

Introduction to Machine Learning – Discriminative Approach and Deep Learning

The 8th KIAS CAC Summer School
2017. 7. 28 (Wed.)

Yung-Kyun Noh
Seoul National University



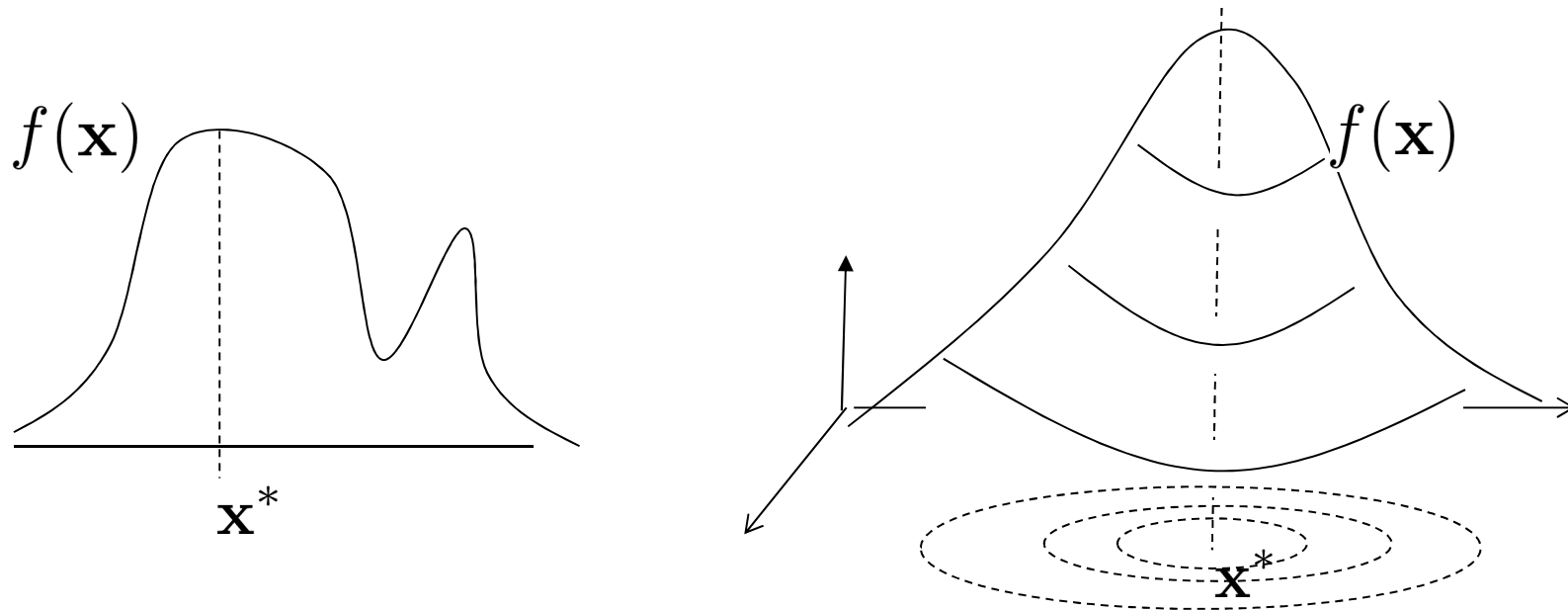
Overview

- Optimization
- Numerical methods, gradient ascent/descent
- Analytical methods
- Constrained optimization
- Convex optimization
- Backpropagation

Motivation - Optimization

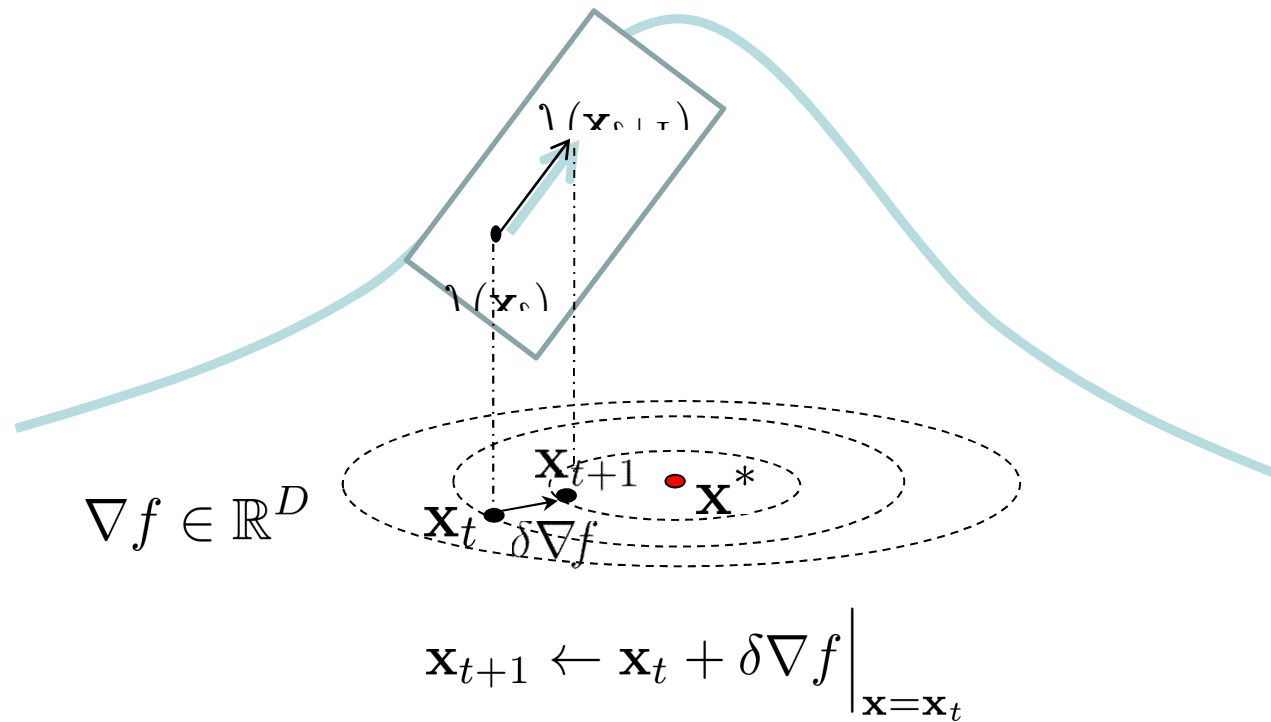
- Optimization

$$\mathbf{x}^* = \arg \max_{\mathbf{x}} f(\mathbf{x}) \quad \mathbf{x} \in \mathbb{R}^D$$



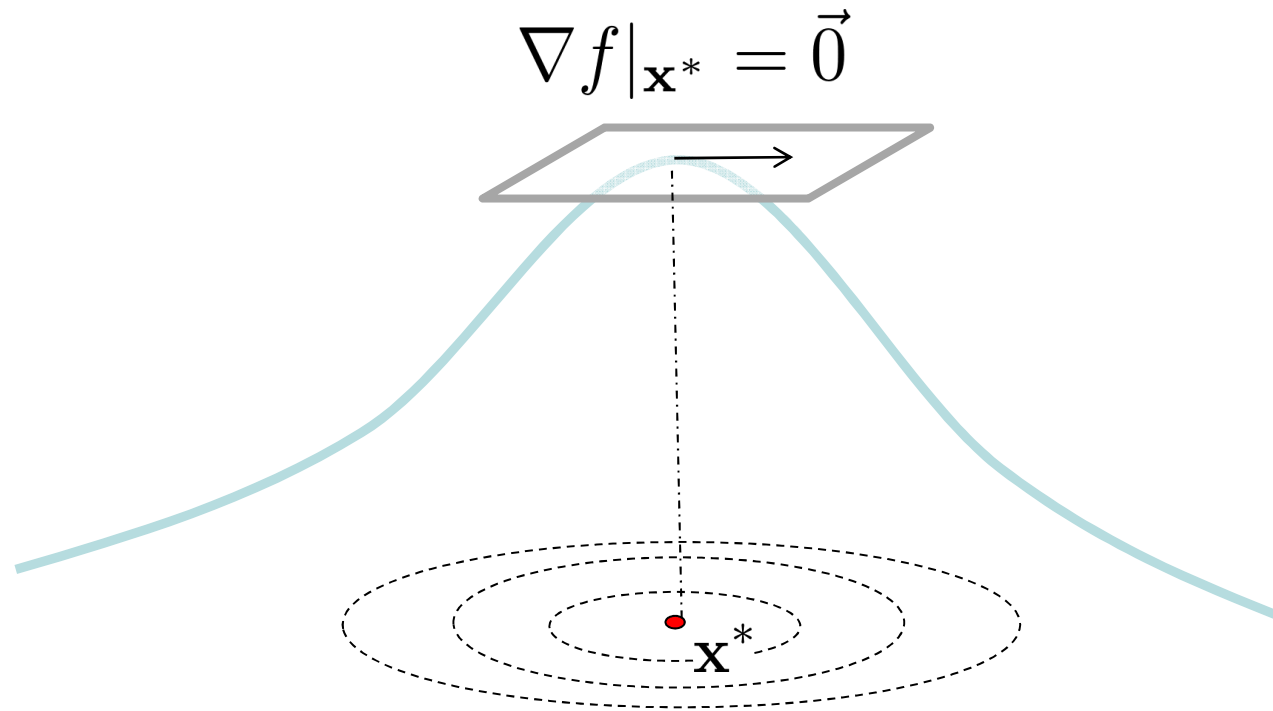
Optimization – Using Gradient

$$\begin{aligned} f(\mathbf{x}) : \mathbb{R}^D &\longrightarrow \mathbb{R} \\ \mathbf{x} &\longmapsto f(\mathbf{x}) \end{aligned}$$



Optimization – Using Gradient

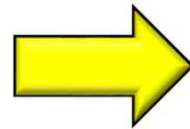
- Analytic solution



Differentiation w.r.t. Vector / Matrix

$$f : \mathbb{R}^D \longrightarrow \mathbb{R}$$
$$\mathbf{x} \longmapsto y$$

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{pmatrix} \in \mathbb{R}^D$$



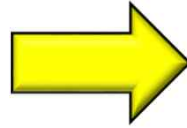
$$\frac{df}{d\mathbf{x}} = \begin{pmatrix} \partial f / \partial x_1 \\ \partial f / \partial x_2 \\ \vdots \\ \partial f / \partial x_D \end{pmatrix}$$

$$\left[\frac{df}{d\mathbf{x}} \right]_i = \frac{\partial f}{\partial x_i}$$

Differentiation w.r.t. Vector / Matrix

$$\begin{aligned} f : \mathbb{R}^{D \times D} &\longrightarrow \mathbb{R} \\ A &\longmapsto y \end{aligned}$$

$$A = \begin{pmatrix} A_{1,1} & \cdots & A_{1,D} \\ A_{2,1} & \cdots & A_{2,D} \\ \vdots & & \\ A_{D,1} & \cdots & A_{D,D} \end{pmatrix} \in \mathbb{R}^{D \times D}$$



$$\frac{df}{dA} =$$

$$\begin{pmatrix} \frac{\partial f}{\partial A_{1,1}} & \cdots & \frac{\partial f}{\partial A_{1,D}} \\ \vdots & & \\ \frac{\partial f}{\partial A_{D,1}} & \cdots & \frac{\partial f}{\partial A_{D,D}} \end{pmatrix}$$

$$\left[\frac{df}{dA} \right]_{i,j} = \frac{\partial f}{\partial A_{i,j}}$$

Differentiation w.r.t. Vector / Matrix

- Ex)

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{w}^\top \mathbf{x} \\ &= \sum w_i x_i \end{aligned}$$

$$\left(\frac{df}{d\mathbf{x}} \right)_i = w_i \quad \frac{df}{d\mathbf{x}} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_D \end{pmatrix} = \mathbf{w} \in \mathbb{R}^D$$

Differentiation w.r.t. Vector / Matrix

• Ex)

$$f(\mathbf{w}) = \sum_{i=1}^N (\mathbf{w}^\top \mathbf{x}_i - b)^2$$

$\mathbf{x}_i, \mathbf{w} \in \mathbb{R}^D$

$$= \sum_{i=1}^N \left(\sum_{j=1}^D w_j x_{ij} - b \right)^2$$

$$\frac{\partial f}{\partial w_k} = \sum_{i=1}^N 2 \left(\sum_{j=1}^D w_j x_{ij} - b \right) \cdot x_{ik}$$

$$\frac{\partial f}{\partial \mathbf{w}} = 2 \sum_{i=1}^N (\mathbf{w}^\top \mathbf{x}_i - b) \cdot \mathbf{x}_i \in \mathbb{R}^D$$

Differentiation w.r.t. Vector / Matrix

- Ex)

$$f(B) = \text{tr}[ABC] = \sum_{ijk} A_{ij} B_{jk} C_{ki}$$

$$\begin{aligned}
 ABC &= \begin{matrix} i \\ \left(\begin{array}{cccc} A_{1,1} & A_{1,2} & \cdots & A_{1,D} \\ \textcolor{red}{-} A_{2,1} \textcolor{red}{-} & \textcolor{red}{-} & \textcolor{red}{-} & \textcolor{red}{-} A_{2,D} \\ \cdots & & & \\ A_{D,1} & \cdots & & A_{D,D} \end{array} \right) \end{matrix} \begin{matrix} \begin{matrix} \textcolor{blue}{-} B_{1,1} \textcolor{blue}{-} & \textcolor{blue}{-} B_{1,2} \textcolor{blue}{-} & \cdots & \textcolor{blue}{-} B_{1,D} \textcolor{blue}{-} \\ \textcolor{blue}{-} B_{2,1} \textcolor{blue}{-} & \textcolor{blue}{-} & \textcolor{blue}{-} & \textcolor{blue}{-} B_{2,D} \textcolor{blue}{-} \\ \vdots & & & \\ \textcolor{blue}{-} B_{D,1} \textcolor{blue}{-} & \textcolor{blue}{-} & \textcolor{blue}{-} & \textcolor{blue}{-} B_{D,D} \textcolor{blue}{-} \end{matrix} \end{matrix} \begin{matrix} j \\ \left(\begin{array}{cccc} C_{1,1} & C_{1,2} & \cdots & C_{1,D} \\ C_{2,1} & \cdots & & C_{2,D} \\ \cdots & & & \\ C_{D,1} & \cdots & & C_{D,D} \end{array} \right) \end{matrix} \\
 &= \begin{pmatrix} \sum_{lm} A_{1l} B_{lm} C_{m1} & \sum_{lm} A_{1l} B_{lm} C_{m2} & \cdots & \sum_{lm} A_{1l} B_{lm} C_{mD} \\ \sum_{lm} A_{2l} B_{lm} C_{m1} & \cdots & & \sum_{lm} A_{2l} B_{lm} C_{mD} \\ \cdots & & & \cdots \\ \sum_{lm} A_{Dl} B_{lm} C_{m1} & \cdots & & \sum_{lm} A_{Dl} B_{lm} C_{mD} \end{pmatrix}
 \end{aligned}$$

$$[ABC]_{ij} = \sum_{lm} A_{il} B_{lm} C_{mj}, \quad \text{tr}[ABC] = \sum_i [ABC]_{ii} = \sum_{ijk} A_{ij} B_{jk} C_{ki}$$

Differentiation w.r.t. Vector / Matrix

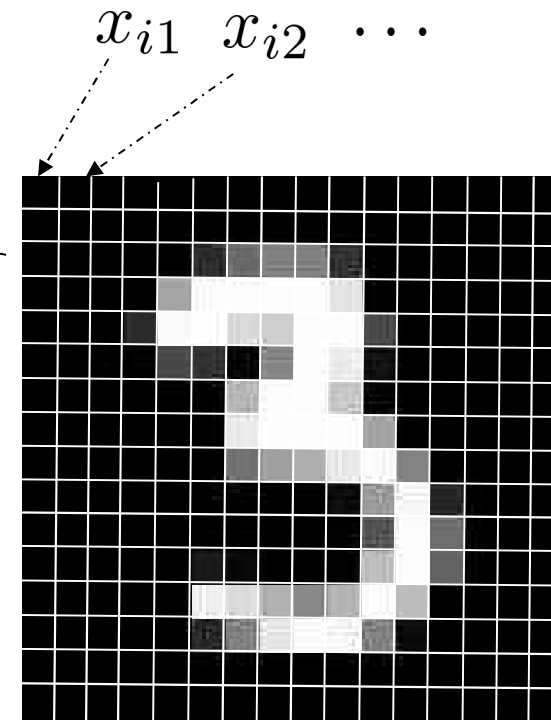
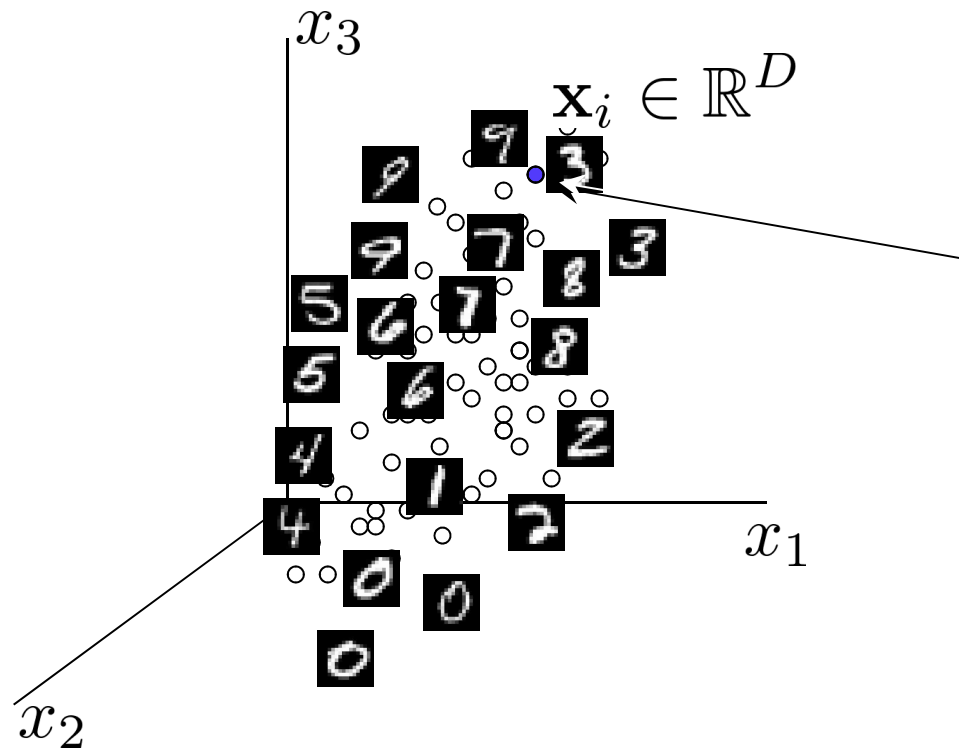
- Ex)

$$f(B) = \text{tr}[ABC] = \sum_{ijk} A_{ij} B_{jk} C_{ki}$$

$$\left[\frac{df(B)}{dB} \right]_{lm} = \sum_i A_{il} C_{mi} = \sum_i [A^\top]_{li} [C^\top]_{im}$$

$$\frac{df}{dB} = A^\top C^\top \in \mathbb{R}^{D \times D}$$

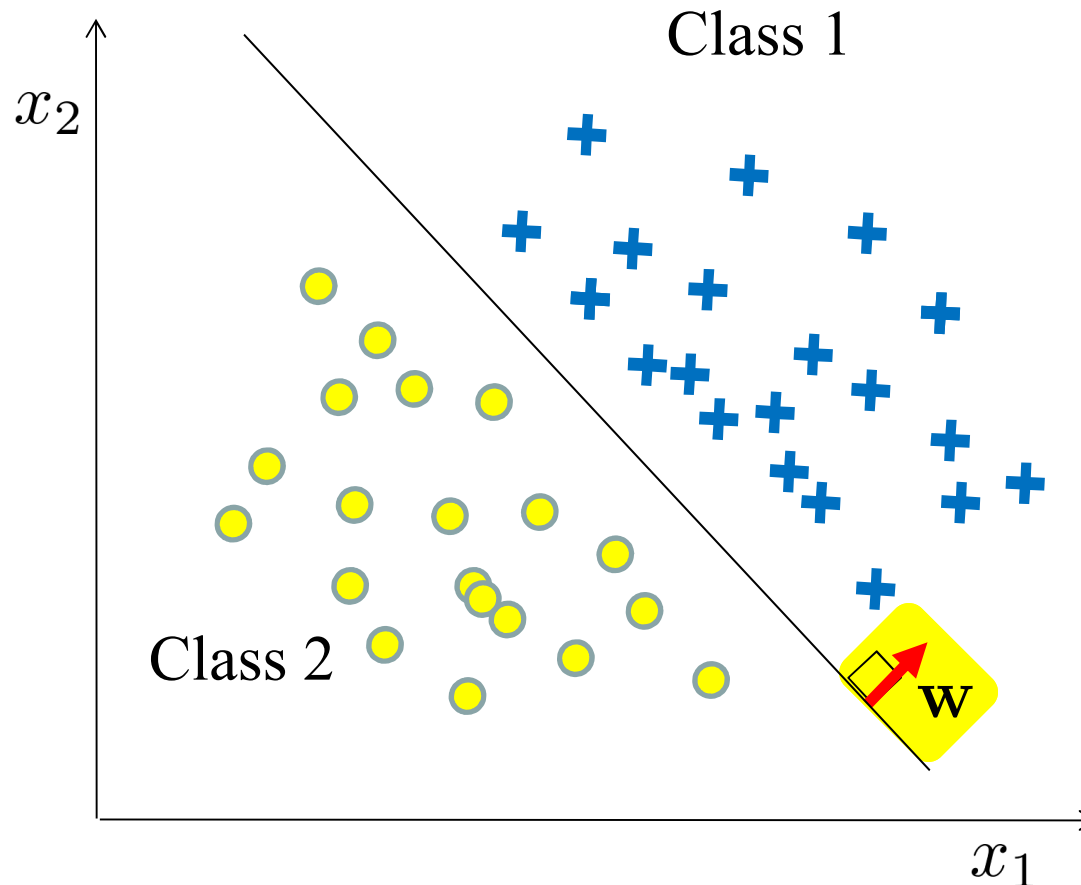
Data Space



$$\begin{aligned}\mathbf{x}_i &= [x_{i1}, \dots, x_{iD}] \\ &= [1, 2, 5, 12, 10, \dots]\end{aligned}$$

- Each datum is one point in a data space

Linear Classifier



Given

$$\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$$

Learn a prediction
function from data

$$y = f(\mathbf{x})$$

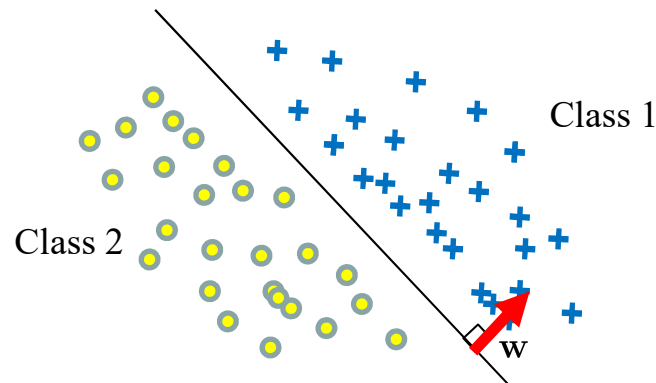
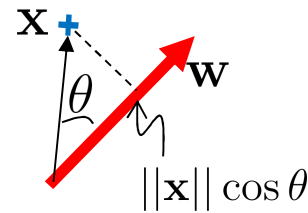
$$= f(\mathbf{x}; \mathbf{w}) \in \mathcal{H}$$

Linear Classifier

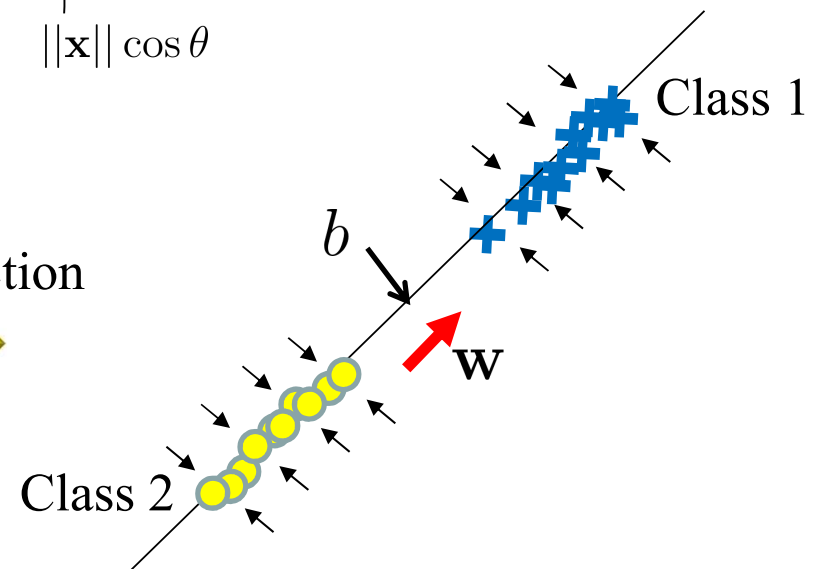
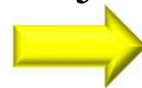
Introduce a vector $\mathbf{w}$

$$\begin{aligned}\mathbf{w}^\top \mathbf{x} &\geq b \rightarrow f(\mathbf{x}) = \text{Class 1} \\ &< b \rightarrow f(\mathbf{x}) = \text{Class 2}\end{aligned}$$

$$\mathbf{w}^\top \mathbf{x} = \|\mathbf{w}\| \|\mathbf{x}\| \cos \theta$$

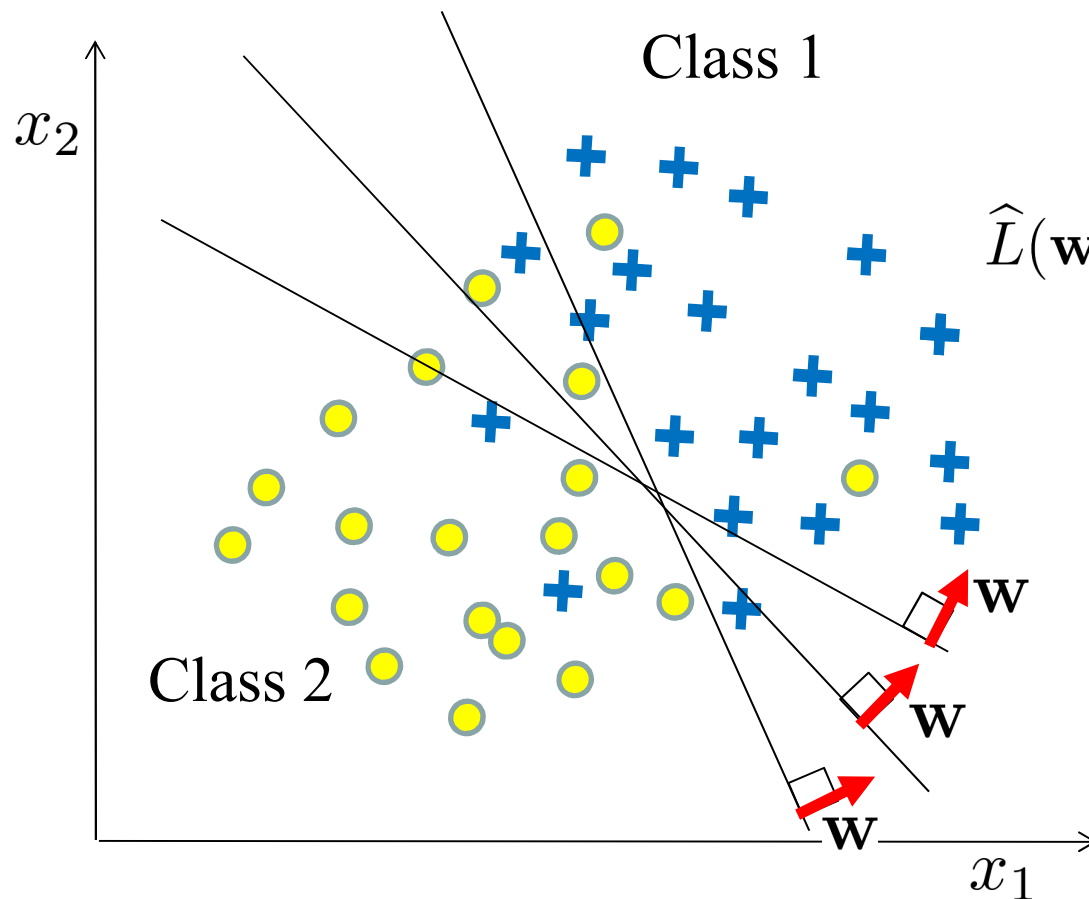


Projection



Linear Classifier

- Find $\mathbf{w} \in \mathbb{R}^D$ which classifies training data correctly as many as possible.



$$\hat{L}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^N \|f(\mathbf{x}_i; \mathbf{w}) - y_i\|^2$$

$$f(\mathbf{x}_i; \mathbf{w}) \in \mathcal{H}$$

$$\mathbf{x}_i \in \mathbb{R}^D \quad y_i \in \{1, 2\}$$

$$\mathbf{w}^* = \arg \min_{\mathbf{w} \in \mathbb{R}^D} \hat{L}(\mathbf{w})$$

Linear Classifier

- A naïve one

$$f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}$$

$$\frac{\partial \hat{L}}{\partial \mathbf{w}} = \frac{1}{2} \frac{\partial}{\partial \mathbf{w}} \sum_{i=1}^N \|f(\mathbf{x}_i; \mathbf{w}) - y_i\|^2$$

$$= \sum_{i=1}^N (f(\mathbf{x}_i; \mathbf{w}) - y_i) \frac{\partial f(\mathbf{x}_i; \mathbf{w})}{\partial \mathbf{w}}$$

$$= \sum_{i=1}^N (\mathbf{w}^\top \mathbf{x}_i - y_i) \mathbf{x}_i = 0$$

$$f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}$$


has a closed-form solution: $\mathbf{w} = ?$

Linear Classifier

$$\sum_{i=1}^N (\mathbf{w}^\top \mathbf{x}_i - y_i) \mathbf{x}_i = 0$$

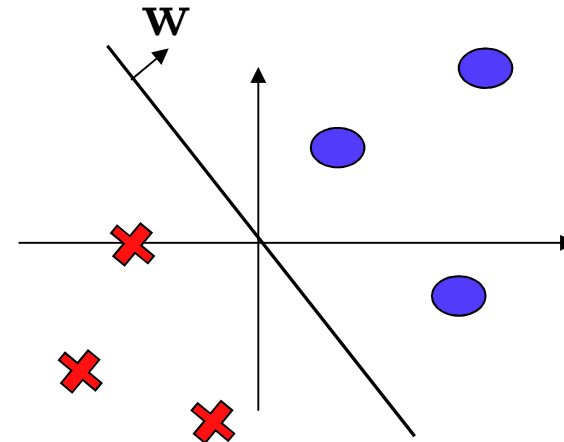
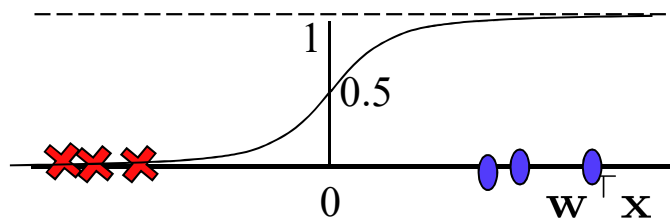
$$X = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots & \mathbf{x}_N \\ | & & | \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$$

$$\sum_{i=1}^N \mathbf{w}^\top \mathbf{x}_i \mathbf{x}_i - \sum_{i=1}^N y_i \mathbf{x}_i = 0 \rightarrow \sum_{i=1}^N \mathbf{x}_i \mathbf{x}_i^\top \mathbf{w} - \sum_{i=1}^N y_i \mathbf{x}_i = 0$$


$$X X^\top \mathbf{w} - X \mathbf{y} = 0 \rightarrow \mathbf{w} = (X X^\top)^{-1} X \mathbf{y}$$

Logistic Regression

$$f(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^\top \mathbf{x})}$$



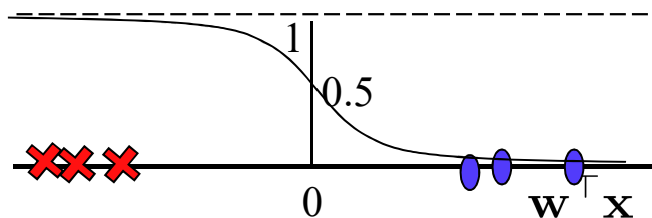
$$\hat{L}(\mathbf{w}) = \prod_{i=1}^N f(\mathbf{x}_i)^{y_i} (1 - f(\mathbf{x}_i))^{1-y_i}$$

Probability for $y_i = 1$
Probability for $y_i = 0$

$$\mathbf{w}^* = \arg \max_{\mathbf{w}} \hat{L}(\mathbf{w}) = \arg \max_{\mathbf{w}} \ln \hat{L}(\mathbf{w})$$

Logistic Regression

$$1 - f(\mathbf{x}) = 1 - \frac{1}{1 + \exp(-\mathbf{w}^\top \mathbf{x})} = \frac{\exp(-\mathbf{w}^\top \mathbf{x})}{1 + \exp(-\mathbf{w}^\top \mathbf{x})}$$



$$\hat{L}(\mathbf{w}) = \prod_{i=1}^N f(\mathbf{x}_i)^{y_i} (1 - f(\mathbf{x}_i))^{1-y_i}$$

Probability for $y_i = 1$ Probability for $y_i = 0$

$$\mathbf{w}^* = \arg \max_{\mathbf{w}} \hat{L}(\mathbf{w}) = \arg \max_{\mathbf{w}} \ln \hat{L}(\mathbf{w})$$

Logistic Regression

$$\begin{aligned} \ln (f(\mathbf{x}_i)^{y_i} (1 - f(\mathbf{x}_i)^{1-y_i}) \\ = y_i \ln f(\mathbf{x}_i) + (1 - y_i) \ln(1 - f(\mathbf{x}_i)) \quad \dots (*) \end{aligned}$$

Consider

$$f(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^\top \mathbf{x})} \longrightarrow \frac{df}{d\mathbf{w}} = \frac{\exp(-\mathbf{w}^\top \mathbf{x}) \mathbf{x}}{(1 + \exp(-\mathbf{w}^\top \mathbf{x}))^2} = f(1 - f)\mathbf{x}$$

$$\begin{aligned} \frac{d(*)}{d\mathbf{w}} &= \frac{y_i}{f} f(1 - f)\mathbf{x}_i - \frac{1 - y_i}{1 - f} f(1 - f)\mathbf{x}_i \\ &= y_i(1 - f)\mathbf{x}_i - (1 - y_i)f\mathbf{x}_i \\ &= (y_i - f)\mathbf{x}_i \end{aligned}$$

Logistic Regression

- Update rule

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \lambda \sum_{i=1}^N (y_i - f(\mathbf{x}_i)) \mathbf{x}_i$$

small constant

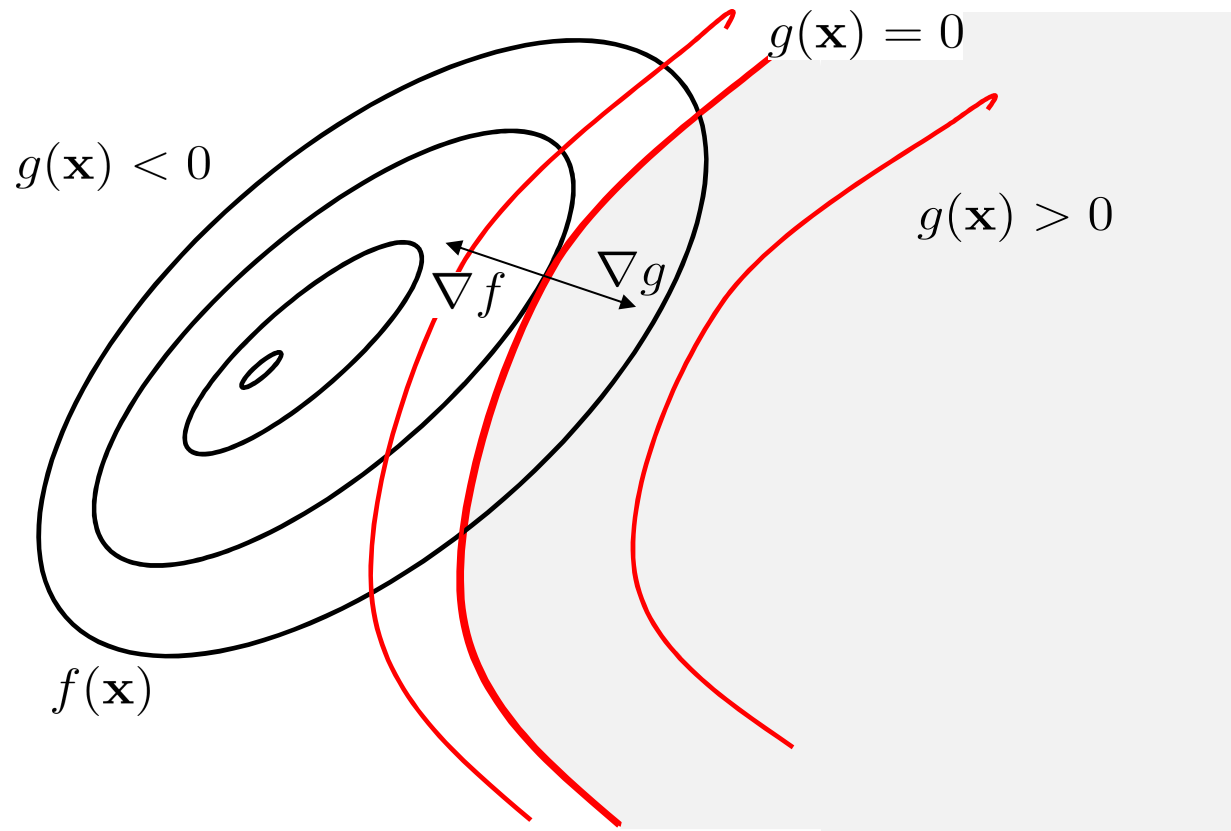
– Perform this operation until converge

- Online algorithm

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \lambda (y_i - f(\mathbf{x}_i)) \mathbf{x}_i$$

Optimization with Constraints

$$\max f(\mathbf{x}) \quad s.t. \quad g(\mathbf{x}) \geq 0$$



$$\nabla f = -\lambda \nabla g \quad \text{for a nonnegative constant } \lambda \ (\lambda \geq 0)$$

Optimization with Constraints

- For $\max f(\mathbf{x}) \quad s.t. \quad g(\mathbf{x}) \geq 0$

$$L(\mathbf{x}) = f(\mathbf{x}) + \lambda g(\mathbf{x}) \quad \longrightarrow \quad \left. \frac{dL(\mathbf{x})}{d\mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}^*} = 0$$

- For $\min f(\mathbf{x}) \quad s.t. \quad g(\mathbf{x}) \geq 0$

$$L(\mathbf{x}) = f(\mathbf{x}) - \lambda g(\mathbf{x}) \quad \longrightarrow \quad \left. \frac{dL(\mathbf{x})}{d\mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}^*} = 0$$

Optimization with Constraints

- Karush–Kuhn–Tucker (KKT) condition

maximize $f(\mathbf{x})$ for $\mathbf{x} \in \mathbb{R}^D$

subject to

$$g_i(\mathbf{x}) \geq 0, \quad i = 1, \dots, m$$

$$h_j(\mathbf{x}) = 0, \quad j = 1, \dots, r$$

Optimization with Constraints

- Karush–Kuhn–Tucker (KKT) condition

$$L(\mathbf{x}, \lambda, \mu) = f(\mathbf{x}) + \sum_{i=1}^m \lambda_i g_i(\mathbf{x}) + \sum_{j=1}^r \mu_j h_j(\mathbf{x})$$

$$\frac{\partial L(\mathbf{x}, \lambda, \mu)}{\partial x_k} = 0, \quad k = 1, \dots, D$$

$$\lambda_i \geq 0, \quad i = 1, \dots, m$$

$$\boxed{\lambda_i g_i(\mathbf{x}) = 0, \quad i = 1, \dots, m}$$

$$g_i(\mathbf{x}) \geq 0, \quad i = 1, \dots, m$$

$$h_j(\mathbf{x}) = 0, \quad j = 1, \dots, r$$

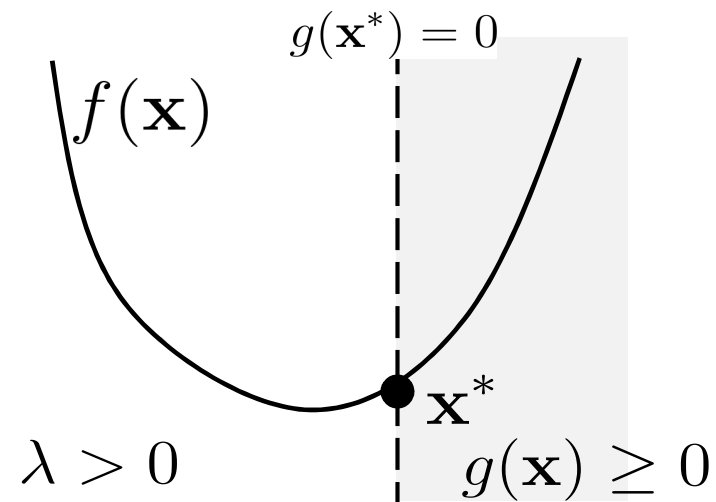
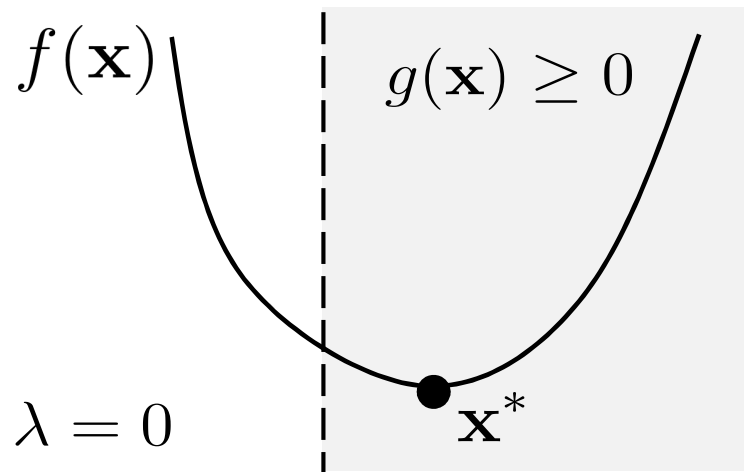
KKT Condition for Inequality Constraint

- The solution satisfies either

$$\lambda = 0 \quad \& \quad g(\mathbf{x}^*) \geq 0$$

or

$$\lambda > 0 \quad \& \quad g(\mathbf{x}^*) = 0$$



KKT Condition for Inequality Constraint

- To find analytic solution, we have to search for the solution that satisfies KKT condition – 2^k cases in the worst case for k constraints.

$$g_1(\mathbf{x}) \geq 0, \dots, g_k(\mathbf{x}) \geq 0$$

$$L(\mathbf{x}) = f(\mathbf{x}) + \lambda_1 g_1(\mathbf{x}) + \dots + \lambda_k g_k(\mathbf{x})$$

$$\begin{array}{l} \lambda_1 = 0, g_1(\mathbf{x}^*) \geq 0 \\ \lambda_2 = 0, g_2(\mathbf{x}^*) \geq 0 \\ \vdots \\ \lambda_k = 0, g_k(\mathbf{x}^*) \geq 0 \end{array}$$

$$\begin{array}{l} \lambda_1 > 0, g_1(\mathbf{x}^*) = 0 \\ \lambda_2 = 0, g_2(\mathbf{x}^*) \geq 0 \\ \vdots \\ \lambda_k = 0, g_k(\mathbf{x}^*) \geq 0 \end{array}$$

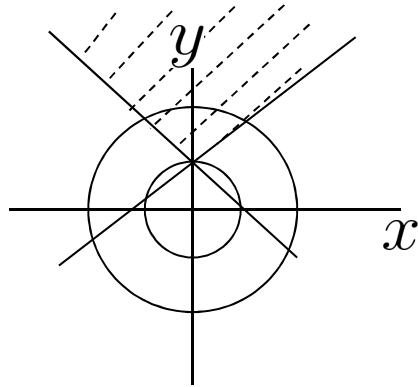
$$\begin{array}{l} \lambda_1 > 0, g_1(\mathbf{x}^*) = 0 \\ \lambda_2 > 0, g_2(\mathbf{x}^*) = 0 \\ \vdots \\ \lambda_k = 0, g_k(\mathbf{x}^*) \geq 0 \end{array}$$

...

2^k cases

Optimization with Constraints

• Ex)



$$f(\mathbf{x}) = x^2 + y^2$$

$$g_1(\mathbf{x}) = y - x - 1 \geq 0$$

$$g_2(\mathbf{x}) = y + x - 1 \geq 0$$

$$L = x^2 + y^2 - \lambda_1(y - x - 1) - \lambda_2(y + x - 1)$$

$$\frac{dL}{dx} = 2x + \lambda_1 - \lambda_2 = 0$$

$$\frac{dL}{dy} = 2y - \lambda_1 - \lambda_2 = 0$$

KKT condition (visit all cases)

$$(1) \lambda_1 = \lambda_2 = 0$$

$$(2) \lambda_1 = 0, y + x - 1 = 0$$

$$(3) y - x - 1 = 0, \lambda_2 = 0$$

$$(4) y - x - 1 = 0, y + x - 1 = 0$$

Optimization with Constraints

$$f(\mathbf{x}) = x^2 + y^2 \quad g_1(\mathbf{x}) = y - x - 1 \geq 0$$

$$g_2(\mathbf{x}) = y + x - 1 \geq 0$$

$$\frac{dL}{dx} = 2x + \lambda_1 - \lambda_2 = 0 \quad \frac{dL}{dy} = 2y - \lambda_1 - \lambda_2 = 0$$

$$(1) \lambda_1 = \lambda_2 = 0$$

$$x = y = 0 \quad \text{Not feasible}$$

$$(2) \lambda_1 = 0, y + x - 1 = 0$$

$$\left. \begin{array}{l} 2x - \lambda_2 = 0 \\ 2y - \lambda_2 = 0 \end{array} \right] \longrightarrow x - y = 0$$

$$x = y = \frac{1}{2} \quad \text{Not feasible}$$



Optimization with Constraints

$$(3) \ y - x - 1 = 0, \lambda_2 = 0$$

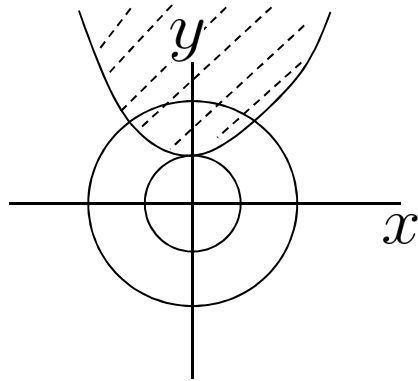
$$\left. \begin{array}{l} 2x + \lambda_1 = 0 \\ 2y - \lambda_1 = 0 \end{array} \right\} \longrightarrow x + y = 0$$
$$x = -\frac{1}{2}, y = \frac{1}{2} \quad \text{Not feasible}$$

$$(4) \ y - x - 1 = 0, y + x - 1 = 0$$

$$x = 0, y = 1 \quad \text{Feasible}$$

Optimization with Constraints

• Ex)



$$f(\mathbf{x}) = x^2 + y^2$$

$$g(\mathbf{x}) = y - x^2 - 1 \geq 0$$

$$L = x^2 + y^2 - \lambda(y - x^2 - 1)$$

$$\frac{dL}{dx} = 2x + 2\lambda x = 0 \quad \rightarrow \quad (1 + \lambda)x = 0$$

$$\frac{dL}{dy} = 2y - \lambda = 0 \quad \rightarrow \quad \lambda = 2y$$

$$\left. \begin{array}{l} (1 + \lambda)x = 0 \\ \lambda = 2y \end{array} \right\} \begin{array}{l} (1 + 2y)x = 0 \\ \dots (1) \end{array}$$

We assume that the solution is at $g(\mathbf{x}) = 0$

$$\text{From (1), } (2x^2 + 3)x = 0 \quad \rightarrow \quad y = 1$$

Optimization with Constraints

- Continued...
 - Candidate solution: $(0, 1)$
 - The candidate solution $(x_1, x_2) = (0, 1)$ satisfies $g(\mathbf{x}) \geq 0$

- Solution:

$$\mathbf{x} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Convex Optimization

- If the objective function is convex & if the feasible region is also convex.

- Objective function is convex

$$\alpha_1 + \alpha_2 = 1, \quad \alpha_1, \alpha_2 \geq 0$$

$$\frac{\alpha_1 f(\mathbf{x}_1) + \alpha_2 f(\mathbf{x}_2)}{\alpha_1 + \alpha_2} \geq f(\alpha_1 \mathbf{x}_1 + \alpha_2 \mathbf{x}_2)$$

- Domain is convex

$$\mathbf{x}_1, \mathbf{x}_2 \in \text{Domain}$$

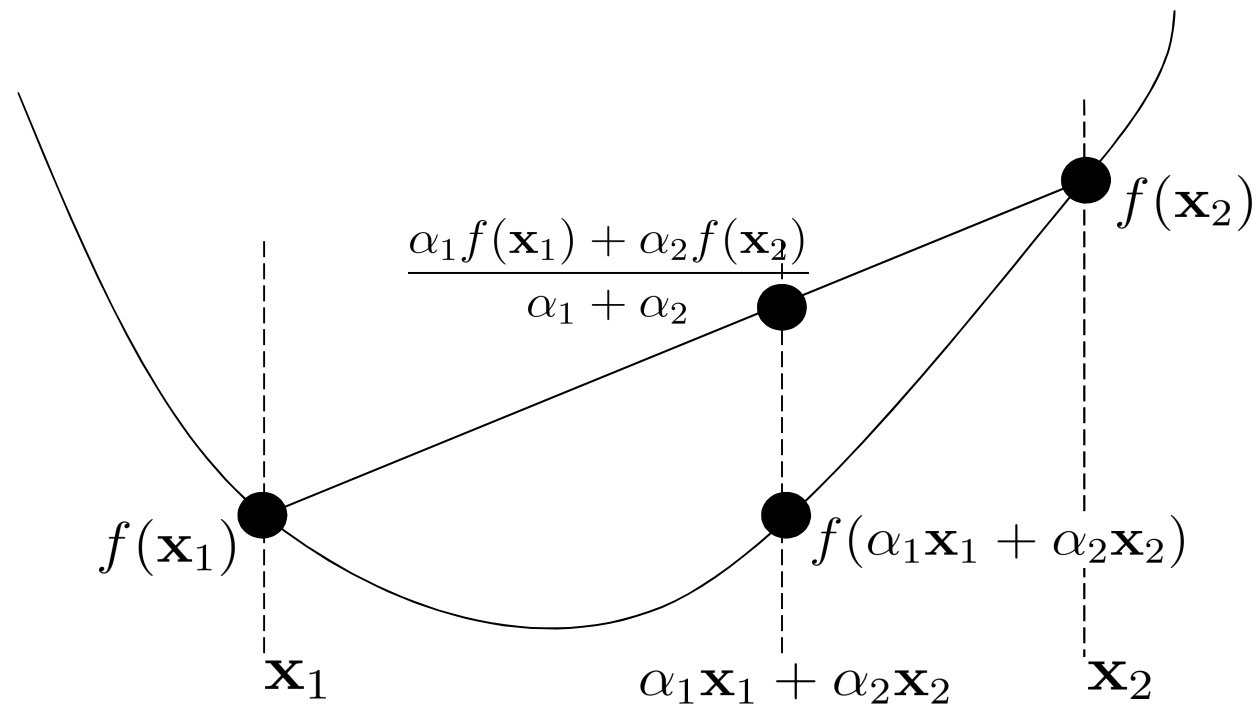
$$\alpha_1 + \alpha_2 = 1, \quad \alpha_1, \alpha_2 \geq 0$$

$$\alpha_1 \mathbf{x}_1 + \alpha_2 \mathbf{x}_2 \in \text{Domain}$$

Convex Optimization

- Function convexity

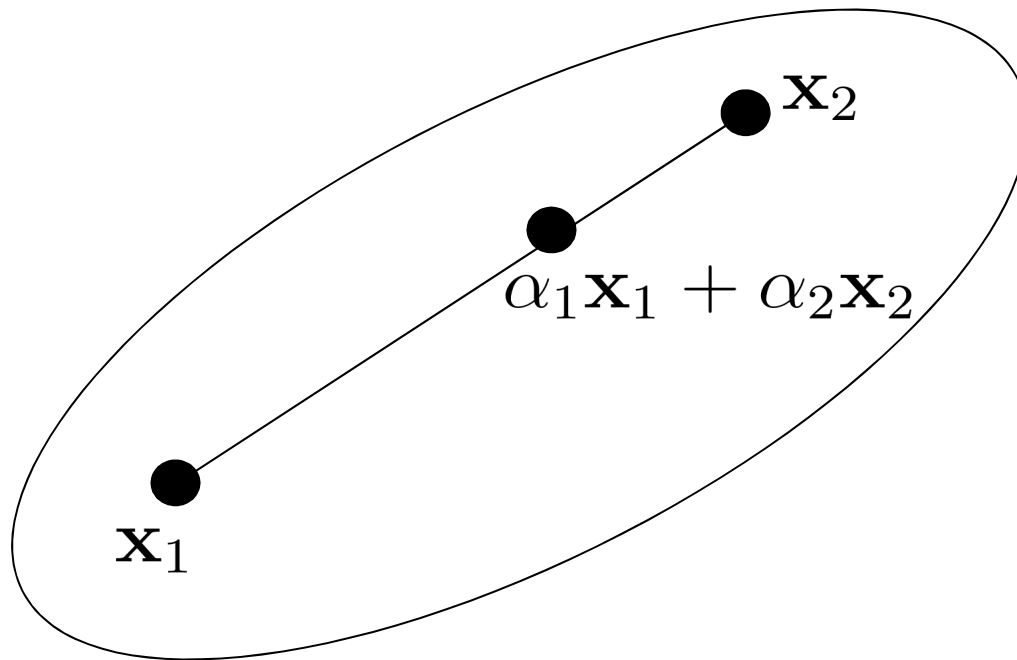
$$\frac{\alpha_1 f(\mathbf{x}_1) + \alpha_2 f(\mathbf{x}_2)}{\alpha_1 + \alpha_2} \geq f(\alpha_1 \mathbf{x}_1 + \alpha_2 \mathbf{x}_2)$$



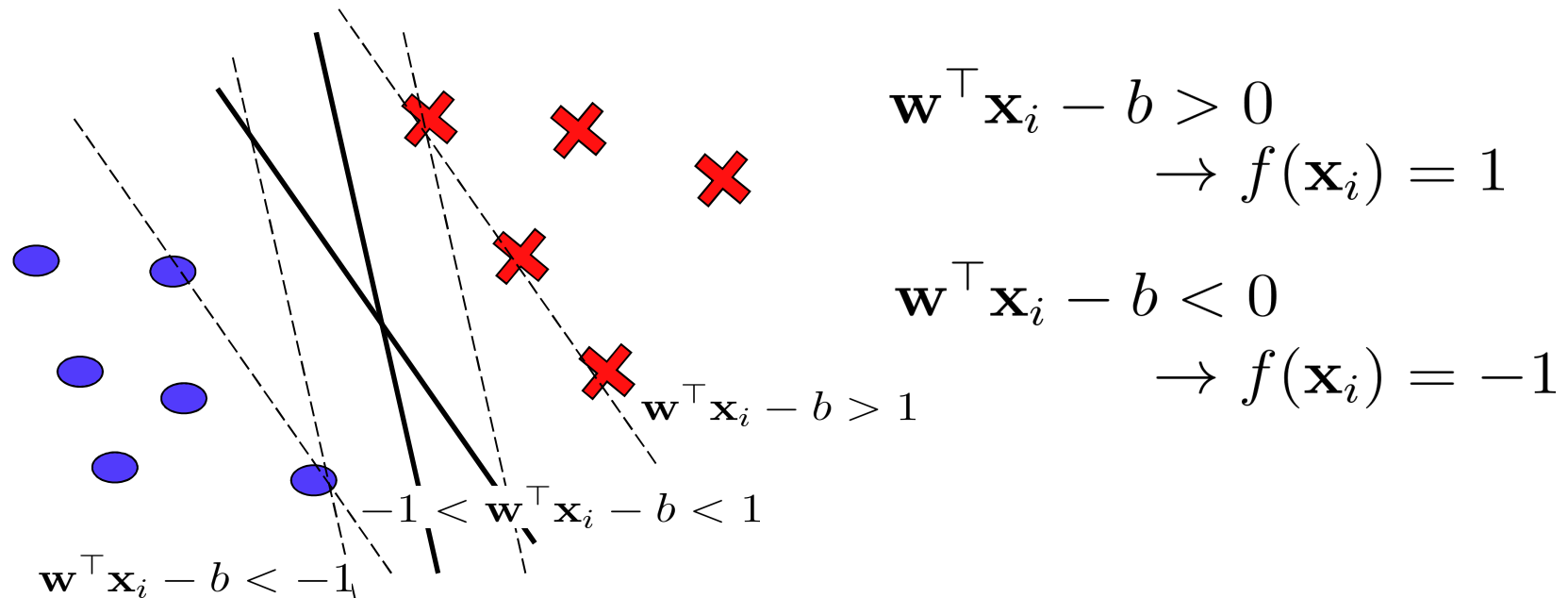
Convex Optimization

- Domain convexity

$$\alpha_1 \mathbf{x}_1 + \alpha_2 \mathbf{x}_2 \in \text{Domain}$$



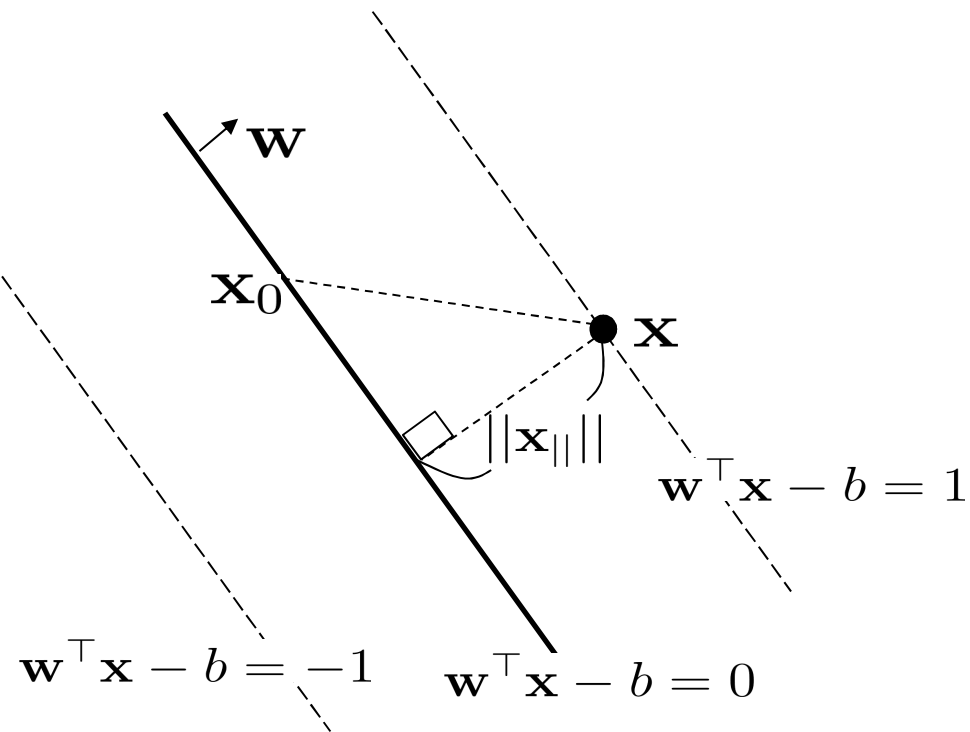
Large Margin Classifier



- We let all data outside margin

$$y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1 \quad y \in \{-1, 1\}$$

Large Margin Classifier



$$\mathbf{w}^\top (\mathbf{x} - \mathbf{x}_0) = \|\mathbf{w}\| \|\mathbf{x}_{||}\| = 1$$

$$\text{Margin: } \|\mathbf{x}_{||}\| = \frac{1}{\|\mathbf{w}\|}$$

Large Margin Classifier

- Support vector machines (SVMs) *Convex problem*

$$\min ||\mathbf{w}||^2 \quad s.t. \quad y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1 \quad \text{for all } i$$

– $D+1$ number of parameters, N constraints

$$L(\mathbf{w}, b) = \frac{1}{2} ||\mathbf{w}||^2 - \sum_{i=1}^N \alpha_i (y_i(\mathbf{w}^\top \mathbf{x}_i - b) - 1)$$
$$\alpha_i \geq 0$$

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \quad \rightarrow \quad \mathbf{w} - \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i = 0$$

$$\frac{\partial L}{\partial b} = 0 \quad \rightarrow \quad \sum_{i=1}^N \alpha_i y_i = 0$$

Large Margin Classifier

- Maximize (Convex dual)

$$L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j$$

$$\alpha_i \geq 0, \quad \sum \alpha_i y_i = 0 \quad \text{Convex dual}$$

orthant *simplex*

- With obtained $\alpha_1, \dots, \alpha_N$,

$$\mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i$$

$$b = -1 + \mathbf{w}^\top \mathbf{x}^{(s)} \quad \mathbf{x}^{(s)}: \text{a support vector}$$

- Classification:

$$y = \text{sgn} [\mathbf{w}^\top \mathbf{x}_{\text{new}} - b]$$

Kernel Idea

- For training:

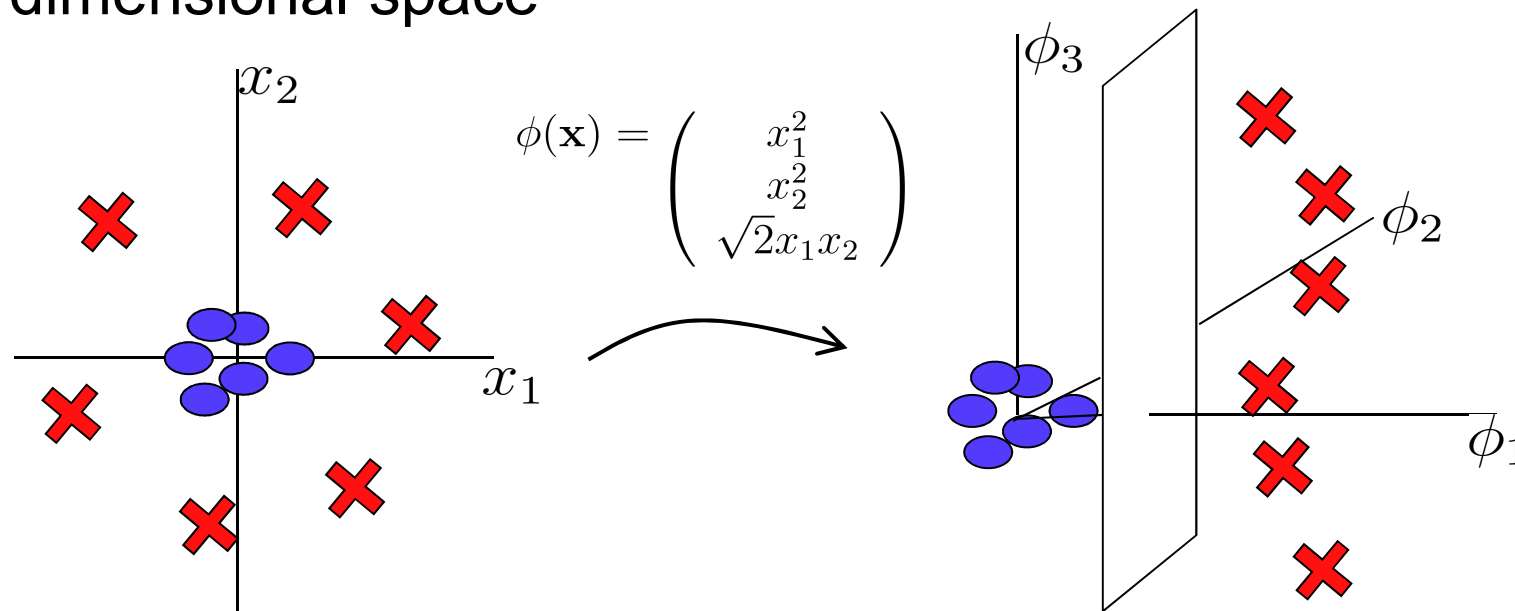
$$L = \sum_i \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \underline{\mathbf{x}_i^\top \mathbf{x}_j}$$

- For classification

$$\begin{aligned} y &= \text{sgn} [\mathbf{w}^\top \mathbf{x}_{\text{new}} - b] \quad \left(\mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i \right) \\ &= \text{sgn} \left[\sum_i \alpha_i y_i \underline{\mathbf{x}_i^\top \mathbf{x}_{\text{new}}} - b \right] \end{aligned}$$

Mapping Data to a High Dimensional Space

- Consider a nonlinear mapping to a higher-dimensional space



- And consider a function

$$\begin{aligned} k(\mathbf{x}, \mathbf{z}) &= (\mathbf{x}^\top \mathbf{z})^2 = (x_1 z_1 + x_2 z_2)^2 = x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2 \\ &= \begin{pmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1 x_2 \end{pmatrix}^\top \begin{pmatrix} z_1^2 \\ z_2^2 \\ \sqrt{2}z_1 z_2 \end{pmatrix} = \phi(\mathbf{x})^\top \phi(\mathbf{z}) \end{aligned}$$

Mapping Data to High Dimensional Space

- Consider functions $k(.,.)$ of which the corresponding mapping $\Phi(.)$ exists:
- Condition: if a function $k(.,.)$ is **positive definite (P.D.)**, there exists a mapping function $\Phi(.)$ satisfying

$$k(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^\top \phi(\mathbf{z}) \quad (\text{Mercer theorem})$$

- Such functions include

$$k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^\top \mathbf{z} + 1)^d \quad (\text{Polynomial kernel})$$

$$k(\mathbf{x}, \mathbf{z}) = \exp \left(-\frac{(\mathbf{x} - \mathbf{z})^2}{\gamma} \right) \quad (\text{Gaussian kernel})$$

⋮

Replace All Inner Products with Kernels

- For training:

$$L = \sum_i \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \underline{k(\mathbf{x}_i, \mathbf{x}_j)}$$

- For classification

$$\begin{aligned} y &= \text{sgn} [\mathbf{w}^\top \mathbf{x}_{\text{new}} - b] \\ &= \text{sgn} \left[\sum_i \alpha_i y_i \underline{k(\mathbf{x}_i, \mathbf{x}_{\text{new}})} - b \right] \end{aligned}$$

Positive Definiteness of a Matrix

- Positive definiteness of a matrix $A \in \mathbb{R}^{D \times D}$

$$A \succ 0$$

- For any vector $\mathbf{c} \in \mathbb{R}^D$

$$\mathbf{c}^\top A \mathbf{c} > 0$$

Positive semi-definiteness $A \succeq 0$ is similarly defined with the corresponding condition $\mathbf{c}^\top A \mathbf{c} \geq 0$ for any vector $\mathbf{c} \in \mathbb{R}^D$.

Positive Definiteness of a Matrix

- Matrix A is P.D. if (and only if)
 - All eigenvalues are positive $U = (\mathbf{u}_1, \dots, \mathbf{u}_D)$

$$\mathbf{c}^\top A \mathbf{c} = \mathbf{c}^\top U \Lambda U^\top \mathbf{c} = \sum_{i=1}^D \lambda_i (\mathbf{u}_i^\top \mathbf{c})^2 > 0$$

- A matrix can be decomposed as $A = Z^\top Z$ using a full rank matrix Z .

$$\mathbf{c}^\top A \mathbf{c} = \mathbf{c}^\top Z^\top Z \mathbf{c} = \sum_{i=1}^D (\mathbf{z}_i^\top \mathbf{c})^2 > 0$$

$$Z = (\mathbf{z}_1, \dots, \mathbf{z}_D)$$

Positive Definiteness of a Function

- Positive definiteness of a function $k(\mathbf{x}_1, \mathbf{x}_2)$ of two variables $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^D$,
 - For any set of vectors $\mathbf{x}_1, \dots, \mathbf{x}_N$ for arbitrary N , the matrix of size $N \times N$ with function values is positive definite.

$$\begin{pmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ k(\mathbf{x}_2, \mathbf{x}_1) & \cdots & k(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{pmatrix} \succ 0$$

Support Vectors

$$L(\mathbf{w}, b) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^N \alpha_i (y_i(\mathbf{w}^\top \mathbf{x}_i - b) - 1)$$

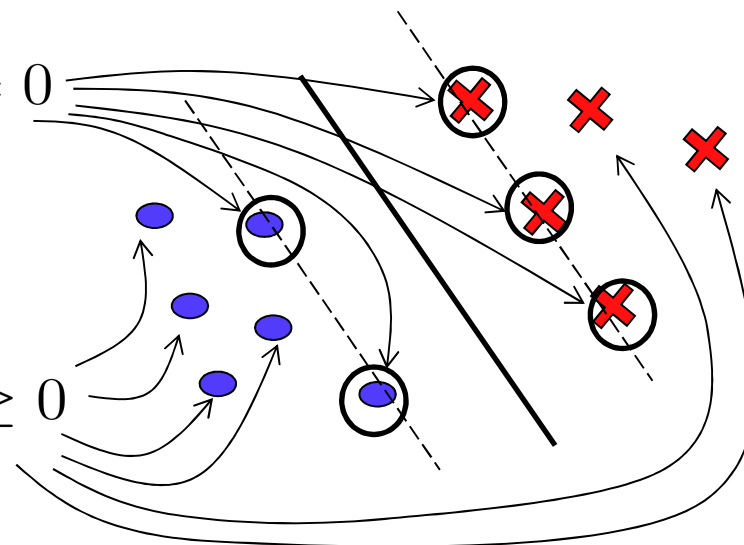
$$y = \text{sgn} \left[\sum_i \alpha_i y_i \mathbf{x}_i^\top \mathbf{x} - b \right]$$

- From KKT conditions, for $i = 1, \dots, N$,

$$\alpha_i \geq 0 \quad \& \quad y_i(\mathbf{w}^\top \mathbf{x}_i - b) - 1 = 0$$

or

$$\alpha_i = 0 \quad \& \quad y_i(\mathbf{w}^\top \mathbf{x}_i - b) - 1 \geq 0$$



Support Vectors and Classification

- Classification function

$$y = \text{sgn} \left[\sum_{i \in \text{Support Vectors}} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}_{\text{new}}) - b \right]$$

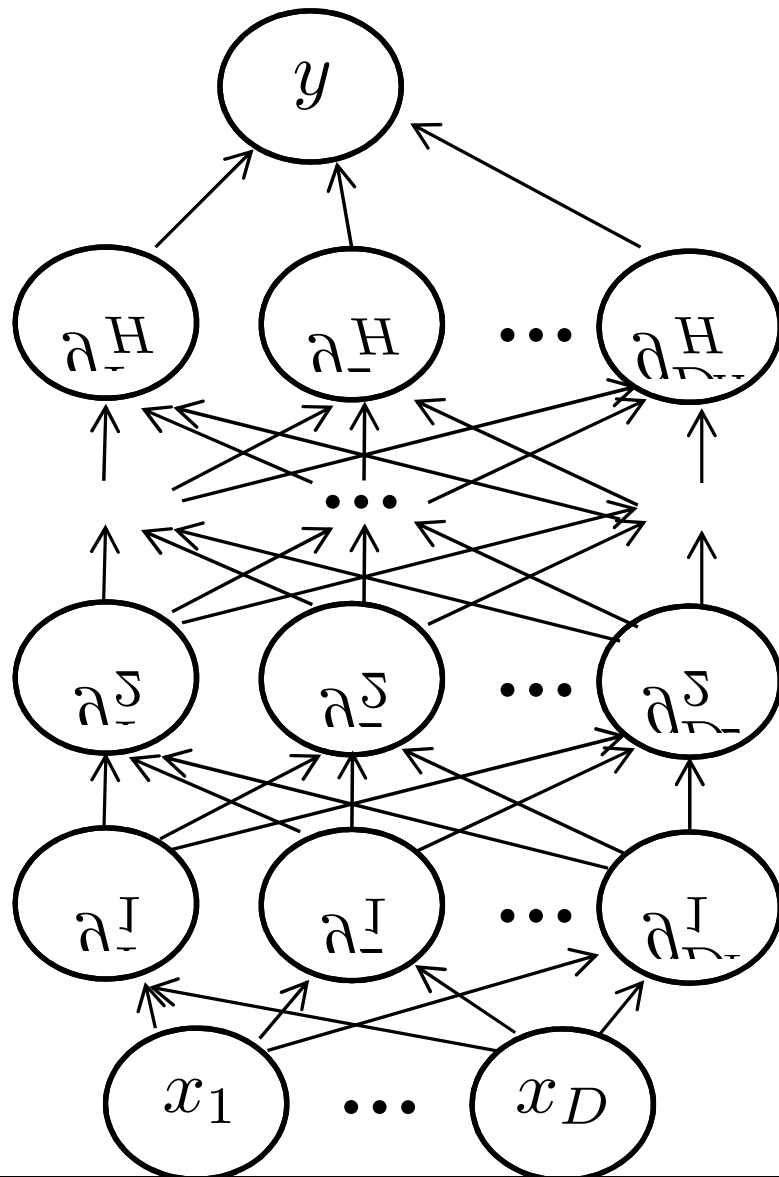
- Keep data $\mathbf{x}_i$ having nonzero α_i .
- We can keep support vectors, and others do not contribute to classification

Advanced Topics

- Reproducing Kernel Hilbert Space (RKHS)
- Representer Theorem
- Kernelization of other algorithms

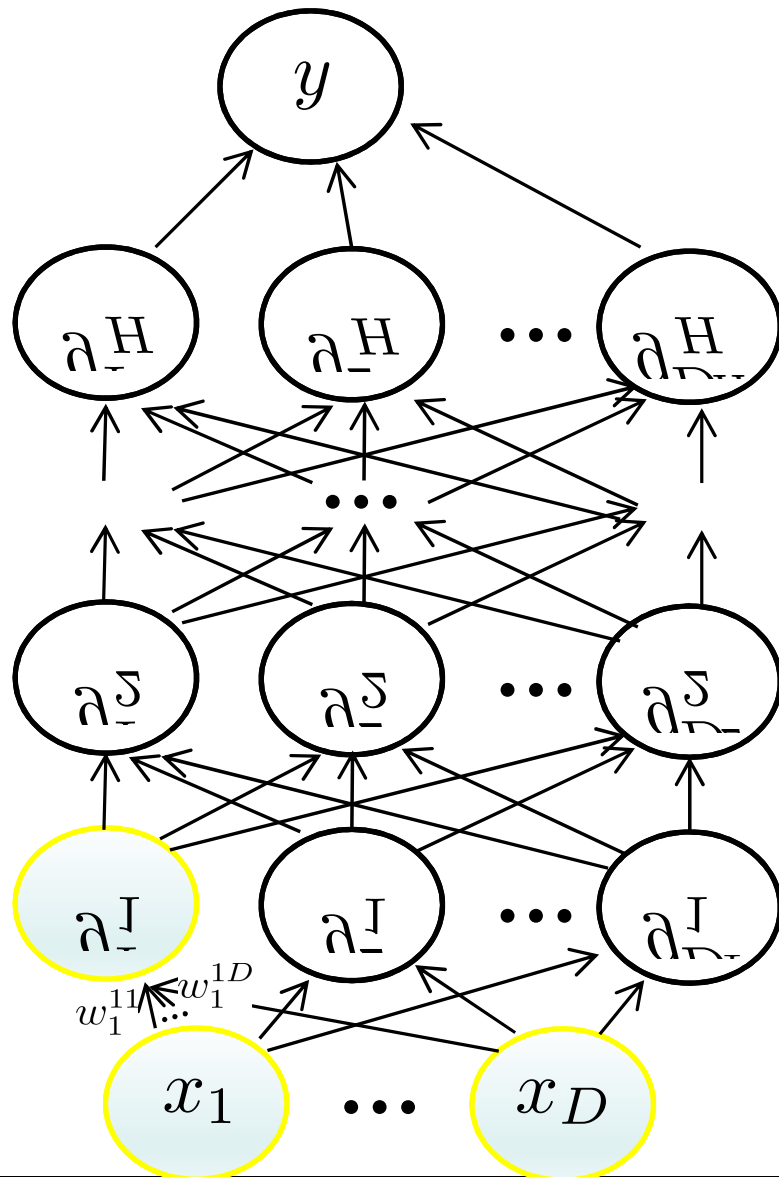


Artificial Neural Networks

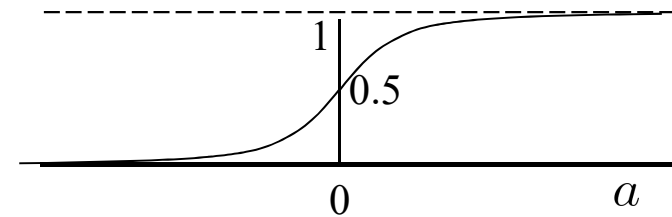


- Feedforward Neural Network
 - No loop (no recurrency)

Artificial Neural Networks

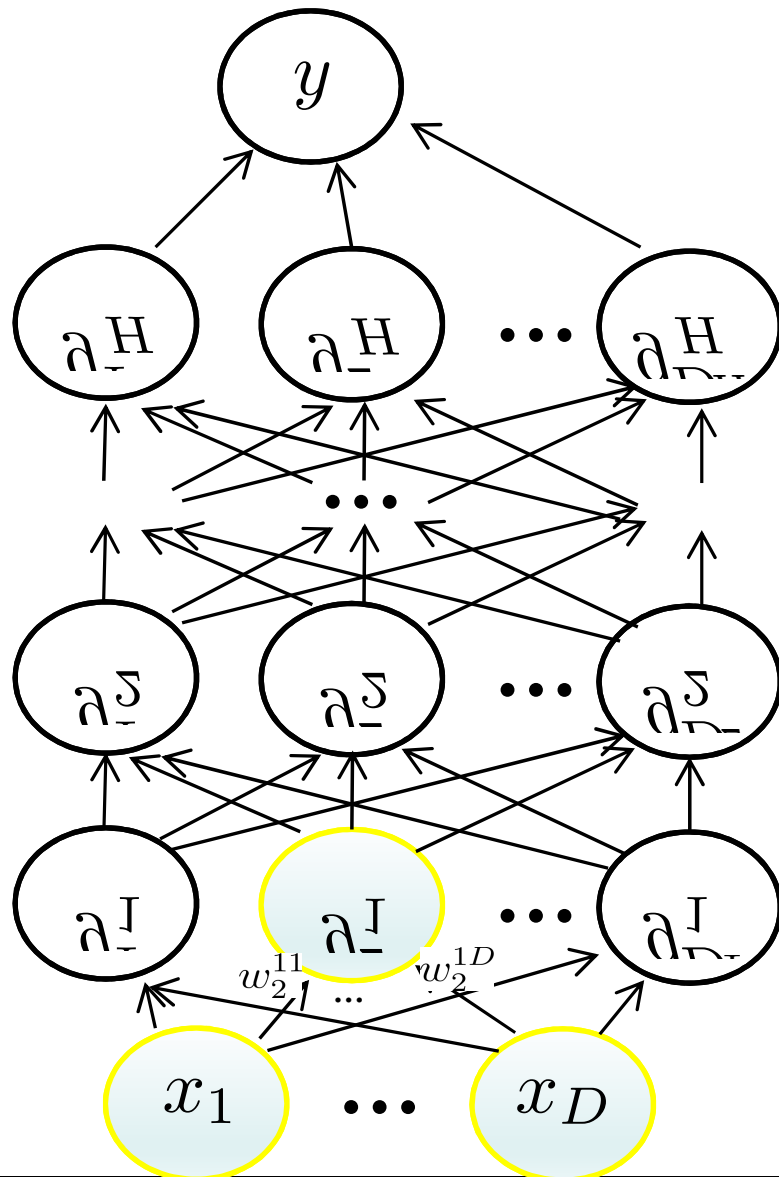


$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

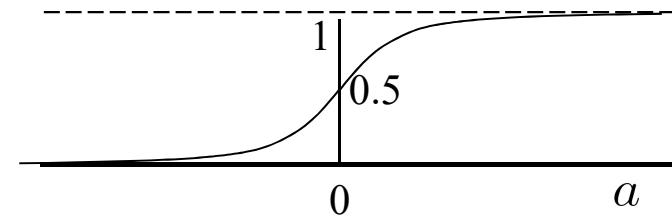


$$g_1^1 = \sigma \left(\sum_{i=1}^D w_1^{1i} x_i \right)$$

Artificial Neural Networks

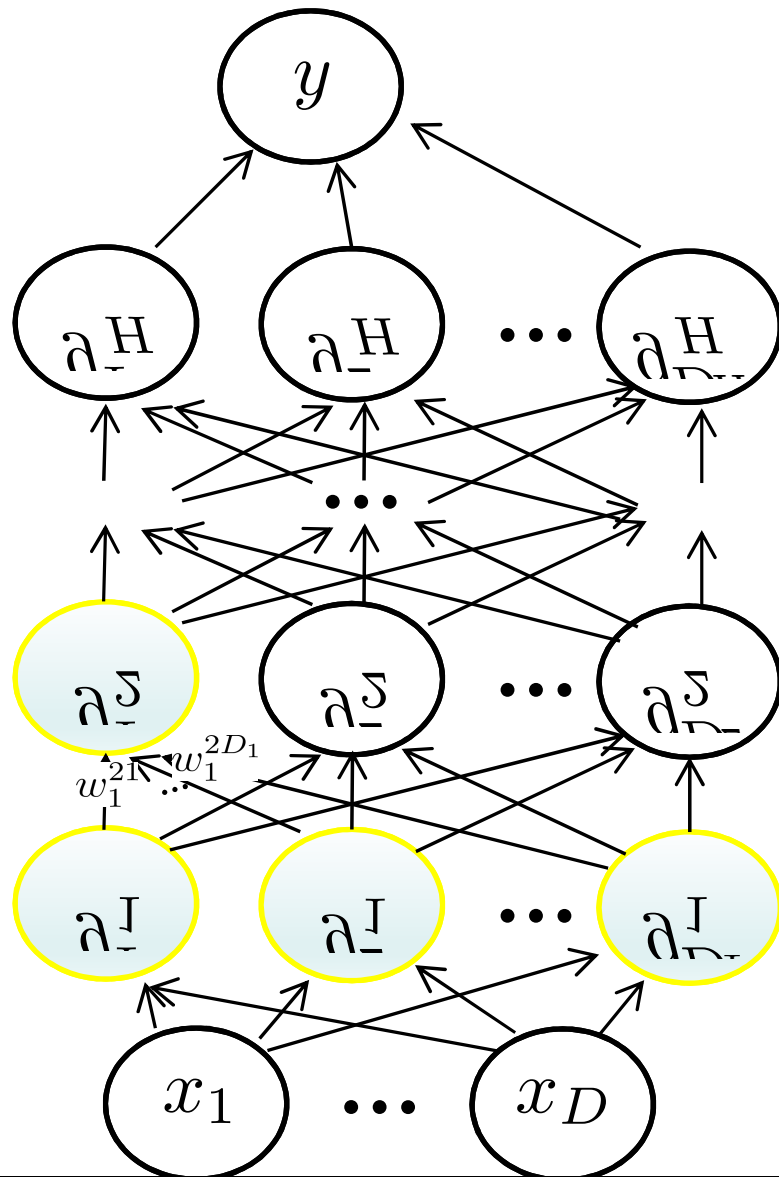


$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

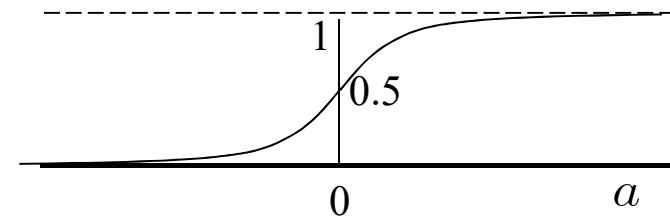


$$g_2^1 = \sigma \left(\sum_{i=1}^D w_2^{1i} x_i \right)$$

Artificial Neural Networks

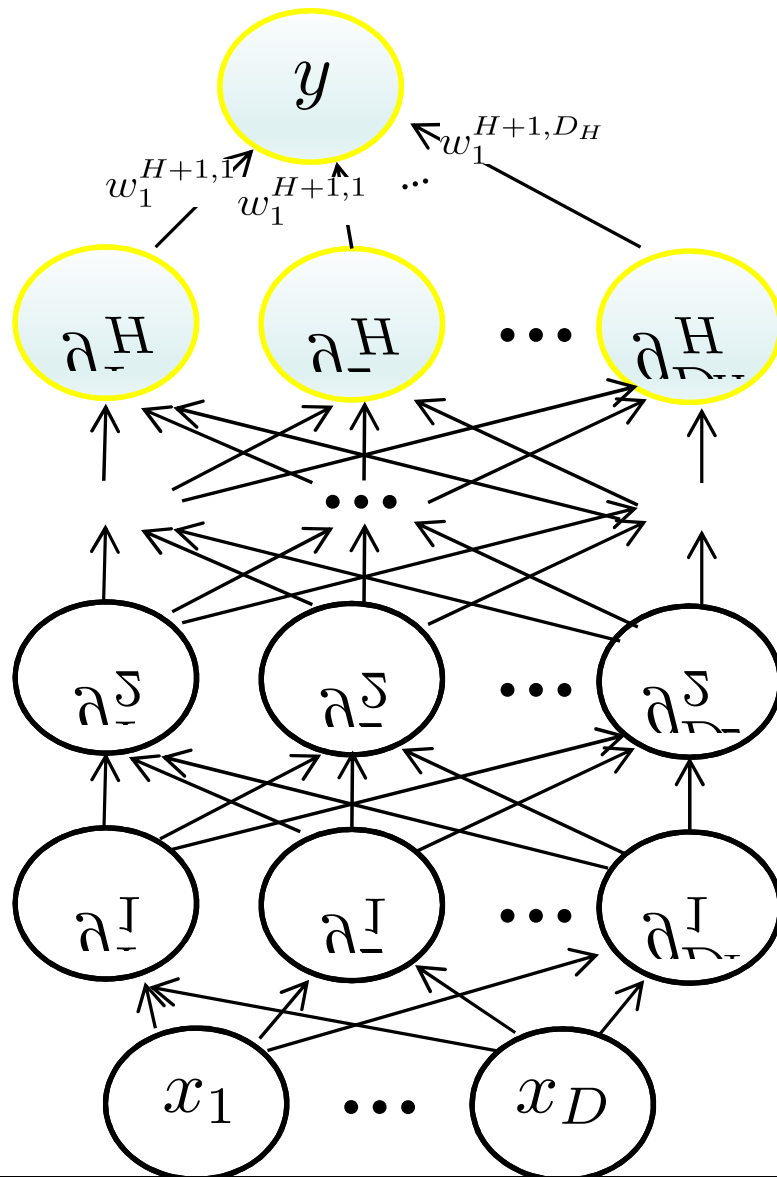


$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$



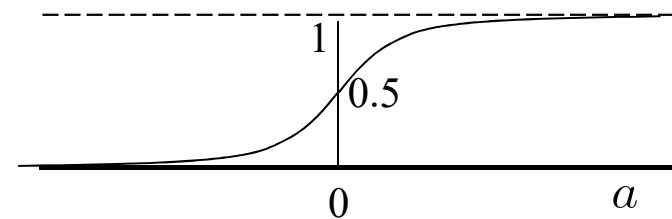
$$g_1^2 = \sigma \left(\sum_{i=1}^{D_1} w_1^{2i} g_i^1 \right)$$

Artificial Neural Networks

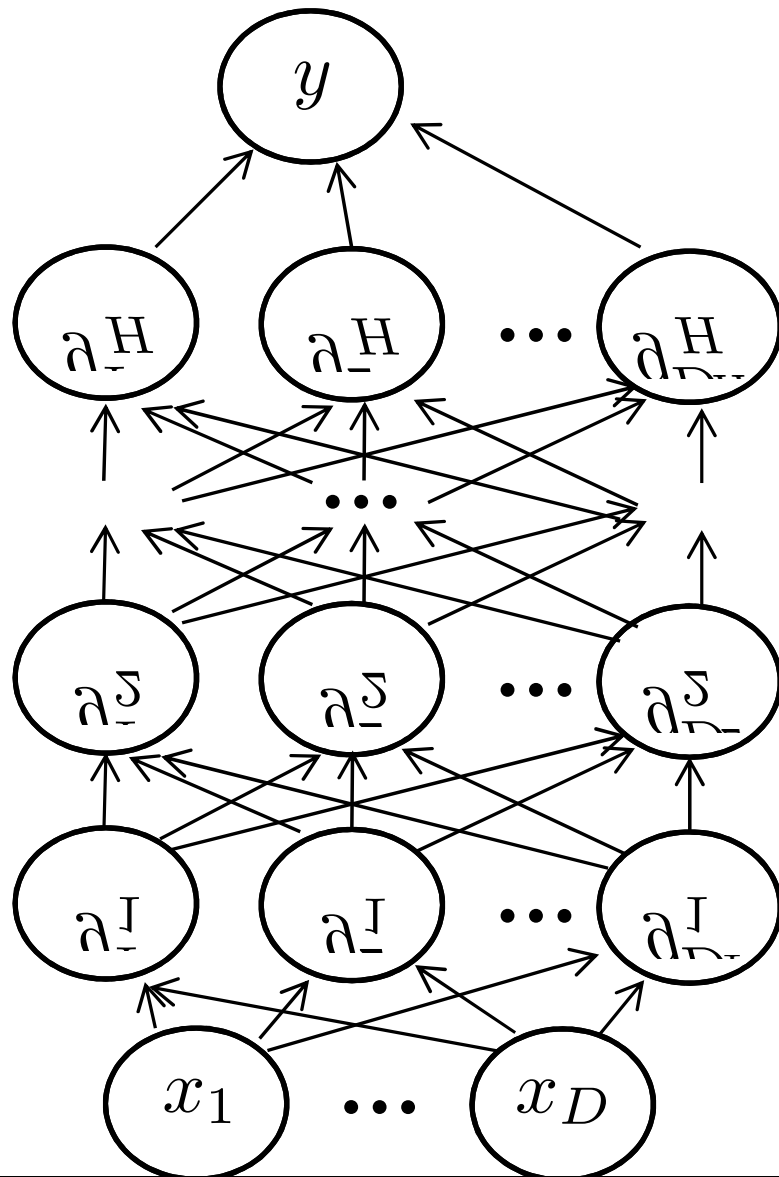


$$y = \sigma \left(\sum_{i=1}^{D_H} w^{out,i} g_i^H \right)$$

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$



Artificial Neural Networks



$$y = \sigma(\mathbf{w}^{out\top} \mathbf{g}^H(\mathbf{x}))$$

$$\mathbf{g}^{h+1}(\mathbf{x}) = \sigma(W^{h+1\top} \mathbf{g}^h(\mathbf{x}))$$

$$W^h = \begin{pmatrix} w_1^{h,1} & \dots & w_{D_h-1}^{h,1} \\ w_1^{h,2} & \dots & w_{D_h-1}^{h,2} \\ \vdots & & \vdots \\ w_1^{h,D_h} & \dots & w_{D_h-1}^{h,D_h} \end{pmatrix}$$

$$\mathbf{g}^1(\mathbf{x}) = \sigma(W^{1\top} \mathbf{x})$$

Optimization in ANN

$$\begin{aligned}\hat{y}(\mathbf{x}) &\leftarrow f(\mathbf{g}^H(\mathbf{g}^{H-1} \dots (\mathbf{g}^1(\mathbf{x})))) \\ &= \sigma(W^{H\top} \sigma(W^{H-1\top} \dots \sigma(W^{1\top} \mathbf{x})))\end{aligned}$$

- Empirical risk

$$L = \frac{1}{2} \sum_{i=1}^N ||y_i - \hat{y}(\mathbf{x}_i)||^2$$

– Derivative of the empirical risk

$$\begin{aligned}\frac{dL}{dW^h} &= \sum_{i=1}^N (y_i - \hat{y}_i) \left(-\frac{\partial \hat{y}_i}{\partial W^h} \right) \in \mathbb{R}^{D_h \times D_h} \\ \hat{y}_i &= \hat{y}(\mathbf{x}_i)\end{aligned}$$

Chain Rule

$$\frac{dL}{dW^h} = \sum_{i=1}^N (y_i - \hat{y}_i) \left(-\frac{\partial \hat{y}_i}{\partial W^h} \right)$$

$$\frac{\partial \hat{y}}{\partial W^h} = \frac{\partial \hat{y}}{\partial \mathbf{g}^H} \frac{\partial \mathbf{g}^H}{\partial \mathbf{g}^{H-1}} \cdots \frac{\partial \mathbf{g}^h}{\partial W^h}$$

$$\frac{\partial \sigma(a)}{\partial a} = \sigma(a)(1 - \sigma(a))$$

$$\frac{\partial \hat{y}}{\partial \mathbf{g}^H} = \frac{\partial \sigma(\mathbf{w}^{out\top} \mathbf{g}^H)}{\partial \mathbf{g}^H} = \sigma(\mathbf{w}^{out\top} \mathbf{g}^H)(1 - \sigma(\mathbf{w}^{out\top} \mathbf{g}^H)) \mathbf{w}^{out}$$

$$\frac{\partial \mathbf{g}_i^{h+1}}{\partial \mathbf{g}^h} = \frac{\partial \sigma([W^{h+1\top} \mathbf{g}^h]_i)}{\partial \mathbf{g}^h} = \sigma([W^{h+1\top} \mathbf{g}^h]_i)(1 - \sigma([W^{h+1\top} \mathbf{g}^h]_i)) W_{:i}^{h+1}$$

$$\frac{\partial \mathbf{g}_i^h}{\partial W_{:i}^h} = \frac{\partial \sigma([W^{h\top} \mathbf{g}^{h-1}]_i)}{\partial W_{:i}^h} = \sigma([W^{h\top} \mathbf{g}^{h-1}]_i)(1 - \sigma([W^{h\top} \mathbf{g}^{h-1}]_i)) \mathbf{g}^{h-1}$$

($\mathbf{g}^0 = \mathbf{x}$)

$$\frac{\partial \mathbf{g}_i^h}{\partial W_{:j}^h} = 0, \quad i \neq j$$

Update Rule

$$\begin{aligned} W^h &\leftarrow W^h - \gamma \frac{dL}{dW^h} \\ &= W^h - \gamma \sum_{i=1}^N (y_i - \hat{y}_i) \left(-\frac{\partial \hat{y}_i}{\partial W^h} \right) \\ &= W^h + \gamma \sum_{i=1}^N (y_i - \hat{y}_i) \frac{\partial \hat{y}_i}{\partial \mathbf{g}^H} \frac{\partial \mathbf{g}^H}{\partial \mathbf{g}^{H-1}} \cdots \frac{\partial \mathbf{g}^h}{\partial W^h} \end{aligned}$$

Delta Rule (Reuse Derivative)

$$\delta_i^h \equiv (y_i - \hat{y}_i) \frac{\partial \hat{y}_i}{\partial \mathbf{g}^H} \frac{\partial \mathbf{g}^H}{\partial \mathbf{g}^{H-1}} \cdots \frac{\partial \mathbf{g}^{h+1}}{\partial \mathbf{g}^h}$$

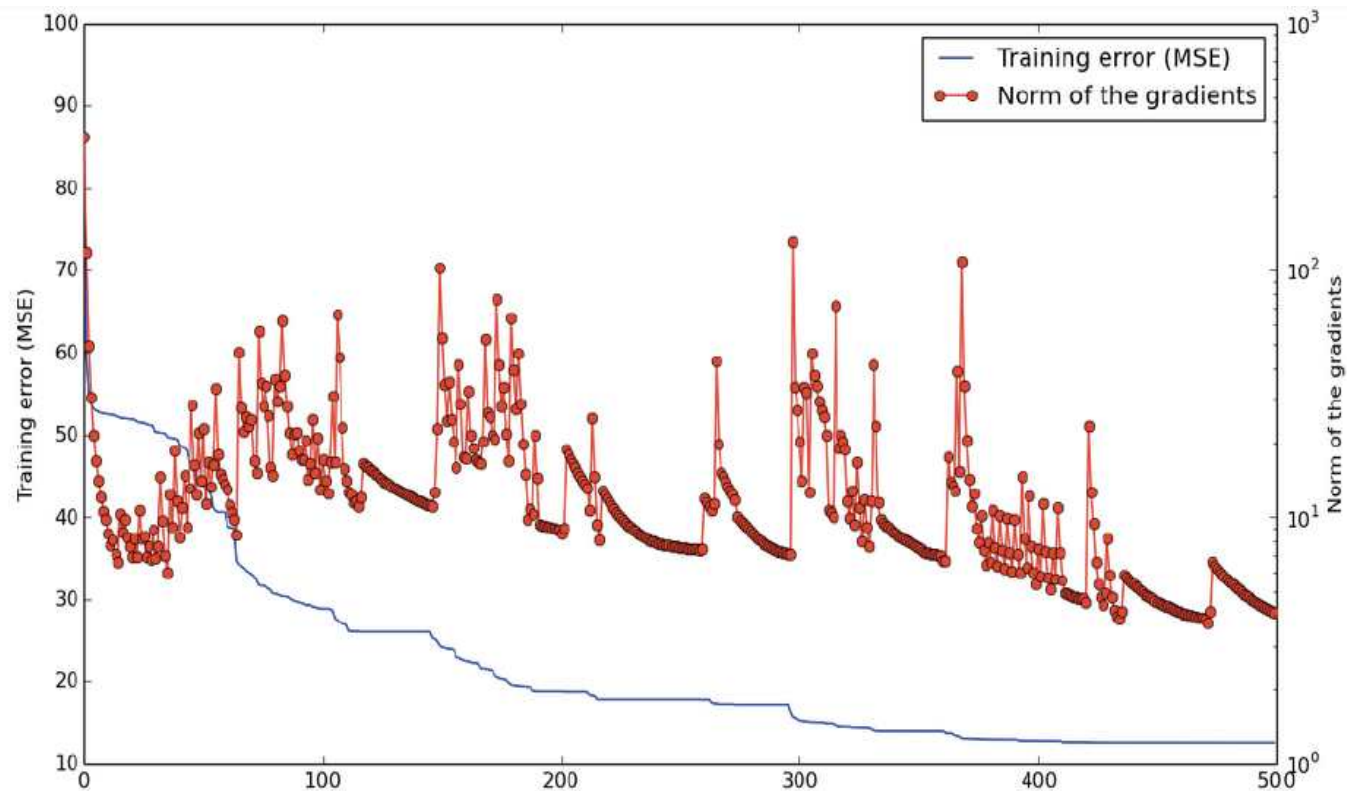
$$\delta_i^h = \delta_i^{h+1} \frac{\partial \mathbf{g}^{h+1}}{\partial \mathbf{g}^h} \quad \textit{Backpropagation of error}$$

- Update of weights of the level h using backpropagated error

$$W^h \leftarrow W^h + \gamma \sum_{i=1}^N \delta_i^h \frac{\partial \mathbf{g}^h}{\partial W^h}$$

Local Minima in ANN

- Saddle point analysis

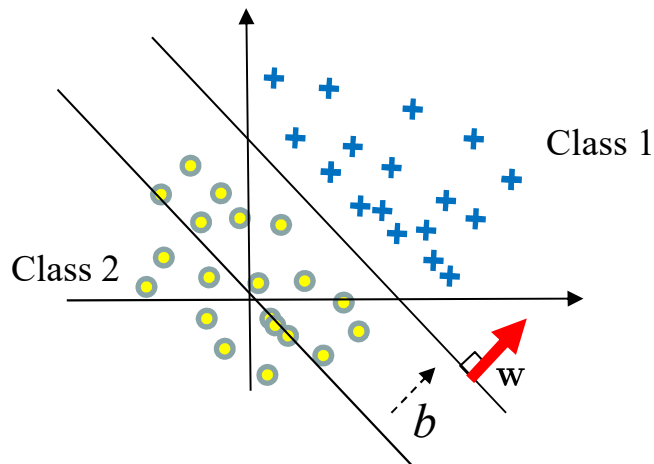


Razvan Pascanu et al. "On the saddle point problem for non-convex optimization." arXiv preprint arXiv:1405.4604 (2014)
Anna Choromanska et al. "The loss surfaces of multilayer networks." arXiv preprint arXiv:1412.0233 (2014)

Bias in ANN

- Linear classifier

$$\mathbf{w}^\top \mathbf{x} - b \leq 0 \quad \rightarrow \quad \mathbf{w}'^\top \mathbf{x}' \leq 0$$



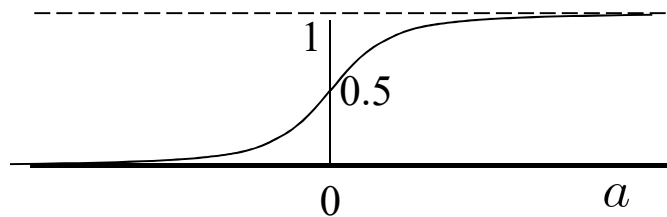
$$\mathbf{w}' = \begin{pmatrix} w_1 \\ \vdots \\ w_D \\ -b \end{pmatrix} \quad \mathbf{x}' = \begin{pmatrix} x_1 \\ \vdots \\ x_D \\ 1 \end{pmatrix}$$

Implement a bias of the affine equation in a linear equation.

Different Activation Functions

- Sigmoid

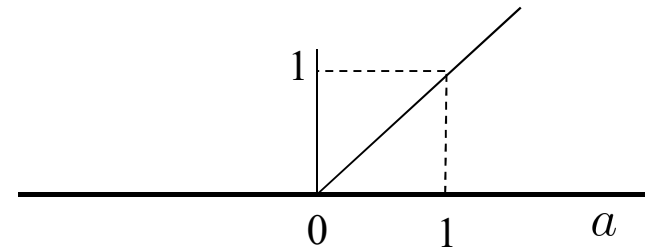
$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$



$$\frac{d\sigma(a)}{da} = \sigma(1 - \sigma)$$

- Rectified Linear Unit (ReLU)

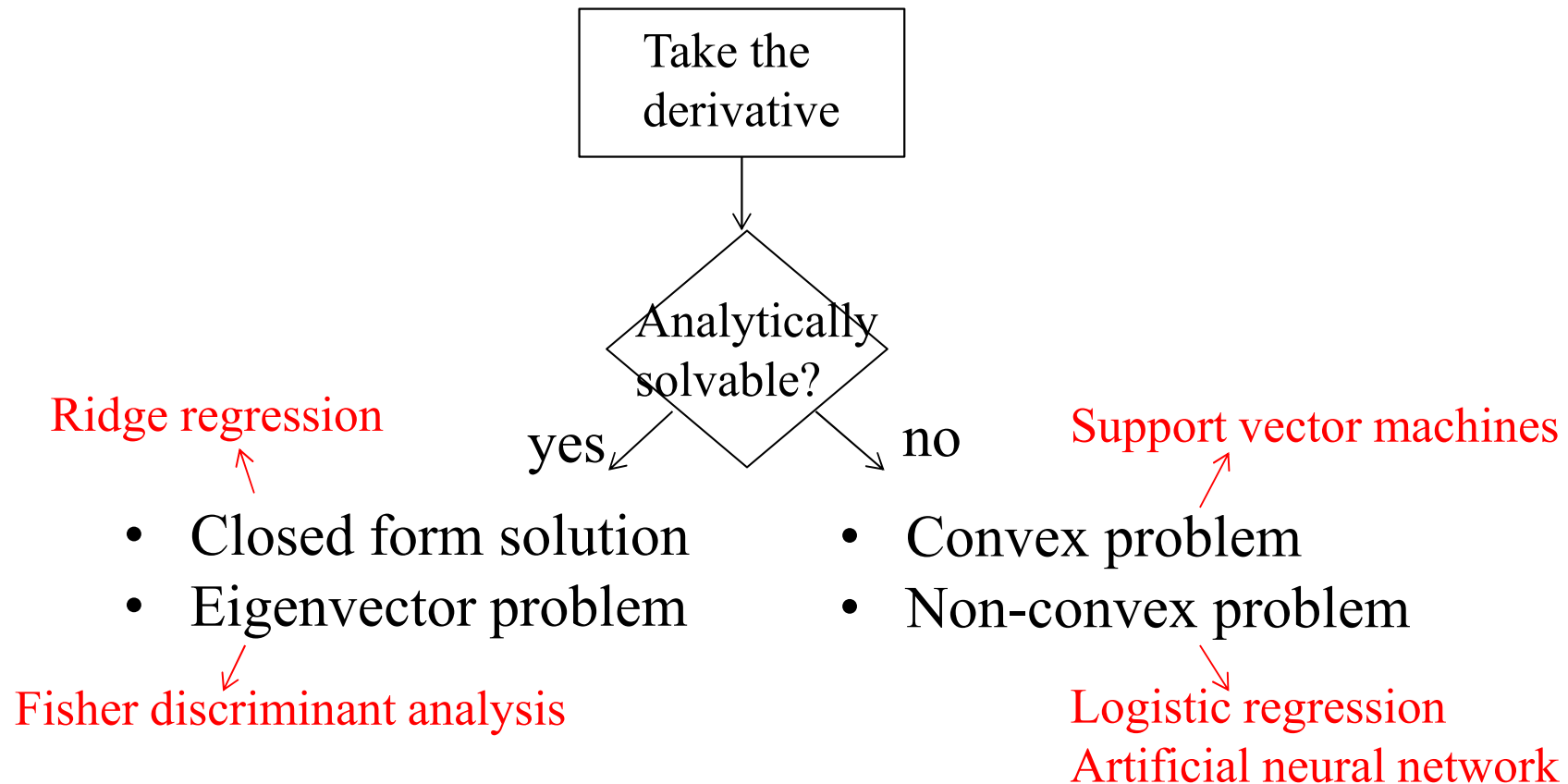
$$\sigma(a) = \max\{0, a\}$$



$$\frac{d\sigma(a)}{da} = \begin{cases} 1 & a > 0 \\ 0 & a \leq 0 \end{cases}$$

Determine the Problem I Am Solving!!

“Take the derivative, first”



ANY QUESTIONS?

