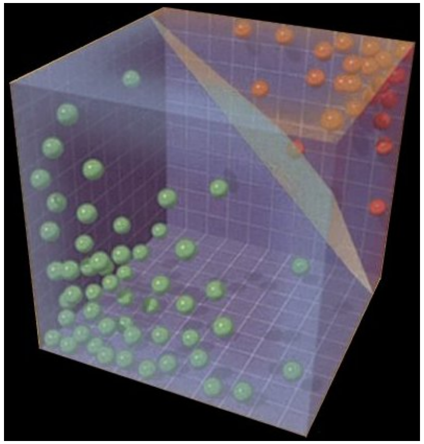
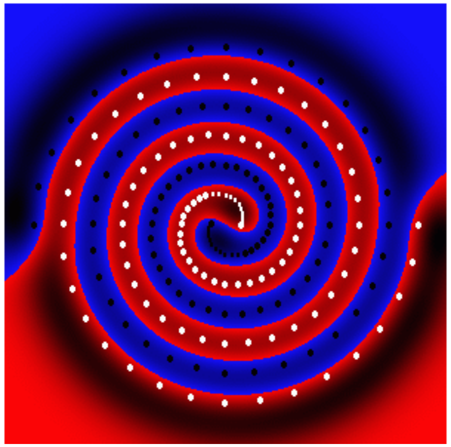
Support Vector Machines

Maximizing the Margin



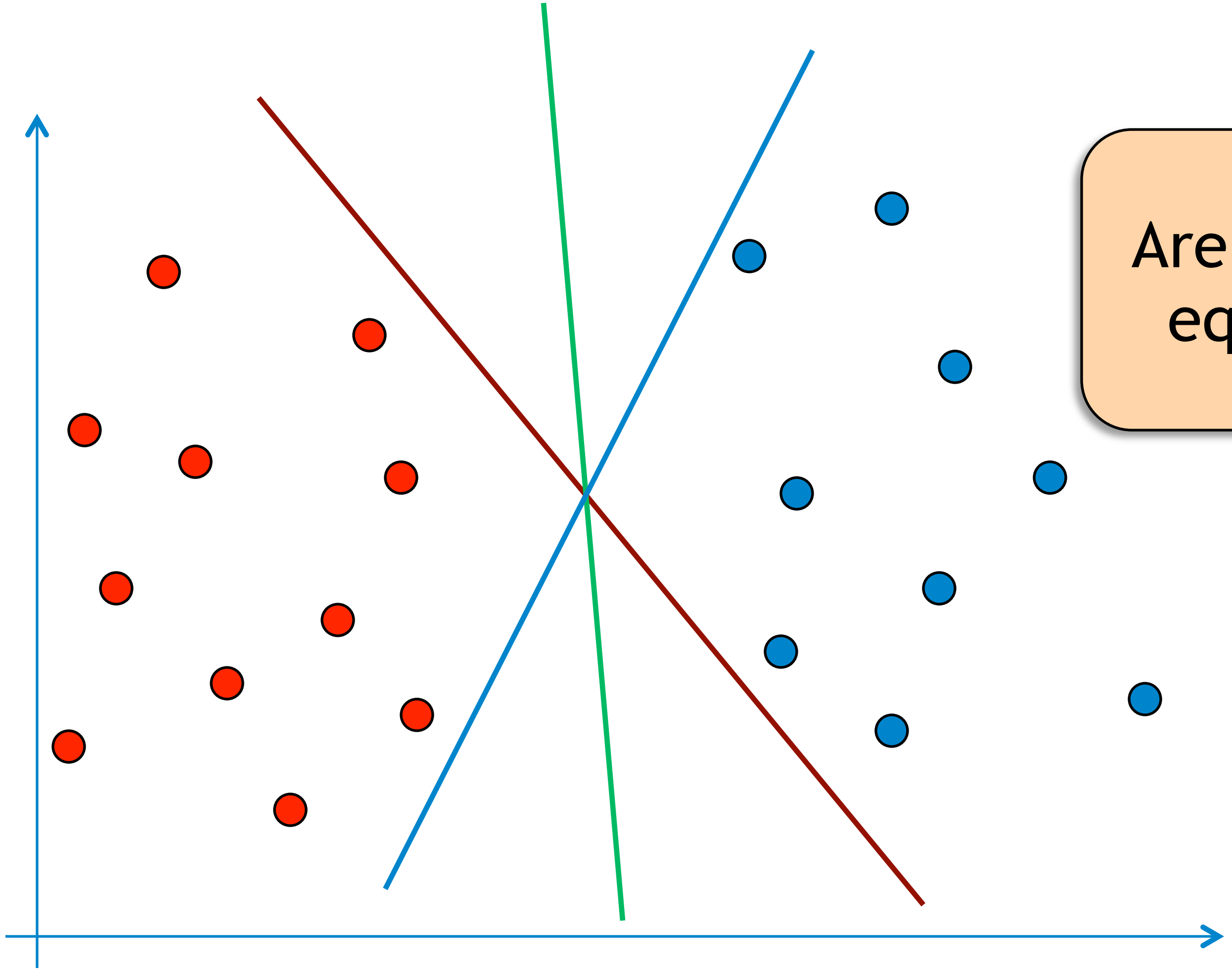


Anoop M. Namboodiri

Centre for Visual Information Technology

IIIT, Hyderabad, INDIA

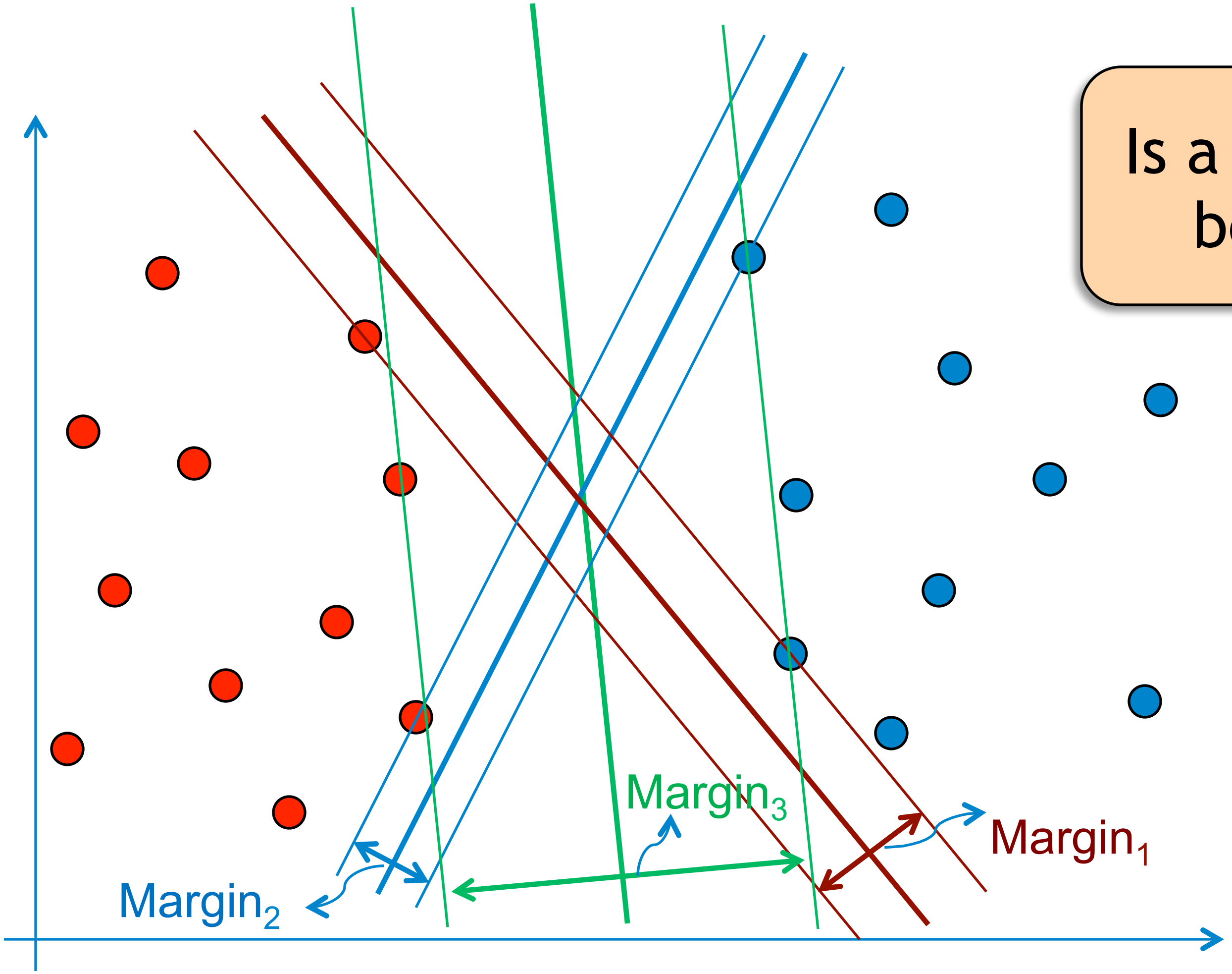
Perceptron Learning



Are all solutions equally good?

- Multiple solutions exist for linearly separable data
- Perceptron learning (any GD) results in a feasible solution

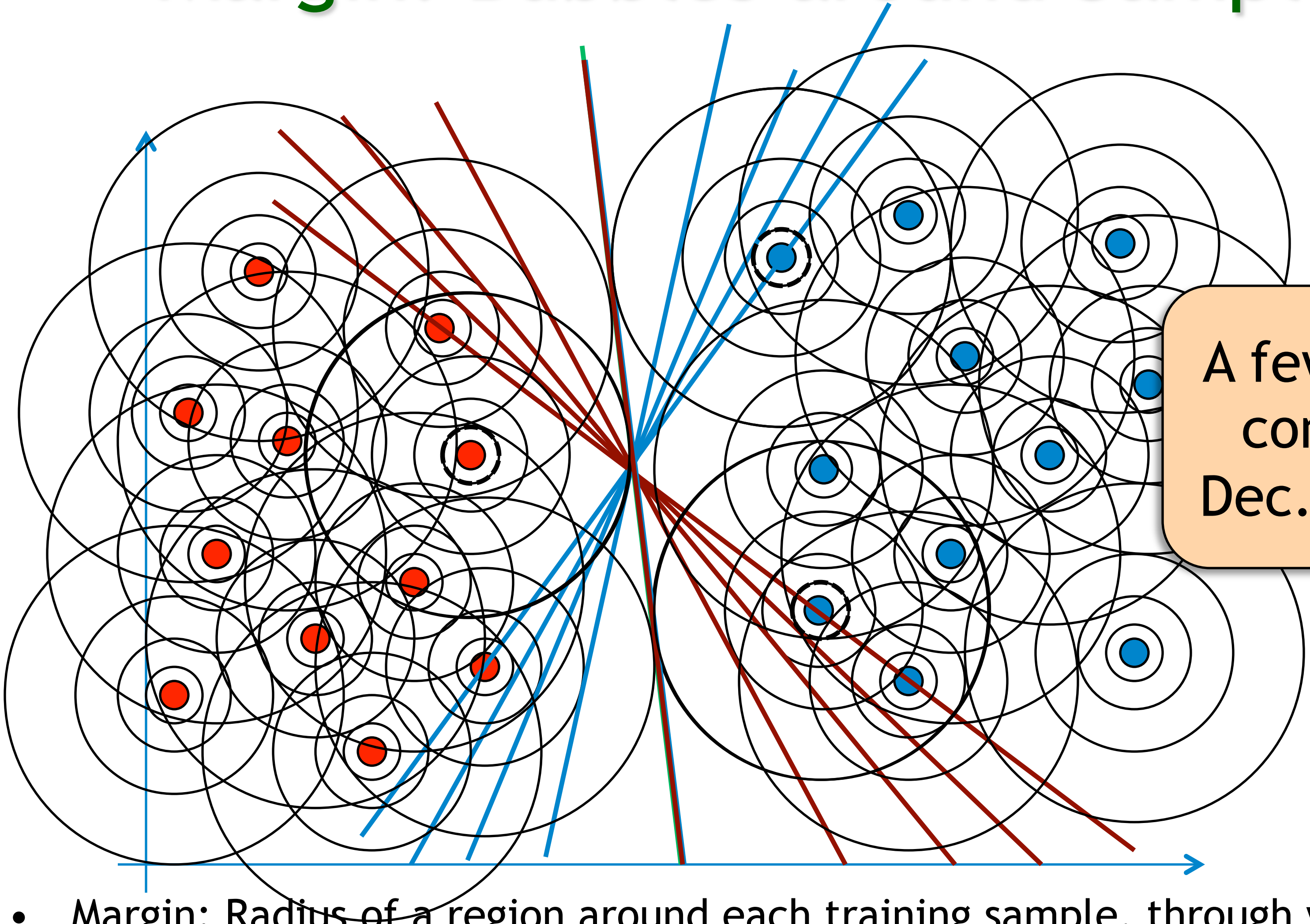
Margin: The No-mans Band



Is a Larger Margin better? Why?

- Margin: Width of a band around decision boundary without any training samples
- Margin varies with the position and orientation of the separating hyperplane

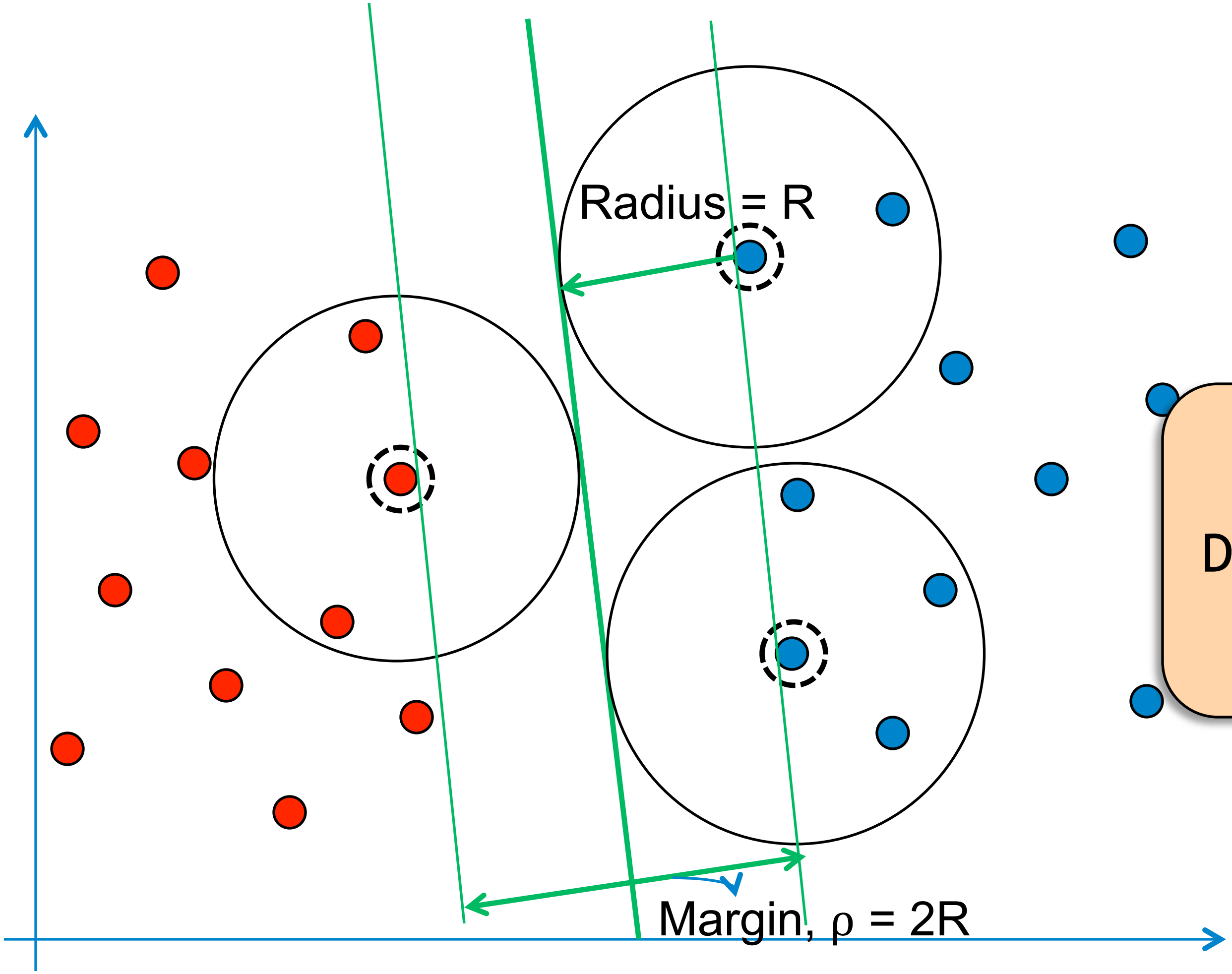
Margin: Bubbles around Samples



A few samples control the Dec. boundary

- Margin: Radius of a region around each training sample, through which the decision boundary cannot pass
- As margin increases, the feasible region reduces

Margin: Band vs. Bubbles

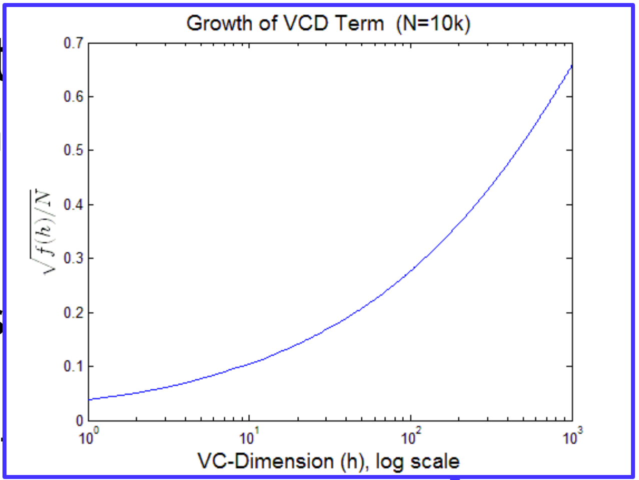


Samples that support the Decision boundary are called ***Support Vectors***

- Margin: Both interpretations yield the same decision boundary

Can we Quantify Generalization?

- Work by Vapnik and Chervonenkis in t
  1. V.N. Vapnik, A.Ya. Červonenkis, “On Uniform Frequencies of Events to their Probabilities”, Primenen, 1971, 16(2), pp. 264–279.
  2. V.N. Vapnik, “Estimation of Dependences Bas Russian]. Nauka, Moscow, 1979.
  3. V.N. Vapnik, “The Nature of Statistical Learn Verlag, New York, 1995.



- Bound on Expected loss [3]: $\boxed{\mathrm{R}(\alpha) \le R_{train}(\alpha) + \boxed{\sqrt{f(h)/N}}}$

- h is the VC dimension, and $f(h)$ is given by:

$$f(h) = h + h\log(2N) - h\log(h) - c$$

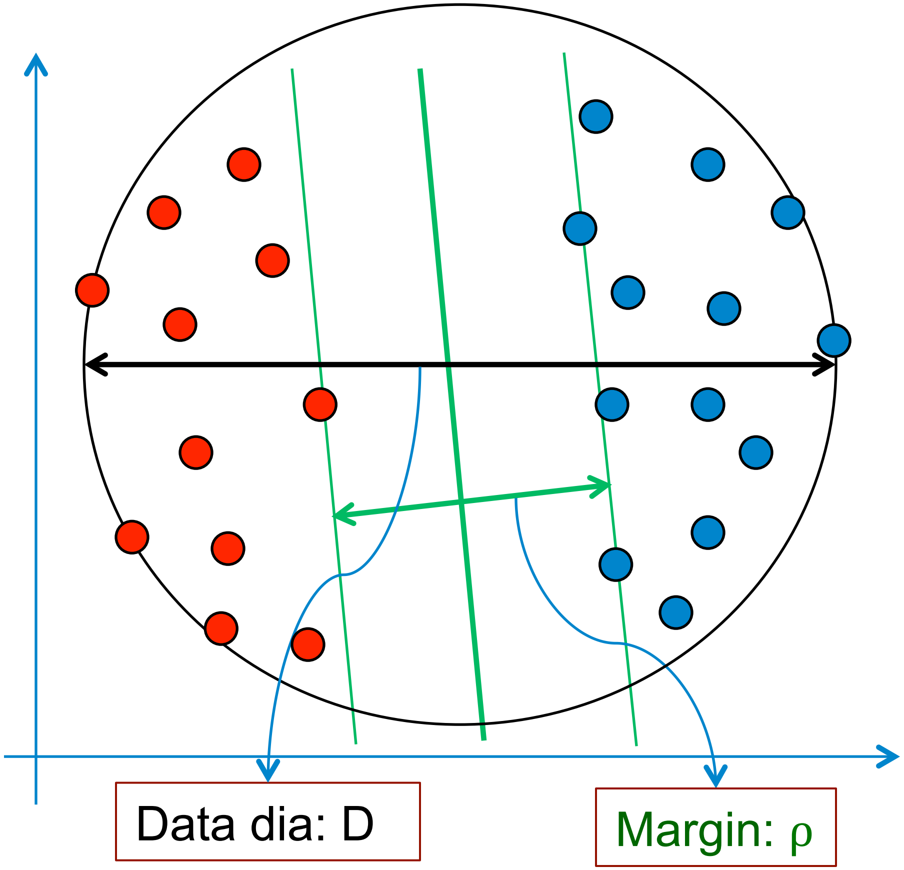
Why Maximize the Margin?

- To reduce test error, keep training error low (say 0), and minimize the VC-dimension, h.

$$\text{Relative Margin}: \ \rho / \mathrm{D}$$

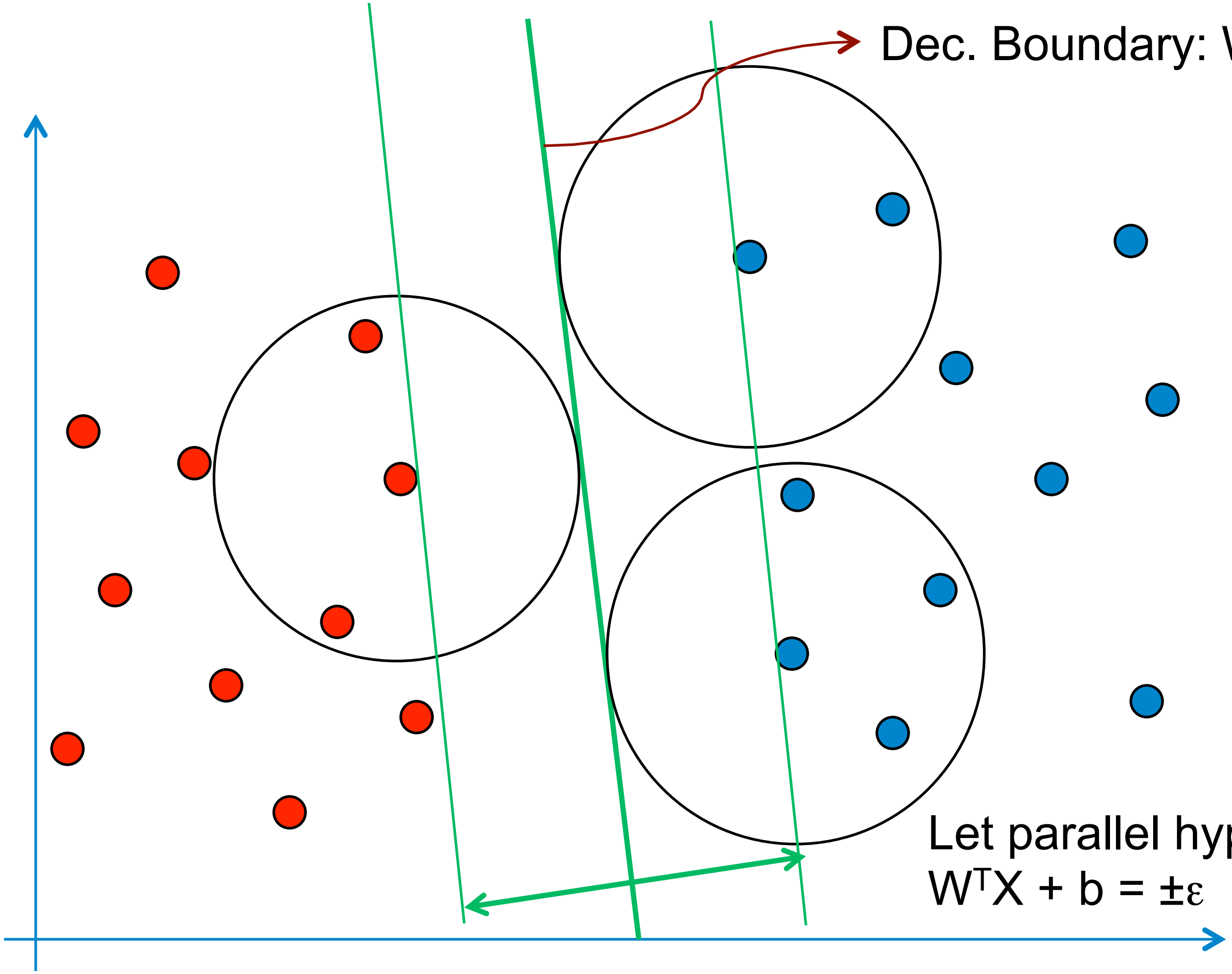
$$\mathrm{VC\text{-}D},\ h \le \min\left\{ d\ ,\ \left\lceil \frac{D^2}{\rho^2} \right\rceil \right\} + 1$$

- Maximizing margin improves generalization.
- h can be made independent of the dimensionality: d.


Data dia: D

Margin: ρ

Formalizing the Margin



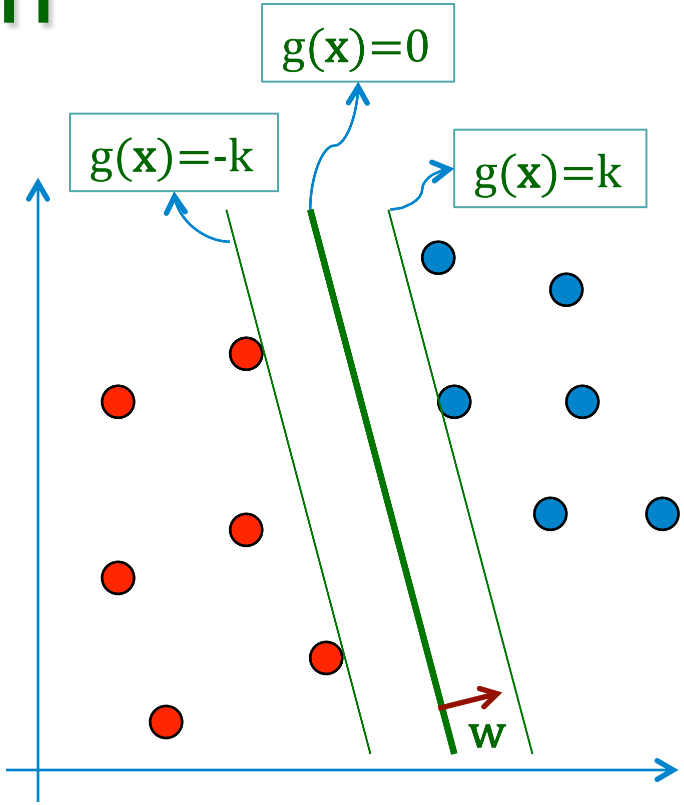
Dec. Boundary: $W^TX + b = 0$

Let parallel hyperplanes be:
$W^TX + b = \pm\varepsilon$

- Note: The value of $W^TX_i + b$ is dependent on the scale of X and W

Formulation

- Let $g(x)=\mathbf{w}^T\mathbf{x}+b$.
- We want to maximize k such that:
  - $\mathbf{w}^T\mathbf{x}_i+b \geq k$ for $d_i=1$
  - $\mathbf{w}^T\mathbf{x}_i+b \leq -k$ for $d_i=-1$
- Value of $g(\mathbf{x})$ dependents on $\|\mathbf{w}\|$:
  1. Keep $\|\mathbf{w}\|=1$, and maximize $g(\mathbf{x})$, or
  2. Let $g(\mathbf{x})\geq 1$, and minimize $\|\mathbf{w}\|$.

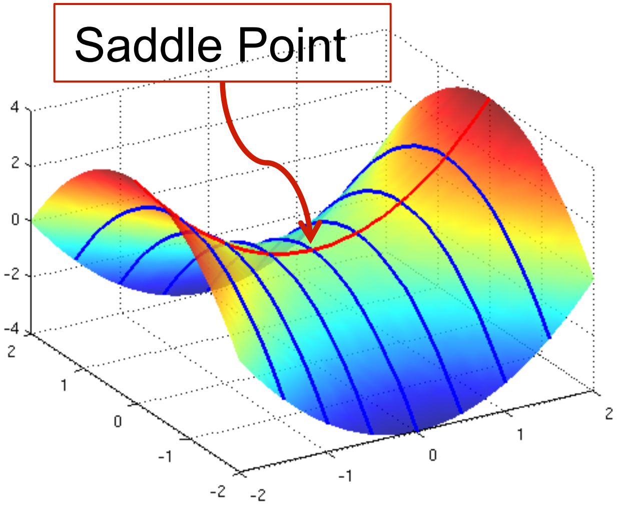


- We use approach (2) and formulate the problem as:
  - Minmize: $\frac{1}{2}\mathbf{w}^T\mathbf{w}$
  - Subject to: $d_i(\mathbf{w}^T\mathbf{x}_i+b) \geq 1$, for $i=1..N$

The Optimization Problem

$$\text{Minimize}: \ \Phi(\mathbf{w}) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{Subject to}: \ d_i(\mathbf{w}^T\mathbf{x_i} + b) - 1 \geq 0 \quad \forall i$$


Saddle Point

- Quadratic objective function with linear inequalities as constraints: QP Solver.
- Integrating the constraints into the Lagrangian form, we get:

$$\text{Minimize}: \ J(\mathbf{w}, b, \boldsymbol{\alpha}) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i d_i(\mathbf{w}^T\mathbf{x_i} + b) + \sum_{i=1}^{N} \alpha_i$$

$$\text{Subject to}: \ \alpha_i \geq 0 \quad \forall i$$

- Minimize J with respect to w and b, and maximize with respect to α.

Converting to the Dual Form

$$\text{Objective}: J(\mathbf{w}, b, \boldsymbol{\alpha}) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i d_i (\mathbf{w}^T\mathbf{x_i} + b) + \sum_{i=1}^{N} \alpha_i$$

$$\text{At the optimum,} \quad \boxed{1: \ \frac{\partial J}{\partial \mathbf{w}} = 0} \quad \text{and} \quad \boxed{2: \ \frac{\partial J}{\partial b} = 0}$$

$$\boxed{1: \ \mathbf{w}_o = \sum_{i=1}^{N} \alpha_i d_i \mathbf{x_i}} \qquad \boxed{2: \ \sum_{i=1}^{N} \alpha_i d_i = 0}$$

KKT Conditions

$$\boxed{3: \alpha_i [d_i (\mathbf{w}_o{}^T \mathbf{x_i} + b_o) - 1] = 0}$$

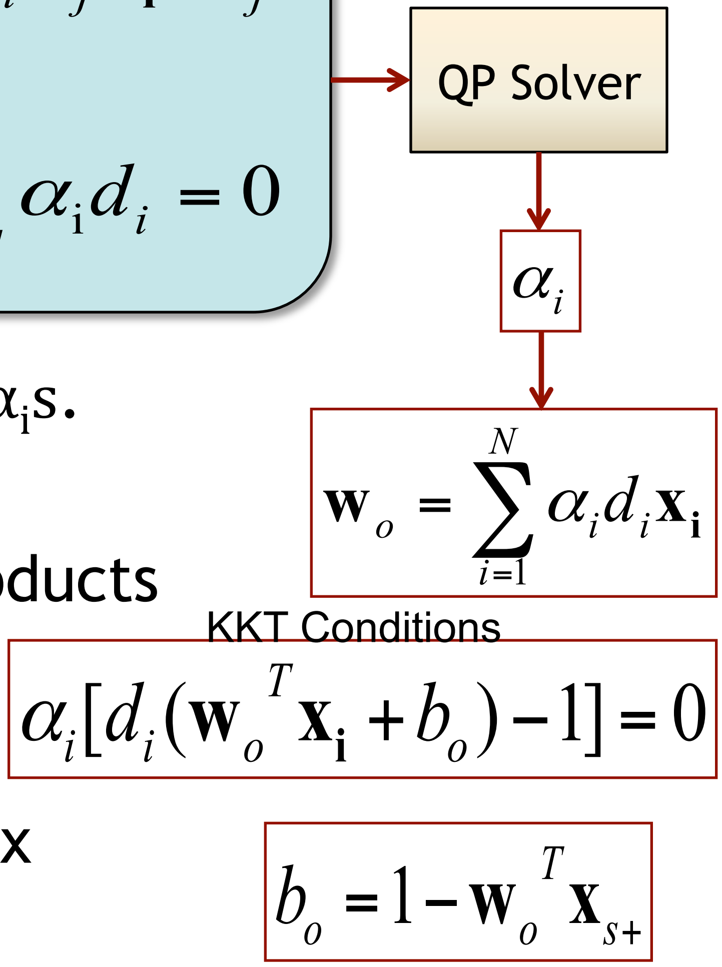
$$\text{Obj}: \ J(\mathbf{w}, b, \boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i + \tfrac{1}{2}\mathbf{w}^T\mathbf{w} - \mathbf{w}^T \sum_{i=1}^{N} \alpha_i d_i \mathbf{x_i} - b \sum_{i=1}^{N} \alpha_i d_i$$

$$\text{Using } 1,2: \ \boxed{Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x_i}^T \mathbf{x}_j}$$

Solving the Dual Form

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x_i}^T \mathbf{x}_j$$

$$\text{Subject to } \alpha_i \geq 0 \quad \forall_i \quad \text{and} \quad \sum_{i=1}^{N} \alpha_i d_i = 0$$



QP Solver

α_i

$$\mathbf{w}_o = \sum_{i=1}^{N} \alpha_i d_i \mathbf{x_i}$$

KKT Conditions

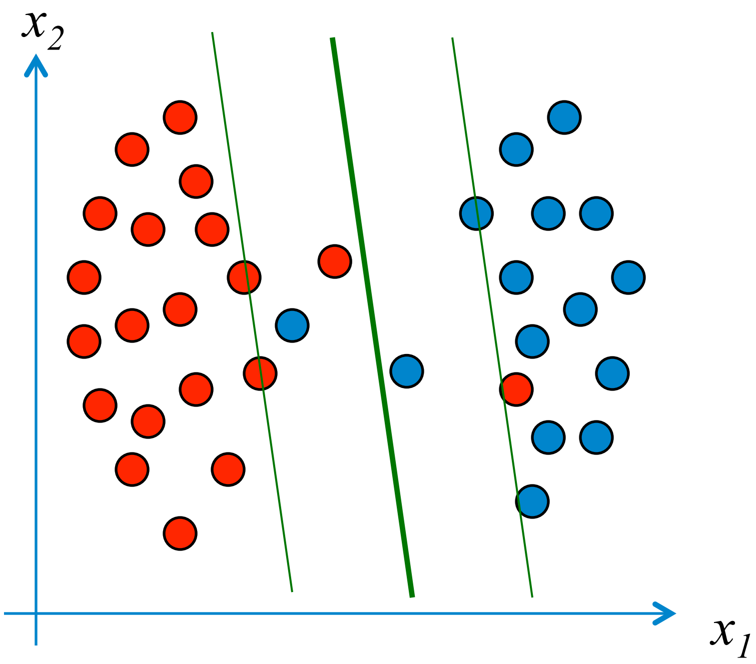
$$\alpha_i [d_i (\mathbf{w}_o{}^T \mathbf{x_i} + b_o) - 1] = 0$$

$$b_o = 1 - \mathbf{w}_o{}^T \mathbf{x}_{s+}$$

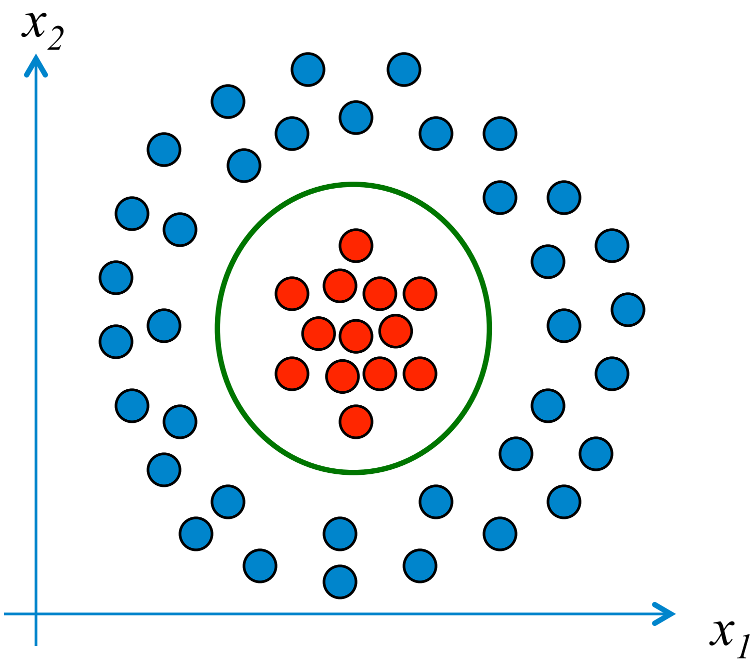
- The only unknowns (variables) are α_is.
- The constraints are also on α_is only.
- Data vectors appear only as dot products
- Objective is convex, subject to linear constraints
- Can be solved using standard convex quadratic program solvers

Non-Separable Data

1: Noisy Data/Bad Features



2: Non-linear Boundary



Noise in Data

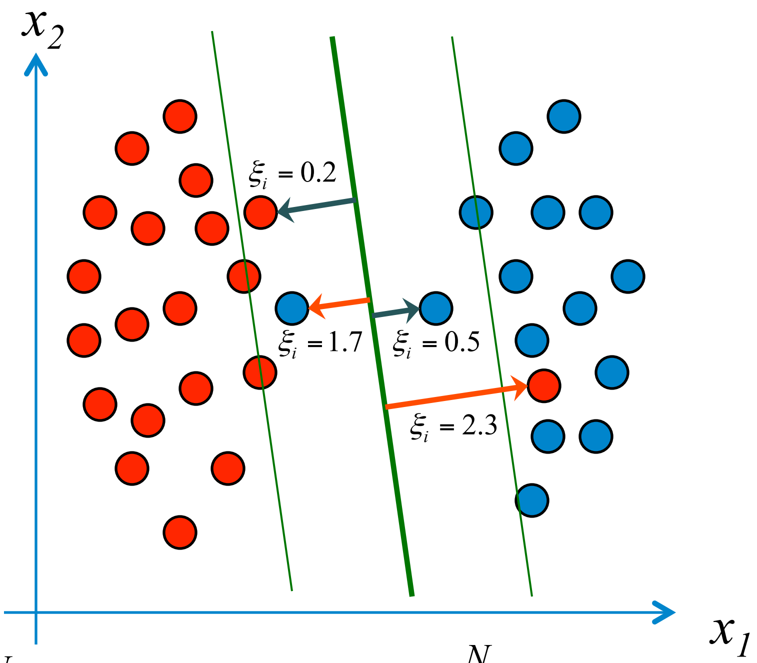
$$\text{Minimize}: \ \Phi(\mathbf{w}) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{Subject to}: \ d_i(\mathbf{w}^T\mathbf{x}_i + b) \ge 1 \quad \forall i$$

Introduce slack variables $\xi_i \ge 0$

$$\text{Minimize}: \ \Phi(\mathbf{w}) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{Subject to}: \ d_i(\mathbf{w}^T\mathbf{x}_i + b) \ge 1 - \xi_i \quad \forall i$$



Also minimize training error $\sum_{i=1}^{N} \mathrm{I}(\xi_i \ge 1)$ or $\sum_{i=1}^{N} \xi_i$

$$\text{Minimize}: \ \Phi(\mathbf{w}) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i$$

$$\text{Subject to}: \ d_i(\mathbf{w}^T\mathbf{x}_i + b) \ge 1 - \xi_i \ ; \quad \xi_i \ge 0 \, , \quad \forall i$$

Dual form with Slack

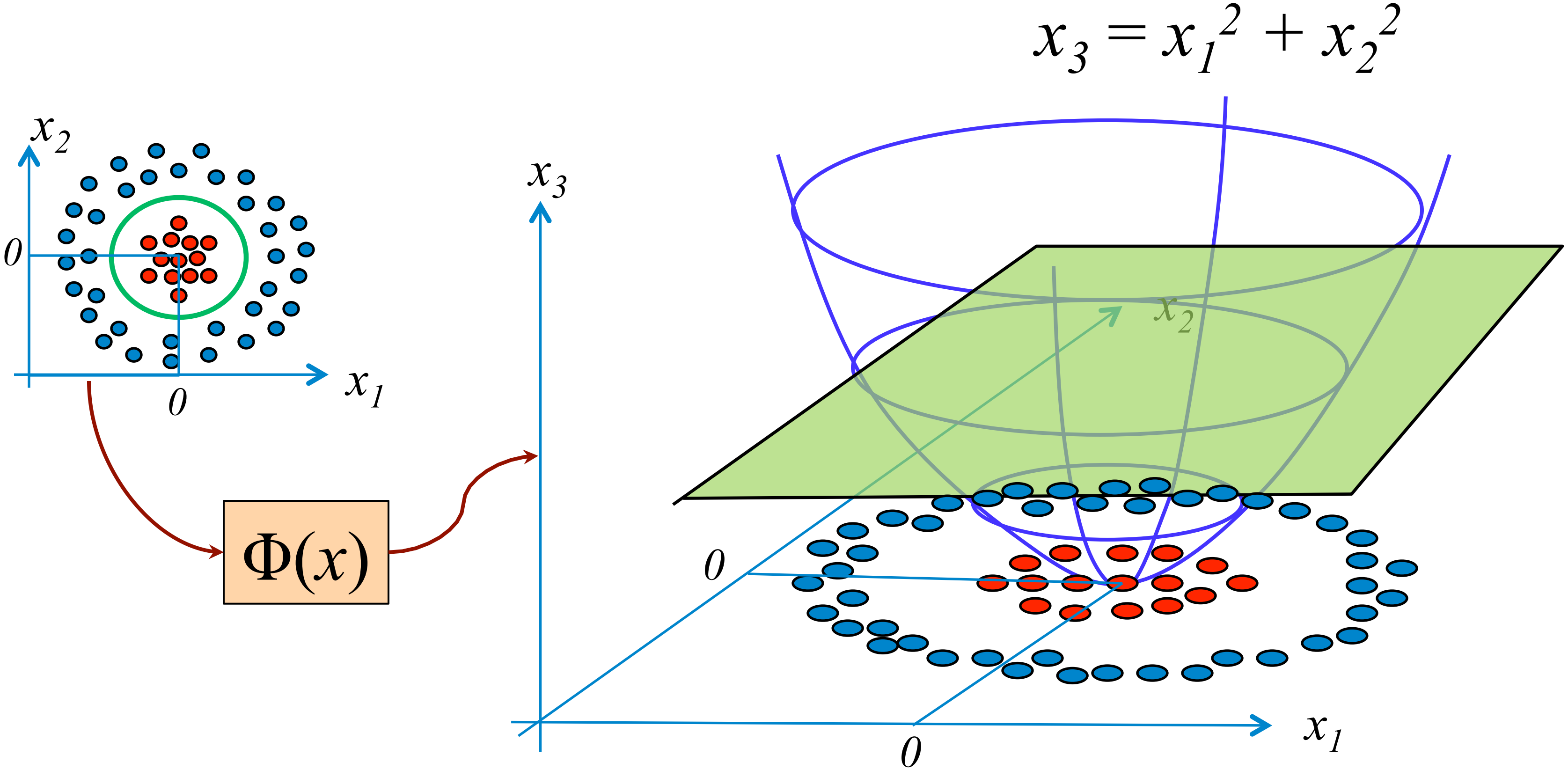
- Forming the Lagrangian and converting to dual, we get:

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x_i}^T \mathbf{x}_j$$

$$\text{Subject to } 0 \le \alpha_i \le C \quad \forall_{\mathrm{i}} \quad \text{and} \quad \sum_{\mathrm{i}=1}^{N} \alpha_{\mathrm{i}} d_i = 0$$

- Note that neither the slack variables, nor their Lagrange multipliers appear in the dual.
- The only change is the additional constraint on α_i
- The parameter C controls the relative weight between training error and the VC dimension.

Non-Linear Boundaries



Φ is a non-linear mapping into a possibly high-dimensional space

Solving the SVM with a Mapping

- Data vectors occur only as dot products in SVM-learning and testing

Testing Phase

Training Phase

-

$$\text{Label} = sign\left(\mathbf{w_o} \bullet \Phi(\mathbf{x_{test}}) + b_o\right)$$

$$\mathbf{w}_o = \sum_{i=1}^{N} \alpha_i d_i \Phi(\mathbf{x_i})$$

$\therefore$ L

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathrm{K}(\mathbf{x_i}, \mathbf{x}_j)$$

$$\text{Label} = sign\left(\sum_{i=1}^{N} \left(\alpha_i d_i \mathrm{K}(\mathbf{x_i}, \mathbf{x_{test}})\right) + b_o \right)$$

A Simple Quadratic Kernel

$$\text{Let } \Phi(\mathbf{X}) = \Phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \begin{bmatrix} x_1x_1 \\ x_1x_2 \\ x_2x_1 \\ x_2x_2 \end{bmatrix}$$

We can compute K(X,Y)=(X.Y)^2 instead of mapping with **Φ** explicitly and then computing dot product.

$$\text{Let } \mathrm{K}(\mathbf{X},\mathbf{Y}) = \Phi(\mathbf{X}) \bullet \Phi(\mathbf{Y}) = \begin{bmatrix} {x_1}^2 \\ x_1x_2 \\ x_2x_1 \\ {x_2}^2 \end{bmatrix} \bullet \begin{bmatrix} {y_1}^2 \\ y_1y_2 \\ y_2y_1 \\ {y_2}^2 \end{bmatrix}$$

$$= {x_1}^2{y_1}^2 + 2x_1x_2y_1y_2 + {x_2}^2{y_2}^2 = (x_1y_1 + x_2y_2)^2 = (\mathbf{X} \bullet \mathbf{Y})^2$$

Similarly for a Cubic Kernel

- Original Space: 2-dimensional

$$\text{Let } \mathrm{K}(\mathbf{X},\mathbf{Y}) = (\mathbf{X} \bullet \mathbf{Y})^3 = (x_1 y_1 + x_2 y_2)^3$$

$$\Phi(\mathbf{X}) = \Phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \begin{bmatrix} x_1^{\ 3} \\ x_1^{\ 2} x_2 \\ x_1 x_2^{\ 2} \\ x_2^{\ 3} \end{bmatrix}$$

- Equivalent to working in an 4-dimensional space
- 4:× and 1:+, instead of 16:× and 3:+

Similarly for a Cubic Kernel

- Original Space: 3-dimensional

Let $\mathrm{K}(\mathbf{X},\mathbf{Y}) = (\mathbf{X}\bullet\mathbf{Y})^3 = (x_1y_1 + x_2y_2 + x_3y_3)^3$

$$\Phi(\mathbf{X}) = \Phi\left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}\right) = \begin{bmatrix} x_1^{\ 3} \\ x_2^{\ 3} \\ x_3^{\ 3} \\ x_1^{\ 2}x_2 \\ x_1x_2^{\ 2} \\ x_1^{\ 2}x_3 \\ x_1x_3^{\ 2} \\ x_2^{\ 2}x_3 \\ x_2x_3^{\ 2} \\ x_1x_2x_3 \end{bmatrix}$$

- Equivalent to working in an 10-dimensional space
- 5:× and 2:+, instead of 38:× and 9:+

A Generic Polynomial Kernel

- Adding two Kernels gives you a new Kernel:

$$\mathrm{K}(\mathbf{X},\mathbf{Y}) = \mathrm{K}_1(\mathbf{X},\mathbf{Y}) + \mathrm{K}_2(\mathbf{X},\mathbf{Y}) \quad : \quad \Phi(\mathbf{X}) = \begin{bmatrix} \Phi_1(\mathbf{X}) \\ \Phi_2(\mathbf{X}) \end{bmatrix}$$

$$\mathrm{K}(\mathbf{X},\mathbf{Y}) = \Phi(\mathbf{X}) \bullet \Phi(\mathbf{Y}) = \Phi_1(\mathbf{X}) \bullet \Phi_1(\mathbf{Y}) + \Phi_2(\mathbf{X}) \bullet \Phi_2(\mathbf{Y})$$

$$\boxed{\mathrm{K}_p(\mathbf{X},\mathbf{Y}) = (1 + \mathbf{X} \bullet \mathbf{Y})^p = 1 + \mathbf{X} \bullet \mathbf{Y} + (\mathbf{X} \bullet \mathbf{Y})^2 + \cdots + (\mathbf{X} \bullet \mathbf{Y})^p}$$

- Just adding a 1 and raising to the power of p maps the input vector into a space containing all original dimensions, all 2-products, 3-products,..,p-products.

Mercer's Theorem

- Using Kernels, we avoid explicit mapping with Φ
- In fact, we do not even have to know what Φ is as long as we are sure there exists a valid Φ.
- Mercer's Theorem:

Any given kernel can be expanded as a series :

$$\mathrm{K}(\mathbf{X},\mathbf{Y}) = \sum_{i=1}^{\infty} \lambda_i \Phi_\mathrm{i}(\mathbf{X}) \bullet \Phi_\mathrm{i}(\mathbf{Y}), \quad \lambda_i > 0; \mathit{iff}$$

$\mathrm{K}(\mathbf{X},\mathbf{Y})$ satisfies the Mercer's conditions

(symmetric, continuous, positive semi - definite)

Popular Kernels

- Polynomial:

$$\mathrm{K}_p\left(\mathbf{X},\mathbf{Y}\right)=\left(1+\mathbf{X}\bullet\mathbf{Y}\right)^p$$

- Radial Basis Function (RBF) or Gaussian:

$$\mathrm{K}_r\left(\mathbf{X},\mathbf{Y}\right)=e^{-\frac{1}{2\sigma^2}\left\|\mathbf{X}-\mathbf{Y}\right\|_2^2}$$

- Hyperbolic Tangent:

$$\mathrm{K}_s\left(\mathbf{X},\mathbf{Y}\right)=\tanh(\beta_0\mathbf{X}\bullet\mathbf{Y}+\beta_1)$$