TTIC31020 HW1 : Regression

Part 1 : Linear Regression

Collaborated with Haichuan Wang, Lily Zhu

Problem 1 [6 points]

Suppose we take the data set from which we learned (estimated) $\mathbf{w}^*$, and for every example $(\mathbf{x}_i, y_i)$, change y_i to $y_i' = ay_i + b$, for some constants a and b (same a and b for all the is). Now you fit least squares model to the new data set $\{\mathbf{x}_i, y_i'\}$, yielding the parameter vector $\mathbf{w}'$.

Can $\mathbf{w}'$ be computed directly from $\mathbf{w}^*$, without looking at the data again? If yes, how exactly? If not, why not?

- Yes.

- $w^* = (X^TX)^{-1}X^Ty$

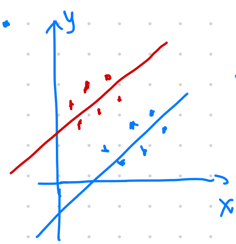
$$w' = (X^TX)^{-1}X^T\Big(ay + b\underbrace{\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix}}_{n\times 1}\Big)$$

$$= a(X^TX)^{-1}X^Ty + b(X^TX)^{-1}X^T\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix}$$

We can write $\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix} = \underbrace{\begin{pmatrix}1 & x_{11} & \cdots & x_{1p}\\ \vdots & & & \\ 1 & x_{n1} & \cdots & x_{np}\end{pmatrix}}_{X} \cdot \begin{pmatrix}1\\ 0\\ 0\\ \vdots\\ 0\end{pmatrix}$,

Then, $w' = a(X^TX)^{-1}X^Ty + b(X^TX)^{-1}X^T \cdot X \cdot \begin{pmatrix}1\\ 0\\ \vdots\\ 0\end{pmatrix}$

$$= a \cdot w^* + b \cdot \begin{pmatrix}1\\ 0\\ \vdots\\ 0\end{pmatrix}$$



$W = [W_0 \;\; W_1]$

Problem 2 [7 points]

Now, instead of modifying ys, we will modify $\mathbf{x}$s. For every feature (dimension of $\mathbf{x}$) we will change x_{ij} to $\tilde{x}_{ij} = c_jx_{ij}$ for some constants c_j (same set of $c_1, \ldots, c_d$ for all is). Again, we fit least squares model to the new data set $\{\tilde{\mathbf{x}}_i, y_i\}$, and get $\mathbf{w}'$.

Can $\mathbf{w}'$ be computed directly from $\mathbf{w}^*$, without looking at the data again? If yes, how exactly? If not, why not?

- Yes.

- Let $X' = \overbrace{\begin{pmatrix}1 & x_{11} & \cdots & x_{1p}\\ \vdots & & & \\ 1 & x_{n1} & \cdots & x_{np}\end{pmatrix}}^{X} \overbrace{\begin{pmatrix}1 & 0 & \cdots & 0\\ 0 & c_1 & & \\ \vdots & & \ddots & \\ 0 & & & c_p\end{pmatrix}}^{D}$

Now, $w' = \left((XD)^T \cdot XD\right)^{-1}(XD)^T \cdot y$

$$= (D^TX^T \cdot XD)^{-1} \cdot D^TX^T \cdot y$$

$$= D^{-1} \cdot (X^TX)^{-1} \cdot (D^T)^{-1} \cdot D^TX^T \cdot y$$

$$= D^{-1}(X^TX)^{-1}X^Ty$$

$$= D^{-1} \cdot w^*$$

$$= \begin{pmatrix}1 & 0 & \cdots & 0\\ \vdots & \frac{1}{c_1} & & \\ \vdots & & \ddots & \\ 0 & & & \frac{1}{c_p}\end{pmatrix} \cdot w^*$$

In Lecture 2 we saw two necessary conditions for the optimal w as given below.

$$\sum_{i=1}^{N}(y_i - \mathbf{w}\cdot\mathbf{x}_i) = 0 \tag{1}$$

$$\forall j = 1,\dots,d: \quad \sum_{i=1}^{N}(y_i - \mathbf{w}\cdot\mathbf{x}_i)x_{ij} = 0 \tag{2}$$

We are going to confirm that these indeed are necessary conditions for optimal linear model $\mathbf{w}$ in a different way, without reasoning about derivatives, instead proving it by explicit contradiction.

First, consider a linear model $\mathbf{w}$ which violates (1),

$$\sum_{i=1}^{N}(y_i - \mathbf{w}\cdot\mathbf{x}_i) \;=\; p \neq 0. \tag{3}$$

Problem 3 [6 points]
Show a model $\mathbf{w}_p$ which satisfies (1) and has a lower empirical squared loss than $\mathbf{w}$.

• Let $W_p = W + \begin{pmatrix} \frac{p}{N} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$,

$$L(W_p) = \frac{1}{N}\sum_{i=1}^{N}(y_i - W_p x_i)^2$$

$$= \frac{1}{N}\cdot\sum_{i=1}^{N}\left(y_i - Wx_i - \begin{pmatrix} \frac{p}{N} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}(1\ x_{i1} \cdots x_{ip})\right)^2$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left(y_i - W\cdot X_i - \frac{p}{N}\right)^2$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left((y_i - wx_i)^2 - 2(y_i - wx_i)\cdot\frac{p}{N} + \frac{p^2}{N^2}\right)$$

$$= L(W) - \frac{2}{N}\cdot p\cdot\frac{p}{N} + \frac{1}{N}\cdot N\cdot\frac{p^2}{N^2}$$

$$= L(W) - \underbrace{\frac{p^2}{N^2}}_{>0}$$

Therefore, $L(W_p) < L(W)$.

Now do the same for the other condition: consider a linear model $\mathbf{w}$ which violates (2),

$$\forall j = 1,\dots,d: \quad \sum_{i=1}^{N}(y_i - \mathbf{w}\cdot\mathbf{x}_i)x_{ij} \;=\; s \neq 0. \tag{4}$$

Problem 4 [7 points]
Show a model $\mathbf{w}_s$ which satisfies (2) and has a lower empirical squared loss than $\mathbf{w}$.

• Pick $k \in [1, d]$, let $a = \frac{s}{N\sum_{i=1}^{N} x_{ik}^2}$

• Define $W_s = \begin{pmatrix} w_0 \\ \vdots \\ w_k + a \\ \vdots \\ w_d \end{pmatrix} = W + \begin{pmatrix} 0 \\ \vdots \\ a \\ \vdots \\ 0 \end{pmatrix}$

Then, $L(W_s) = \frac{1}{N}\sum_{i=1}^{N}(y_i - W_s x_i)^2$

$$= \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \left(W + \begin{pmatrix} 0 \\ \vdots \\ a \\ \vdots \\ 0 \end{pmatrix}\right)x_i\right)^2$$

$$= \frac{1}{N}\sum_{i=1}^{N}(y_i - Wx_i - a\cdot x_{ik})^2$$

$$= \frac{1}{N}\sum_{i=1}^{N}(y_i - Wx_i)^2 - 2\cdot\frac{1}{N}\sum_{i=1}^{N}(y_i - Wx_i)\cdot x_{ik}\cdot a + \frac{1}{N}\sum_{i=1}^{N}(a\cdot x_{ik})^2$$

$$= L(W) - 2\cdot\frac{1}{N}s\cdot a + \frac{1}{N}a^2\cdot\sum_{i=1}^{N} x_{ik}^2$$

$$= L(W) - 2\cdot\frac{1}{N}s\cdot a + \frac{1}{N}a\cdot\frac{s}{N\cdot\sum x_{ik}^2}\cdot\sum x_{ik}^2$$

$$= L(W) - \frac{1}{N}s(2a - a)$$

$$= L(W) - \frac{sa}{N} \;\rightarrow\; = \frac{s^2}{N^2\cdot\sum_{i=1}^{N} x_{ik}} > 0$$

Therefore, $L(W_s) < L(W)$

Part2: Asymmetric Loss

In class we have discussed the idea of learning (via empirical risk minimization) by finding the parameter values that minimize the loss function on the training data. In the case of least squares regression there is a closed form solution. We will now consider a modification, and develop a gradient descent solution for minimizing it.

Consider an *asymmetric loss* function:

$$\ell_\alpha(y, \widehat{y}) = \begin{cases} \alpha\,(y-\widehat{y})^2 & \text{if } y \geq \widehat{y}, \\ (y-\widehat{y})^2 & \text{if } y < \widehat{y}. \end{cases} \tag{5}$$

Problem 5 [9 points]
Assuming $\alpha > 1$, what kind of desired properties of the predictor does this loss function capture? When would you want to use it instead of the standard least squares? Feel free to provide a specific (possibly made up!) application scenario, or describe such a scenario in general terms.

- paneltíes under-predictions ($\hat{y} \leq y$)
- For example, when predicting the chance of a person getting infected, we can afford overpredicting a little, i.e. providing them with a harmless vaccine. But underpredicting the risk will cause the person not protected by vaccine.

Problem 6 [10 points]
Write down the expression for the gradient of the loss ℓ_α with respect to the parameters of a linear predictor $\widehat{y}(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x}$.

- $\hat{y}(x) = wx$

$$L(w) = \frac{1}{N}\sum_{i=1}^{N}(y_i - wx_i)^2$$

- $$\frac{\partial l_\alpha}{\partial w_0} = \begin{cases} \dfrac{\partial(\alpha\cdot(y-w_0-w_1x_1-\cdots-w_px_p)^2)}{\partial w_0}, & y \geq \hat{y} \\ \dfrac{\partial(y-w_0-w_1x_1-\cdots-w_px_p)^2}{\partial w_0}, & y < \hat{y} \end{cases}$$

$$= \begin{cases} 2\alpha(y-wx), & y \geq \hat{y} \\ 2(y-wx), & y < y \end{cases}$$

- $$\frac{\partial l_\alpha}{\partial w_j},\ j \neq 0 = \begin{cases} 2\alpha(y-wx)x_j, & y \geq \hat{y} \\ 2(y-wx)x_j, & y < \hat{y} \end{cases}$$

- Therefore,

$$\nabla l_\alpha = \begin{cases} \begin{pmatrix} 2\alpha(y-wx) \\ 2\alpha(y-wx)x_1 \\ \vdots \\ 2\alpha(y-wx)x_p \end{pmatrix}, & y \geq \hat{y} \\ \begin{pmatrix} 2(y-wx) \\ 2(y-wx)x_1 \\ \vdots \\ 2(y-wx)x_p \end{pmatrix}, & y < \hat{y} \end{cases}$$

Problem 8 **[20 bonus points]**
In this problem, we will flesh out the analysis of the convergence rate of the Perceptron algorithm. It's a bonus problem: it's optional, and if you solve it you may get more than 100 points for the homework assignment.

Let's make the following assumptions. Suppose that our data lies in the unit ball and is linearly separable by a *unit-norm, origin-containing* hyperplane $\mathbf{w}^*$ with margin γ. This means that for every training data point $\mathbf{x}_i$ and corresponding label y_i, we have:

$$\|\mathbf{x}_i\| \leq 1$$
$$y_i \mathbf{w}^* \cdot \mathbf{x}_i \geq \gamma$$

Note that we can always ensure (by manipulating the data) that $\|\mathbf{x}_i\| \leq 1$, and $b = 0$ (think why this is true).

Let M denote the number of mistakes Perceptron makes. Recall from lecture that we want to show that $M \leq 1/\gamma^2$.

1. Without loss of generality, we'll assume that all our datapoints have label $+1$. Explain why this is a valid assumption.

- We can transfrom the dataset as

$$X_i = \begin{cases} x_i, & \text{if } y_i = 1 \\ -x_i, & \text{if } y_i = -1, \end{cases}$$

Then, we would still have the same prediction, because

$$\text{sign}(-X_i W) = -\text{sign}(X_i W).$$

2. In showing the convergence of Perceptron, we'll need two lemmas. The first roughly says, "we make decent progress towards a good solution after each mistake." The second roughly says, "the norm of our solution vector isn't too big." Think about why these might be good things to show (no need to write anything here).

3. OK, let's formalize the statement of the first lemma. Let w_i denote our weight vector after we've made i mistakes. Show that:

$$\mathbf{w_{i+1}} \cdot \mathbf{w}^* - \mathbf{w_i} \cdot \mathbf{w}^* \geq \gamma$$

Then, think about how this mathematical statement fits with the English description above.

- After making the $i+1^{th}$ mistake,

$$W_{i+1} = W_i + y_i X_i$$

- now we get

$$w^*(W_{i+1} - W_i) = w^* y_i X_i$$

- as $y_i W^* X_i \geq \gamma$, we obtained the desired inequality.

4. Before we move on to the next lemma, let's try a naive bound to capture the idea of the second lemma. Using the triangle inequality, bound $\|\mathbf{w_M}\|$.
Hint: The triangle inequality states that for any two vectors $\mathbf{a}$ and $\mathbf{b}$:

$$\|\mathbf{a} + \mathbf{b}\| \leq \|\mathbf{a}\| + \|\mathbf{b}\|$$

- Let $\alpha(i)$ be the index of $X_{(i)}$ at the i^{th} mistake.

then
$$\|W_M\| = \|W_0 + \sum_{i=1}^{M} y_{\alpha(i)} \cdot X_{\alpha(i)}\|$$

$y_i = 1$

$$= \|\sum_{i=1}^{M} X_{\alpha(i)}\|$$

$$\leq \underbrace{\|X_{\alpha(1)}\|}_{\leq 1} + \cdots + \underbrace{\|X_{\alpha(M)}\|}_{\leq 1}$$

$$\leq M$$

5. Let's now strengthen this. In general, the following "squared triangle inequality" doesn't hold (if you're unconvinced, find some counterexample):

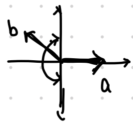
$$\|\mathbf{a}+\mathbf{b}\|^2 \leq \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2$$

However, this is true for the iterates of our Perceptron algorithm! To see this, show that if $\mathbf{a}\cdot\mathbf{b} \leq 0$, then the above "squared triangle inequality" is actually true. Use this to show that:

$$\|\mathbf{w_M}\| \leq \sqrt{M}$$

Contrast this with the bound we saw in the previous part.

- As $a \cdot b = \|a\| \cdot \|b\| \cdot \cos\theta \leq 0$, it must be that $\cos\theta \leq 0$, meaning $\theta \in [\frac{\pi}{2}, \frac{3\pi}{2}]$.



- Therefore, a and b are pointing at "opposing directions". Thus, a+b will be "shorter" than both a and b:

$$\|a+b\| \leq \|a\| \text{ and } \|a+b\| \leq \|b\|,$$

so $\|a+b\|^2 \leq \|a\|^2$ and $\|a+b\|^2 \leq \|b\|^2$,

so $\|a+b\|^2 \leq \|a\|^2 + \|b\|^2$.

Last question:

- $\|W_M\| = \|X_{\alpha(1)} + \cdots + X_{\alpha(M)}\|$,

now

$$\|X_{\alpha(1)} + \cdots + X_{\alpha(M)}\|^2 \leq \underbrace{\|X_{\alpha(1)}\|^2}_{\leq 1} + \cdots + \underbrace{\|X_{\alpha(M)}\|^2}_{\leq 1}$$

$$\leq M$$

Thus, $\|W_M\| \leq \sqrt{M}$.

6. Let's combine everything. Get an upper and lower bound on the quantity $\mathbf{w}^* \cdot \mathbf{w_M}$, simplify appropriately, and conclude that $M \leq 1/\gamma^2$. Here, it should become clear why the bound from part (4) is not sufficient to tell us anything interesting.

Hint: You might find it useful to know the *Cauchy-Schwarz inequality*, which states that for any two vectors $\mathbf{a}$ and $\mathbf{b}$:

$$\mathbf{a}\cdot\mathbf{b} \leq \|\mathbf{a}\| \cdot \|\mathbf{b}\|$$

The proof of this is straightforward:

$$\begin{aligned}\mathbf{a}\cdot\mathbf{b} &= \|\mathbf{a}\| \cdot \|\mathbf{b}\| \cdot \cos(\theta_{\mathbf{a},\mathbf{b}}) \\ &\leq \|\mathbf{a}\| \cdot \|\mathbf{b}\|\end{aligned}$$

- Use telescoping:

$$W_M \cdot W^* = \left.\begin{aligned} &\overbrace{W_M W^* - W_{M-1} W^*}^{\geq \gamma} \\ &+ W_{M-1}W^* - W_{M-2}W^* \\ &\vdots \\ &+ W_1 W^* - W_0 W^* \end{aligned}\right\} M$$

$$\geq \boxed{M \cdot \gamma} \text{ lower bound}$$

- In addition,

$$W_M \cdot W^* \leq \|W_M\| \cdot \underbrace{\|W^*\|}_{=1\text{, from class slide}}$$

$$= \|W_M\|$$

$$\leq \boxed{\sqrt{M}} \text{ upperbound}$$

- Combine:

$$M \cdot \gamma \leq \sqrt{M}$$

$$M \leq \frac{1}{\gamma^2}$$

Problem 9 [10 points]
Write down a modified algorithm for the multi-way Perceptron, as pseudo-code. Explain the motivation for your proposed modification.

pseudo-code:

```
while epoch < max_epoch:
    for i in 1...N:
        each_class_score = x_i · W + b
        predicted_class = each_class_score.argmax()

        if predicted_class != actual_class:
            weight vector for predicted class -= lr * x_i
            weight vector for actual class += lr * x_i
            b[predicted_class] -= lr
            b[actual_class] += lr

    if number of wrong predicts < stop_threshold:
        return
    else:
        epoch += 1    # continue to next epoch
        lr = lr * decay
```

★ • for lr in 1 0.5 0.3:

decay = 0.95 (call it ξ)

⇒ results same:

• because: take learning rate η_1 and η_2, say $\eta_2 = k\eta_1$, then

$$W_2 = \sum_{t=1}^{T} \xi^{i(t)} \eta_2 \cdot y_{i(t)} \cdot X_{i(t)}$$

$$= \underbrace{\xi^{T} \cdot \eta_2}_{\text{a constant}} \cdot \sum_{t=1}^{T} \cdot y_{i(t)} \cdot X_{i(t)}$$

so W_2 is proportional to W..

★Thus, we need to change the decay rate.

$$\begin{array}{c} x_1 \\ \vdots \\ x_N \end{array} \begin{pmatrix} [x_{i1} \cdots x_{iD}] \end{pmatrix} \quad \begin{array}{c} c_i \\ \begin{pmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_D \end{bmatrix} \end{pmatrix} \end{array} + [b_0 \cdots b_q]$$

$X \quad W \quad b$

$$x_i W + b = [c_0 \cdots c_q]$$

↑ each_class_score

Motivation:

• $\text{old\_score} = x_i W_c + b_c$ ← predicted class

$$\text{new\_score} = x_i(W_c - lr \cdot x_i^T) + b_c - lr$$
$$= x_i W_c - lr\, x_i \cdot x_i^T + b_c - lr$$
$$= \text{old\_score} - lr(\|x_i\|^2 + 1)$$
$$< \text{old\_score}$$

Therefore, the update will make new_score < old_score.

• Similarly, for the actual class:

$$\text{new\_score} = x_i(W_c + lr \cdot x_i^T) + b_c + lr$$
$$= \text{old\_score} + lr(\|x_i\|^2 + 1)$$
$$> \text{old\_score}.$$

which means that after the update, the actual class will be more likely to chosen.