Why Nonlinear Optimization?

Supervised Learning. (x_1, y_1) (x_2, y_2) ... (x_n, y_n) — training data $\sim p(x, y)$

input output labels

$x_i \in \mathbb{R}^{N_x}$

(x, y)

x $\boxed{0}$ $\boxed{1}$

$\left.\begin{array}{c}\boxed{1}\\ \boxed{1}\\ \boxed{1}\end{array}\right\} \to 1$

$y =$

(x, y) $\quad$ $p(x, y)$

Recognition $\quad p(y \mid x) = \dfrac{p(x,y)}{\int p(x,y)\,dy = p_X(x)}$

training data $(x_1, y_1) \cdots (x_n, y_n)$ i.i.d. $\sim p(x, y)$

$$
\begin{tikzcd}
x \arrow[r] \arrow[dr] & \boxed{p(y|x)} \arrow[r] & y \\
& g(x;w) \arrow[ur] &
\end{tikzcd}
$$

$g(x;w) \approx y$

error $= |g(x;w) - y|$

$$\mathbb{E}_{(x,y)\sim p(x,y)} |g(x, w) - y| = \int_y \int_x \underline{|g(x,w) - y|}\, p(x,y)\, dx\, dy$$ "population loss"

$$= L(w)$$

Learning $w^* = \boxed{\min_w L(w)}$

Framework $(x, y) \sim p(x, y)$

$$\underline{g(x; w)} \begin{cases} = wx \\ w^T x = w_1 x_1 + \dots + w_n x_n \end{cases}$$

$$\boxed{L(w) = \mathbb{E}_{(x,y)\sim p(x,y)}\, \ell(g(x,w); y)}$$

$\ell(\underline{g; y})$ — Loss function

Monday, July 13, 2020 9:18 AM

$$X \longrightarrow g(x; w^*) - y \neq 0$$

$g(x; w)$

$$\mathbb{E}_{(x,y)\sim p(x,y)} \ell(g(x;w), y) \approx \frac{\ell(g(x_1;w), y_1) + \dots + \ell(g(x_n;w), y_n)}{n}$$

"empirical loss"

$$L_n(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(g(x_i; w), y_i) \xrightarrow{n\to\infty} \mathbb{E}_{(x,y)\sim p(x,y)} \ell(g(x,w), y)$$

Monday, July 13, 2020 9:29 AM

$(x_1, y_1) \cdots (x_n, y_n)$ training data

$$\arg\min_{w} \frac{1}{n} \sum_{i=1}^{n} \ell(\underline{g}(x_i, w), y_i) = w_n^* \qquad \text{"Optimization"}$$

Empirical Risk Minimization (ERM).

$$w_n^* \xrightarrow{n \to \infty} w^*$$

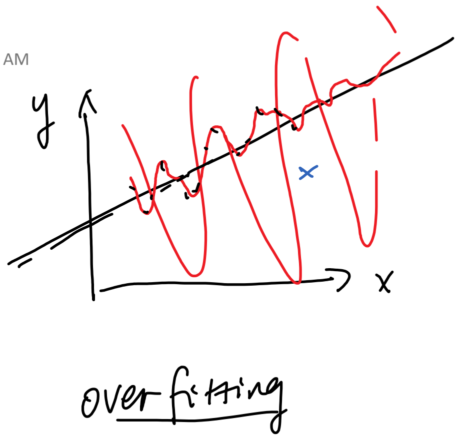
$$\arg\min L_n(w) \qquad \arg\min \underline{L(w)}$$

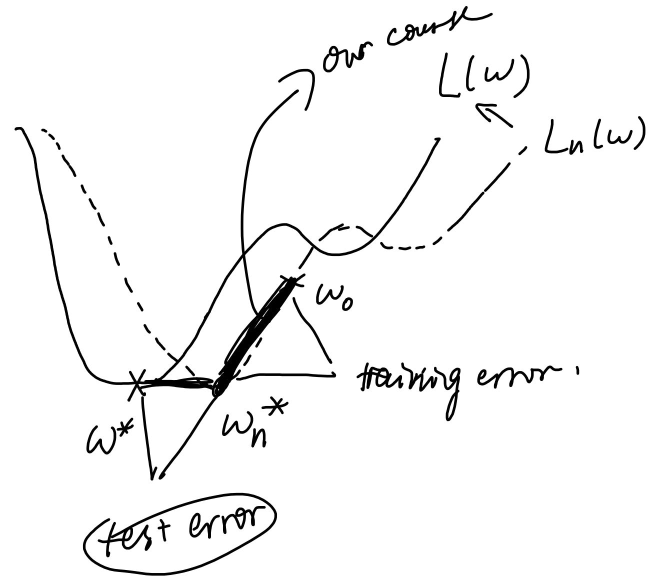
$$L_n \longrightarrow L$$

$$\mathbb{E}_{(x,y)} \ell(g(x, w_n^*); y)$$

"Statistics" $(x, y) \sim \cancel{p(x,y)}$

test data


y
x
overfitting


our course
L(w)
Ln(w)
w0
training error
w*
wn*
test error

Empirical Risk Minimization $L_n(w) = \frac{1}{n}\sum_{i=1}^{n} \ell(g(x_i; w), y_i)$

Learning Problem $\min_{w} L_n(w) = w_n^*$

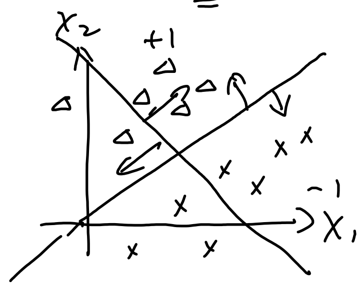
Concrete Stuff

"Loss function" $\ell(g, y)$ $\begin{cases} \text{classification} \\ \text{estimation} \end{cases}$

0-1 Loss function

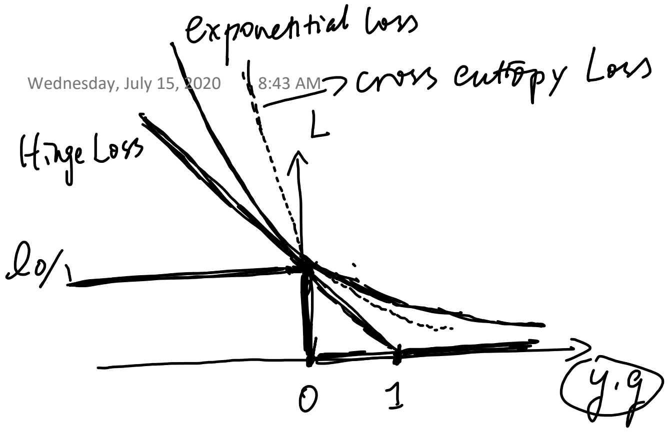
$x = (x_1, x_2)$

$y = (1, -1)$



$\ell_{0/1}(y, g) = \begin{cases} 1 & \text{if } y \cdot g < 0 \\ 0 & \text{otherwise} \end{cases}$

Wednesday, July 15, 2020 8:43 AM



$$L = l_{0/1} = \begin{cases} 0 & \text{if } y \cdot g > 0 \\ 1 & \text{if } y \cdot g \le 0 \end{cases}$$

(1) Hinge Loss

$$L(g, y) = \max(0, 1 - yg)$$

(2) Exponential Loss $\quad L(g, y) = e^{-y \cdot g}$

(3) Cross Entropy Loss

$$L(g, y) = -\left(\mathbb{1}_{y=1} \ln \underbrace{\frac{e^{g}}{e^{g} + e^{-g}}}_{p_1} + \mathbb{1}_{y=-1} \ln \underbrace{\left(\frac{e^{-g}}{e^{g} + e^{-g}}\right)}_{p_{-1}}\right) = -\ln\left(\frac{e^{y g}}{e^{y \cdot g} + e^{-y \cdot g}}\right)$$

$z_1 \, ... \, z_n$

$$\frac{e^{z_j}}{e^{z_1} + \cdots + e^{z_n}}$$

$y = (1, 2, 3, 4)$
$g = (1, 3, 1, 4)$
$y - g$

Estimation (Regression)

$$L(y, g) = |y - g|_2^2 \quad (L^2\text{-Loss})$$

$$|x|_2^2 = x_1^2 + \cdots + x_d^2$$

$$L(y, g) = |y - g|_1 \quad (L^1\text{-Loss})$$

$$|x|_1 = |x_1| + \cdots + |x_d|$$

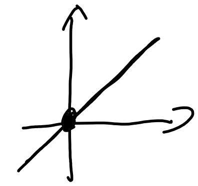
$$L(y, g) = |y - g|_0 \quad (L_0\text{-Loss}) \qquad |x|_0 = \#\{i : x_i \neq 0\}$$

$x = (1, 0, 2, 0, 0)$ $\qquad$ $x = (x_1 \ldots x_d)$

Learning Functions

(1) Linear Regression

$$g(x; w) = \underline{w^T x}$$



$$= w_1 x_1 + \cdots + w_d x_d$$

$$x = (x_1 \cdots x_d)$$

$$w = (w_1 \cdots w_d)$$

$x \longrightarrow y$

$\hookrightarrow g(x; w) \rightarrow \uparrow$

$w^T x$

$$w^T x + \beta$$

$$\left.\begin{array}{l} \tilde{x} = (x, 1) \\ \tilde{w} = (w, \beta) \end{array}\right\} \quad w^T x + \beta = \tilde{w}^T \tilde{x}$$

Least Squares Problem

(x_j, y_j) $\quad \underline{x_j \in \mathbb{R}^d}$ $\quad \underline{y_j \in \mathbb{R}}$

$y = (y_1, \dots y_n)$

$$L(g, y) = \frac{1}{2}(g - y)^2$$

$$g(x; w) = w^T x \qquad w \in \mathbb{R}^d \qquad w^T x = x^T w$$

$$L(g(x_j; w), y_j) = \frac{1}{2}(w^T x_j - y_j)^2$$

<u>ERM</u> $\quad w_n^* = \min_{w} \frac{1}{n} \sum_{j=1}^{n} \frac{1}{2}(\underline{\underline{w^T x_j - y_j}})^2$

Wednesday, July 15, 2020 9:04 AM

$A = (x_1^T \dots x_n^T)$

$(x_1^T w - y_1, \dots, x_n^T w - y_n)$

$= Aw - y$

ERM $\quad w_n^* = \arg\min_w \frac{1}{2n} |Aw - y|_2^2$

Big data $\quad$ "High-dimensional Stat": $d \gg 1$ $\quad |w|_1 = |w_1| + \dots + |w_d|$

L¹-regularization

(LASSO Least Absolute Shrinkage and Selection Operator)

$$\min_w \left[\frac{1}{2n} |Aw - y|_2^2 + \lambda |w|_1 \right]$$

tuning parameter

$\lambda > 0$ $\quad$ penalization

Regularized ERM

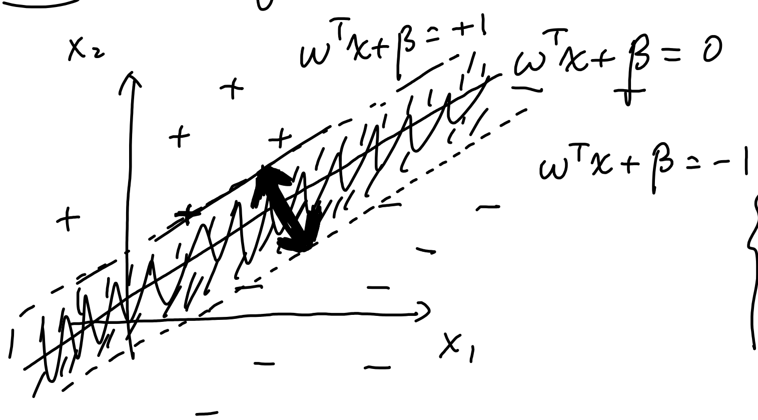
$$w_n^* = \underset{w}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \Big[\underbrace{L(g(x_i, w), y_i) + \lambda R(w)}_{l_i(w)} \Big]$$

$$L_n(w) = \frac{1}{n} \sum_{i=1}^{n} l_i(w)$$

SVM — Support Vector Machines

Neural Network

SVM $x_j \in \mathbb{R}^d$ $y_j = \{1, -1\}$



"hyperplane"

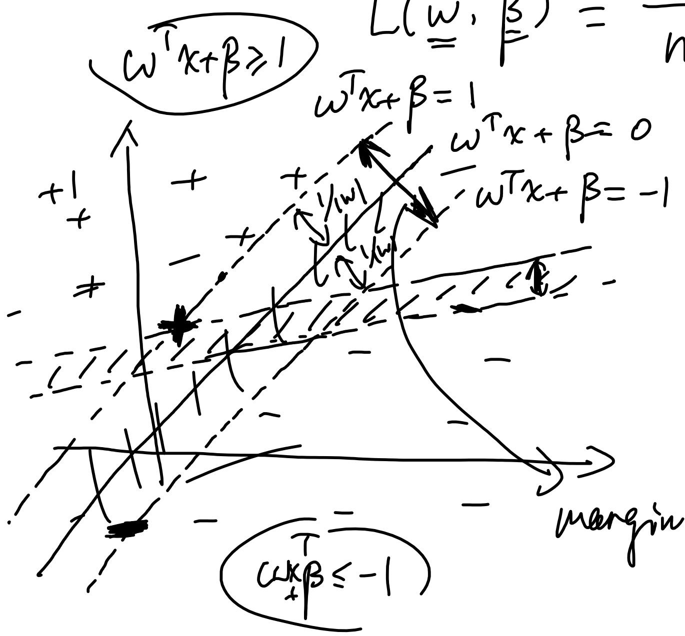
Seek for $\omega \in \mathbb{R}^d$ $\beta \in \mathbb{R}$ s.t.

$$\begin{cases} \omega^T x_j + \beta \geq 1 & \text{if } y_j = 1 \\ \omega^T x_j + \beta \leq -1 & \text{if } y_j = -1 \end{cases}$$

Hinge Loss

$$L(y, \hat{y}) = \max(0, 1 - y\hat{y})$$

$$L(\underline{w}, \beta) = \frac{1}{N} \sum_{i=1}^{n} \max(0, 1 - y_i (x_i^T w + \beta))$$



$x \in \mathbb{R}^d \quad \Pi: w^T x + \beta = 0$

$$\text{dist}(x, \Pi) = \frac{|w^T x + \beta|}{|w|}$$

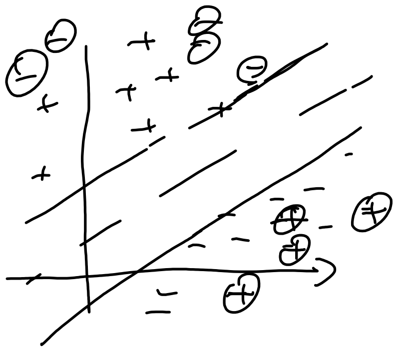
$$\text{width of the strip} = d = \frac{2}{|w|}$$

SVM

$$\max_{w \in \mathbb{R}^d, \beta \in \mathbb{R}} \frac{2}{|w|} \quad \text{s.t.} \quad y_j(w^T x_j + \beta) \geq 1 \quad \text{for } j = 1 \ldots n$$

$$\min_{w \in \mathbb{R}^d, \beta \in \mathbb{R}} \frac{1}{2}|w|^2 \quad \text{s.t.} \quad y_j(w^T x_j + \beta) \geq 1 \quad \text{for } j = 1 \ldots n$$

Wednesday, July 15, 2020 9:36 AM



"Soft margin"

$$\min_{\substack{w \in \mathbb{R}^d \\ \beta \in \mathbb{R}}} \left[\frac{1}{2}|w|^2 + C \underbrace{\sum_{j=1}^{n} \ell_{0/1}\left(y_j(w^T x_j + \beta) - 1\right)} \right]$$

Hinge

$$\max_{\substack{w \in \mathbb{R}^d \\ \beta \in \mathbb{R}}} \frac{1}{2}|w|^2 + C \sum_{j=1}^{n} \xi_j \quad \text{s.t.} \quad \underline{y_j(w^T x_j + \beta)} \geq 1 - \xi_j$$

$$\xi_j \geq 0$$

slack variables

Monday, July 20, 2020 8:27 AM

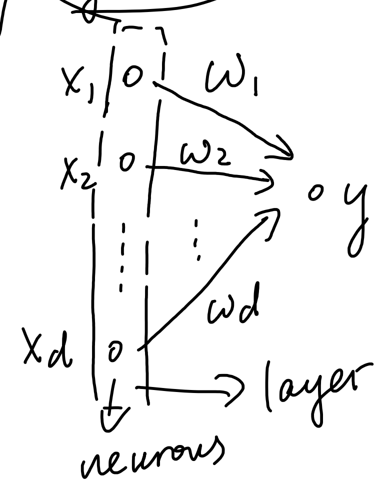
Neural Networks

Least Squares

$$y \approx \underline{\underline{A}}x \Longleftrightarrow \min_{A} \frac{1}{2}|y - Ax|^2$$

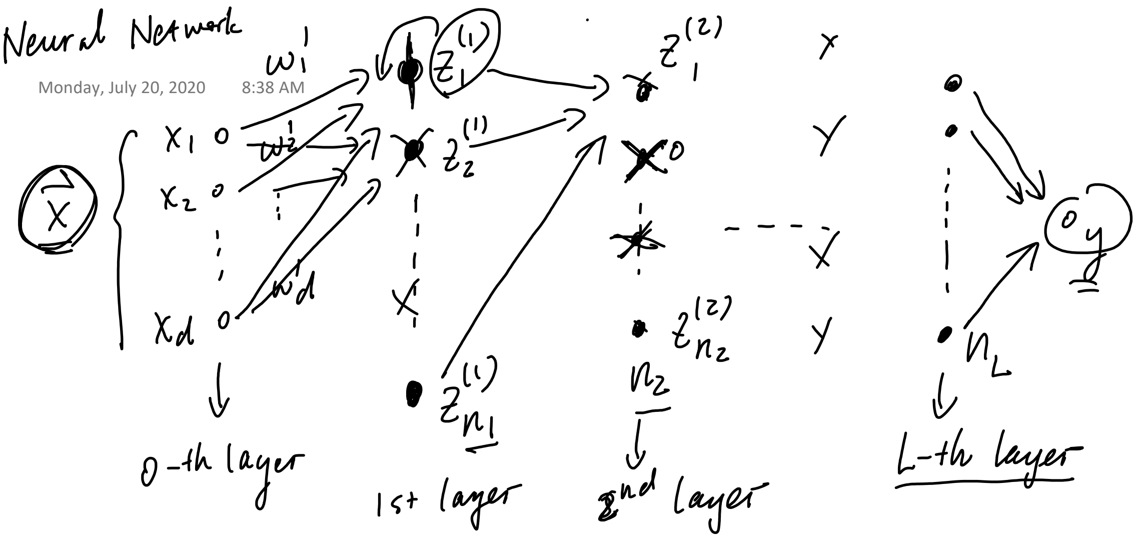
$$y \approx \underline{\underline{w_1}}x_1 + \underline{\underline{w_2}}x_2 + \dots + \underline{\underline{w_d}}x_d = g(x, w)$$

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix} \qquad A = (w_1, \dots, w_d)$$



Neural Network

Monday, July 20, 2020 8:38 AM



$$z^{(1)} = W^{(1)}x + b^{(1)}$$

$$z^{(2)} = W^{(2)}(\sigma(z^{(1)})) + b^{(2)}$$

$$\vdots \quad \sigma$$

$$z^{(L)} = W^{(L)}(\sigma(z^{(L-1)})) + b^{(L)}$$

$$y = a^T z^{(L)} = \langle a, z^{(L)} \rangle$$

$$z_k^{(1)} = w_1^k x_1 + w_2^k x_2 + \dots + w_d^k x_d \qquad k = 1 \dots n_1$$

$$z^{(1)} = \begin{pmatrix} z_1^{(1)} \\ \vdots \\ z_{n_1}^{(1)} \end{pmatrix} = \underbrace{\boxed{\begin{pmatrix} w_1^1 & \cdots & w_d^1 \\ w_1^k & \cdots & w_d^k \\ w_1^{n_1} & \cdots & w_d^{n_1} \end{pmatrix}}}_{n_1 \times d} \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix}$$

$W^{(1)}$ — weight.

$b^{(1)} \in \mathbb{R}^{n_1}$ — bias

$$y = a^T \sigma\left[W^{(L)}\left(\cdots \sigma\left[W^{(3)}\left[\sigma\left(W^{(2)}\left[\sigma\left(W^{(1)}x + b^{(1)}\right)\right] + b^{(2)}\right)\right] + b^{(3)}\right] \cdots \right) + b^{(L)} \right]$$

$$= \underbrace{a^T W^{(L)} W^{(L-1)} \cdots W^{(1)}}_{W} x + \cdots = Wx + b$$

$$\min_{W^{(i)}} \frac{1}{2} \left| y - a^T \hat{W}^{(L)} \hat{W}^{(L-1)} \cdots \hat{W}^{(1)} x \right|^2$$

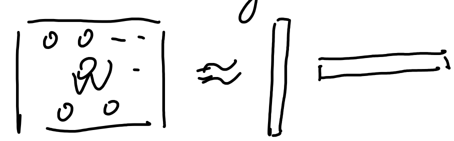
$\downarrow$

adaptive

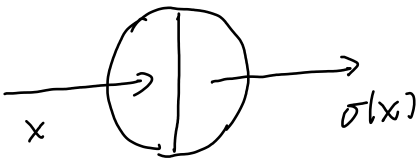
"matrix factorization"

$W = [W_1 \; W_2]$

$x \longrightarrow y$



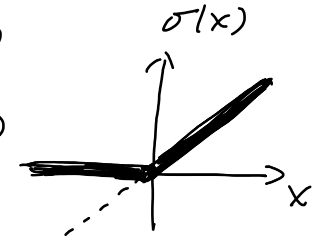
Monday, July 20, 2020 8:58 AM



$$\sigma(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \le 0 \end{cases}$$

$\downarrow$

activation function



ReLU

(Rectified Linear Unit)

$$\sigma(x_1 \dots x_d) = (\sigma(x_1), \dots, \sigma(x_d))$$

tanh

Sigmoid

Neural-Network "Linear" + "Activation"

Monday, July 20, 2020 9:07 AM

I. Goodfellow --- "Deep Learning" MIT

$$y \approx \underbrace{a^T\left(\sigma\left(\underline{\underline{W}}^{(L)}\sigma\left(\underline{\underline{W}}^{(L-1)} \cdots \sigma\left(\underline{\underline{W}}^{(1)}x + b^{(1)}\right) \cdots + b^{(L-1)}\right) + b^{(L)}\right)\right)}_{g(x,\omega)}$$

$$\underset{\triangle}{\omega} = \left(W^{(1)} \cdots W^{(L)}\right.$$
$$\left. b^{(1)} \cdots b^{(L)}\right)$$

$$\min_{\omega} \frac{1}{2}\left(y - g(x; \underline{\omega})\right)^2$$

"Interpretable Machine/Deep Learning"

ReLU $\sigma(x_1, \dots x_d) = (x_1, x_2, 0, x_3 \dots 0 \dots 0\ x_d)$

"adaptive piecewise linear model"

$$\underline{\underline{F(w)}} = \frac{1}{2}\left(y - g(x; w)\right)^2$$

"loss landscape"

$\underline{w_1, w_2}$

Optimization

- convex functions
- Gradient descent, $\underline{\text{neural networks}}$ back-propagation
- accelerated GD (nesterov)
- SGD / VR-SGD
- Practical Issues

$$w^{(0)} \longrightarrow w^{(1)} \longrightarrow w^{(2)} \longrightarrow \cdots \longrightarrow \underline{\underline{w^{(2020)}}} = w.$$