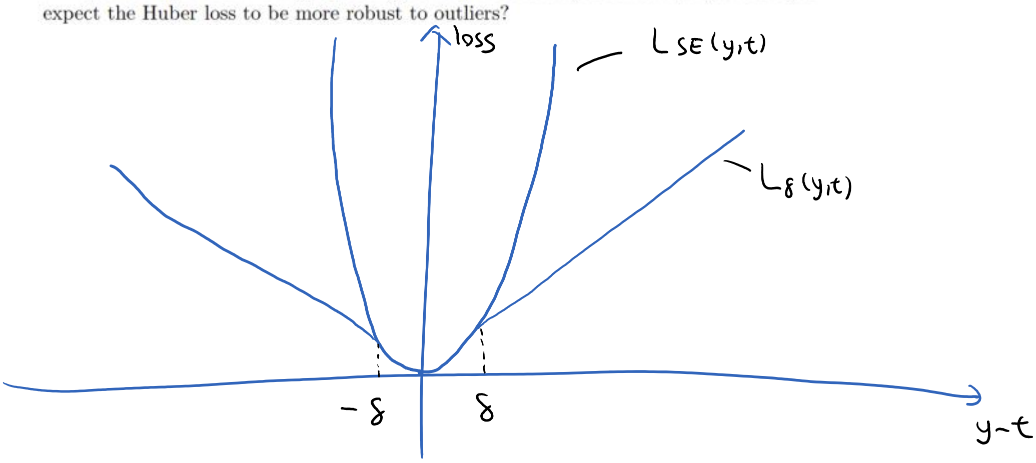
Homework 3

February 7, 2019 11:06 PM

1. **[3pts] Robust Regression.** One problem with linear regression using squared error loss is that it can be sensitive to outliers. Another loss function we could use is the *Huber loss*, parameterized by a hyperparameter δ:

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

(a) **[1pt]** Sketch the Huber loss $L_\delta(y,t)$ and squared error loss $L_{\mathrm{SE}}(y,t) = \frac{1}{2}(y-t)^2$ 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?



Huber loss is more robust to outlier because it assigns less penalty/loss to instances where y (prediction) and t (label) are different by a large margin (larger than δ), usually y and t will be different by a lot if (x,t) is an outlier, thus the Huber loss function is less affected by outlier, and more robust.

(b) **[1pt]** Just as with linear regression, assume a linear model:

$$y = \mathbf{w}^\top \mathbf{x} + b.$$

Give formulas for the partial derivatives $\partial L_\delta / \partial \mathbf{w}$ and $\partial L_\delta / \partial b$. (We recommend you find a formula for the derivative $H'_\delta(a)$, and then give your answers in terms of $H'_\delta(y-t)$.)

$$\frac{\partial H_\delta}{\partial a} = \begin{cases} a & \text{if } |a| < \delta \\ \delta & \text{if } a > \delta \\ -\delta & \text{if } a < -\delta \end{cases}$$

$$\frac{\partial L_\delta}{\partial w} = \frac{\partial L_\delta}{\partial y}\frac{\partial y}{\partial w} = L'_\delta(y,t)\,x = \boxed{H'_\delta(y,t)\,x}$$

$$\frac{\partial L_\delta}{\partial b} = \frac{\partial L_\delta}{\partial y}\frac{\partial y}{\partial b} = L'_\delta(y,t) = \boxed{H'_\delta(y,t)}$$

1c):

Here is the output for gradient descent, which shows the loss is decreasing and converging as the predictor is being trained, as intended

```
Commands execute without debug.  Use arrow keys for history.                                    Options
Python Type "help", "copyright", "credits" or "license" for more information.
>>> [evaluate HW3.py]
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\externals\cloudpickle\cloudpickle.py:47: DeprecationWarning: the imp module is de
  import imp
the total loss is 31.23430341506266
the total loss is 21.983101968123847
the total loss is 17.604053794834957
the total loss is 15.856703798243407
the total loss is 15.086196086936722
the total loss is 14.58849890902105
the total loss is 14.19815481367193
the total loss is 13.861972616371643
the total loss is 13.561045649348188
the total loss is 13.289631426799723
the total loss is 13.045789622990402
the total loss is 12.826000785931248
the total loss is 12.630067329319466
the total loss is 12.454416689824123
the total loss is 12.297326194459894
the total loss is 12.157944724119089
the total loss is 12.035090360037826
the total loss is 11.927163549933566
the total loss is 11.83291381989587
the total loss is 11.750906887120479
the total loss is 11.679201124993273
the total loss is 11.616631541864503
the total loss is 11.56200570600776
the total loss is 11.514556975300232
the total loss is 11.4731825041131
the total loss is 11.436410751491684
the total loss is 11.403380680425794
the total loss is 11.37366373004653
the total loss is 11.346901937463446
the total loss is 11.322779574911962
the total loss is 11.301017555404556
the total loss is 11.281379291790717
the total loss is 11.26349670727444
the total loss is 11.247097983422378
the total loss is 11.231963140982925
the total loss is 11.217885106605037
the total loss is 11.204712277859729
the total loss is 11.192349190634044
```

2. **[6pts] Locally Weighted Regression.**

(a) **[2pts]** Given $\{(\mathbf{x}^{(1)}, y^{(1)}), .., (\mathbf{x}^{(N)}, y^{(N)})\}$ and positive weights $a^{(1)}, ..., a^{(N)}$ show that the solution to the *weighted* least squares problem

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

is given by the formula

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{A}\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{A}\mathbf{y} \tag{2}$$

where $\mathbf{X}$ is the design matrix (defined in class) and $\mathbf{A}$ is a diagonal matrix where $\mathbf{A}_{ii} = a^{(i)}$.

define J as the cost function that we want to minimize:

$$J = \frac{1}{2}\sum_{i=1}^{N} a^{(i)}(y^{(i)} - w^T x^{(i)})^2 + \frac{\lambda}{2}||w||^2$$

convert it into Matrix form:

$$J = \frac{1}{2}(y - Xw)^T A (y - Xw) + \frac{\lambda}{2}||w||^2$$

where A, y, w, X are all matrices/vectors, as set up by the problem.

$$J = \frac{1}{2} y^T A y - \frac{1}{2} y^T A X w - \frac{1}{2} w^T X^T A y + \frac{1}{2} w^T X^T A X w + \frac{\lambda}{2} \|w\|^2$$

$$J = \frac{1}{2} y^T A y - w^T X^T A y + \frac{1}{2} w^T X^T A X w + \frac{\lambda}{2} \|w\|^2$$

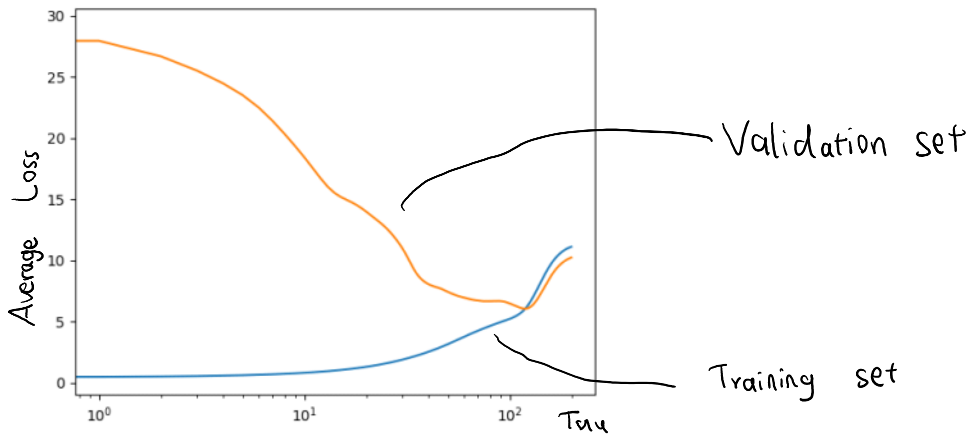
to minimize J, set $\frac{\partial J}{\partial w} = 0$

$$\frac{\partial J}{\partial w} = -X^T A y + X^T A X w + \lambda w = 0$$

$$(X^T A X + \lambda I) w = X^T A y$$

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

QED



(d) **[1pt]** How would you expect this algorithm to behave as $\tau \to \infty$? When $\tau \to 0$? Is this what actually happened?

As $\tau \to \infty$, the algorithm will produce approximately the same weight for all training data, thus the predictor will behave more and more like a standard linear regression with squared error loss function. This can cause underfitting as linear regression may not be able to fit the data well.

As $\tau \to 0$, the training loss will be close to zero. That is because with a small τ, the calculated weights will increase dramatically as $\|x - x^{(i)}\|^2$ increase. This will cause overfitting as the model will be too sensitive to outliers. As a result of overfitting, the training loss with be small, and the validation loss will be large.

This agrees with the graph, as we can see, in general as increases, training loss increases and validation loss decreases.