Loss Function

Regression:

① Squared loss

$$L(f) = (Y - f(x))^2$$

r

- inner product
- differentiable
- not robust

② absolute loss

$$L(f) = |Y - f(x)|$$

r

- not differentiable
- more robust

$$r = Y - f(x) \leftarrow$$

Classification: (Binary)

Two ways of parameterizing binary classification

① $Y \in \{0, 1\}$ $\xrightarrow{(Y+1)/2}$ ② $Y \in \{-1, 1\}$

$2Y - 1$

① 0-1 Loss

$$L(f) = \mathbb{1}(Y \neq f(x)) = \begin{cases} 0 & Y = f(x) \\ 1 & Y \neq f(x) \end{cases}$$

aside: $Y \in \{-1, 1\}$ and $f(x) \in \{-1, 1\}$

signs match = correct classification
signs don't match = incorrect classification

for any f there is some h so that

$$f(x) = \text{sign}(h(x))$$

real-valued function h,

$$\text{sign}(a) = \begin{cases} 1 & a > 0 \\ -1 & a < 0 \\ 0 & a = 0 \end{cases}$$

$\rightarrow h(x) \gg 0$ if $Y = 1$

$\rightarrow h(x) \ll 0$ if $Y = -1$

margin: $Yh(x) > 0$ = correct classification

< 0 = incorrect classification

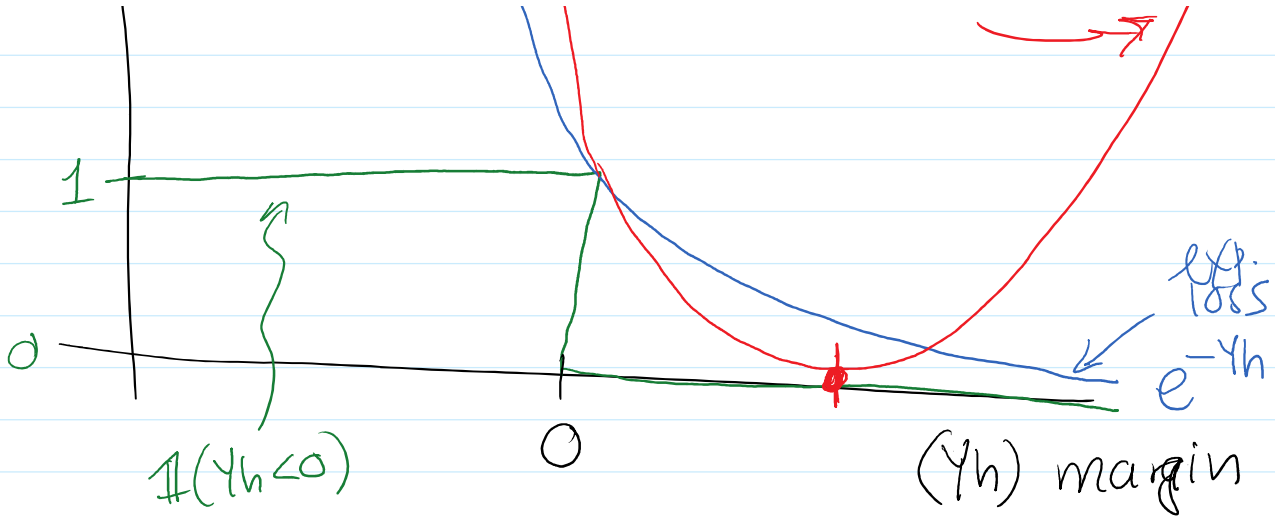
$|Yh(x)|$ = in/correct amount

Similar to a residual in the regression case

We can write loss functions as a fn of margin.

0-1 Loss $L(f) = \mathbb{1}(Yh(x) < 0)$

$= \mathbb{1}(f(x) \neq Y)$



(2) Exponential Loss

$$L(f) = \exp(-Yh(x))$$

Boosted Methods: orginally motivated as an ensemble method, consensus method

Idea: Combine a series of weak methods to form a stroger one.

Weak Classifier — Classification Setting (Binary)

slightly better than random guessing

Boosting: (1) sequentially learn weak class. to a series of modified trainig data

$\hat{f}_1, \hat{f}_2, \hat{f}_3, \hat{f}_4, \ldots$

→ focus train on examples that were misclass in prev. iter.

$t_1, t_2, t_3, t_4, \ldots$ examples that were misclass in prev. iter.

(2) $\hat{f}(x) = \text{sign}\left(\sum_{b=1}^{B} \alpha_b \hat{f}_b(x)\right)$

weighted vote

weights: higher for a good classifier, lower for a bad classifier

---

Alg: Ada Boost

(1) $w_n = 1/N$ for each train example (x_n, y_n)

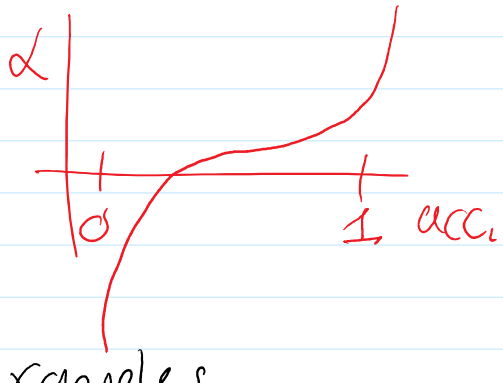
(2) For $b = 1, \ldots, B$

(a) fit $\hat{f}_b$ on weighted (w_n) train data — weak method "a stump" — 1 split class tree

(b) Compute err for $\hat{f}_b$

e_b = weighted misclass error

(c) $\alpha_b = \log\left(\frac{1-e_b}{e_b}\right)$



(d) re-weight train examples

(d) re-weight train examples

$w_n \leftarrow w_n$ if I classified correct

or $w_n \leftarrow e^{\alpha_b} w_n$ if " incorrect

(3) $$\hat{f}(x) = \text{Sign}\left(\sum_{b=1}^{B} \alpha_b \hat{f}_b(x)\right)$$

What is boosting (really) doing?

Why does it work?

Boosted method: additive model

$$\hat{f}(x) = \text{Sign}\left(\underbrace{\sum_{b=1}^{B} \alpha_b \hat{f}_b(x)}_{h(x)}\right)$$

$$h(x) = \sum_{b=1}^{B} \alpha_b \hat{f}_b(x)$$

Generalized additive method:

$$h(x) = \sum_{b=1}^{B} \alpha_b \beta_b(x; \gamma_b)$$

γ_b = parameters for basis fn

β = basis function

$b = 1$ ... B = basis function

Ideally: find h by minimizing

$$\min_{\{\alpha_b, \gamma_b\}_{b=1}^{B}} L\left(\sum_{b=1}^{B} \alpha_b \beta_b(x; \gamma_b)\right)$$

loss fn

Complicated problem. Soln: Stagewise Additive Modeling

Only optimize over one b at a time

For $b = 1, \ldots, B$

(*) → (a) $$\alpha_b, \gamma_b = \underset{\alpha, \gamma}{\operatorname{argmin}} \sum_{n=1}^{N} L\left(f_{b-1}(x_n) + \alpha \beta(x_n, \gamma_b)\right)$$

(b) $$f_b(x) = f_{b-1}(x) + \alpha_b + \beta(x; \gamma_b)$$

Punchline: AdaBoost =
Forward Stagewise Additive Modeling
w/ L = exponential Loss

Generalization: Can we do for Regression?

Yes: L = squared loss

at each step, fit new model to residuals of prev. (in 2@)

More general losses?

Harder to do, Square loss or exp. loss lead to simple algos.

Solv:

Gradient Boosting: (for any loss)

gradient descent to solve (*)

Analogy: $t = 1, 2, 3, \ldots$ $x_t \leftarrow x_{t-1} - \alpha \nabla f(x_{t-1})$

ensemble methods

RF: reduce variance, hard to overfit, easy to tune

Boosting: reduce bias, can overfit, needs tuning