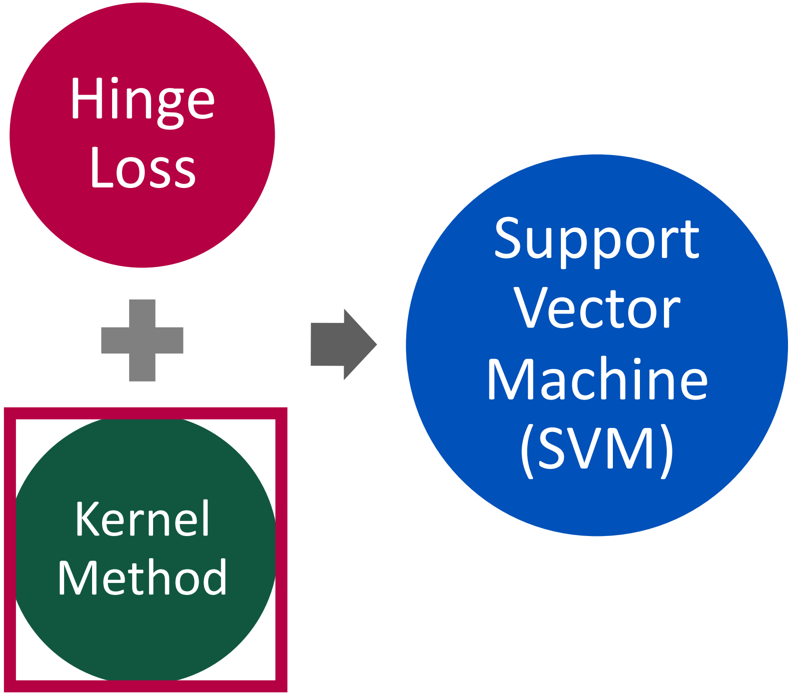
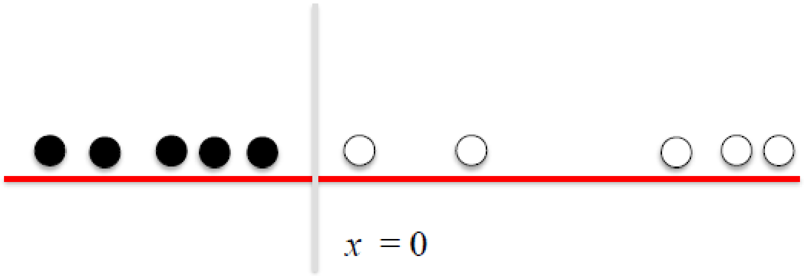
Kernelized SVM

Part I

Outline

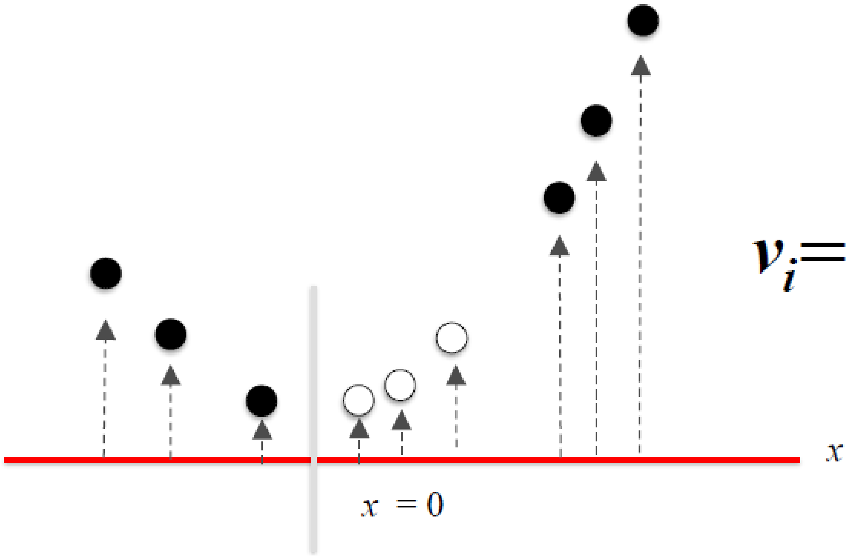


A Simple 1-Dimensional Classification Problem



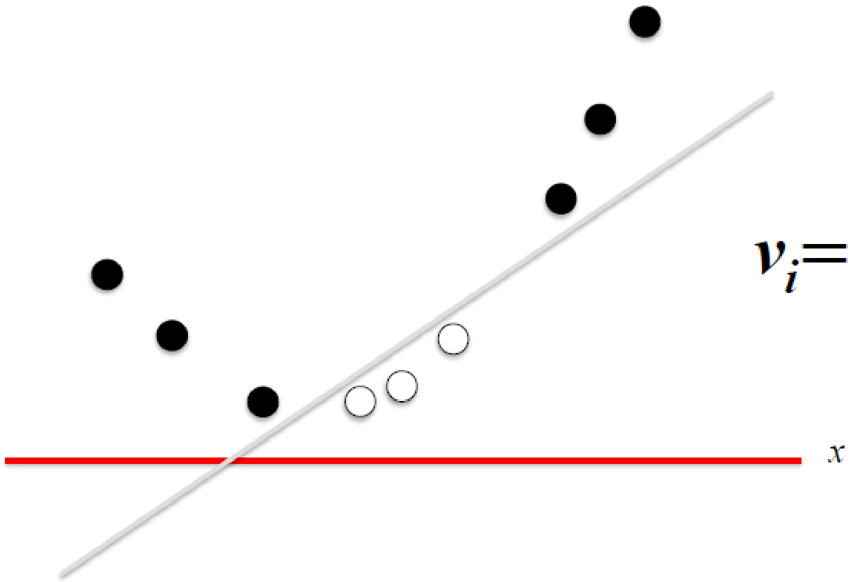
A More Perplexing 1-Dimensional Classification Problem

Transform the Data by Adding a Second Dimension/Feature



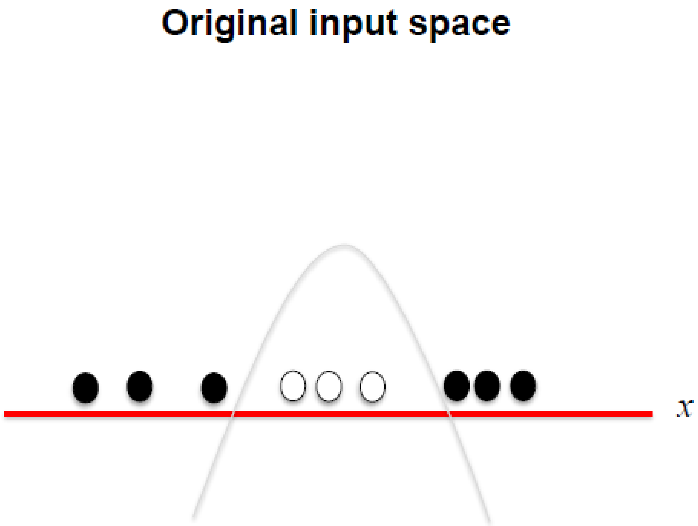
$$v_i = (x_i, x_i^2)$$

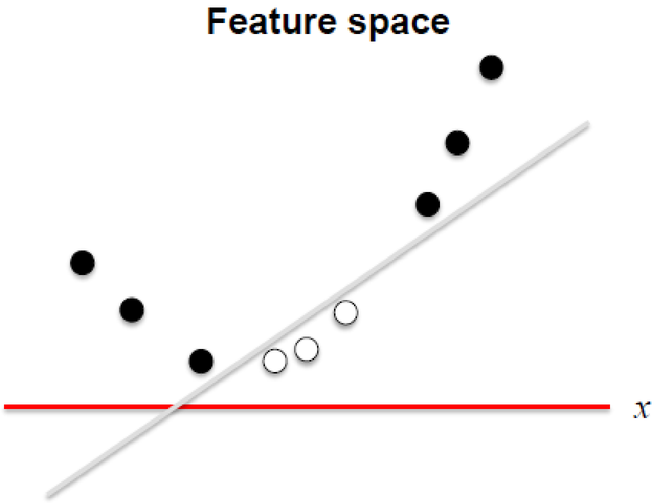
The Data Transformation Makes it Possible With a Linear Classifier



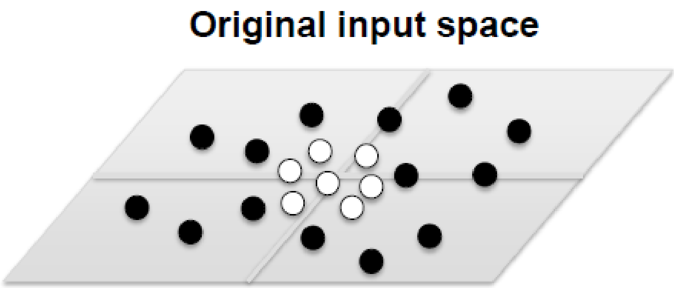
$$v_i = (x_i, x_i^2)$$

What Does the Linear Decision Boundary in Feature Space Correspond to in the Original Input Space?



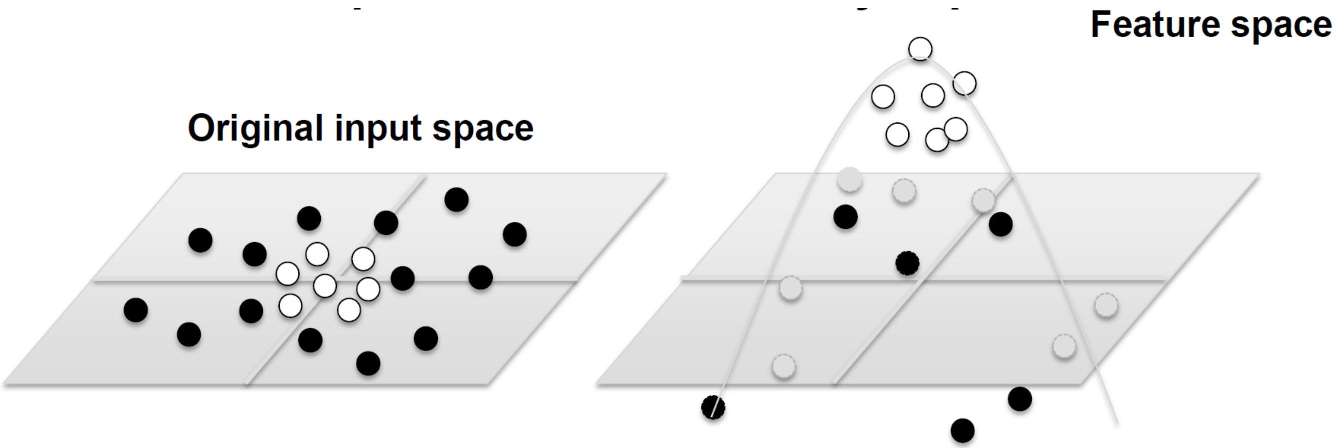


Mapping From 2D to 3D: Part I



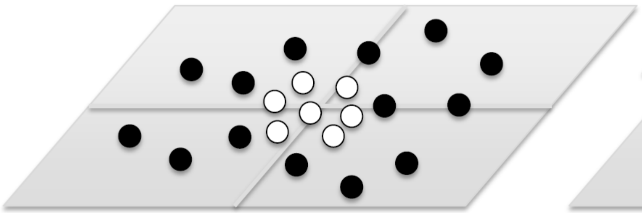
$$x_i = (x_0, x_1)$$

Mapping from 2D to 3D: Part II

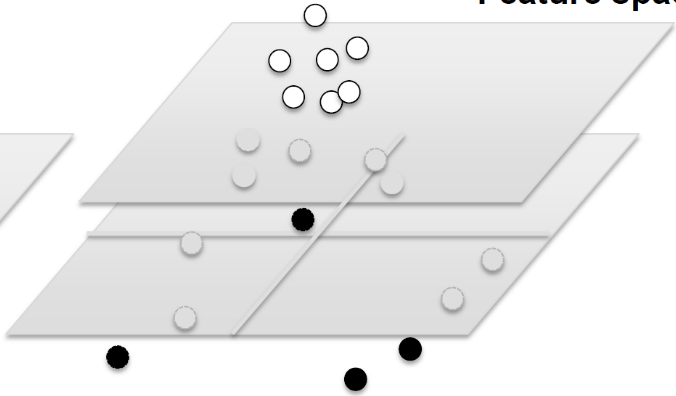


$\boldsymbol{x_i} = (x_0, x_1)$ $\qquad$ $\boldsymbol{v_i} = (x_0, x_1, 1 - (x_0^2 + x_1^2))$

Mapping from 2D to 3D: Part III

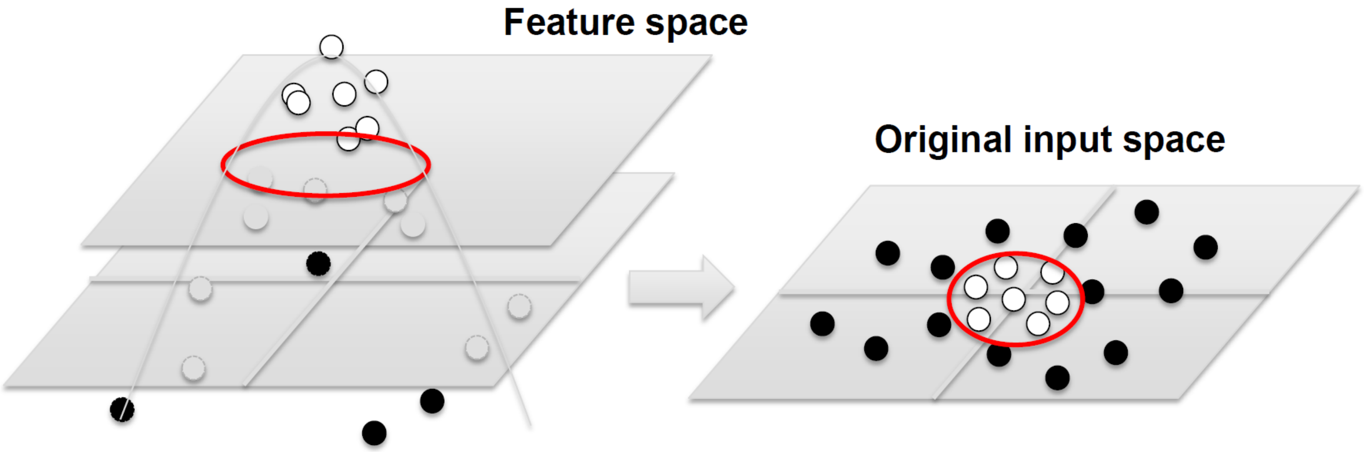


$x_i = (x_0, x_1)$



$v_i = (x_0, x_1, 1 - (x_0^2 + x_1^2))$

Mapping from 2D to 3D: Part IV



$$\boldsymbol{v_i} = (x_0, x_1, 1\ - (x_0^2 + x_1^2))$$

$$\boldsymbol{x_i} = (x_0, x_1)$$

SVM With Non-Linear Decision Boundary

- **The motto:** instead of tweaking the definition of SVM to accommodate non-linear decision boundaries, we map the data into a feature space in which the classes are linearly separable (or nearly separable):

- Apply transform $\phi: \mathbb{R}^J \to \mathbb{R}^{J'}$ on training data

$$x_n \to \phi(x_n)$$

where typically J' is much larger than J.

- Train an SVM on the transformed data

$$\{(\phi(x_1), y_1), (\phi(x_2), y_2), \ldots, (\phi(x_N), y_N)\}$$

The Kernel Trick

The *inner product* between two vectors is a measure of the similarity of the two vectors.

Definition

Given a transformation $\phi : \mathbb{R}^J \to \mathbb{R}^{J'}$, from input space $\mathbb{R}^J$ to feature space $\mathbb{R}^{J'}$, the function $K : \mathbb{R}^J \times \mathbb{R}^J \to \mathbb{R}$ defined by

$$K(x_n, x_m) = \phi(x_n)^\top \phi(x_m), \quad x_n, x_m \in \mathbb{R}^J$$

is called the ***kernel function*** of ϕ.

Generally, ***kernel function*** may refer to any function $K : \mathbb{R}^J \times \mathbb{R}^J \to \mathbb{R}$ that measure the similarity of vectors in $\mathbb{R}^J$, without explicitly defining a transform ϕ.

Dual Representation: Part I

$$w^* = \sum_n \alpha_n^* x^n$$ Linear combination of data points

α_n^* may be sparse ➡ x^n with non-zero α_n^* are support vectors

Dual Representation: Part II

$$w = \sum_n \alpha_n \, x^n = X\boldsymbol{\alpha} \qquad X = \boxed{\begin{matrix} x^1 & x^2 & \ldots\ldots & x^N \end{matrix}} \qquad \boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}$$

$$w = X\boldsymbol{\alpha}$$

Step 1: $f(x) = w^T x \Longrightarrow f(x) = \boldsymbol{\alpha}^T X^T x$

$$\begin{bmatrix} \alpha_1 & \cdots & \alpha_N \end{bmatrix} \qquad \boxed{\begin{matrix} x^1 \\ x^2 \\ \vdots \\ x^N \end{matrix}} \qquad x$$

$$\begin{bmatrix} x^1 \cdot x \\ x^2 \cdot x \\ \vdots \\ x^N \cdot x \end{bmatrix}$$

$$f(x) = \sum_n \alpha_n (x^n \cdot x)$$

$$= \sum_n \alpha_n K(x^n, x)$$

Dual Representation: Part III

Step 1: $$f(x) = \sum_n \alpha_n K(x^n, x)$$

Step 2, 3: Find $\{\alpha_1^*, \cdots, \alpha_n^*, \cdots, \alpha_N^*\}$, minimizing loss function L

$$L(f) = \sum_n l(\underline{f(x^n)}, \hat{y}^n)$$

$$= \sum_n l\left(\underline{\sum_{n'} \alpha_{n'} K\left(x^{n'}, x^n\right)}, \hat{y}^n\right)$$

We don’t really need to know vector x

We only need to know the inner product between a pair of vectors x and z

$$K(x, z)$$

Kernel Trick

Kernel Trick

Directly computing $K(x, z)$ can be faster than "feature transformation + inner product" sometimes.

Kernel trick is useful when we transform all x to $\phi(x)$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\phi(x) = \begin{bmatrix} {x_1}^2 \\ \sqrt{2}x_1x_2 \\ {x_2}^2 \end{bmatrix}$$

$$K(x, z) = \phi(x) \cdot \phi(z) = \begin{bmatrix} {x_1}^2 \\ \sqrt{2}x_1x_2 \\ {x_2}^2 \end{bmatrix} \cdot \begin{bmatrix} {z_1}^2 \\ \sqrt{2}z_1z_2 \\ {z_2}^2 \end{bmatrix}$$

$$= {x_1}^2{z_1}^2 + 2x_1x_2z_1z_2 + {x_2}^2{z_2}^2$$

$$= (x_1z_1 + x_2z_2)^2 = \left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \cdot \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}\right)^2$$

$$= (x \cdot z)^2$$