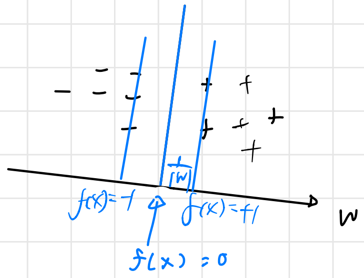
Lecture 14

- SVM:

Linear:

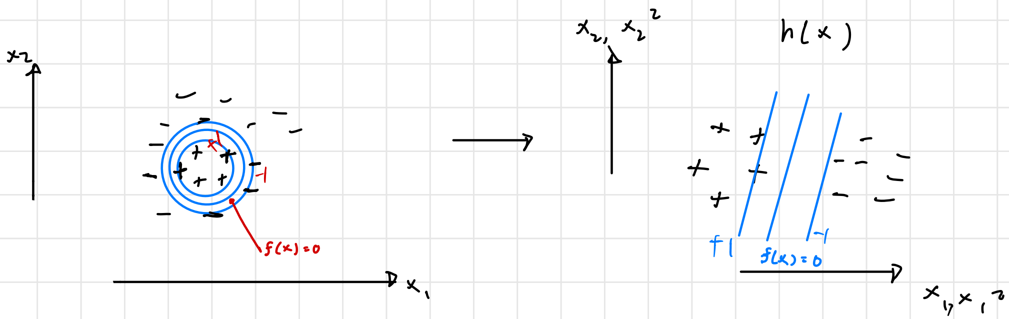
$s = f(x) = \langle x, w \rangle + b$



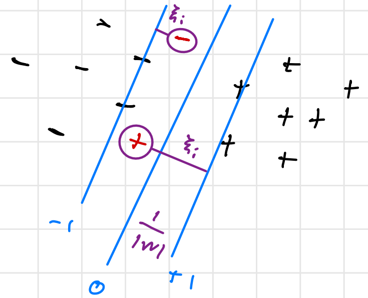
Kernel: $s = f(x) = \langle h(x), w \rangle + b$

$$s = f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b, \quad K(x, x') = \langle h(x), h(x') \rangle$$

Interpolative Memorization & Retrival



- Back to linear, non-seperable



- $\min\limits_{(w,b)} \quad \frac{1}{2}|w|^2 + C\sum_{i=1}^{n} \xi_i$

  subject to $\quad y_i(\langle x_i, w\rangle + b) \geq 1 - \xi_i$

  $\xi_i \geq 0, \quad i = 1, \ldots, n$

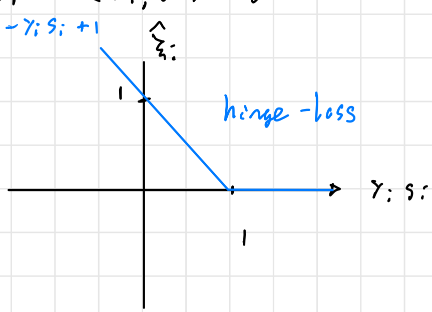
- for each ξ_i, min ξ_i

  if $\quad y_i(\underbrace{\langle x_i, w\rangle + b}_{s_i}) \geq 1 \Rightarrow \hat{\xi}_i = 0$

  if $\quad$ " $\quad$ " $\quad$ " $\quad$ " $\quad < 1 \Rightarrow \hat{\xi}_i = 1 - y_i(\langle x_i, w\rangle + b)$

So $\quad \hat{\xi}_i = \max(0, 1 - y_i s_i)$

$s_i = \langle x_i, w\rangle + b$



- So we only need to minimize:

$$\min_{(w,b)} \frac{1}{2}|w|^2 + C\sum_{i=1}^{n} \max(0, 1 - y_i s_i)$$

- Recall logistic regression

$$\begin{array}{c|c|c} & p & \\ \hline 1 & & \\ \vdots & & \\ i & x_i^T & y_i \in \{-1, +1\} \\ \vdots & & \\ n & & \end{array}$$

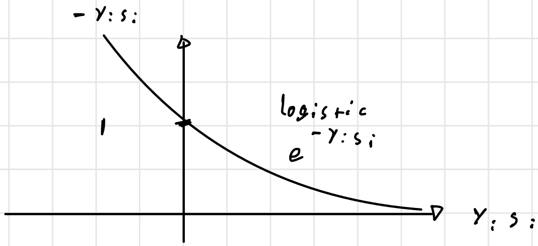
$$\begin{cases} p(y_i = +1 \mid s_i) = \dfrac{e^{s_i}}{1+e^{s_i}} = \dfrac{1}{1+e^{-s_i}} \\ p(y_i = -1 \mid s_i) = \dfrac{1}{1+e^{s_i}} \end{cases}$$

$$p(y_i \mid s_i) = \frac{1}{1+e^{-y_i s_i}}$$

- log-likelihood:

$$\sum_{i=1}^{n} \log(y_i \mid s_i) = -\sum_{i=1}^{n} \log(1+e^{-y_i s_i})$$

$$\text{Logistic Loss} \equiv \sum_{i=1}^{n} \log(1+e^{-y_i s_i})$$



$\log(1+e^{-y_i s_i}) =$

- $y_i s_i \to \infty$: $e^{-y_i s_i}$ exponential loss (adaboost)
- $y_i s_i \to -\infty$: $-y_i s_i$

$\log(1+\delta) \approx \delta$

- Unified : Loss + Regularization

ridge : $\frac{1}{2}|w|_2^2$

Lasso : $|w|_1$

- primal - dual :

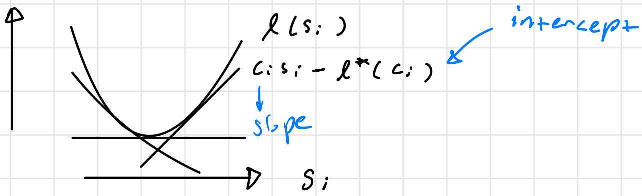
$$\min_{(w,b)} \sum_{i=1}^{n} \ell(s_i) + \frac{\lambda}{2}|w|^2$$

or
$\ell(y_i s_i)$
classification

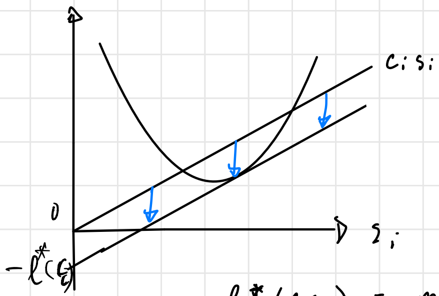
$\min\limits_{(w,b)} \max\limits_{c} \longrightarrow \max\limits_{c} \boxed{\min\limits_{(w,b)}}$

Representer $\longrightarrow$ Kernel Trick

- Legendre Transform

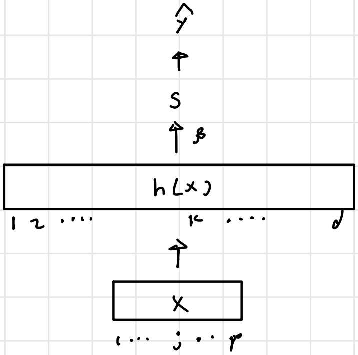


$$\ell(s_i) = \max_{c_i} \left[c_i s_i - \ell^*(c_i) \right]$$



$$\ell^*(c_i) = \max_{s_i} \left(c_i s_i - \ell(s_i) \right)$$

- Part 4: Deep Learning (neural network)
  - Unified Framework for supervised learning:



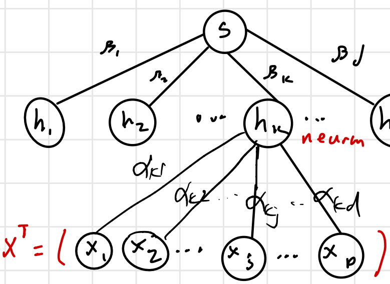
kernel: $K(x, x') = \langle h(x), h(x') \rangle \quad d \to \infty$

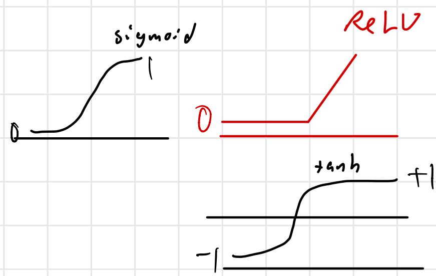
XGB: $h_k(x)$ is a tree

tree: $h(x)$ is one-hot

- features are designed.

- Neural Network:





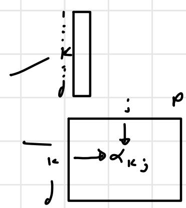
$$\boxed{\begin{aligned} p &= \sigma(s) \\ s &= \sum_{k=1}^{d} \beta_k h_k \end{aligned}}$$

recursively

$$h_k = \sigma(s_k)$$

$$s_k = \sum_{j=1}^{p} \alpha_{kj} x_j = x^T \alpha_k$$

$$\Theta = \begin{cases} \beta_k, & k = 1 \cdots d \\ \alpha_{kj}, & k = 1 \cdots d, \; j = 1 \cdots p \end{cases}$$

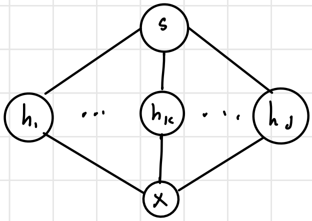


connection weights

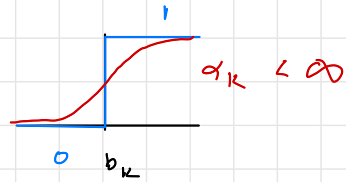
- 1D x

$$s = f(x) = \sum_{k=1}^{d} \beta_k h_k(x)$$

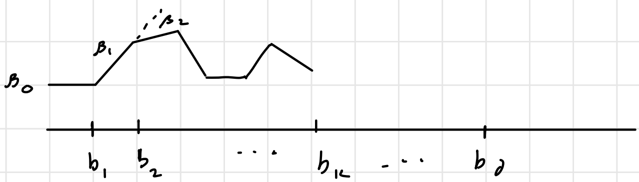
$$= \sum_{k=1}^{d} \beta_k \, \sigma(\alpha_k(x - b_k)) + \beta_0$$



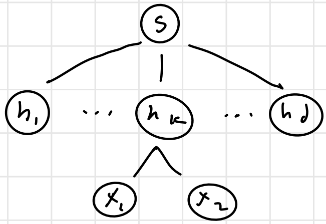
If $\alpha_k \to \infty$



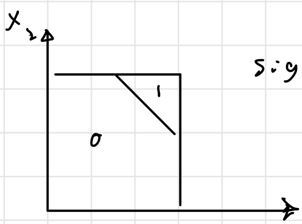
$$s = f(x) = \sum_{k=1}^{d} \beta_k \, \mathrm{ReLU}(x - b_k) + \beta_0$$



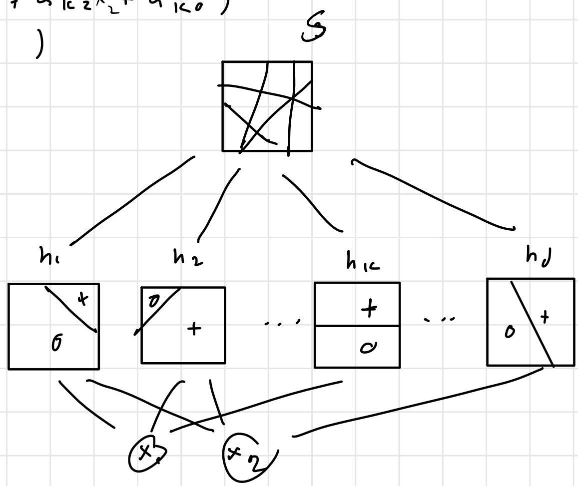
• 2D x :



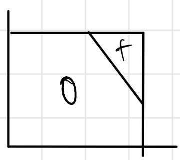
$$h_k = \text{ReLU}(\alpha_{k1}x_1 + \alpha_{k2}x_2 + \alpha_{k0})$$
$$\text{sigmoid}(\; \ldots \;)$$



sigmoid



ReLU : paper folding



- Random initialization:

$$s = f(x) = \frac{1}{\sqrt{d}} \sum_{k=1}^{d} \beta_k \, \sigma(x^T \alpha_k)$$

$$\alpha_k \overset{iid}{\sim} \mathcal{N}(0, I_d)$$

$$\beta_k \overset{iid}{\sim} \mathcal{N}(0, 1) \qquad \beta_k \perp \alpha_k$$

$$f(x) \xrightarrow[\infty]{d} GP$$ (central limit Theorem)

- Gradient Learning

$$s = f_\theta(x) \quad , \quad \theta = (\alpha, \beta)$$

$$\text{Loss} = \frac{1}{2n} \sum_{i=1}^{n} (y_i - f_\theta(x_i))^2$$

$$\theta_{t+1} = \theta_t + \eta_t \underbrace{\frac{1}{n} \sum_{i=1}^{n} (\underbrace{y_i - f_{\theta_t}(x_i)}_{\text{error}}) \frac{\partial f_{\theta_t}(x_i)}{\partial \theta}}_{\Delta\theta}$$

- $f_{\theta_{t+1}}(\overset{\text{query}}{x}) = f_{\theta_t}(x) + \left\langle \frac{\partial_{\theta_t} f(x)}{\partial \theta}, \Delta\theta \right\rangle$

$$= f_{\theta_t}(x) + \left\langle \frac{\partial f_{\theta_t}(x)}{\partial \theta}, \eta_t \frac{1}{n} \sum_{i=1}^{n} \left(y_i - f_{\theta_t}(x_i) \right) \overset{\text{ker}}{\frac{\partial_{\theta_t} f(x_i)}{\partial \theta}} \right\rangle$$

$$= f_{\theta_t}(x) + \eta_t \frac{1}{n} \sum_{i=1}^{n} \left(y_i - f_{\theta_t}(x_i) \right) \left\langle \overset{\text{ker}}{\frac{\partial f_{\theta_t}(x_i)}{\partial \theta}}, \overset{\text{query}}{\frac{\partial_{\theta_t}(x)}{\partial \theta}} \right\rangle$$

$$\Downarrow$$

Neural Tangent $K(x_i, x)$ → learned as opposed to designed in classical kernel machines.