Assignment 1 Report

This is an outline for your report to ease the amount of work required to create your report. Jupyter notebook supports markdown, and I recommend you to check out this cheat sheet. If you are not familiar with markdown.

Before delivery, **remember to convert this file to PDF**. You can do it in two ways:

1. Print the webpage (ctrl+P or cmd+P)
2. Export with latex. This is somewhat more difficult, but you'll get somehwat of a "prettier" PDF. Go to File -> Download as -> PDF via LaTeX. You might have to install nbconvert and pandoc through conda; `conda install nbconvert pandoc`.

Task 1

task 1a)

$$ C(w) = - \frac{1}{N} \sum_{n = 1}^{N} y^n \ln(\hat{y}^n) + (1-y^n) \ln(1-\hat{y}^n) , \hat{y} = f(x) = \frac{1}{1+e^{-w^Tx}} $$
Hint gives us that $ \frac{\partial f(x^n)}{\partial w_{i}} = x_i^n f(x^n)(1-f(x^n))$.

Partial derivate $$ \frac{\partial}{\partial w_j} \ln(\hat{y}^n)= \frac{\partial}{\partial w_j} \ln(f(x^n)) = \frac{1}{f(x^n)} \frac{\partial}{\partial w_j}f(x^n) = \frac{1}{f(x^n)} x_j^n f(x^n)(1-f(x^n)) = x_{j}^{n} (1-f(x^n)) $$

$$ \frac{\partial}{\partial w_j} \ln(1-\hat{y}^n) = \frac{\partial}{\partial w_j} \ln(1-f_w(x^n))= \frac{1}{1-f_w(x^n)} \frac{\partial }{\partial w_j} (1-f_w(x^n))= \frac{-1}{1-f(x^n)} x_j^n f(x^n)(1-f(x^n)) =- x_{j}^{n} f(x^{n}) $$
This gives $$ \frac{\partial}{\partial w_j} C^n(w)= -\frac{\partial}{\partial w_j}(y^n \ln(\hat{y}^n) + (1-y^n) \ln(1-\hat{y}^n)) = -y^n x_{j}^{n} (1-f(x^n)) + (1-y^n) x_{j}^{n} f(x^{n}) = -(y^n - \hat{y}^n) x_j^n $$

$$ \frac{\partial}{\partial w_j} C^n(w)= -(y^n - \hat{y}^n) x_j^n $$

task 1b)

$$ \frac{\partial}{\partial w_{kj}} C^n(w) = -\frac{1}{K}x_j^n (y_k^n-\hat{y}_k^n). $$$$ C(w) = \frac{1}{N} \sum_{n=1}^N C^n(w)= -\frac{1}{N\cdot K} \sum_{n=1}^N \sum_{k=1}^K y_k^n \ln(\hat{y_k^n}) $$
We observe $ \hat{y}_k^n = \frac{e^{z_k^n}}{\sum_{k'=1}^K e^{z_{k'}^n}} $ where $ z_k^n = w_k^T x^n $

$$ \frac{\partial}{\partial w_{kj}} C^n(w) = \frac{\partial C^n(w)}{\partial \hat{y}_k^n} \frac{\partial \hat{y}_k^n}{\partial z_k^n} \frac{\partial z_k^n}{\partial w_{kj}} $$
Hints give us that $ \sum_{k=1}^K y_k^n =1 $ and $ \ln(\frac{a}{b}) = \ln(a)-\ln(b) $

Two different cases:

case i) $i = k $

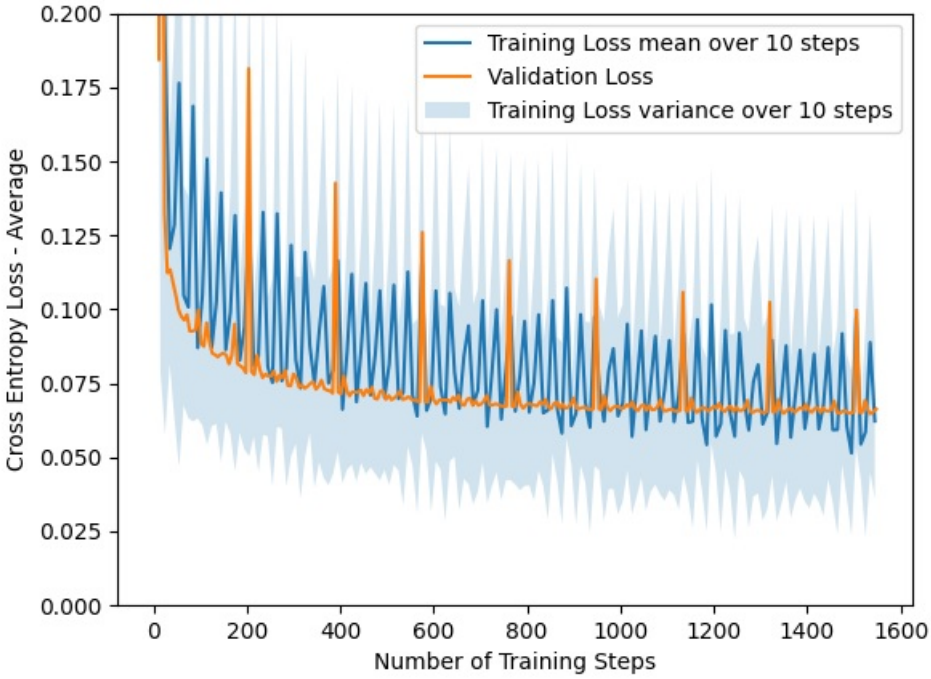
$$ \partial \frac{\hat{y}_i^n}{z_k^n} = \frac{e^{z_k^n}}{\sum_{k'=1}^K e^{z_{k'}^n}} -\frac{e^{z_k^n} e^{z_k^n}}{(\sum_{k'=1}^K e^{z_{k'}^n})^2} = \hat{y}_k^n(1-\hat{y}_k^n) $$
and case ii)$ i != k$ $$ \partial \frac{\hat{y}_i^n}{z_k^n} = \frac{-e^{z_i^n} e^{z_k^n}}{(\sum_{i} e^{z_k^n})^2} = - \hat{y}_i^n *\hat{y}_k^n $$

Finally we get the following:

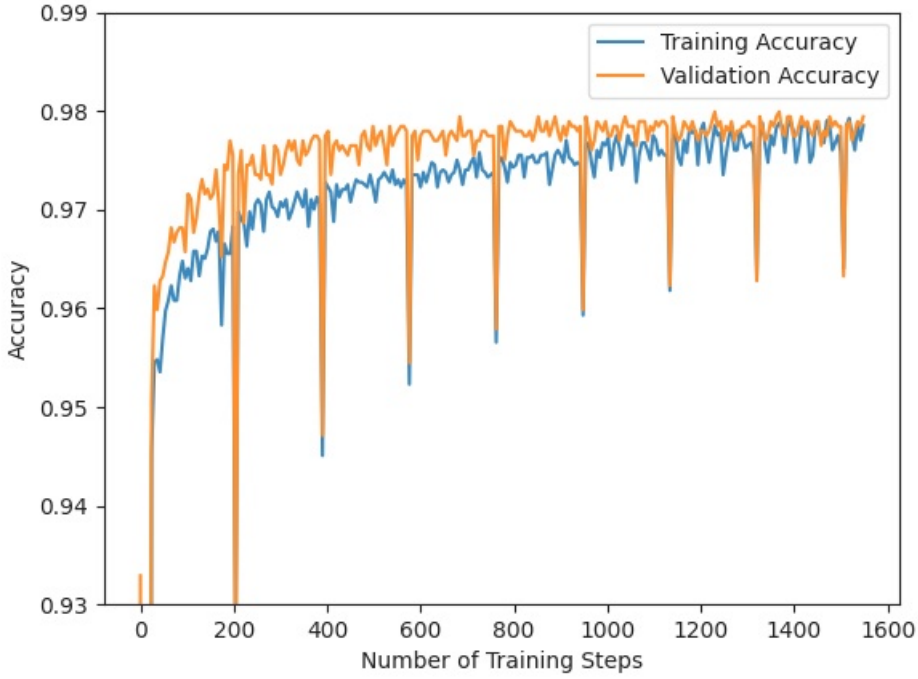
$$ \frac{\partial}{\partial w_{kj}} C^n(w) =-\frac{1}{N}\sum_{n=1}^N y_k^n (x_i^n - \frac{e^{a_k^n}x_i^n}{\sum_{k'}{e^{a_k^n}}} - y_k^n\frac{e^{a_k^n}x_i^n}{\sum_{k'}{e^{a_k^n}}}) $$
$=-\frac{1}{N}\sum_{n=1}^N x_i^n(y_k^n-\hat{y}_k^n)$ $$ =-x_i^n(y_k^n-\hat{y}_k^n)

Task 2

Task 2b)



Task 2c)

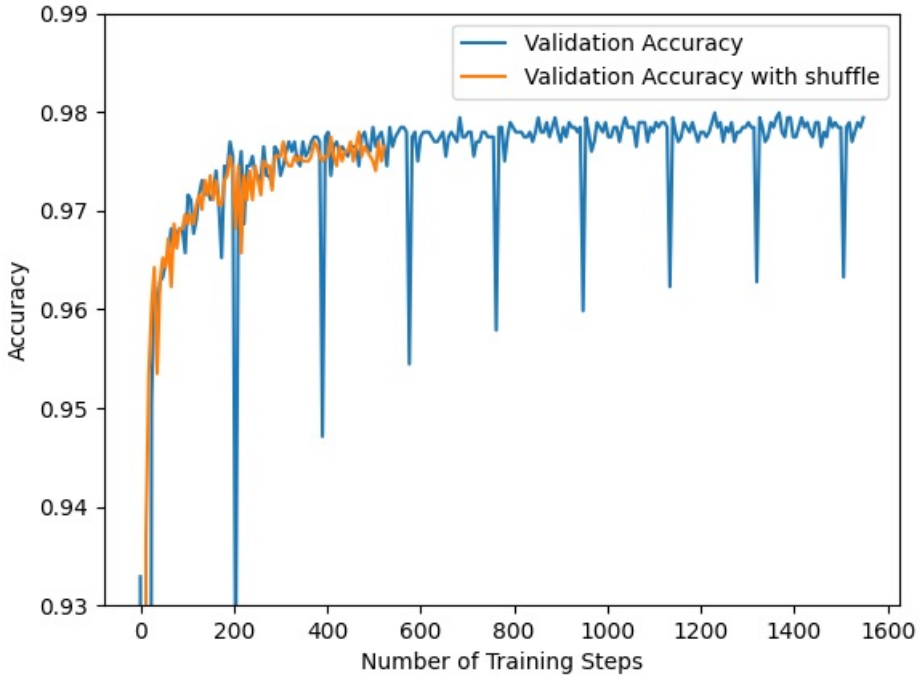


Task 2d)

The early stopping kicked in after 16 epochs when running with shuffle=True, and when shuffle=False it stops at epoch 33. What we see in the diagram is with shuffling set to true

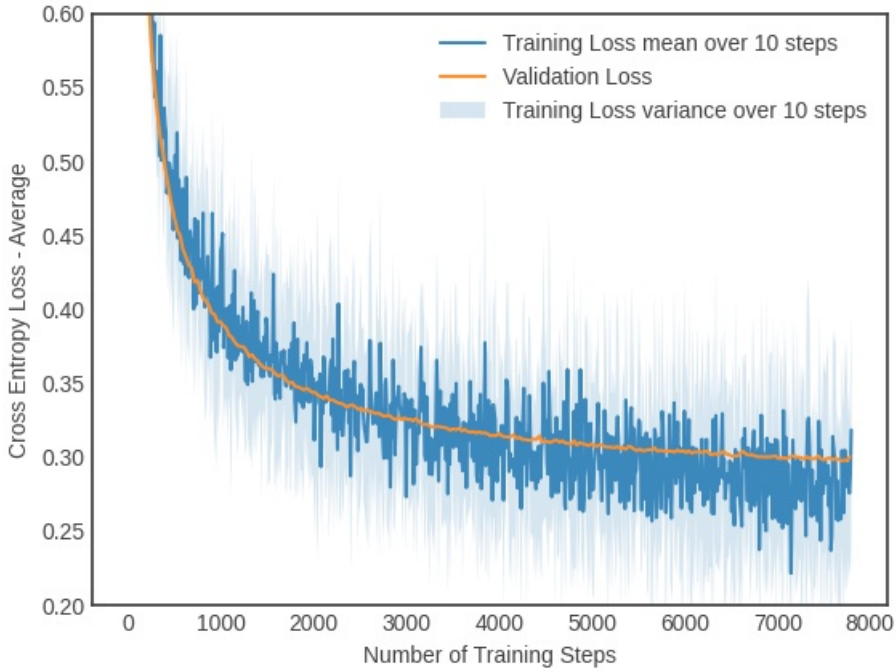
Task 2e)

In the plot we see that the spikes are almost nonexistent when using shuffling. This is because when shuffling is set to true the model receives a completely new random batch at each epoch. When there isn't shuffling the model spikes whenever it receives a batch that it seemingly does not predict particularly well. We say that shuffling helps keep the model general. Shuffling data means that we reduce the amount of times we go into a local minima during gradient descent since the points are chosen more at random.

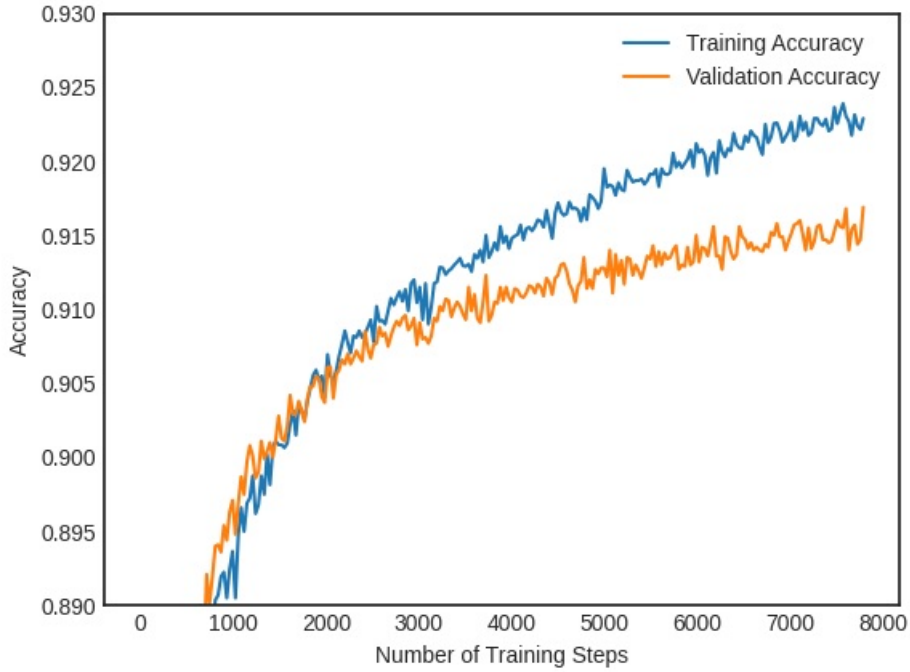


Task 3

Task 3b)



Task 3c)



Task 3d)

We see tendencies of overfitting, since the accuracy grows higher on the training data than on the validation data. This means that the model becomes better at predicting on the training data but does not improve at predicting validation on the test data. In our case however, the validation accuracy still has a growing trend for 500 epochs so while it is showing a general trend of growing towards overfitting, I would argue it is not overfitting. Some deviation between validation data and training data accuracy will always be a reality. I would argue this model has a good amount of generalization after training.

Task 4

Task 4a)

The new cost function is as given in (9) $ J(w) = C(w) + \lambda R(w) $ where $R(w)$ is the complexity penalty and λ is the strength of the regularization.

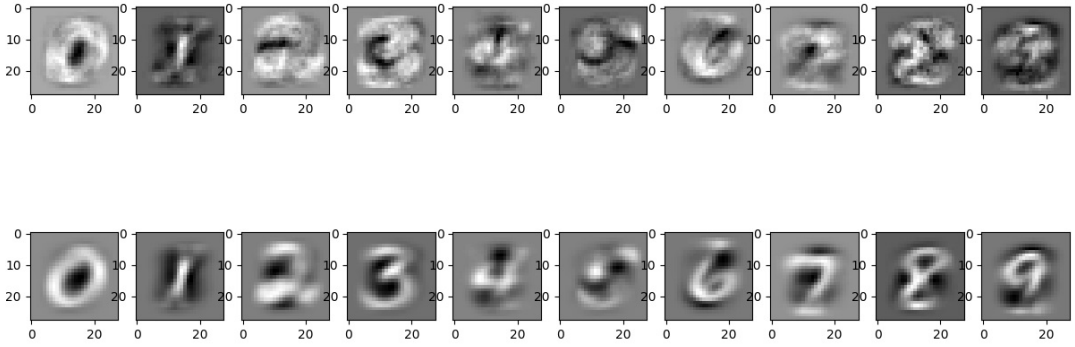
L2 ridge regression is implemented as $$ R(w)=\|w\|^2= \sum_{i,j} w_{i,j}^2 = \sum_{i=1}^{I} \sum_{j=1}^{J} w_{i,j}^2. $$ Its gradient can be calculated as $$ \frac{\partial}{\partial w}R(w) = \sum_{i=1}^{I} \sum_{j=1}^{J} w_{i,j}^2 \frac{\partial}{\partial w}w_{i,j} = \sum_{i=1}^{I} 2 w_i = 2\sum_{i=1}^{I} w_i $$

Which gives our cost function derivative the following form $$ \frac{\partial}{\partial w}J(w) = -x_i^n(y_k^n-\hat{y}_k^n) + 2 \lambda \sum_{i=1}^{I} w_i $$

Task 4b)

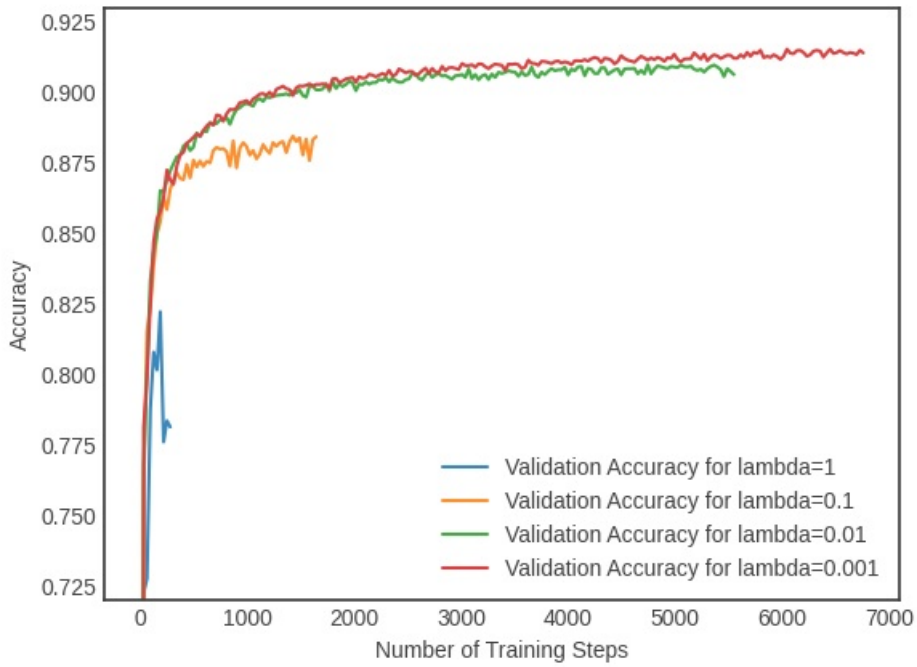
In the plot below we see the weights with lambda=1.0 and lambda=0.0, after a L2 term is added to the cost function. Without L2 penalization the weights have more noise as we see below. However, when we set a lambda=1.0 we see a reduced noise due to the L2 function penalizing the weight function thus reducing the noise in the model. To reiterate, each input is given less weight and the cost function is greedier in its allocation. This is what results in the denoising.

Digit weights. Top lambda=0, bottom lambda=1



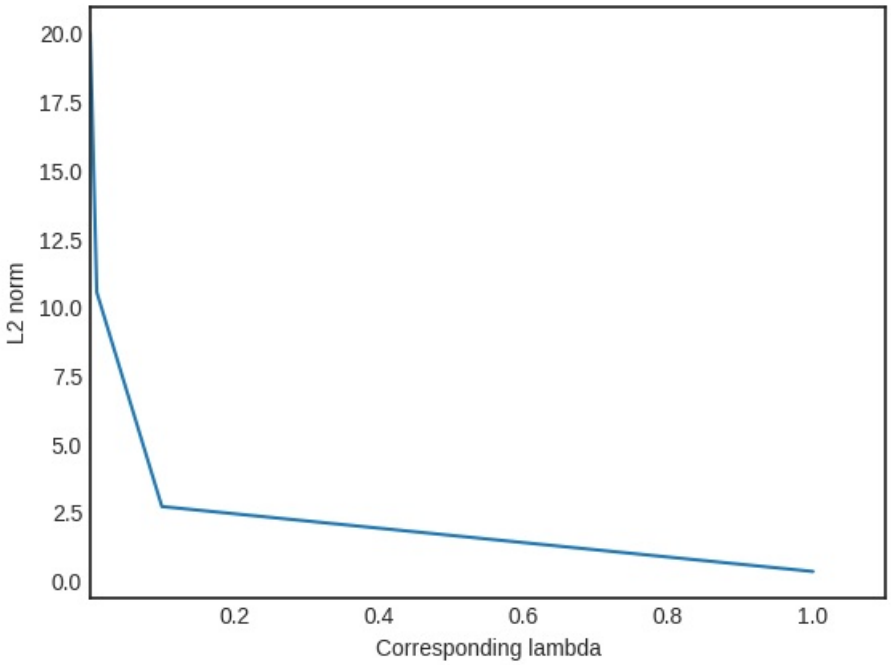
Task 4c)

Below we find the plot with different lambda values. note that early stopping is turned on which is why some stop earlier. Low lambda values give better results as we see. However, lambda=0 is worse than small lambda values meaning the L2 regularization is helping the model.



Task 4d)

By adding the L2 function to the cost we punish the weight sizing, and this in turn results in lower accuracy. It may seem counterproductive to use an L2 function for this reason, but in turn it makes the model much more generalized and reduces overfitting. Lambda needs to be chosen with care as too high or too low can result in under- or overfitting.



Task 4e)

Increasing lambda decreases the length of the L2 norm