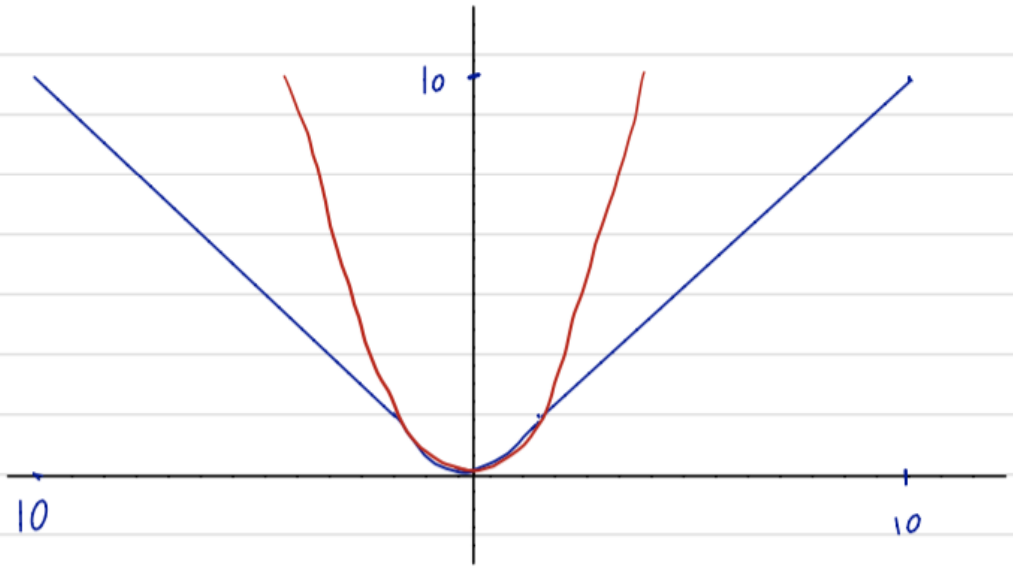
1. a) huber loss. Squared error loss.



Huber loss should be more robuse to outliers. Because outside of range $[-\delta, \delta]$, it inereases or decrease linearly. However, squared error has an increasing rate of change. The error of prediction affects the squared loss more than huber loss.

b) $L_\delta(y,t) = H_\delta(y-t)$

$$L_\delta(y,t) = \begin{cases} \frac{1}{2}(y-t)^2 & |y-t| \le \delta \\ \delta|y-t| - \frac{1}{2}\delta) & |y-t| > \delta \end{cases}$$

$$\therefore \frac{dL_\delta}{dw} = \frac{dL_\delta}{da} \times \frac{da}{dw}$$

$$a = w \cdot x + b \quad \therefore \frac{da}{dw} = x$$

$$\therefore \frac{dL_\delta}{dw} = x \cdot H_\delta'(y-t)$$

$$= \begin{cases} (y-t)x & |y-t| \le \delta \\ -\delta x & y-t < -\delta \\ \delta x & y-t > \delta \end{cases}$$

$$\frac{dL_\delta}{db} = \frac{dH_\delta}{da} \times \frac{da}{db}$$

$$= H_\delta'(y-t)$$

$$= \begin{cases} y-t & |y-t| \le \delta \\ -\delta & y-t < -\delta \\ \delta & y-t > \delta \end{cases}$$

2. $\vec{w}^* = \arg\min \frac{1}{2} \sum_{i=1}^{N} a^{(i)} \left(y^{(i)} - \vec{w}^T \vec{x}^{(i)}\right)^2 + \frac{\lambda}{2} \|\vec{w}\|^2$

$= \cdots\cdots + \frac{\lambda}{2} w^T \cdot w$

derivative $\Rightarrow$

$0 = -\frac{1}{2} \sum_{i=1}^{N} a^{(i)} \cdot 2 \cdot \vec{x}^{(i)} \cdot \left(y^{(i)} - \vec{w}^T \vec{x}^{(i)}\right) + \frac{\lambda}{2} \vec{w} \cdot 2$

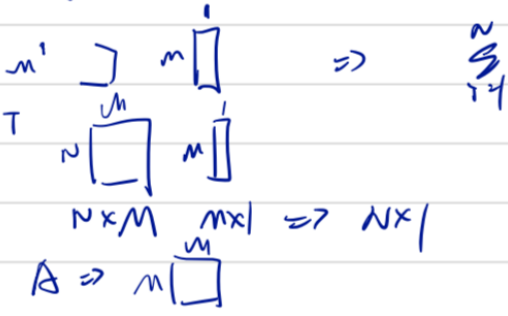
$= -\sum_{i=1}^{N} a^{(i)} \left(\vec{x}^{(i)}\right)\left(y^{(i)} - \vec{w}^T \vec{x}^{(i)}\right) + 2 \cdot \frac{1}{2} \lambda \vec{w}$

$= -\sum_{i=1}^{N} \vec{x}^{(i)} \cdot a^{(i)} y^{(i)} + \sum_{i=1}^{N} \vec{x}^{(i)} a^{(i)} \cdot \vec{w}^T \cdot \vec{x}^{(i)} + \lambda \vec{w}$

$\because x, w$ vector $\Rightarrow \quad 0 = -\sum_{i=1}^{N} \vec{x}^{(i)} a^{(i)} \cdot y^{(i)} + \sum_{i=1}^{N} \vec{x}^{(i)} \cdot a^{(i)} \cdot (\vec{x}^{(i)})^T \vec{w}$

$$\sum_{i=1}^{N} \vec{x}^{(i)} \cdot a^{(i)} \cdot \vec{x}^{(i)T} \vec{w} + \lambda \vec{w} = \sum_{i=1}^{N} x^{(i)} \cdot a^{(i)} \cdot y^{(i)}$$

$\sum_{i=1}^{N} a^{(i)} \cdot \vec{x}^{(i)} \cdot y^{(i)} = a^{(1)} \vec{x}^{(1)} y^{(1)} + a^{(2)} \vec{x}^{(2)} y^{(2)} \cdots\cdots a^{(N)} \vec{x}^{(N)} y^{(N)}$



$\Rightarrow \quad \sum_{i=1}^{N} a^{(i)} \cdot x^{(i)} y^{(i)} = X^T \cdot A \cdot Y$

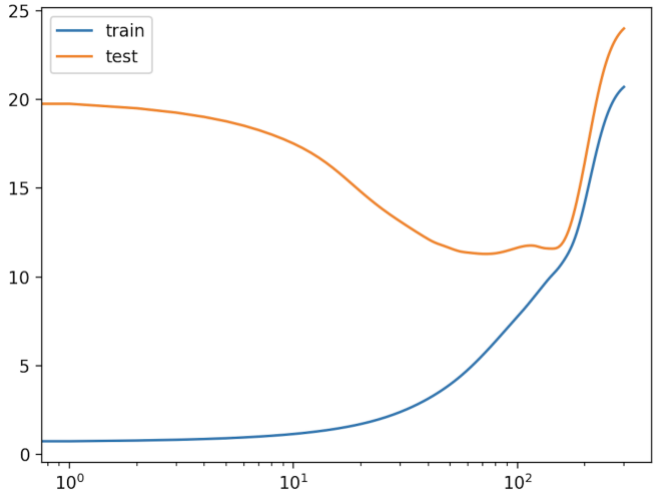
$\sum_{i=1}^{N} \vec{x}^{(i)} \cdot a^{(i)} \cdot \vec{x}^{T(i)} \cdot \vec{w} = X \cdot A \cdot X^T \cdot w$

$\because \quad 0 = X^T A y \cdot (X^T A X w - \lambda I w)$

$\because \quad 0 = X^T A y \cdot (X^T A X + \lambda I) w \Rightarrow$

$w = (X^T A X + \lambda I)^{-1} X^T A y$

Question 2 C)



Question:

Based on our understanding of overfitting and underfitting, how would you expect the training error and the validation error to vary as a function of τ? (I.e., what do you expect the curves to look like?)
Now run the experiment. Randomly hold out 30% of the dataset as a validation set. Compute the average loss for different values of τ in the range [10,1000] on both the training set and the validation set. Plot the training and validation losses as a function of τ (using a log scale for τ). Was your guess correct?

Answer:

First, we should know the τ is used to decide the weight for each point. When the τ is small, the training error is small, which also means the algorithm pays more attention to the closest point to the datum. The algorithm would be overfit to the training dataset. Therefore, the validation error should be huge when the τ is small. Conversely, when the τ increases, the algorithm will consider the weight of each point fairly. And it will turn to be underfit the features of training dataset. Therefore, the training error will increase and validation error should be decrease. However, if the τ is so huge. Both validation error and training error should be increase.

My guess is almost correct. For the small value of τ, the algorithm had low training loss error but high validation loss error. Because the algorithm overfitted the training data and lost more generalization. When the τ value increased over 100, the algorithm underfit the training data too much, and it cannot generate a linear regression model to fit the training dataset. Therefore, the validation losses also increase. When the τ value is about 50(seen from the graph above), the algorithm reached a suitable situation, it didn't overfit the training dataset, and the generalization increased. The weight for each point is perfect for us to predict the result. So we can see from the graph above, the validation error reached the minimum value.

Question 3: to prove $err' = \frac{\sum_{i=1}^{N} w_i' I(h_t(x_i) \neq t^{(i)})}{\sum_{i=1}^{N} w_i'} = \frac{1}{2}$

given: target set $t^{(i)} \in \{-1, 1\}$
$h(x_i)$ classifier return the value $\{-1, 1\}$

We set $W_n = \sum_{i=1}^{N} w_i$ $\quad W_E = \sum_{i \in E} w_i$ $\quad I\{h_t(x_i) \neq t_i\}$
$W_{cE} = \sum_{i \in cE} w_i$ $\quad I^c\{h_t(x_i) = t_i\}$

$\therefore i \in E$, $h_t(x_i) t_i = -1$ means the classifier did wrong
$i \in E_c$ $h_t(x_i) t_i = 1$ means the classifier did right

① $err_t = \frac{\sum_{i=1}^{N} w_i I(h_t(x_i) \neq t^{(i)})}{\sum_{i=1}^{N} w_i}$

$= \frac{\sum_{i \in E} w_i}{\sum_{i=1}^{N} w_i} = \frac{W_E}{W_E + W_{cE}}$

② $err' = \frac{\sum_{i=1}^{N} w_i' I(h_t(x_i) \neq t_i)}{\sum_{i=1}^{N} w_i'} = \frac{\sum_{i=1}^{n} w_i' I(E)}{\sum_{i=1}^{N} w_i \alpha_t} = \frac{\sum_{j \in E} w_j'}{\sum_{j \in E} w_j' + \sum_{j \in E_c} w_j'}$

③ $\because w_i' = w_i \cdot \exp\{-\alpha_t t^{(i)} h_t(x^{(i)})\}$

we seperate w_i' equation for $i \in E$ and $i \in E_c$

$\therefore w_i' = \begin{cases} \text{(a)} & i \in E \\ \text{(b)} & i \in E_c \end{cases}$

when $i \in E$ $\quad t(i) \cdot h_t(x_i) = -1$

$\therefore w_i' = w_i \cdot \exp(\alpha_t)$

$\alpha_t = \frac{1}{2}\left(\log \frac{1-err}{err}\right)$

$\therefore w_i' = w_i \cdot \exp\left(\frac{1}{2}\log\frac{1-err_t}{err_t}\right) = e^{\frac{1}{2}} = \sqrt{e}$

$\therefore w_i' = w_i \sqrt{\frac{1-err_t}{err_t}}$

using the same method, when $i \in E_C$

$i \in E_C \quad w_i' = w_i \cdot \exp(-\alpha_t)$

$= w_i \exp\left(-\frac{1}{2}\log\frac{1-err_t}{err_t}\right)$

$= w_i \exp\left(\frac{1}{2}\log\frac{err_t}{1-err_t}\right) = w_i \cdot \sqrt{\frac{err_t}{1-err_t}}$

Continous with (2)

$$err_t' = \frac{\sum_{i\in E} w_i'}{\sum_{i\in E} w_i' + \sum_{i\in E_C} w_i'} = \frac{\sum_{i\in E} w_i \sqrt{\frac{1-err_t}{err_t}}}{\sum_{i\in E} w_i \sqrt{\frac{1-err_t}{err_t}} + \sum_{i\in E_C} w_i \sqrt{\frac{err_t}{1-err_t}}}$$

from (1) we know $err_t = \frac{W_E}{W_E + W_{E_C}}$ $\quad \therefore 1-err_t = \frac{W_{E_C}}{W_E + W_{E_C}}$

$$\therefore err_t' = \frac{W_E \cdot \sqrt{\frac{W_{E_C}}{W_E}}}{W_E \cdot \sqrt{\frac{W_{E_C}}{W_E}} + W_{E_C} \cdot \sqrt{\frac{W_E}{W_{E_C}}}}$$

$$= \frac{\sqrt{W_{E_C} \cdot W_E^2}}{\sqrt{W_E}} \Big/ \left(\frac{\sqrt{W_{E_C} \cdot W_E^2}}{\sqrt{W_E}} + \frac{\sqrt{W_E W_{E_C}^2}}{\sqrt{W_{E_C}}}\right) = \frac{\sqrt{W_{E_C} \cdot W_E}}{2\sqrt{W_{E_C} \cdot W_E}}$$

$$\therefore = \frac{1}{2}$$