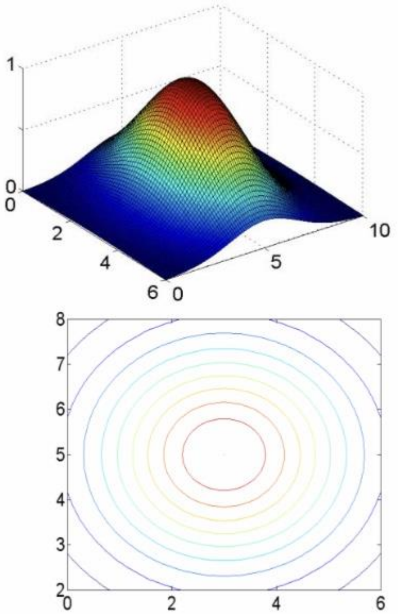
$\sigma^2 = 1$



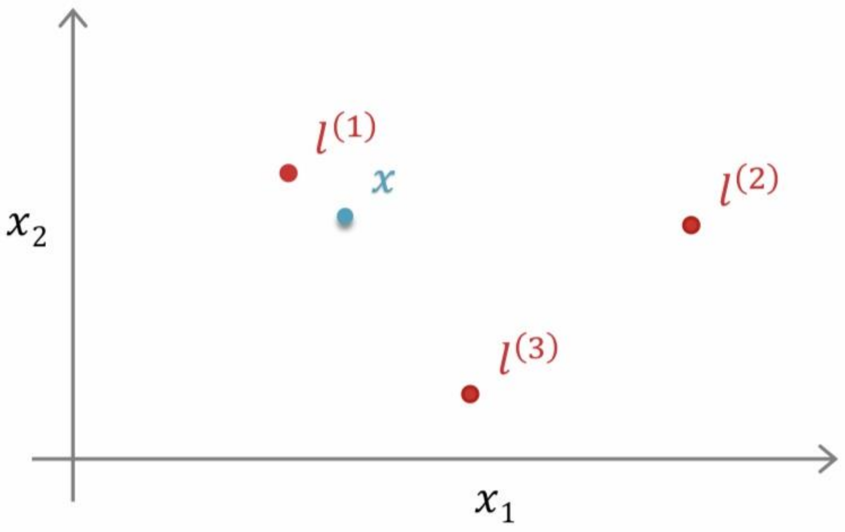
$\sigma^2 = 0.5$



$\sigma^2 = 3$



Kernels as a measure of similarity



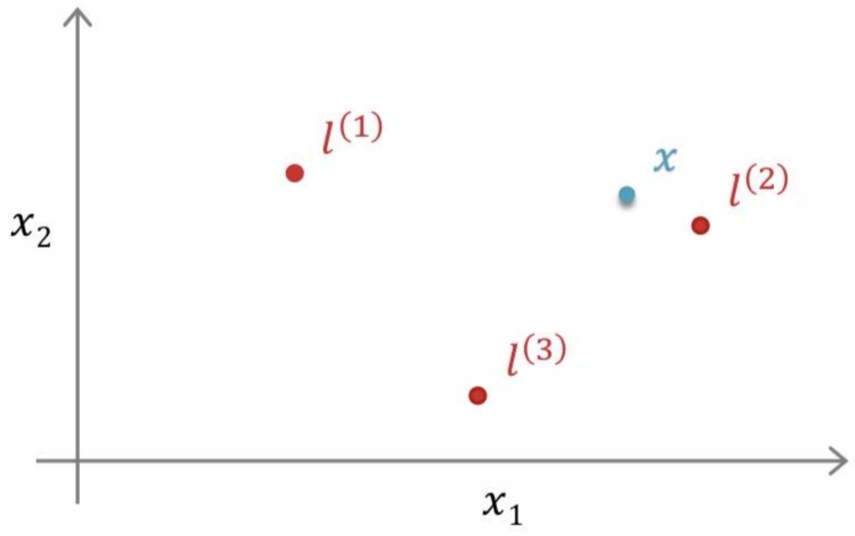
Prediction: y = 1 if:

$$\boxed{\underset{\substack{\uparrow\\ -0.5}}{\theta_0} + \underset{\substack{\uparrow\\ 1.0}}{\theta_1} f_1 + \underset{\substack{\uparrow\\ 1.0}}{\theta_2} f_2 + \underset{\substack{\uparrow\\ 0.0}}{\theta_3} f_3 \geq 0}$$

$$f_1 \approx 1, f_2 \approx f_3 \approx 0$$

$$h_\theta(f) \approx -0.5 + (1.0)(1.0) + (1.0)(0.0) + (0.0)(0.0) = 0.5 \geq 0 \Rightarrow y = 1$$

Kernels as a measure of similarity



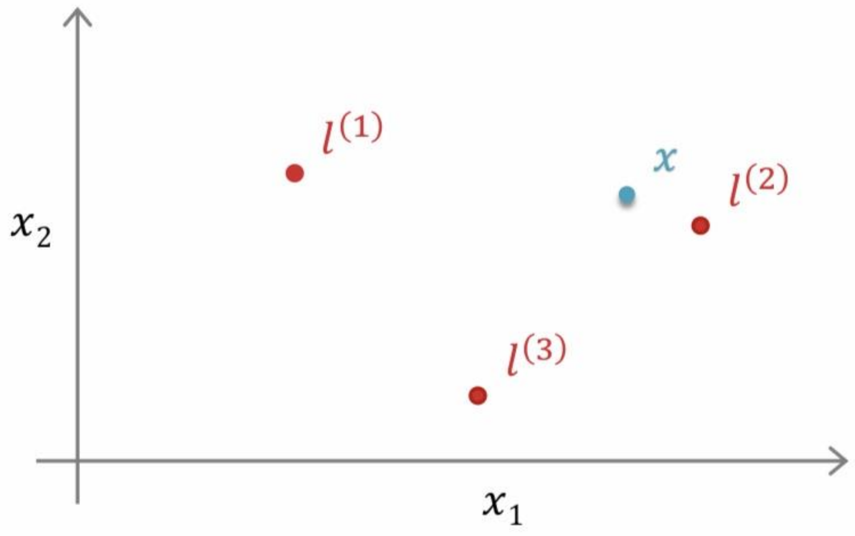
Prediction: y = 1 if:

$$\underset{-0.5}{\theta_0} + \underset{1.0}{\theta_1} f_1 + \underset{1.0}{\theta_2} f_2 + \underset{0.0}{\theta_3} f_3 \geq 0$$

$$f_1 \approx f_3 \approx 0, f_2 \approx 1$$

$$h_\theta(f) \approx -0.5 + (1.0)(0.0) + (1.0)(1.0) + (0.0)(0.0) = 0.5 \geq 0 \Rightarrow y = 1$$

Kernels as a measure of similarity



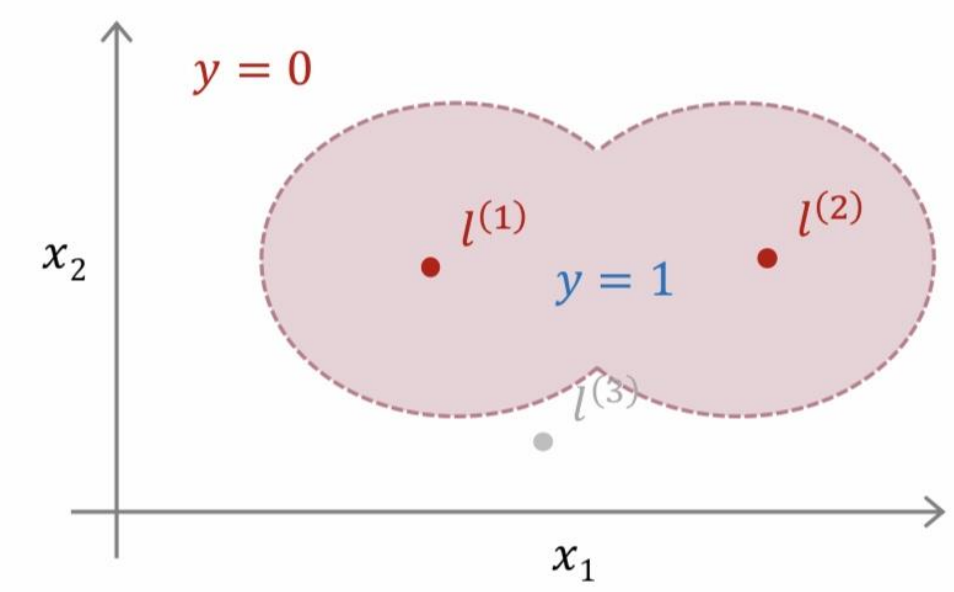
Prediction: y = 1 if:

$$\underset{-0.5}{\theta_0} + \underset{1.0}{\theta_1} f_1 + \underset{1.0}{\theta_2} f_2 + \underset{0.0}{\theta_3} f_3 \geq 0$$

$$f_1 \approx f_3 \approx 0, f_2 \approx 1$$

$$h_\theta(f) \approx -0.5 + (1.0)(0.0) + (1.0)(1.0) + (0.0)(0.0) = 0.5 \geq 0 \Rightarrow y = 1$$

Kernels as a measure of similarity

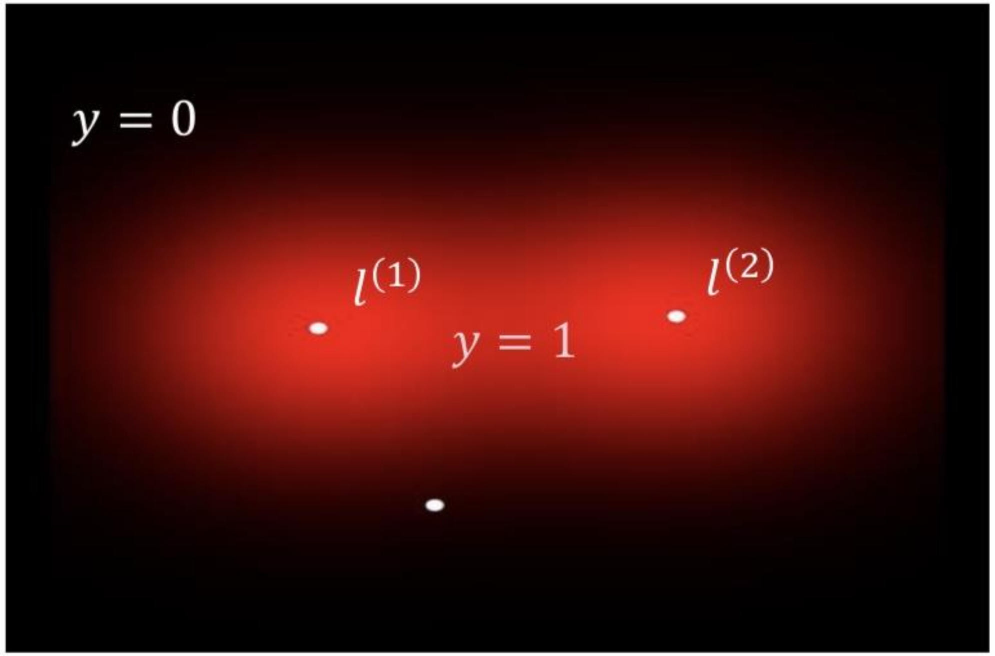


Prediction: y = 1 if:

$$\underset{-0.5}{\theta_0} + \underset{1.0}{\theta_1} f_1 + \underset{1.0}{\theta_2} f_2 + \underset{0.0}{\theta_3} f_3 \geq 0$$

- Decision boundary: it classifies points close to $l^{(1)}$ and $l^{(2)}$ in class 1 and other points in class zero.

Kernels as a measure of similarity



Prediction: y = 1 if:

$$\underset{-0.5}{\theta_0} + \underset{1.0}{\theta_1} f_1 + \underset{1.0}{\theta_2} f_2 + \underset{0.0}{\theta_3} f_3 \geq 0$$

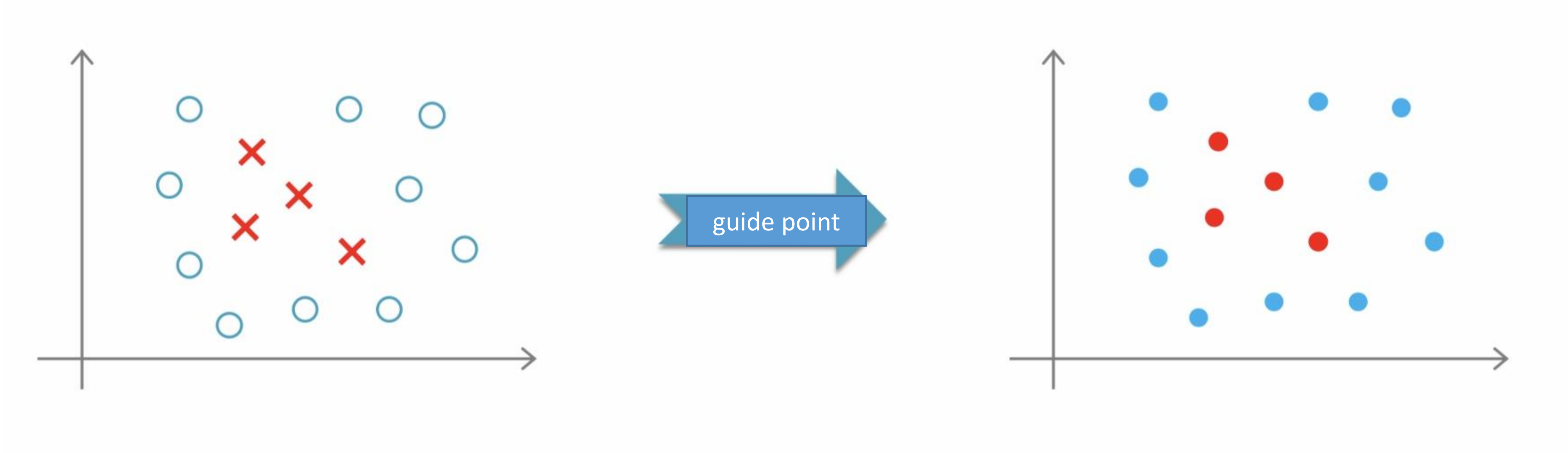
- Decision boundary: it classifies points close to $l^{(1)}$ and $l^{(2)}$ in class 1 and other points in class zero.

Remaining details:

- Question: How does the learning algorithm automatically select guide points?
- Question: How are the appropriate values for kernel function parameters determined?
- Question: Are there other types of kernels?

Selection of guide points

- Question: How does the learning algorithm automatically select guide points?
  - For each sample in the training set, a guide point equal to that sample is selected.



Mapping features

Training Set:

$$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$$

Guide Point:

$$l^{(1)} = x^{(1)},\ l^{(2)} = x^{(2)}, \cdots,\ l^{(m)} = x^{(m)}$$

Feature Area Mapping:

$$x = \begin{bmatrix} x_0 = 1 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \xrightarrow{\text{Feature Area Mapping}} f = \begin{bmatrix} f_0 = 1 \\ f_1 = K(x, x^{(1)}) \\ f_2 = K(x, x^{(2)}) \\ \vdots \\ f_m = K(x, x^{(m)}) \end{bmatrix}$$

Kernel Trick

- Kernel function: preprocessing x data using kernel functions:

$$\begin{aligned} \boldsymbol{z} &= \varphi(\boldsymbol{x}) \\ &= (\varphi_1(\boldsymbol{x}), \varphi_2(\boldsymbol{x}), \dots, \varphi_k(\boldsymbol{x})) \end{aligned}$$

$$g(\boldsymbol{z}) = \boldsymbol{\theta}^T \boldsymbol{z} + \theta_0$$

$$g(\boldsymbol{x}) = \boldsymbol{\theta}^T \varphi(\boldsymbol{x}) + \theta_0$$

It may be infinite!!

- SVM solution:

$$\boldsymbol{\theta} = \sum_{t=1}^{m} \alpha^t y^t \boldsymbol{z}^t = \sum_{t=1}^{m} \alpha^t y^t \varphi(\boldsymbol{x}^t)$$

- New data classification:

$$g(\boldsymbol{x}) = \boldsymbol{\theta}^T \varphi(\boldsymbol{x}) + \theta_0 = \left(\sum_{t=1}^{m} \alpha^t y^t \varphi(\boldsymbol{x}^t)^T\right) \varphi(\boldsymbol{x}) + \theta_0 = \left(\sum_{t=1}^{m} \alpha^t y^t \varphi(\boldsymbol{x}^t)^T \varphi(\boldsymbol{x})\right) + \theta_0$$

$$g(\boldsymbol{x}) = \left(\sum_{t=1}^{m} \alpha^t y^t k(\boldsymbol{x}^t, \boldsymbol{x})\right) + \theta_0 \longleftarrow$$

Decision Boundary

Training data new data

Kernel Functions

$$L_p = \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{t=1}^{m}\varepsilon^t$$

s.t. $\quad y^t\boldsymbol{\theta}^T\varphi(\boldsymbol{x}^t) \geq 1 - \varepsilon^t$

$$\varepsilon^t \geq 0$$

$$L_p = \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{t=1}^{m}\varepsilon^t - \sum_{t=1}^{m}\alpha^t[y^t\boldsymbol{\theta}^T\varphi(\boldsymbol{x}^t) - 1 + \varepsilon^t] - \sum_{t=1}^{m}\mu^t\varepsilon^t$$

Lagrange factors

Lagrange factors

Kernel Functions: main issue

$$L_p = \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{t=1}^{m}\varepsilon^t - \sum_{t=1}^{m}\alpha^t[y^t\boldsymbol{\theta}^T\varphi(\boldsymbol{x}^t) - 1 + \varepsilon^t] - \sum_{t=1}^{m}\mu^t\varepsilon^t$$

$$\frac{\partial L_p}{\partial \boldsymbol{\theta}} = 0 \Rightarrow \boldsymbol{\theta} = \sum_{t=1}^{m}\alpha^t y^t \varphi(\boldsymbol{x}^t)$$

$$\frac{\partial L_p}{\partial \varepsilon^t} = 0 \Rightarrow C - \alpha^t - \mu^t = 0 \Rightarrow 0 \leq \alpha^t \leq C$$

Kernel functions: Dugan's problem

$$L_d = -\frac{1}{2}\sum_{t=1}^{m}\sum_{s=1}^{m}\alpha^t\alpha^s y^t y^s \varphi(\boldsymbol{x}^t)^T\varphi(\boldsymbol{x}^s) + \sum_{t=1}^{m}\alpha^t$$

subject to $\sum_{t=1}^{m}\alpha^t y^t = 0$ and $0 \le \alpha^t \le C \; \forall t$

The idea of kernel machines (kernel trick) :
Replacing the inner product of basis functions with a kernel function in the form of $K(x^t, x^s)$

$$L_d = -\frac{1}{2}\sum_{t=1}^{m}\sum_{s=1}^{m}\alpha^t\alpha^s y^t y^s K(\boldsymbol{x}^t, \boldsymbol{x}^s) + \sum_{t=1}^{m}\alpha^t$$

Geram matrix: A symmetric and positive definite matrix (for linear separability)

Kernel functions: polynomial kernel

- Polynomial Kernel: A polynomial of degree q.

$$K(x^t, x) = (x^T x^t + 1)^q$$

- Example: [q = 2, d = 2]

$$K(x, y) = (x^T y + 1)^2$$

3 multiplication, 2 sum

$$= (x_1 y_1 + x_2 y_2 + 1)^2$$

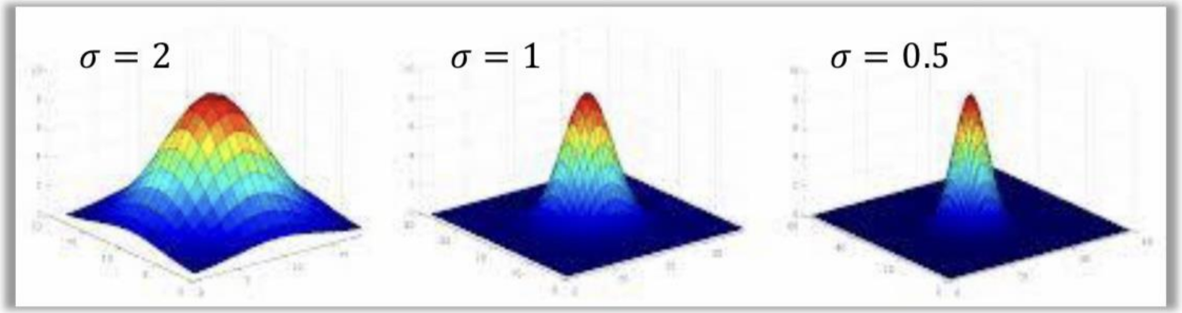
$$= 1 + 2x_1 y_1 + 2x_2 y_2 + 2x_1 x_2 y_1 y_2 + x_1^2 y_1^2 + x_2^2 y_2^2$$

6 multiplication, 5 sum

$$\varphi(x) = \left[1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2, x_1^2, x_2^2\right]^T$$

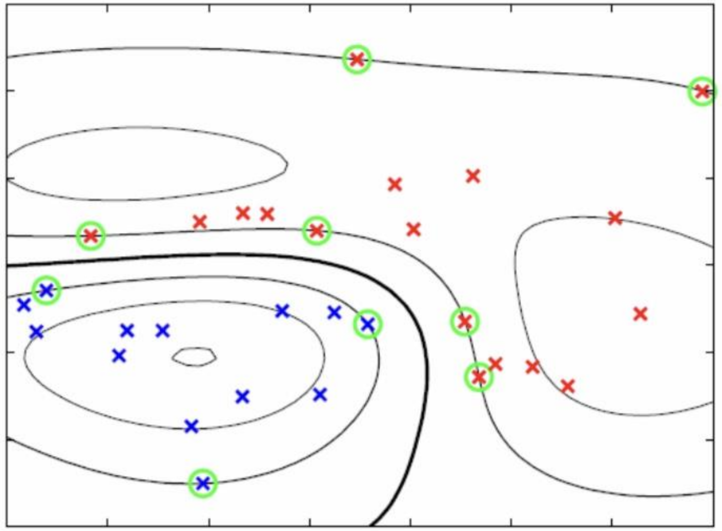
$$\varphi(y) = \left[1, \sqrt{2}y_1, \sqrt{2}y_2, \sqrt{2}y_1 y_2, y_1^2, y_2^2\right]^T$$

Gaussian Kernel Function



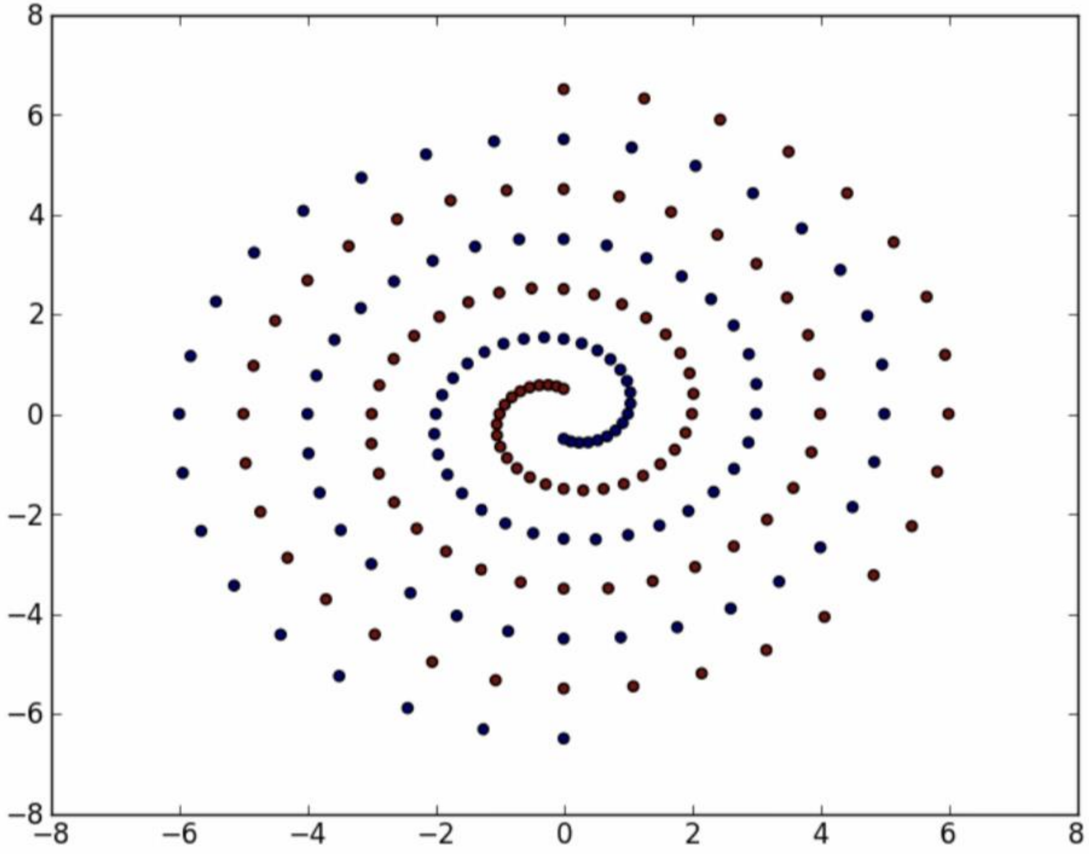
Gaussian Kernel Function:

$$K(\boldsymbol{x}^t, \boldsymbol{x}) = \exp\left(-\frac{\|\boldsymbol{x}^t - \boldsymbol{x}\|^2}{2\sigma^2}\right)$$

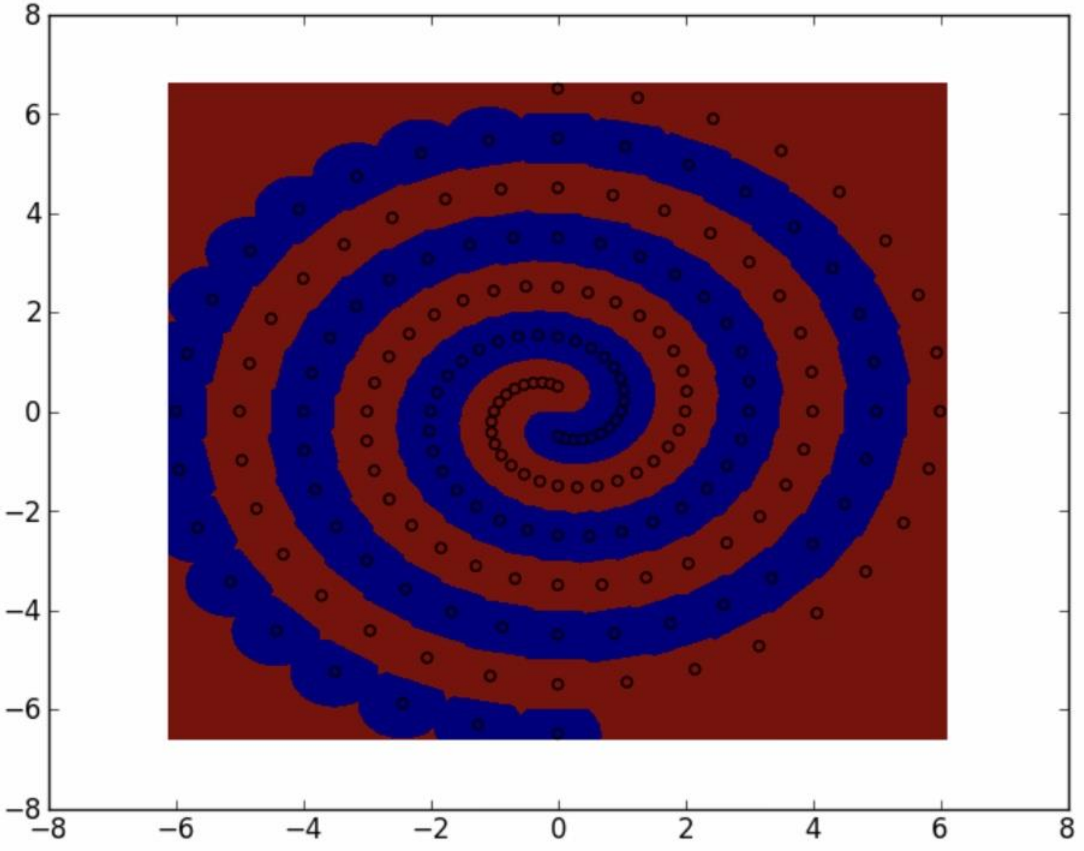


Finding a suitable value for sigma:
Using the validation set (model selection)
Larger values: smoother decision boundary

Example: Gaussian Kernel Function

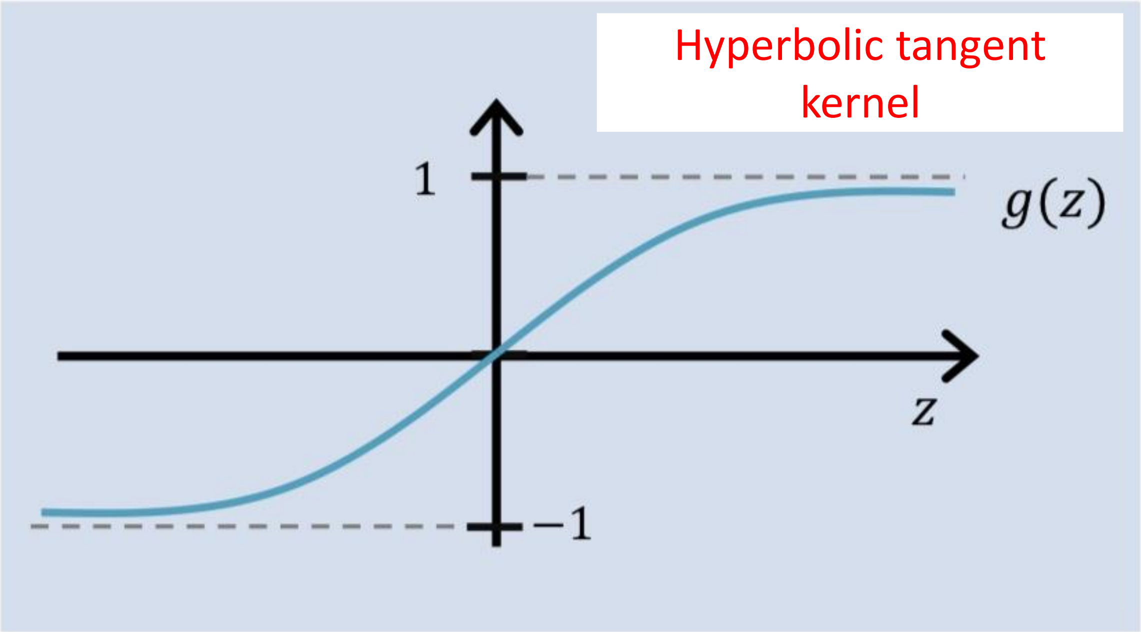


Example: Gaussian Kernel Function



Kernel Functions: Other Kinds

- Kernel function: a measure to calculate the similarity between x and y data.
- Other types:
- Hyperbolic tangent kernel
- Threaded kernel
- tree kernel
- Cornellography



$$K(x^t, x) = \tanh(2x^T x^t + 1)$$

SVM Parameters

- Question: How are the appropriate values for kernel function parameters determined?
- Parameter C:
- Smaller values: more bias, less variance
- Larger values: less bias, more variance

- Sigma parameter:
- Smaller values: less bias, more variance
- Larger values: more bias, less variance

Determining the values of both parameters:
Search networking

Multiclass classification

- First method: one against all (preferred method)
  - Training: training k support vector machines, one for each class
  - Experiment: Calculate $g_i(x)$ for $0 \leq I \leq k$ and choose the largest value

$$\arg\max_i g_i(x)$$

- The second method: separating two by two
  - Training k(k-1)/2 support vector machine such that $g_{ij}(x)$ separates examples of two classes C_i and C_j.
  - Simpler and faster
- The third method: solving a multi-class optimization problem

$$\min \frac{1}{2}\sum_{i=1}^{k} \|\boldsymbol{\theta}_i\|^2 + C\sum_i \sum_t \varepsilon_i^t$$

$$\text{s.t.} \quad \boldsymbol{\theta}_{z^t}\boldsymbol{x}^t + \theta_{z^t 0} \geq \boldsymbol{\theta}_i \boldsymbol{x}^t + \theta_{i0} + 2 - \varepsilon_i^t, \forall i \neq z^t$$

A guide to using SVM

- Implementation:
  - Using existing software packages such as LIBSVM and SVM light
- Determine the kernel function:
  - Linear kernel (not using kernel): when n is much larger than m.
  - Gaussian, polynomial, string and...
- Determining the value of parameters: grid search
  - Select a value for the C parameter
  - Value selection for kernel function parameters (such as sigma)

	σ					
C	err	.01	.1	1	10	100
	.01					
	.1					
	1					
	10					
	100					

SVM, Logistic Regression or Neural Network?

- Case 1: (n much larger than m)
  - Example: Spam detection (1000 training examples, 50000 features)
  - Logistic regression or linear SVM
- Situation 2: (low number of features, high number of training samples)
  - SVM with Gaussian kernel
- Note: Neural networks can be used in all the above situations, but they may need more time for training.

More about Kernels

- Question: How do we know that the use of kernels helps us in separating data?
- In n-dimensional space, any set of n independent vectors are linearly separable.
- If the matrix K is a positive definite matrix, then the data are linearly separable.
- Theorem: The matrix K is a positive definite matrix, because K = $L^T L$
- column i in matrix L is equal to vector $\Phi(x^{(i)})$
- Proof: Consider a non-zero vector v. in this case:

$$v^T K v = v^T L^T L v = (Lv)^T (Lv) = w^T w = \|w\|^2 \geq 0$$

- And since L and v are both opposite to zero,

vector W is also opposite to zero. that's mean:

$$\|w\|^2 > 0 \Rightarrow v^T K v > 0 \Rightarrow K \text{ is positive definite}$$