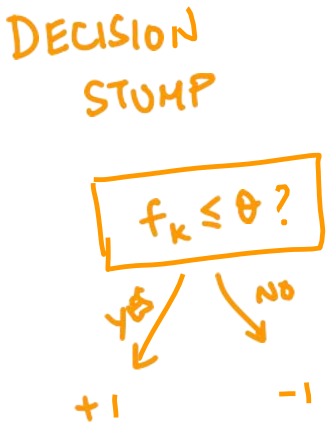
META CLASSIFIERS (or)

ENSEMBLE CLASSIFIERS.

WEAK CLASSIFIERS → STRONG CLASSIFIERS.

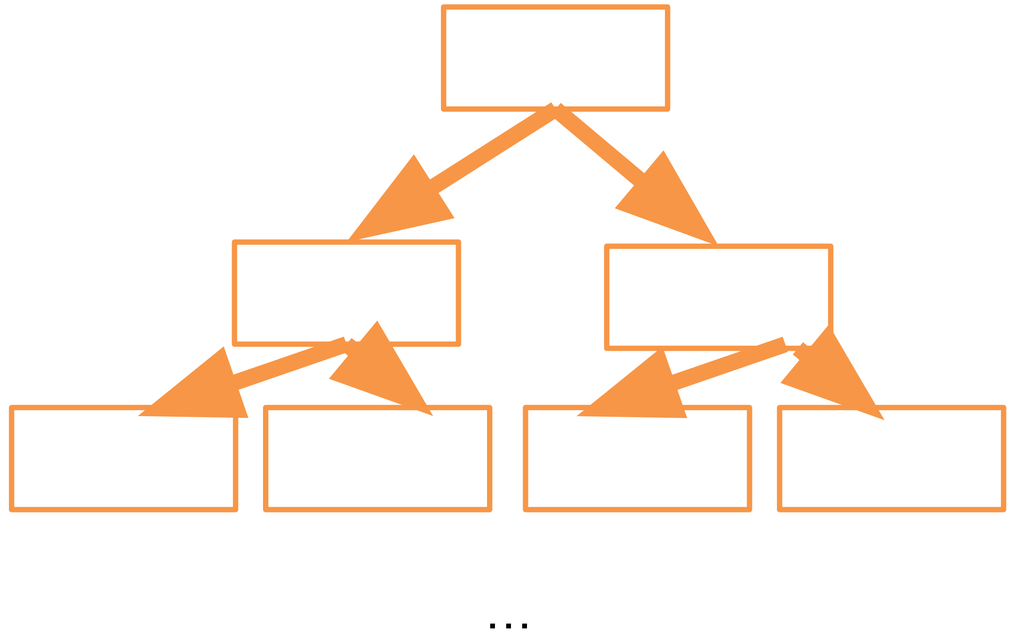
[better than random]

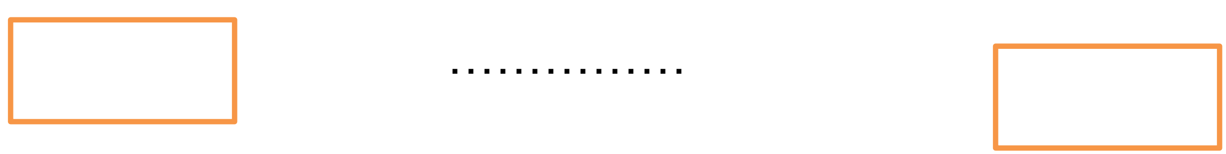
# Weak classifiers




high bias, low variance

Overfit decision tree





low bias, high variance

$$X_1, X_2, \dots X_n \sim N(\mu, 1)$$

$$\hat{\mu}_1 = X_1 \qquad \hat{\mu}_2 = X_2, \quad \dots, \quad \hat{\mu}_N = X_n \qquad \hat{\mu}_{ML} = \frac{1}{n}\sum X_i$$

$$\begin{tikzcd}
\boxed{D_1} \arrow[d] & \boxed{D_2} & \cdots & \boxed{D_K} \\
\boxed{\phantom{D}} \arrow[d] & \boxed{\phantom{D}} \arrow[d] & \cdots & \boxed{\phantom{D}} \arrow[d] \\
h_1 & h_2 & & h_k
\end{tikzcd}$$

Overfit decision trees

$$h_i : \mathbb{R}^d \to \{\pm 1\}$$

$$h^*(x) = \text{majority}(h_1(x), \dots, h_k(x))$$

BAGGING - Bootstrap Aggregation.

$D = \{(x_1, y_1) \dots (x_n, y_n)\}$

Chance that a point appears in a dataset

$1 - \underbrace{\left(1-\frac{1}{n}\right)\left(1-\frac{1}{n}\right)\cdots\left(1-\frac{1}{n}\right)}$

$1 - \underbrace{\left(1-\frac{1}{n}\right)^n}$

$1 - \frac{1}{e}$ (as $n \to \infty$)

$\approx 66\%$

- Create datasets $D_1, \dots, D_k$ from $D$ by "Sampling with replacement".
- Run weak classifier on $D_1, \dots, D_k$ to get $h_1, \dots, h_k$
- Aggregate $h_1, \dots, h_k$ using majority.

FEATURE BAGGING → Bag the features in addition to datapoints

Feature bagged decision trees -> RANDOM FOREST

BOOTSTRAP - Sampling with Replacement ?

AGGREGATION - Majority. ?

BOOSTING

[Freund & Schapire. 1995 Godel Prize]

ADA-BOOST

Distribution $D$ over $(\mathcal{X} \times \mathcal{Y})$ $\mathcal{X}$: $\mathbb{R}^d$, $\mathcal{Y}$: $\{+1,-1\}$

unknown but fixed.

$x_1, \ldots, x_n$ are iid from $D$.

$h: \underbrace{\mathbb{R}^d}_{\mathcal{X}} \rightarrow \underbrace{\{\pm 1\}}_{\mathcal{Y}}$.

Measure performance using

$$P_{(x,y)\sim D}\left( h(x) \neq y \right)$$

Misclassification probability.

A weak learner is one which outputs a classifier

Strong

$h$ for which

$$P_{x,y\sim D}\left( h(x) = y \right) \geq \frac{1}{2} + \gamma \quad 1-\epsilon$$

$\boxed{\gamma > 0}$

for any unknown but fixed distribution $D$.

# BOOSTING

Weak learner $\rightarrow$ Strong learner.

$\{(x_1, y_1), \ldots (x_n, y_n)\} \rightarrow$ $\boxed{\text{weak learner}}$ $\rightarrow h : \mathbb{R}^d \rightarrow \{\pm 1\}$

$p_1, \ldots p_n$

$$\boxed{\sum_i p_i = 1}$$

$$\boxed{\sum_{i=1}^{n} p_i \mathbb{1}(h(x_i) \neq y_i) \leq \frac{1}{2} - \gamma} \qquad \boxed{\gamma > 0}.$$

Strong learner classifies all data points correctly.

# BOOSTING [ADABOOST]

$$S = \{ (x_1, y_1) \cdots\cdots (x_n, y_n) \}$$

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

$y_i \in \{\pm 1\}$.

Initialize $D_1(i) = \frac{1}{n} \quad \forall i$

iteration

for $t = 1, \ldots, T$

$h_t$ = [Input $\boxed{S, D_t}$ to the weak learner to get $h_t$]

$$D_{t+1}(i) = \frac{D_t(i) \cdot e^{-\alpha_t y_i h_t(x_i)}}{\sum_{j=1}^{n} D_t(j)\, e^{-\alpha_t y_j h_t(x_j)}} \quad \forall i$$

$$\alpha_t > 0 \quad \forall t.$$

end.

$$\underline{H(x)} = \sum_{t=1}^{T} \underline{\alpha_t}\, \boxed{h_t(x)}$$

$$\boxed{C(x) = \text{sign}(H(x))}$$

# Analysis of Boosting

$$D_{t+1}(i) = \frac{D_t(i)\, e^{-\alpha_t y_i h_t(x_i)}}{Z_t} \quad \rightarrow \quad \text{update rule.}$$

$$Z_t = \sum_{j=1}^{n} D_t(j)\, e^{-\alpha_t y_j h_t(x_j)}$$

$$D_{t+1}(i) = \left( \frac{D_{t-1}(i)\, e^{-\alpha_{t-1} y_i h_{t-1}(x_i)}}{Z_{t-1}} \right) \cdot \frac{e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$$

$$D_{t+1}(i) = \frac{\frac{1}{n} e^{-\sum_{l=1}^{t} \alpha_l \underline{y_i} h_l(x_i)}}{\prod_{l=1}^{t} Z_l}$$

$$D_{T+1}(i) = \frac{\frac{1}{n} e^{-y_i H(x_i)}}{\prod_{l=1}^{T} Z_l}$$

$$H(x_i) \longrightarrow \sum_{l=1}^{T} \alpha_l h_l(x_i)$$

$$\underbrace{\sum_{i=1}^{n} D_{t+1}(i)}_{=1} = \frac{\sum_{i=1}^{n} \frac{1}{n} e^{-y_i H(x_i)}}{\prod_{l=1}^{T} Z_l}$$

$$\boxed{\prod_{l=1}^{T} Z_l = \frac{1}{n} \sum_{i=1}^{n} e^{-y_i H(x_i)}} \quad \text{— (A)}$$

Recall, $c(x) = \text{sign}(H(x))$

$$\mathbb{1}(c(x_i) \neq y_i) = \mathbb{1}(H(x_i) y_i < 0) \leq e^{-H(x_i) y_i} \quad \text{— (B)}$$

$$\Rightarrow \prod_{l=1}^{T} Z_l = \frac{1}{n} \sum_{i=1}^{n} e^{-y_i H(x_i)} \quad (\text{from } A)$$

$$\geq \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\left( H(x_i)\, y_i < 0 \right) \quad (\text{from } B)$$

$$\geq \frac{1}{n} \underbrace{\sum_{i=1}^{n} \mathbb{1}\left( c(x) \neq y_i \right)}_{\text{error}(c)}$$

$$\boxed{\text{error}(c) \leq \prod_{l=1}^{T} Z_l} \leq \square$$

$$\prod_{l=1}^{T} Z_l = \prod_{l=1}^{T} \left[ \sum_{j=1}^{n} D_l(j)\, e^{-\alpha_l y_j h_l(x_j)} \right]$$

$$= \prod_{l=1}^{T} \left[ \sum_{j=1}^{n} e^{-\alpha_l} D_l(j)\, \mathbb{1}\left(y_j = h_l(x_j)\right) + \sum_{j=1}^{n} e^{\alpha_l} D_l(j)\, \mathbb{1}\left(y_j \neq h_l(x_j)\right) \right]$$

$$= \prod_{l=1}^{T} e^{-\alpha_l} \left( 1 - \text{error}(h_l) \right) + e^{\alpha_l} \left( \text{error}(h_l) \right)$$

Choose $\alpha_\ell$ s.t

$$e^{-\alpha_\ell}(1-\text{error}(h_\ell)) + e^{\alpha_\ell}(\text{error}(h_\ell))$$ is

as small as possible.

$$\boxed{\alpha_\ell^* = \ln\sqrt{\frac{1-\text{er}(h_\ell)}{\text{er}(h_\ell)}}}$$

↑

$$= \prod_{\ell=1}^{T} 2\underbrace{\sqrt{\text{error}(h_\ell)(1-\text{error}(h_\ell))}}$$

$$\leq \prod_{\ell=1}^{T} 2\sqrt{\left(\tfrac{1}{2}-\gamma\right)\left(\tfrac{1}{2}+\gamma\right)}$$

[show this]

$$= \prod_{\ell=1}^{T} 2\sqrt{\frac{1-4\gamma^2}{4}}$$

$$e^{-\alpha_\ell}(1-\theta) + e^{\alpha_\ell}\theta$$

$$(1-\theta)e^{-\alpha_\ell}(-1) + e^{\alpha_\ell}\theta = 0$$

$$-1 + \theta e^{-\alpha_\ell} + \theta e^{\alpha_\ell} = 0$$

$$= \prod_{l=1}^{T} \sqrt{\underbrace{1-4\gamma^2}}$$

$$\leq \prod_{l=1}^{T} \left(e^{-4\gamma^2}\right)^{1/2} = \prod_{l=1}^{T} e^{-2\gamma^2}$$

$$= e^{-2T\gamma^2} \quad \text{--- } (2)$$

$$\frac{1}{n} \sum_i \mathbb{1}(c(x_i) \neq y_i)$$

$$\downarrow$$

$$\text{error}(c) \leq \prod_{l=1}^{T} z_l \leq \underbrace{e^{-2T\gamma^2}}_{} \uparrow \leq \frac{1}{2n}$$

if

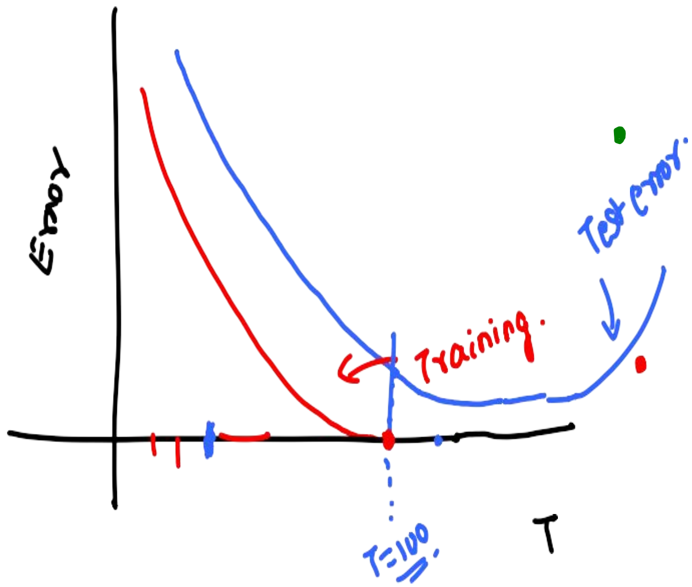
$$\boxed{T \geq \frac{1}{2\gamma^2} \ln(2n)}$$

$$\Rightarrow \text{error}(c) = 0$$

$$\Downarrow$$

strong learner!

# Boosting- Summary




- Every bag depends on previous bag's performance
- Next bag "focusses" on previous bag's mistakes
- Can show training error goes to 0 as number of rounds increases

Cannot run in parallel – hence sometimes bagging is preferred in practice

Usually weak learners are decision stumps -> high bias, low variance
Boosting reduces bias without affecting variance a lot

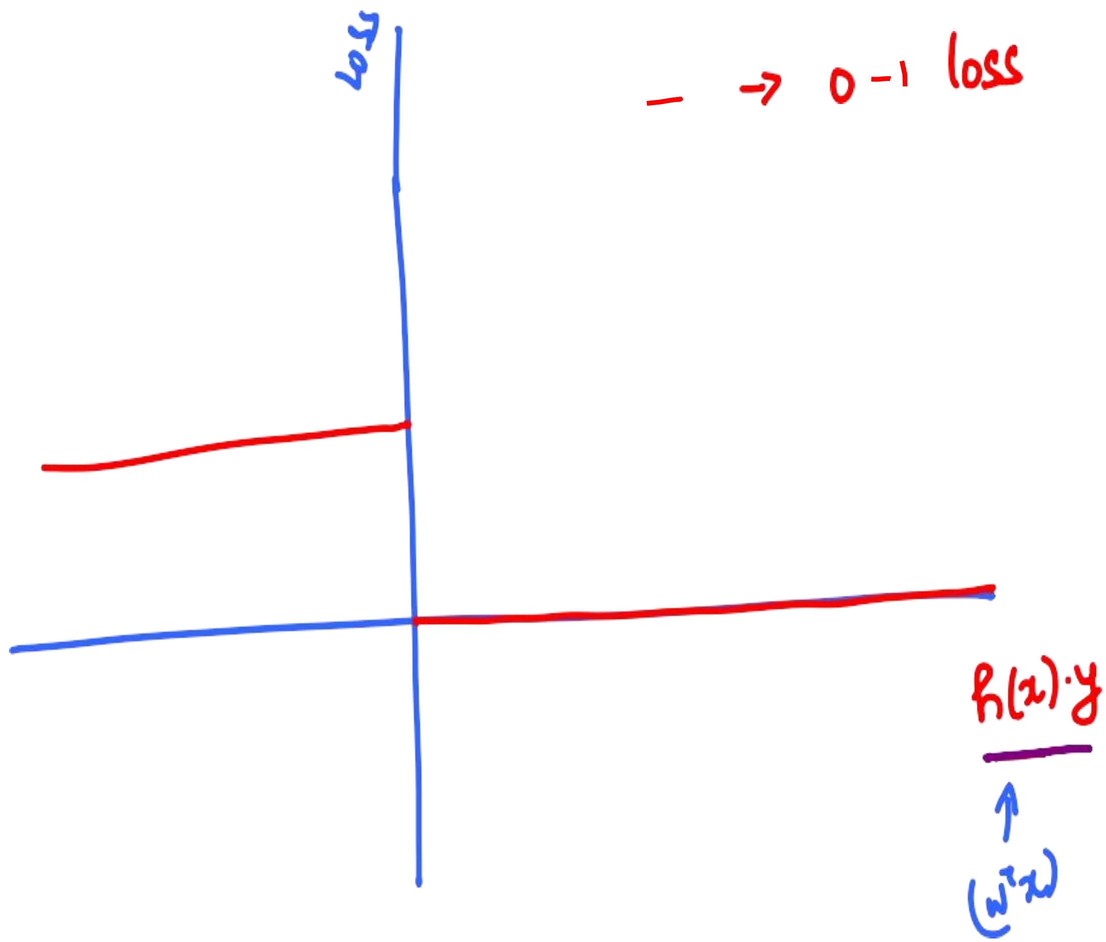
Why are there so many binary classification algorithms?

$x \in \mathbb{R}^d$

$y \in \{\pm 1\}$

$$\min_{w} \sum_{i=1}^{n} \mathbb{1}\left( w^T x \cdot y < 0 \right)$$

NP-Hard problem



Linear Regression for classification

$$\min_{h} \sum_{i=1}^{n} (h(x) - y_i)^2$$

Final classifier: $\text{sign}(h(x))$

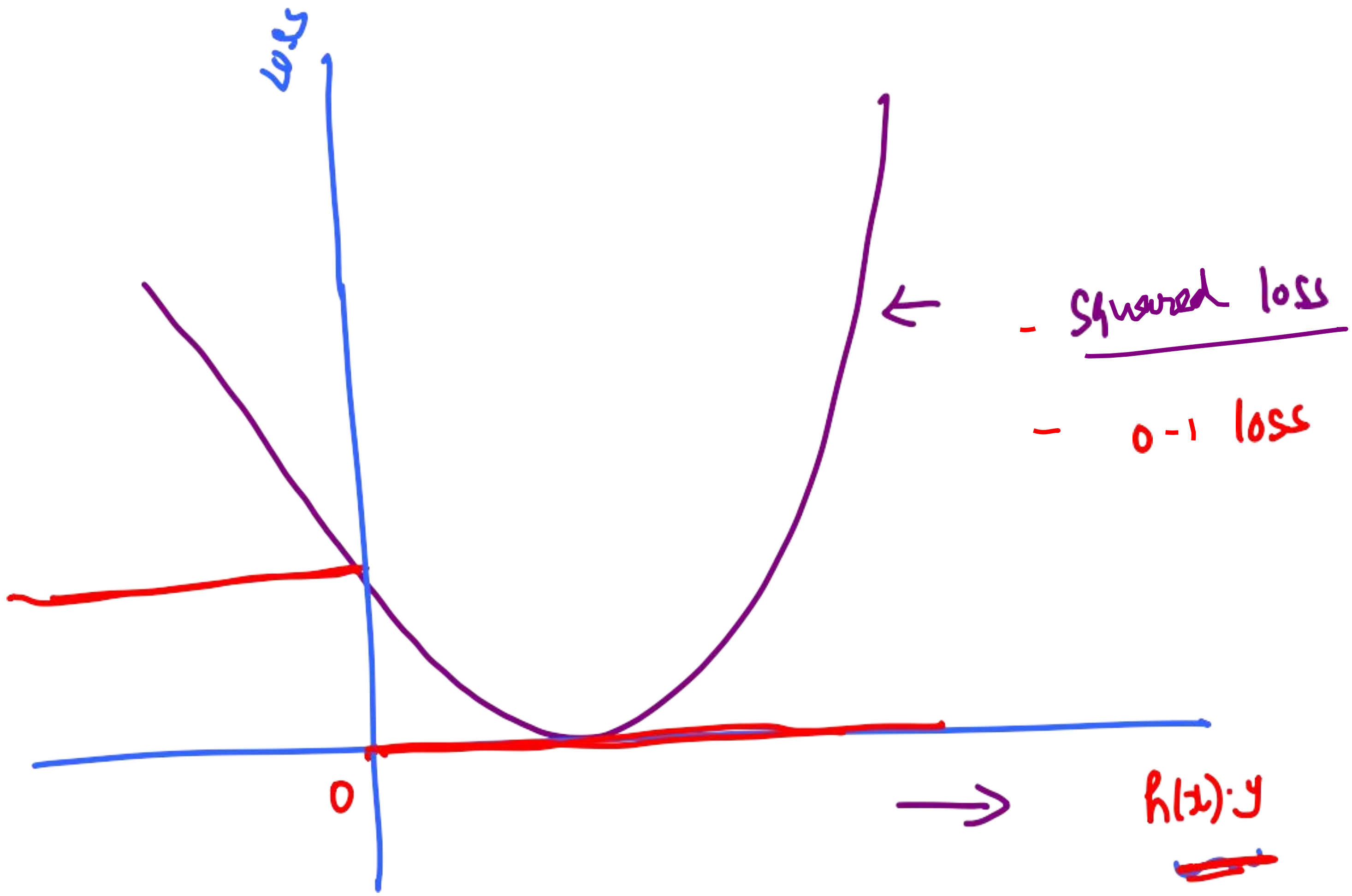
Loss per point for h

$$(h(x) - y)^2 = \begin{cases} (h(x) - 1)^2 & \text{if } y = 1 \\ (h(x) + 1)^2 & \text{if } y = -1 \end{cases}$$

If $y = 1$ $\quad (h(x))^2 + 1 - 2h(x) = (h(x))^2 + 1 - 2h(x)y$

$y = -1$ $\quad (h(x))^2 + 1 + 2h(x) = (h(x))^2 + 1 - 2h(x)\cdot y.$

$$= (h(x)\cdot y - 1)^2$$


loss
- squared loss
- 0-1 loss
0
h(x)·y

# SUPPORT VECTOR MACHINES

$$\min_{w,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \underline{\xi_i}$$

$$\underline{w^T x_i y_i + \xi_i \geq} 1 \quad \forall i$$

$$\xi_i \geq 0. \quad \forall i$$

---

Equivalently

$$\xi_i \geq 1 - w^T x_i y_i \quad \forall i$$

$$\xi_i \geq 0 \quad \forall i$$

---

Equivalently $\xi_i \geq \max\left(1 - w^T x_i y_i, 0\right)$

$$\min_{w,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i$$

$$\xi_i \geq \max(1 - w^T x_i y_i, 0)$$

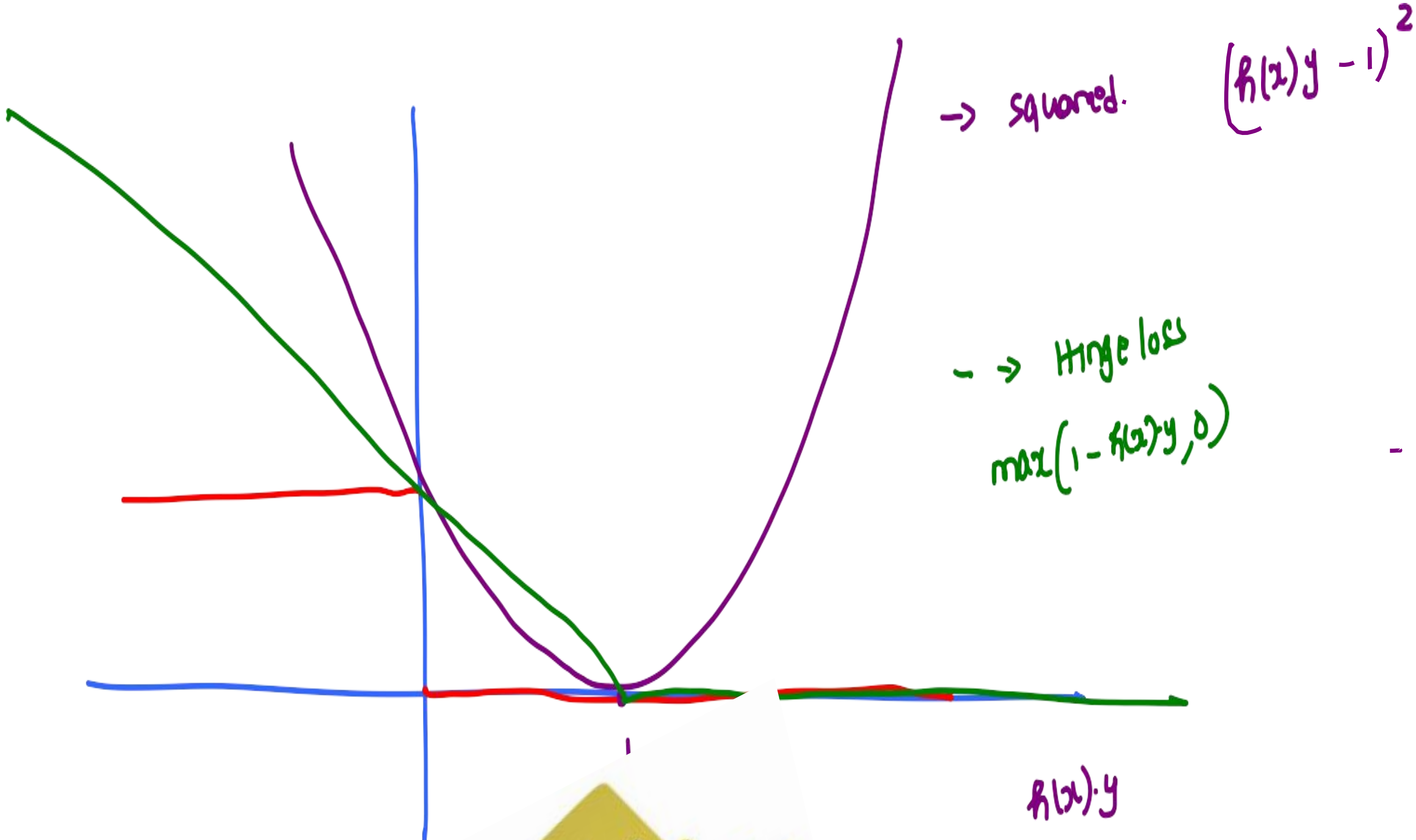
Equivalently,

$$\min_{w,\xi} \quad \underbrace{\frac{1}{2}\|w\|^2}_{\text{Model Regularization term}} + \underbrace{C\sum_{i=1}^{n} \max(1 - w^T x_i y_i, 0)}_{\text{Data Loss term}}$$

So what exactly is the loss?

$$\ell(w, (x,y)) = \max(1 - w^T x y, 0)$$

**HINGE LOSS**




Hinge

Courtesy: Google images

# Logistic regression

$$\max_{w} \prod_{i=1}^{n} \underline{\left(\underline{g(w^T x_i)}\right)^{z_i} \left(1 - \underline{g(w^T x_i)}\right)^{(1-z_i)}}$$

$z_i = 1$ if $y_i = 1$

$z_i = 0$ if $y_i = -1$

$$g(\theta) = \frac{1}{1 + e^{-\theta}}$$

$$\max_{w} \prod_{i=1}^{n} \underline{\left(g(\underline{w^T x_i})\right)^{z_i} \left(1 - \underline{g(w^T x_i)}\right)^{(1-z_i)}}$$

$$\max_{w} \sum_{i=1}^{n} z_i \log\left(g(w^T x_i)\right) + (1-z_i) \log\left(1 - g(w^T x_i)\right)$$

$$= \min_{W} \sum_{i=1}^{n} \left[ -z_i \log\left(g(w^T x_i)\right) + (z_i - 1) \log\left(1 - g(w^T x_i)\right) \right]$$

Loss for a single point when $z_i = 1$ $(y_i = 1)$

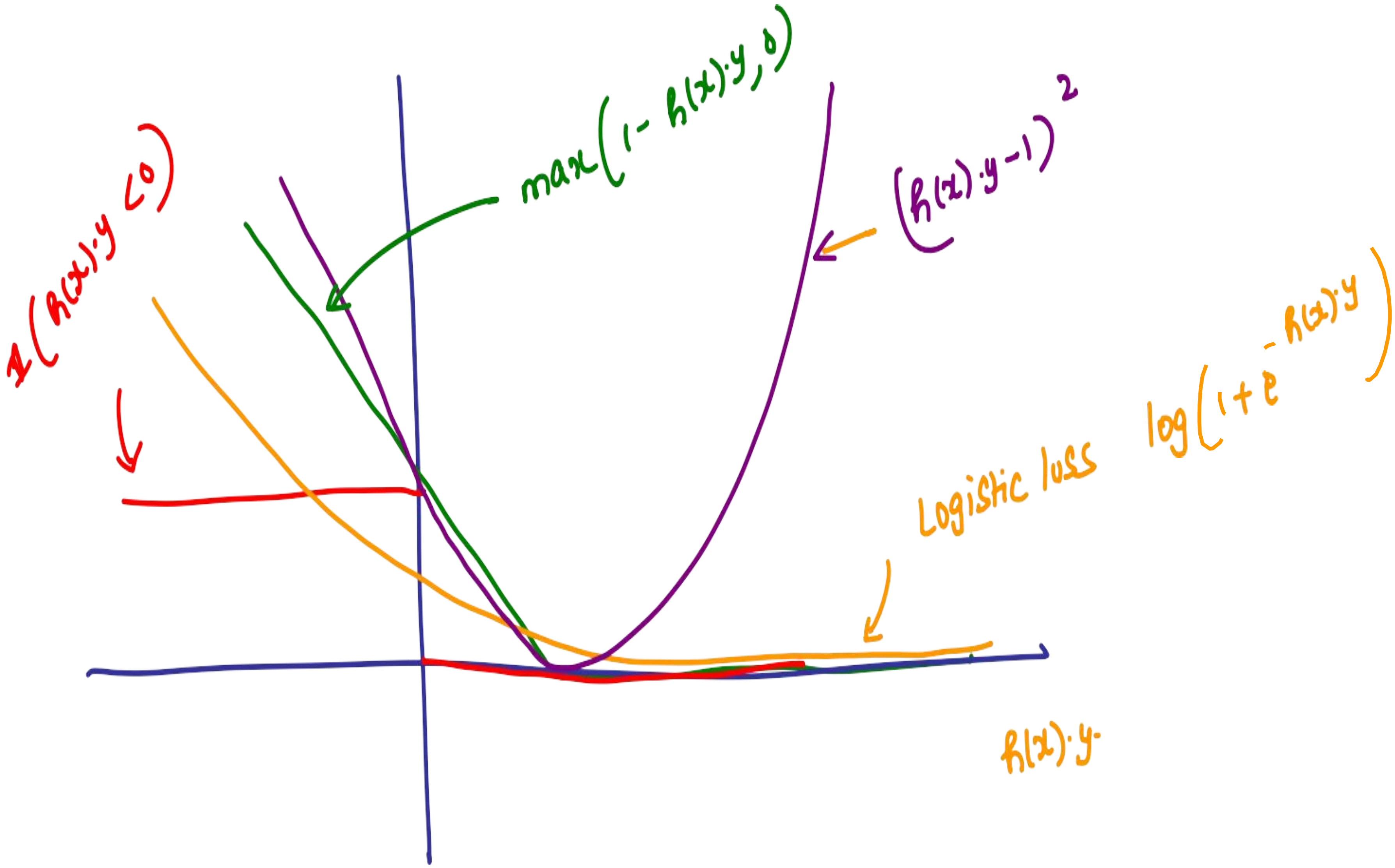
$$-\log(g(w^T x_i)) = -\log\left(\frac{1}{1+e^{-w^T x_i}}\right)$$

$$= \log\left(1+e^{-w^T x_i}\right) = \boxed{\log\left(1+e^{-y_i w^T x_i}\right)}$$

Loss for a single point when $z_i = 0$ $(y_i = -1)$

$$= -\log(1-g(w^T x_i)) = -\log\left(1-\frac{1}{1+e^{-w^T x_i}}\right)$$

$$= -\log\left(\frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}\right) = \log\left(1+e^{w^T x_i}\right) = \boxed{\log\left(1+e^{-y_i w^T x_i}\right)}$$



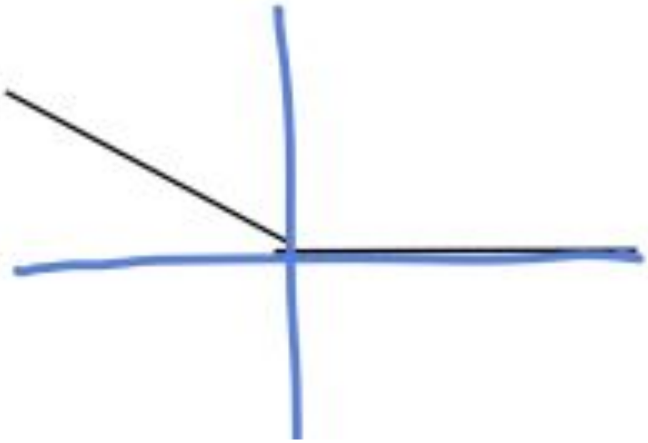
1(h(x)·y <0)
max(1- h(x)·y, 0)
(h(x)·y -1)^2
Logistic loss log(1+e^-h(x)·y)
h(x)·y

## Perceptron

$$w_{t+1} = w_t + x_t y_t$$

consider the hinge loss

$$\mathrm{loss}(w, (x, y)) = \max(0, 1 - w^T x y)$$



$$loss(w,(x,y)) = \max(0,\ 1 - w^T x y)$$

$$\frac{\partial loss}{\partial w} = \begin{cases} -x \cdot y \text{ if } (w^T x)y < 0 \\ = 0 \quad \text{if } (w^T x)y > 0 \\ = \underbrace{[-1, 0]}\ x \cdot y \quad \text{if } (w^T x)y = 0 \end{cases}$$

sub-gradient

↳ choose $-xy$ if mistake.

When mistake

$$w_{t+1} = w_t - \underbrace{\eta_t}_{=1}(-x_t y_t)$$

- Perceptron can be viewed as S.G.D with Hinge loss with $\underset{=1}{\text{step size}}$

# BOOSTING

Recall

$$\prod_{t=1}^{T} Z_t = \frac{1}{n}\sum_{i=1}^{n} e^{-\left(\sum_{t=1}^{T}\alpha_t h_t(x_i)\right)y_i}$$

$$:= \frac{1}{n}\sum_{i=1}^{n} e^{-H_T(x_i)\,y_i}$$

at round T-1, we so far have

$$\sum_{t=1}^{T-1}\alpha_t h_t(x)$$

We need to add one more

classifier in round T to get

$$\sum_{i=1}^{T-1} \alpha_t h_t(x) + \underline{\alpha_T}\ \underline{h_T(x)}$$

greedy choice
to minimize

$$\frac{1}{n}\sum_{i=1}^{n} e^{-\left[\sum_{i=1}^{T-1}\alpha_t h_t(x_i) + \alpha_T h_T(x)\right] y_i}$$

Define $\text{loss}(h, x, y) = e^{-y h(x)}$

- At every round, one chooses $\alpha_t$ to minimize $Z_t$.
- Thus Ada-boost can be viewed as "greedy / co-ordinate descent" on exponential loss

## What about Peceptron and Boosting?

- one can argue perceptron update rule is equivalent to doing a SSGD on hinge loss with stepsize = 1

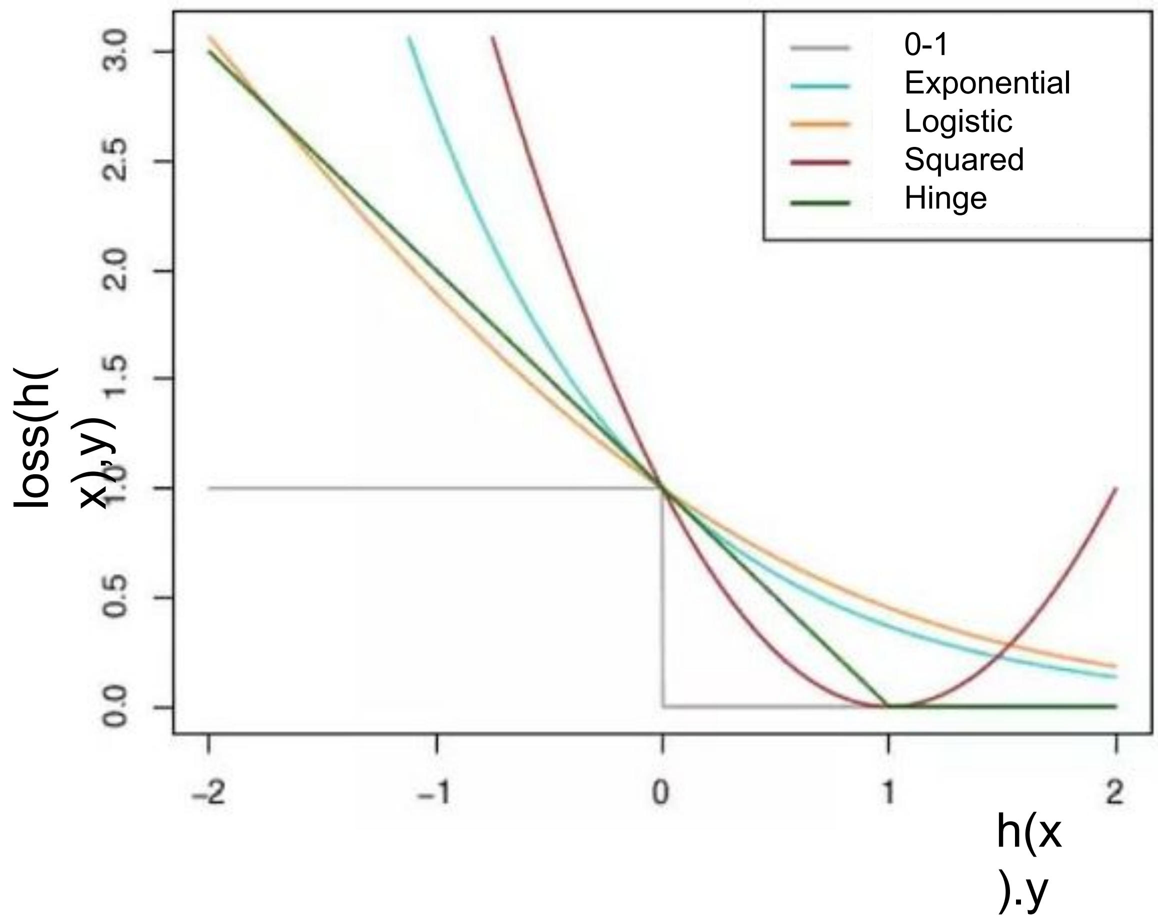
SSGD → Stochastic sub-gradient descent.

$$W_{t+1} = W_t + \underline{x_i y_i}$$

- Boosting (Adaboost) can be viewed as "greedy / co-ordinate wise" descent of exponential loss

$e^{-h(x)\cdot y}$

# SUMMARY




-> The 0-1 loss is NP-hard to optimize even for linear classifiers

-> Different algorithms get around this by using a "surrogate" loss function

-> Surrogates are usually ***"convex" surrogates*** that are easy to optimize