Question 1 Robust Regression.

(a) *Sketch the Huber loss $L_\delta(y,t)$ and squared error loss $L_{SE}(y,t) = \frac{1}{2}(y-t)^2$ for $t = 0$, either by hand or using a plotting library. Based on your sketch, why would you expect the Huber loss to be more robust to outliers?*

When $t = 2$,

$$L_{SE}(y,t) = \frac{1}{2}(y-t)^2 = \frac{1}{2}y^2$$

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

With the help of matplotlib, their sketches are shown in Figure 1.

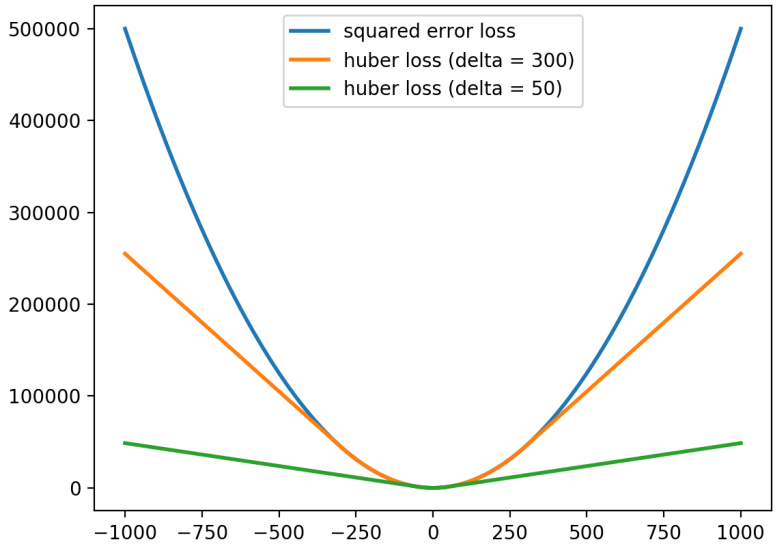

Figure 1: Sketches of squared error loss and Huber loss function.

As you can see from the Figure 1, for LSE functions, when the value $|y|$ is large (which is also the case with outliers), $L_{SE}(y,t)$ is very sensitive to $|y|$ (and outliers). On the contrary, for Huber loss function, when $|y|$ exceeds δ, $L_\delta(y,t)$ increases linearly to $|y|$, which shows that it is relatively less sensitive to outliers.

(b) *Give formulas for the partial derivatives $\partial L_\delta / \partial \boldsymbol{w}$ and $\partial L_\delta / \partial b$.*

The formula for the derivative $H'_\delta(a)$ is

$$H'_\delta(a) = \begin{cases} a & if\ |a| \le \delta \\ \delta & if\ a > \delta \\ -\delta & if\ a < -\delta \end{cases}$$

Then, in terms of $a = y - t$,

$$H'_\delta(y-t) = \begin{cases} y-t & if\ |y-t| \le \delta \\ \delta & if\ y > t+\delta \\ -\delta & if\ y < t-\delta \end{cases}$$

Question 1 [(b)] continued on next page...

As for the linear model,
if $\mathbf{x}$ and $\mathbf{w}$ are both D dimensional vectors.

$$y = \mathbf{w}^T\mathbf{x} + b = \sum_j w_j x_j + b$$

$$H'_\delta(y-t) = \frac{\partial H_\delta}{\partial y} = \begin{cases} \sum_j w_j x_j + b - t & if\ |y-t| \le \delta \\ \delta & if\ y > t+\delta \\ -\delta & if\ y < t-\delta \end{cases}$$

$$\frac{\partial L_\delta}{\partial w_j} = \frac{\partial H_\delta}{\partial w_j} = \frac{\partial H_\delta}{\partial y}\frac{\partial y}{\partial w_j} = \begin{cases} x_j(\sum_{j'} w_{j'} x_{j'} + b - t) & if\ |y-t| \le \delta \\ \delta x_j & if\ y > t+\delta \\ -\delta x_j & if\ y < t-\delta \end{cases}$$

$$\frac{\partial L_\delta}{\partial b} = \frac{\partial H_\delta}{\partial b} = \frac{\partial H_\delta}{\partial y^{(i)}}\frac{\partial y^{(i)}}{\partial b} = \begin{cases} \sum_j w_j x_j + b - t & if\ |y-t| \le \delta \\ \delta & if\ y > t+\delta \\ -\delta & if\ y < t-\delta \end{cases}$$

$$\frac{\partial L_\delta}{\partial \mathbf{w}} = \begin{cases} \mathbf{x}(\mathbf{x}^T\mathbf{w} + b - t) & if\ -\delta\mathbf{1} \le \mathbf{y}-\mathbf{t} \le \delta\mathbf{1} \\ \delta\mathbf{x} & if\ \mathbf{y} > \mathbf{t}+\delta\mathbf{1} \\ -\delta\mathbf{x} & if\ \mathbf{y} < \mathbf{t}-\delta\mathbf{1} \end{cases}$$

When $\mathbf{X}$ is a $N \times D$ matrix and $\mathbf{w}$ is a D dimensional vector. Then,

$$\mathbf{y} = \mathbf{X}\mathbf{w} + b\mathbf{1},$$
$$y^{(i)} = \sum_j w_j x_j^{(i)} + b$$

We define $\mathscr{E}$ as the model's cost function.

$$\mathscr{E}_\delta = \frac{1}{N}\sum_{i=1}^N L_\delta = \frac{1}{N}\sum_{i=1}^N H_\delta$$

We apply $y^{(i)}$ in the partial derivatives,

$$\frac{\partial H_\delta}{\partial y^{(i)}} = \begin{cases} \sum_j w_j x_j^{(i)} + b - t^{(i)} & if\ |y^{(i)} - t^{(i)}| \le \delta \\ \delta & if\ y^{(i)} > t^{(i)} + \delta \\ -\delta & if\ y^{(i)} < t^{(i)} - \delta \end{cases}$$

$$\frac{\partial \mathscr{E}_\delta}{\partial w_j} = \frac{\partial \mathscr{E}_\delta}{\partial y^{(i)}}\frac{\partial y^{(i)}}{\partial w_j} = \frac{1}{N}\sum_{i=1}^N \begin{cases} x_j^{(i)}(\sum_{j'} w_{j'} x_{j'}^{(i)} + b - t^{(i)}) & if\ |y^{(i)} - t^{(i)}| \le \delta \\ \delta x_j^{(i)} & if\ y^{(i)} > t^{(i)} + \delta \\ -\delta x_j^{(i)} & if\ y^{(i)} < t^{(i)} - \delta \end{cases}$$

$$\frac{\partial \mathscr{E}_\delta}{\partial b} = \frac{\partial \mathscr{E}_\delta}{\partial y^{(i)}}\frac{\partial y^{(i)}}{\partial b} = \frac{1}{N}\sum_{i=1}^N \begin{cases} (\sum_j w_j x_j^{(i)} + b - t^{(i)}) & if\ |y^{(i)} - t^{(i)}| \le \delta \\ \delta & if\ y^{(i)} > t^{(i)} + \delta \\ -\delta & if\ y^{(i)} < t^{(i)} - \delta \end{cases}$$

The partial derivatives can also be represented by matrix-vector format,

$$\frac{\partial \mathscr{E}_\delta}{\partial \mathbf{w}} = \begin{cases} \frac{1}{N}\mathbf{X}^T(\mathbf{X}\mathbf{w} + b\mathbf{1} - \mathbf{t}) & if\ -\delta\mathbf{1} \le \mathbf{y}-\mathbf{t} \le \delta\mathbf{1} \\ \frac{\delta}{N}\mathbf{X}^T\mathbf{1} & if\ \mathbf{y} > \mathbf{t}+\delta\mathbf{1} \\ -\frac{\delta}{N}\mathbf{X}^T\mathbf{1} & if\ \mathbf{y} < \mathbf{t}-\delta\mathbf{1} \end{cases}$$

Question 1 continued on next page...

(c) *Write Python code to perform (full batch mode) gradient descent on this model.*

The update rule of gradient descent method is as follows,

$$w_j \leftarrow w_j - \alpha \frac{\partial \mathscr{E}_\delta}{\partial w_j}$$
$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \frac{\partial \mathscr{E}_\delta}{\partial \mathbf{w}}$$

Without using **for** loop, we applied the **np.where** function to realize the gradient descent on the Huber loss function. First, we generalized three vectors using $\mathbf{y} - \mathbf{t}$, $\delta\mathbf{1}$ and $-\delta\mathbf{1}$ separately; then we used **np.where** to determine which line of the three vectors should be chosen.
I sampled 50000 samples through $y = 3x_1 + 5x_2 - 9x_3 + 28x_4 - 10x_5 + 1$. Through 1000 iterations, the loss dropped to 0.001384. The predicted w was $[3.000181, 5.00041, -9.00045, 28.000627, -10.000461, 1.000123]$. However, I evaluated the efficiency of the algorithms with Huber loss function or square error loss function, and found that they took 6.6 and 0.24 seconds, respectively. I guessed the **np.where** function takes up most of the total time.

Code is in q1.py.

Question 2 Locally Weighted Regression.

(a) *Show the solution to the weighted least squares problem.*

We define $\mathscr{E}$ as the function of the weighted least squares problem.

$$\begin{aligned}
\mathscr{E} &= \frac{1}{2}\sum_{i=1}^{N} a^{(i)}(y^{(i)} - \mathbf{w}^T\mathbf{x}^{(i)})^2 + \frac{\lambda}{2}||\mathbf{w}||^2 \\
&= \frac{1}{2}(\mathbf{y} - \mathbf{X}\mathbf{w})^T\mathbf{A}(\mathbf{y} - \mathbf{X}\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} \\
&= \frac{1}{2}[(\mathbf{y}^T\mathbf{A} - \mathbf{w}^T\mathbf{X}^T\mathbf{A})(\mathbf{y} - \mathbf{X}\mathbf{w})] + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} \\
&= \frac{1}{2}(\mathbf{y}^T\mathbf{A}\mathbf{y} - \mathbf{y}^T\mathbf{A}\mathbf{X}\mathbf{w} - \mathbf{w}^T\mathbf{X}^T\mathbf{A}\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{A}\mathbf{X}\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} \\
&= \frac{1}{2}[\mathbf{y}^T\mathbf{A}\mathbf{y} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{A}\mathbf{y} + (\mathbf{X}\mathbf{w})^T\mathbf{A}\mathbf{X}\mathbf{w}] + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}
\end{aligned}$$

To get solution for the problem, $\mathbf{w}^*$, we need partial derivatives to zero.

$$\begin{aligned}
\frac{\partial \mathscr{E}}{\partial \mathbf{w}} = -\mathbf{X}^T\mathbf{A}\mathbf{y} + \mathbf{X}^T\mathbf{A}\mathbf{X}\mathbf{w} + \lambda\mathbf{w} &= 0 \\
(\mathbf{X}^T\mathbf{A}\mathbf{X} + \lambda\mathbf{I})\mathbf{w} &= \mathbf{X}^T\mathbf{A}\mathbf{y} \\
\mathbf{w} &= (\mathbf{X}^T\mathbf{A}\mathbf{X} - \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{A}\mathbf{y}
\end{aligned}$$

(b) *Complete the implementation of locally reweighted least squares by providing the missing parts for q2.py.*

See q2.py.

(c) *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?)*
Run the experiment.

Question 2 [(c)] continued on next page...

I think the curve of both of them falls first and then rises. The curve of validation set is above the curve of training set. Too little τ or too much τ is underfitting for both training set and validation set. However, they may reach the lowest point at different τs, because when the train set curve reaches the lowest point, it may have been overfitting, and the lowest point of validation set needs to be found near this value.

We've drawn the training and validation losses as a function of τ (using a log scale for τ, which is between 10^1 and 10^3), as shown in Figure 2.
The picture is not exactly what I guesses. As can be seen in the Figure 2, the smaller the τ is, the better the result of training set is, but the result of the validation set is not the best, which indicates that the model has been overfitted when τ is small. When τ is around 10^2, the results of validation set are optimized, and then the losses of both sets kept increasing, which indicates that the model is underfitting then.

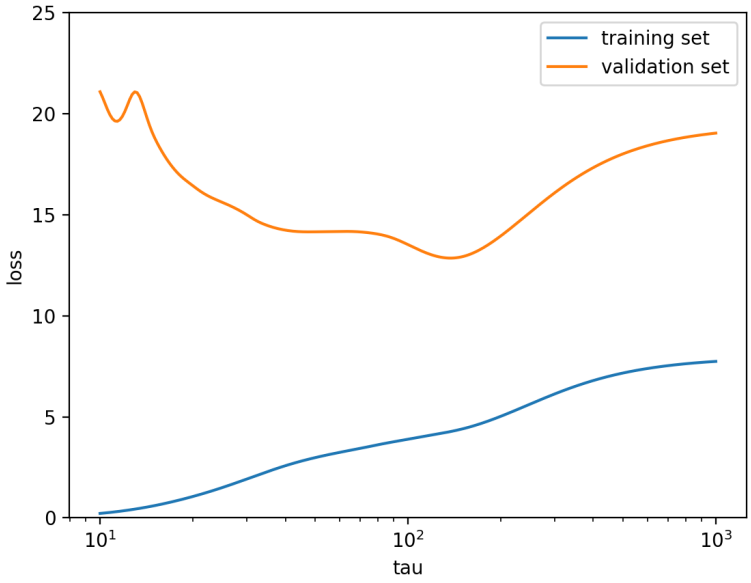

Figure 2: The losses of the training and validation dataset.

Question 3 AdaBoost.

The goal of this question is to show that the AdaBoost algorithm changes the weights in order to force the weak learner to focus on difficult data points.
Show that the error w.r.t. $(w'_1, ..., w'_N)$ *is exactly* $\frac{1}{2}$. *That is, show that*

$$err'_t = \frac{\sum_{i=1}^N w'_i \mathbb{I}\{h_t(\mathbf{x}^{(i)}) \neq t^{(i)}\}}{\sum_{i-1}^N w'_i} = \frac{1}{2}.$$

Firstly, we deducted the $e^{2\alpha}$,

$$e^{2\alpha} = \frac{1 - err_t}{err_t} = \frac{\sum_{i=1}^N w_i \mathbb{I}\{h_t(\mathbf{x}^{(i)}) = t^{(i)}\}}{\sum_{i-1}^N w_i} \frac{\sum_{i-1}^N w_i}{\sum_{i=1}^N w_i \mathbb{I}\{h_t(\mathbf{x}^{(i)}) \neq t^{(i)}\}}$$
$$= \frac{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i}{\sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i}$$

Question 3 continued on next page...

Apply the deductions of w_i' and $e^{2\alpha}$,

$$\begin{aligned}
err_t' &= \frac{\sum_{i=1}^{N} w_i' \mathbb{I}\{h_t(\mathbf{x}^{(i)}) \neq t^{(i)}\}}{\sum_{i-1}^{N} w_i'} \\
&= \frac{\sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i e^{\alpha_t}}{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i e^{-\alpha_t} + \sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i e^{\alpha_t}} \\
&= \frac{\sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i e^{2\alpha_t}}{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i + \sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i e^{2\alpha_t}} \\
&= \frac{\sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i \frac{1-err_t}{err_t}}{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i + \sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i \frac{1-err_t}{err_t}} \\
&= \frac{\frac{1-err_t}{err_t} \sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i}{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i + \frac{1-err_t}{err_t} \sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i} \\
&= \frac{\frac{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i}{\sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i} \sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i}{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i + \frac{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i}{\sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i} \sum_{i \in N, t^i \neq h_t(x^{(i)})} w_i} \\
&= \frac{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i}{\sum_{i \in N, t^i = h_t(x^{(i)})} w_i + \sum_{i \in N, t^i = h_t(x^{(i)})} w_i} \\
&= \frac{1}{2}
\end{aligned}$$