1. a) $y = \text{Keep}$:

$$E[J(y,t)] = \sum_{t' \in \{\text{Spam}, \text{NonSpam}\}} J(y,t') \cdot P(t=t')$$

$$= J(\text{Keep}, \text{Spam}) \cdot \underbrace{P(t=\text{Spam})}_{=0.2}$$

$$+$$

$$J(\text{Keep}, \text{NonSpam}) \cdot \underbrace{P(t=\text{NonSpam})}_{=0.8}$$

$$= 1 \times 0.2 + 0 \times 0.8$$

$$= 0.2$$

$y = \text{remove}$:

$$E[J(y,t)] = J(\text{remove}, \text{Spam}) \cdot P(t=\text{Spam})$$

$$+$$

$$J(\text{remove}, \text{NonSpam}) \cdot P(t=\text{NonSpam})$$

$$= 0 \times 0.2 + 500 \times 0.8$$

$$= 400$$

b) Let $P(t = \text{Spam} \mid x) = P_s$, $P(t = \text{NonSpam} \mid x) = 1 - P_s$

$y = \text{Keep}$:

$$E[\mathcal{J}(y,t) \mid x] = \sum_{t' \in \{\text{Spam}, \text{NonSpam}\}} \mathcal{J}(y,t') \cdot P(t' \mid x)$$

$$= \underbrace{\mathcal{J}(\text{Keep}, \text{Spam})}_{=1} \cdot P_s + \underbrace{\mathcal{J}(\text{Keep}, \text{NonSpam})}_{=0} \cdot (1 - P_s)$$

$$= P_s$$

$y = \text{Remove}$:

$$E[\mathcal{J}(y,t) \mid x] = \underbrace{\mathcal{J}(\text{Remove}, \text{Spam})}_{=0} \cdot P_s + \underbrace{\mathcal{J}(\text{Remove}, \text{NonSpam})}_{=500} \cdot (1 - P_s)$$

$$= 500(1 - P_s)$$

We keep if $P_s \leq 500(1 - P_s)$ (i.e. higher expected loss to remove)

$$\Rightarrow 501 P_s \leq 500$$

$$\Rightarrow P_s \leq \frac{500}{501}$$

remove if $P_s > \frac{500}{501}$ (i.e. higher expected loss to keep)

$$\Rightarrow y_* = \begin{cases} \text{Keep if } P_s \leq \frac{500}{501} \\ \text{Remove otherwise} \end{cases}$$

c) 4 cases: $(x_1, x_2) \in \{(0,0), (0,1), (1,0), (1,1)\}$

(0,0): $P((x_1, x_2) = (0,0) \mid t = Spam) = 0.45$

$P((x_1, x_2) = (0,0) \mid t = NonSpam) = 0.996$

Bayes rule: $P(A|B) = \frac{P(A \cap B)}{P(B)}$

$= \frac{P(B|A) \cdot P(A)}{P(B)}$

$= \frac{P(B|A) \cdot P(A)}{\sum_{x \in S} P(B|x) \cdot P(x)}$

$\Rightarrow P(t = Spam \mid (x_1, x_2) = (0,0)) = \frac{0.45 \cdot P(t = Spam)}{0.45 \cdot P(t = Spam) + 0.996 \cdot P(t = NonSpam)}$

$= \frac{0.45 \times 0.2}{0.45 \times 0.2 + 0.996 \times 0.8}$

≈ 0.1015

$\leq \frac{500}{501}$

So by b), $y_* =$ Keep

(0,1): $P((x_1, x_2) = (0,1) \mid t = Spam) = 0.25$

$P((x_1, x_2) = (0,1) \mid t = NonSpam) = 0.002$

$$\Rightarrow P(t = Spam \mid (x_1, x_2) = (0,0)) = \frac{0.25 \cdot P(t = Spam)}{0.25 \cdot P(t = Spam) + 0.002 \cdot P(t = NonSpam)}$$

$$= \frac{0.25 \times 0.2}{0.25 \times 0.2 + 0.002 \times 0.8}$$

$$\approx 0.9690$$

$$\leq \frac{500}{501}, \text{ so } y_* = \text{Keep}$$

(1,0): $P((x_1, x_2) = (1,0) \mid t = Spam) = 0.18$

$P((x_1, x_2) = (1,0) \mid t = NonSpam) = 0.002$

$$\Rightarrow P(t = Spam \mid (x_1, x_2) = (1,0)) = \frac{0.18 \times 0.2}{0.18 \times 0.2 + 0.002 \times 0.8}$$

(by formula similar to above cases)

$$\approx 0.9574$$

$$\leq \frac{500}{501}, \text{ so } y_* = \text{Keep}$$

(1,1): $P((x_1, x_2) = (1,1) \mid t = \text{Spam}) = 0.12$

$P((x_1, x_2) = (1,1) \mid t = \text{NonSpam}) = 0$

$$\Rightarrow P(t = \text{Spam} \mid (x_1, x_2) = (1,1)) = \frac{0.12 \times 0.2}{0.12 \times 0.2 + 0 \times 0.8}$$

(by formula similar to above cases)

$$= 1$$

$$> \frac{500}{501}, \text{ so } y_* = \text{Remove}$$

d) Again, consider 4 cases:

(0,0): $y_* = \text{Keep}$: Let $P(t = \text{Spam} \mid (x_1, x_2) = (0,0)) \cong 0.1015 = p_{s,(0,0)}$

$$E[J(y,t) \mid (x_1, x_2) = (0,0)] = \sum_{t' \in \{\text{Spam}, \text{NonSpam}\}} J(\text{Keep}, t = t') \times P(t = t' \mid (x_1, x_2) = (0,0))$$

$$= 1 \times p_{s,(0,0)} + 0 \times (1 - p_{s,(0,0)})$$

$$\cong 0.1015$$

(0,1):

$y_* = \text{Keep}$: Let $P(t = \text{Spam} \mid (x_1, x_2) = (0,1)) \cong 0.9690 = p_{s,(0,1)}$

$$E[\mathcal{J}(y,t) \mid (X_1,X_2)=(0,1)] = 1 \times p_{s,(0,1)} + 0 \times (1-p_{s,(0,1)})$$

$$\cong 0.9690$$

(1,0):

y = Keep : Let $P(t=\text{Spam} \mid (X_1,X_2)=(1,0)) \cong 0.9574 = p_{s,(1,0)}$

$$E[\mathcal{J}(y,t) \mid (X_1,X_2)=(1,0)] = 1 \times p_{s,(1,0)} + 0 \times (1-p_{s,(1,0)})$$

$$\cong 0.9574$$

(1,1):

y_* = Remove :

$$E[\mathcal{J}(y,t) \mid (X_1,X_2)=(1,1)] = 0 \times p_{s,(1,1)} + 500 \times (1-p_{s,(1,1)})$$

$$= 0$$

Also, $P((X_1,X_2)=(0,0)) = P((X_1,X_2)=(0,0) \mid t=\text{Spam}) \times P(t=\text{Spam})$

$$+$$

$$P((X_1,X_2)=(0,0) \mid t=\text{NonSpam}) \times P(t=\text{NonSpam})$$

$$= 0.45 \times 0.2 + 0.996 \times 0.8$$

$$= 0.8868$$

$$P((X_1, X_2) = (0,1)) = 0.25 \times 0.2 + 0.002 \times 0.8$$

$$= 0.0516$$

$$P((X_1, X_2) = (1,0)) = 0.18 \times 0.2 + 0.002 \times 0.8$$

$$= 0.0376$$

$$P((X_1, X_2) = (1,1)) = 0.12 \times 0.2 + 0 \times 0.8$$

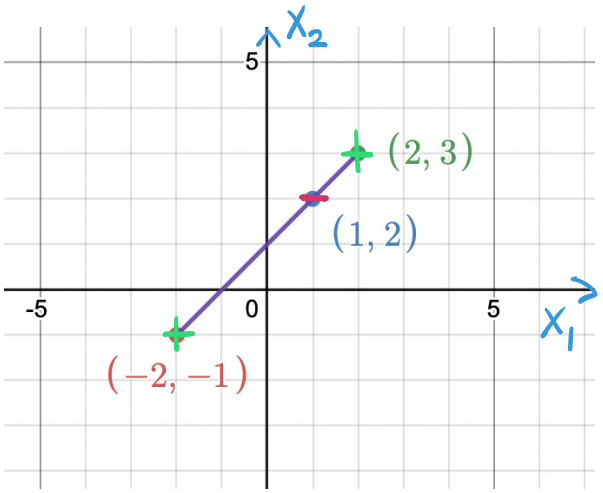
$$= 0.024$$

$$\text{So } E[J(y_*, t)] = \sum_{(k_1,k_2) \in \{(0,0),(0,1),(1,0),(1,1)\}} E[J(y,t) | (X_1, X_2) = (k_1, k_2)] \cdot P((X_1, X_2) = (k_1, k_2))$$

$$= 0.1015 \times 0.8868 + 0.9690 \times 0.0516 + 0.9574 \times 0.0376 + 0$$

$$= 0.1760$$

2. a)



Half spaces are convex, i.e. if 2 points lie in a half space, the line segment connecting them also lie in that half space.

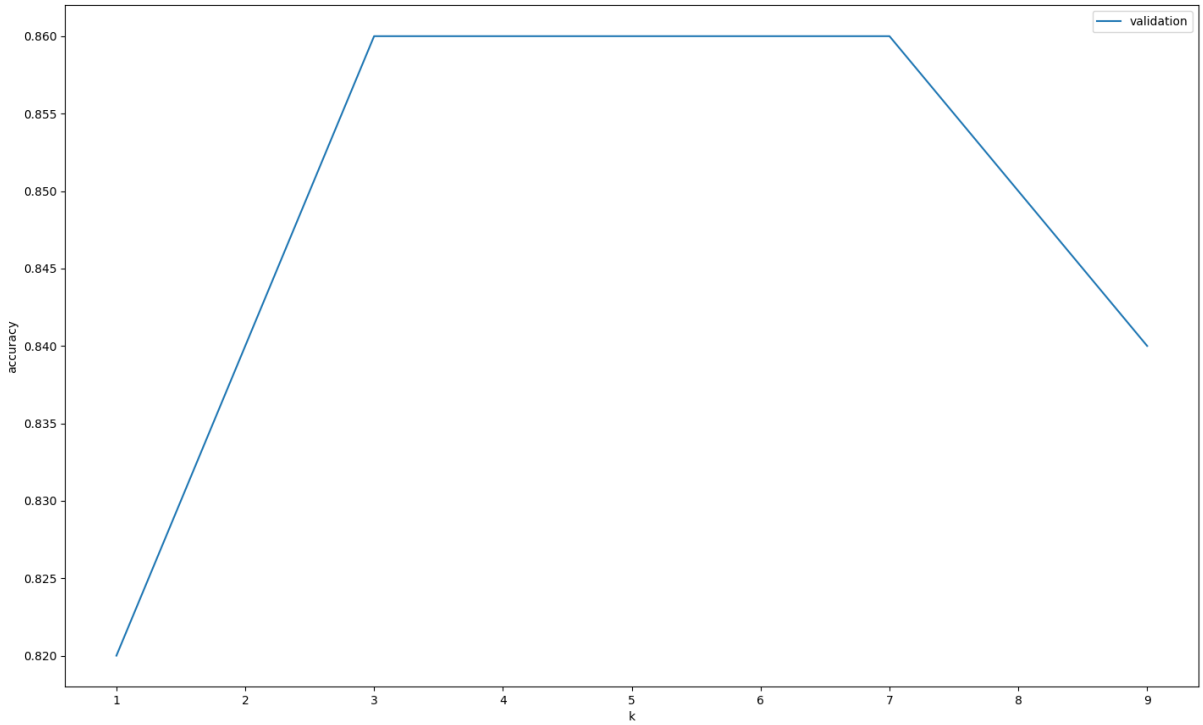
(Given in lec05)

Let a point be (x_1, x_2), the points (-2,-1) and (2,3) lie in a half space $y=1$, while the point (1,2) lie in half space $y=0$

However, (1,2) also lies on the line segment connecting (-2,-1) and (2,3), so (1,2) is also in half space $y=1$

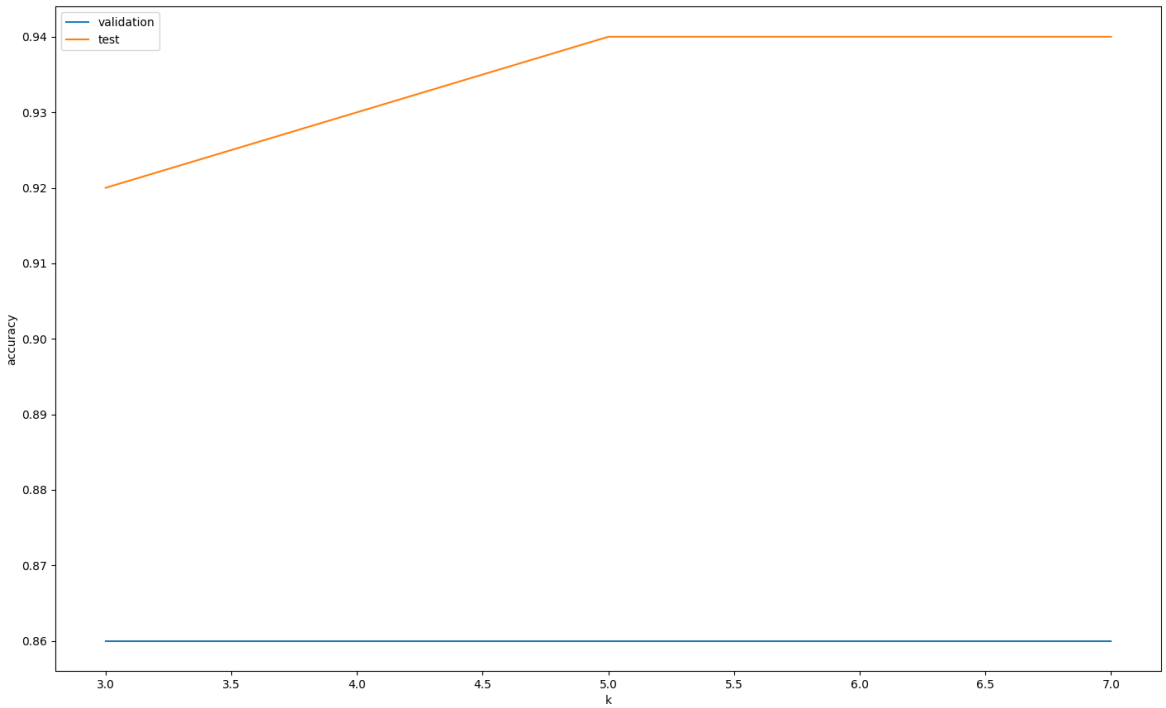
But a point can't lie in both half spaces, hence data set not linearly separable.

3.1 a) Graph is as follows:



b) I choose $k^* = 5$ since it provides the highest validation accuracy of 0.860, and we can conveniently access k^*-2 and k^*+2 (from results in a))

Validation & Test Accuracies:



According to graphs from a) and b), as k increases, validation accuracy first increases, then stops, and eventually decreases. Given k^*, k^*-2 and k^*+2, test accuracy only increases and stops. However, it is expected to decrease similar to the validation accuracy, since a large k gives a model high bias and low variance.

3.2 b)

For mnist_train:

After some testing, I decided on the following hyperparams.

- Learning rate = 0.3
- Initial weight = 0.0
- Num. iterations = 200

The performance is as follows:

```
diff = 8.447481221231437e-09

            Train: cross entropy = 0.033703160470255164, accuracy = 1.0,
            Validation: cross entropy = 0.19481483365572416, accuracy = 0.88,
            Test: cross entropy = 0.20475866552462577, accuracy = 0.92
```

For mnist_train_small:

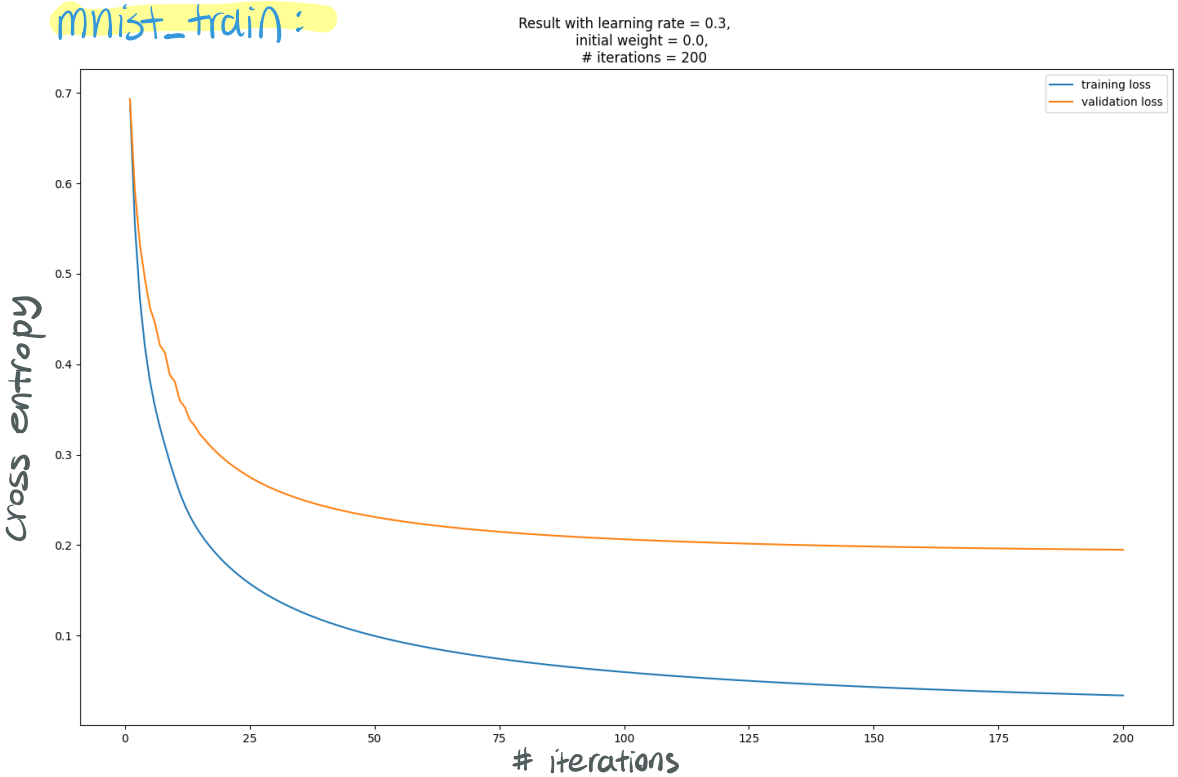
- Learning rate = 0.01
- Initial weight = 0.0
- Num. iterations = 200

Performance:

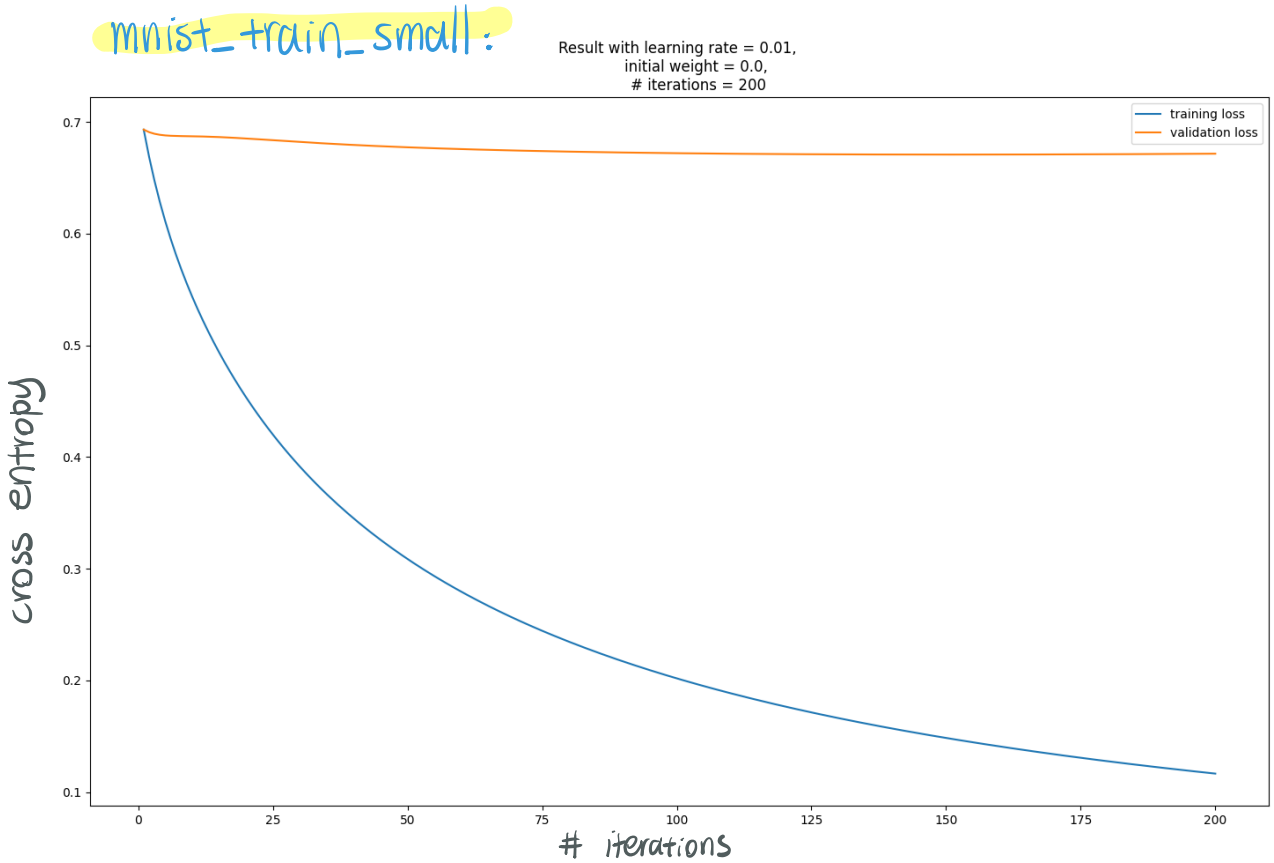
```
diff = 2.6548066960307165e-08

            Train: cross entropy = 0.11659436923096458, accuracy = 1.0,
            Validation: cross entropy = 0.6715799219714644, accuracy = 0.64,
            Test: cross entropy = 0.5411473066396657, accuracy = 0.76
```

c)

mnist_train:



mnist_train_small:



The results do not change after running the code several times on the same hyperparams., so the current setting gives consistent results.

⇒ If result inconsistent given the same training data, then we can tune the parameters based on the general trend of the result.

That is, we run the code many times and take an avg. of the performance on the same parameters, so that we may compare the avg. performance for different parameters.

4. a) Want to minimize $J(w) = \frac{1}{2}\sum_{i=1}^{N} a^{(i)}(y^{(i)} - \underbrace{w \cdot x^{(i)}}_{=\sum_{j'=1}^{D} w_{j'} x_{j'}^{(i)}})^2 + \frac{\lambda}{2}\underbrace{\|w\|^2}_{=\sum_{j'=1}^{D} w_{j'}^2}$

$\Rightarrow$ for each w_j, $\frac{\partial J}{\partial w_j} = 0$:

$$\Rightarrow \frac{\partial J}{\partial w_j} = \frac{1}{2}\sum_{i=1}^{N} a^{(i)} \cdot 2\left(y^{(i)} - \sum_{j'=1}^{D} w_{j'} x_{j'}^{(i)}\right)\left(-x_j^{(i)}\right) + \frac{\lambda}{2} \cdot 2w_j$$

$$= \sum_{i=1}^{N} a^{(i)} \cdot \left(\left(\sum_{j'=1}^{D} w_{j'} x_{j'}^{(i)}\right) - y^{(i)}\right)\left(x_j^{(i)}\right) + \lambda w_j$$

$$= \sum_{i=1}^{N} x_j^{(i)} a^{(i)} \left(\left(\sum_{j'=1}^{D} w_{j'} x_{j'}^{(i)}\right) - y^{(i)}\right) + \lambda w_j$$

So for all D columns:

$$\underbrace{\begin{pmatrix} | & & | \\ x^{(1)} & \cdots & x^{(N)} \\ | & & | \end{pmatrix}}_{= X^T} \underbrace{\begin{pmatrix} a^{(1)} & & 0 \\ & \ddots & \\ 0 & & a^{(N)} \end{pmatrix}}_{= A} \left[\begin{pmatrix} — & x^{(1)} & — \\ & \vdots & \\ — & x^{(N)} & — \end{pmatrix} \begin{pmatrix} w_1 \\ \vdots \\ w_D \end{pmatrix} - \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix} \right] + \lambda \begin{pmatrix} w_1 \\ \vdots \\ w_D \end{pmatrix} = 0$$

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

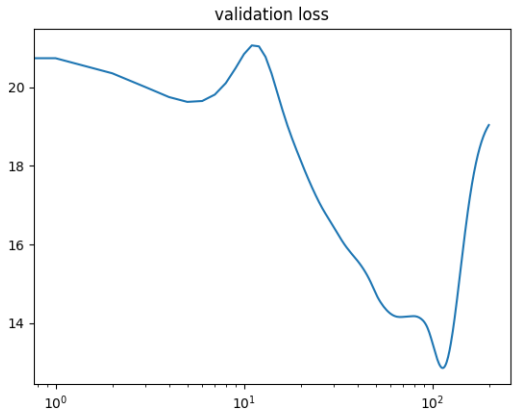
$$= X^T A X w - X^T A y + \lambda w$$

$$= (X^T A X + \lambda I) w - X^T A y = 0$$

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

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

c)



d) As $\tau \to \infty$, the algorithm will perform poorly due to overfitting, as shown by the validation performance for large τ in c). Similarly, $\tau \to 0$ also results in bad performance due to underfitting, as shown by small τ in c).

e) Traditional linear regression is limited to linear relationships, but Locally weighted regression is more flexible (works for non-linear relationships)

However, Locally weighted regression requires the storage of entire training data, while the traditional version only stores the weights.