Homework 3

Monday, November 18, 2024 12:09 PM

Problem 1)

$$P(y=1|x_1,\dots,x_d) = \frac{P(y=1)\prod_{i=1}^{d} P(x_i|y=1)}{P(x_1,\dots,x_d)} \alpha\, \theta_1 \prod_{i=1}^{d} \theta_{i1}^{x_i}(1-\theta_{i1})^{1-x_i}$$

$$P(y=0|x_1,\dots,x_d) = \sim\sim\sim\sim$$

↳ classification rule

$$\frac{P(y=1|x_1,\dots,x_d)}{P(y=0|x_1,\dots,x_d)} > 1$$

$$\rightarrow \frac{P(y=1|x_1,\dots,x_d)}{P(y=0|x_1,\dots,x_d)} > 1 \Rightarrow b + \sum_{i=1}^{d} \omega_i x_i > 0$$

$$\rightarrow \frac{\theta_1 \prod_{i=1}^{d} \theta_{i1}^{x_i}(1-\theta_{i1})^{1-x_i}}{\theta_0 \prod_{i=1}^{d} \theta_{i0}^{x_i}(1-\theta_{i0})^{1-x_i}} > 1$$

$$\rightarrow \left(\frac{\theta_1}{\theta_0}\right) \frac{\prod_{i=1}^{d} \theta_{i1}^{x_i}(1-\theta_{i1})^{1-x_i}}{\prod_{i=1}^{d} \theta_{i0}^{x_i}(1-\theta_{i0})^{1-x_i}} > 1$$

$$\rightarrow \left(\frac{\theta_1}{\theta_0}\right) \frac{\prod_{i=1}^{d} \left(\frac{\theta_1}{1-\theta_{i1}}\right)^{x_i}(1-\theta_{i1})}{\prod_{i=1}^{d} \left(\frac{\theta_0}{1-\theta_{i0}}\right)^{x_i}(1-\theta_{i0})} > 1$$

now the classic take the log

$$\rightarrow \log\left(\left(\frac{\theta_1}{\theta_0}\right) \frac{\prod_{i=1}^{d} \theta_{i1}^{x_i}(1-\theta_{i1})^{1-x_i}}{\prod_{i=1}^{d} \theta_{i0}^{x_i}(1-\theta_{i0})^{1-x_i}}\right) > \log(1)$$

$$\rightarrow \log\left(\frac{\theta_1}{\theta_0}\right) \sum_{i=1}^{d} \log\left(\frac{\theta_{i1}^{x_i}(1-\theta_{i1})^{1-x_i}}{\theta_{i0}^{x_i}(1-\theta_{i0})^{1-x_i}}\right) > 0$$

$$\rightarrow \log\left(\frac{\theta_1}{\theta_0}\right) + \sum_{i=1}^{d} \left[x_i \log\left(\frac{\theta_{i1}}{\theta_{i0}}\right) + (1-x_i)\log\left(\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)\right] > 0$$

$$\rightarrow \log\left(\frac{\theta_{i1}}{\theta_{i0}}\right) + \sum_{i=1}^{d} \left[x_i \log\left(\frac{\theta_{i1}}{\theta_{i0}}\right) + \log\left(\frac{1-\theta_{i1}}{1-\theta_{i0}}\right) - x_i \log\left(\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)\right] > 0$$

$$\rightarrow \log\left(\frac{\theta_1}{\theta_0}\right) + \sum_{i=1}^{d} \left(x_i \log\left(\frac{\theta_{i1}(1-\theta_{i0})}{\theta_{i0}(1-\theta_{i1})}\right) + \log\left(\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)\right) > 0$$

$$\rightarrow \log\left(\frac{\theta_1}{\theta_0}\right) + \sum_{i=1}^{d} x_i \log\left(\frac{\theta_{i1}(1-\theta_{i0})}{\theta_{i0}(1-\theta_{i1})}\right) + \sum_{i=1}^{d} \log\left(\frac{1-\theta_{i1}}{1-\theta_{i0}}\right) > 0$$

$$\rightarrow \underbrace{\log\left(\frac{\theta_1}{\theta_0}\right) + \sum_{i=1}^{d} \log\left(\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)}_{b} + \sum_{i=1}^{d} \underbrace{x_i}_{x_i} \underbrace{\log\left(\frac{\theta_{i1}(1-\theta_{i0})}{\theta_{i0}(1-\theta_{i1})}\right)}_{\omega_i} > 0$$

This looks now the same as $b + \sum_{i=1}^{d} \omega_i x_i > 0$

$$b = \log + \sum_{i=1}^{d} \log\left(\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)$$

$$b = \log + \sum_{i=1}^{d} \log\left(\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)$$

$$x_i = x_i$$

$$w_i = \log\left(\frac{\theta_{i1}(1-\theta_{i0})}{\theta_{i0}(1-\theta_{i0})}\right) = \log\left(\frac{\theta_{i1}}{\theta_{i0}}\right) - \log\left(\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)$$

Problem 2)

assume a uniform prior so that
$P(y=0) = P(y=1)$

$$P(y=1 | X_1 = x_1) > P(y=0 | X_1 = x_1)$$

$$P(y=1 | X_1 = x_1, X_2 = x_2) > P(y=1 | X_1 = x_1)$$

we know that $P(X_2 = x_2 | y) = P(X_1 = x_1 | y)$

Single feature case

$$P(y=1 | X_1 = x_1, X_2 = x_2) = P(y=1 | X_1 = x_1) \cdot P(y=1 | X_2 = x_2)$$

Pasta for single feature

$$P(y=1 | X_1 = x_1) = \frac{P(y=1 | X_1 = x_1) P(y=1)}{P(X_1 = x_1)}$$

Duplicate feature

$$P(y=1 | X_1 = x_1, X_2 = x_2) = P(y=1 | X_1 = x_1)^2$$

Pasta for Duplicate feature

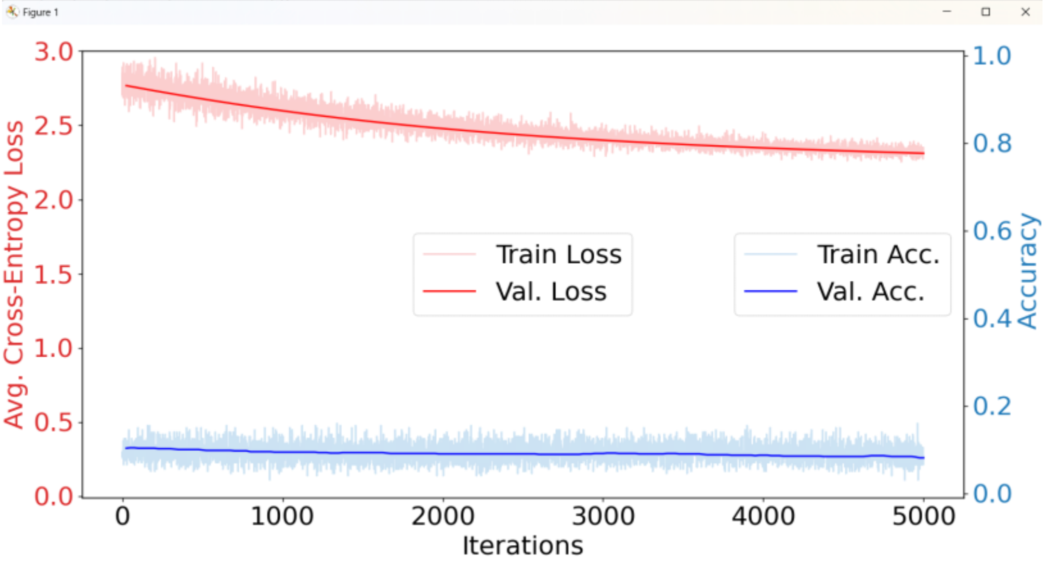
$$P(y=1 | X_1 = x_1, X_2 = x_2) = \frac{P(y=1 | X_1 = x_1)^2 P(y=1)}{P(X_1 = X_1)^2}$$

this shows that a second feature grows faster, we know this from the $P(y=1 | X_1 = x_1)^2$ comparitive to the single feature.

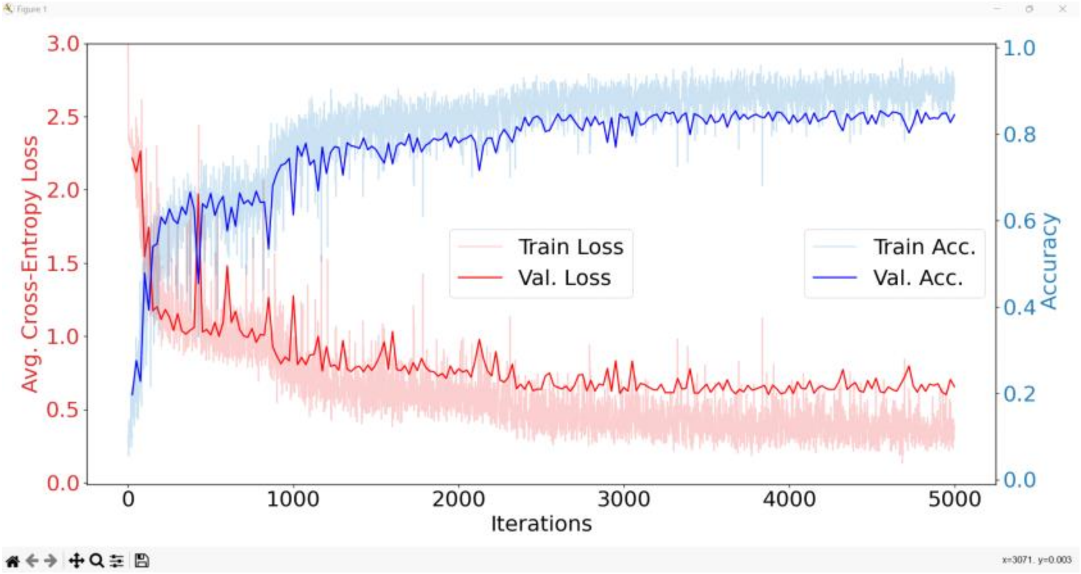
Problem 3

Problem 4)

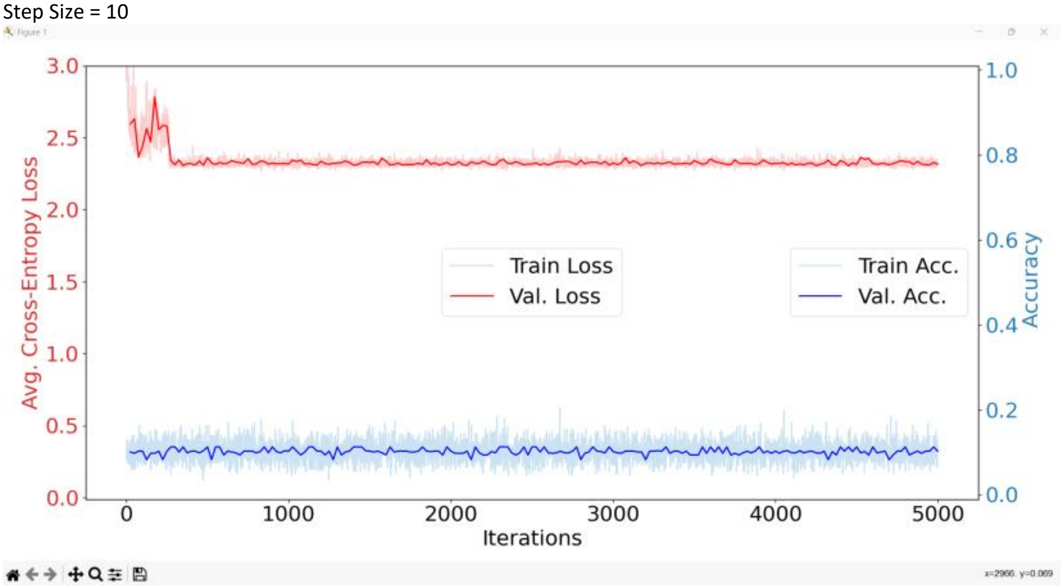
Step Size = 0.0001



Step Size = 5



Step Size = 10



a) Compare and contrast the learning curves with your curve using the default parameters. What do you observe in terms of smoothness, shape, and what performance they reach?

Comparative to the default graph with the default parameter we can see a few interesting things. On the step size of 0.0001 we see a much smooth line showing a slower progression. The loss decreases gradually and the accuracy shows slight improvement over an extended period of time. The shape of the curve is mostly flat due to the tiny step size and neither the training or accuracy approaches acceptable levels. The values at the end of the iterations reach nowhere near the values of the default graph and values.

For the graph with the step size of 5 we see the curve is less smooth and it jumps/fluctuates during training. The shape also is reflective of this where we see the loss drops quickly. This being said we see the loss oscillation showing that we may have overfitting or instability within the model. The accuracy reaches acceptable levels on train but we also see that validation accuracy is lower than the training accuracy. This could mean that in the performance aspect it would be accelerating the initial progress but it’s a widely unstable model seen by the noisy loss and validation lines.

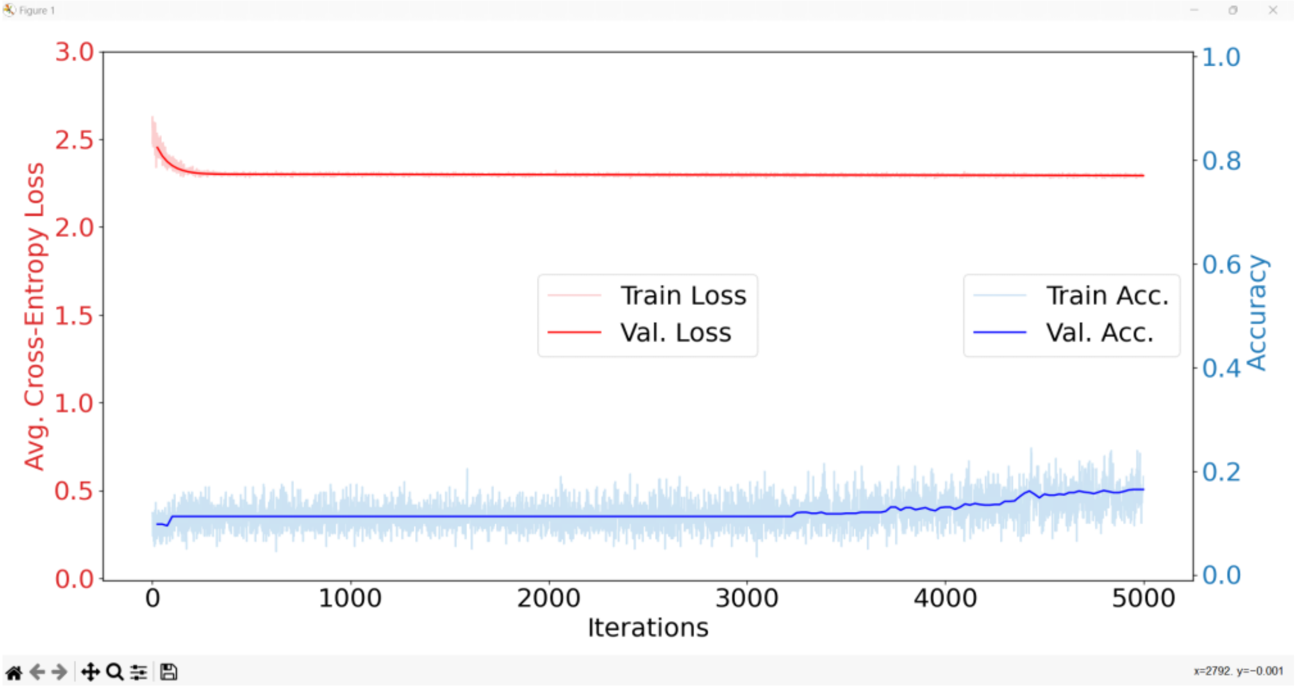
Lastly the step size of 10 seems to be the worst out of all of them with the smoothness being that of a straight parallel lines. With minimal if not any improvement in loss and accuracy. As for the shape the loss barely decrease and shows significant noise, the shape of the validation accuracy does not show many real change and stays close to the starting value. As for the performance the large step size causes divergence causing the model to not learning anything.

b) For (a), what would you expect to happen if the max epochs were increased?
Its not hard to say that increasing the max epochs would greatly benefit smaller step sizes then that of larger step sizes due to larger step sizes stepping over the true goal. We can see this where larger step sizes suffer from instability later in the iterations while only having improvement in the beginning if not no improvement. This

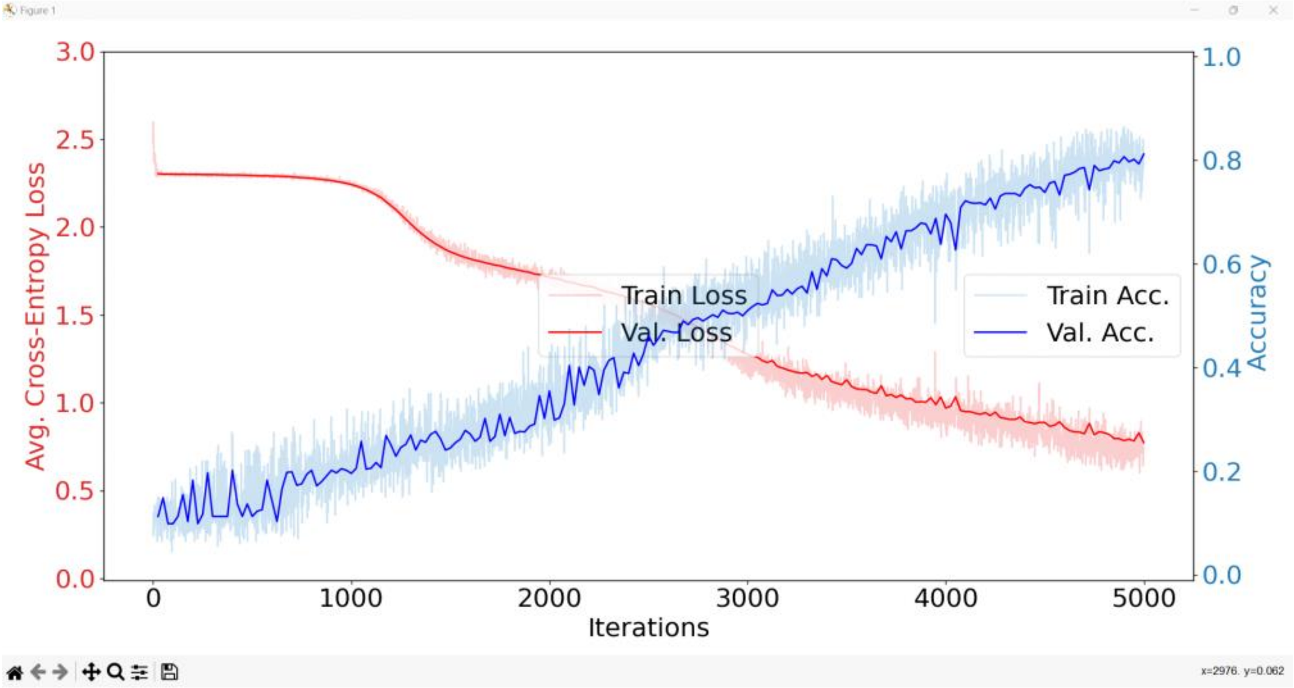
means that even if the iterations is increased we still will continue to have instability or divergence within the models with large step sizes.

Question 5)

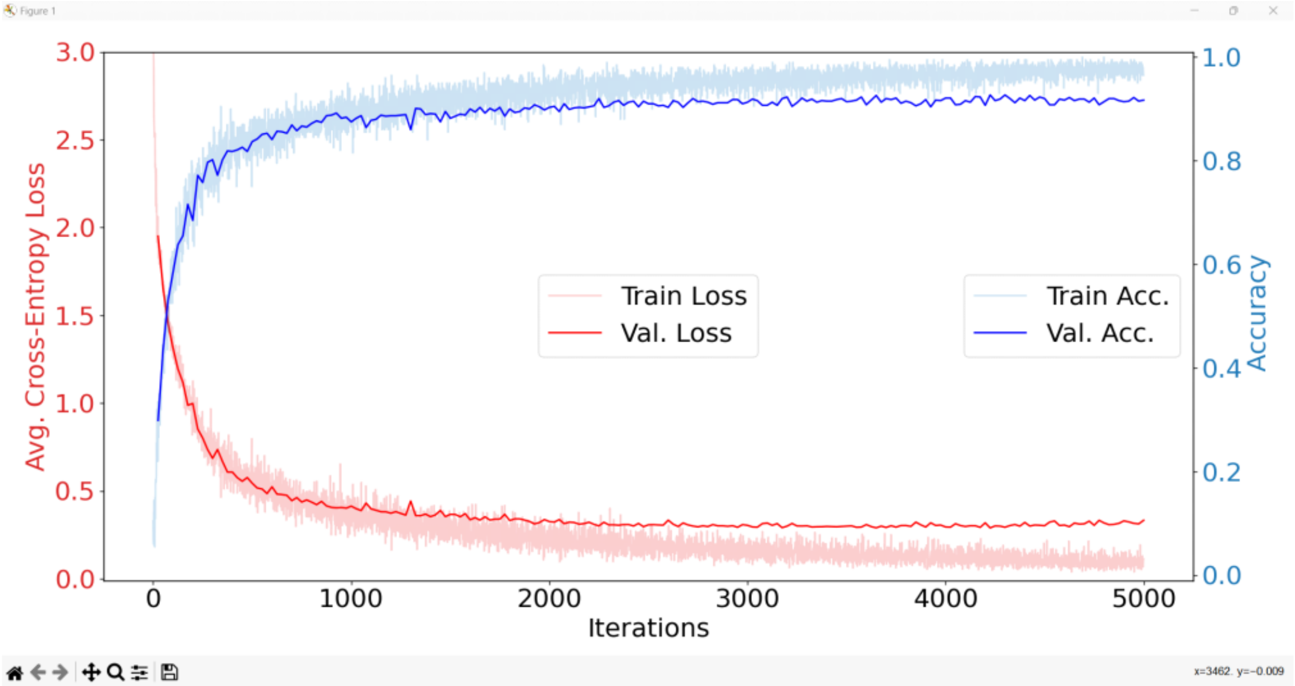
5-layer with sigmoid activation



5-layer with Sigmoid Activation with 0.1 step size



5-layer with ReLU activation



a) Compare and contrast the learning curves you observe and the curve for the default parameters in terms of smoothness, shape, and what performance they reach. Do you notice any differences in the relationship between the train and validation curves in each plot?

5-layer with sigmoid activation: The graph for this one is really not combative at all to the default graph in the bad way. Once again we see the sigmoid action to be much slower than that of the default. Both training loss and validation accuracy are much lower but show gradual improvement. The curves remain relatively smooth but fail to reach any meaningful values. This means overall the performance was overall worse. The training loss only decreased slowly and the gap between the train and validation shows underfitting.

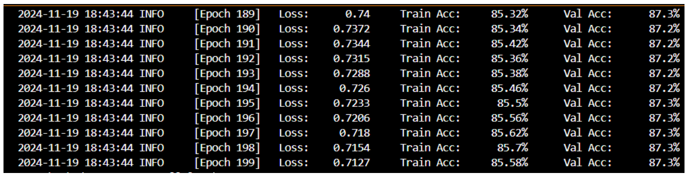
5-layer with sigmoid activation with 0.1 step size: We see the curve to be somewhat smooth but it to be less smooth then 1. For the shape we see the training and validation losses to decrease much more quickly comparative to 1, and the validation accuracy to improve at a somewhat linear rate. For the Performance we see an improvement to 1 but there seems to be some issues still as it reaches a low 80%. Interestingly here we can see that it took longer for the model to reach end results that comparable to the default value via iterations. We also notice that validation accuracy is still much more unsmooth then that of the default values despite reaching similar end values within the same iterations.

5-layer with ReLU activation: For the last one we saw a curve that is the smoothest and also much faster than that of 1 and 2 and the default. Shape wise we see the training and validation losses to drop faster and more significantly and the validation accuracy to improve rapidly and reach a higher value. For the performance we see that the ReLU avoid the previous plateaus being a lower undesirable amount for every category. Comparted to the default values we also see that we are seeing something of a better overall accuracy in both losses and acc meaning that overall our model maybe better.

b) If you observed increasing the learning rate in (2) improves over (1), why might that be?
This is most likely because a bigger learning rate compensates for the slow convergence leading to vanishing gradient. With small learning rates weights becomes extremely small and make it so the learning happens slower and slower making it so that it will stop learning evenly if not immediately. By having a bigger step size it makes these values bigger and makes it longer to reach this state. This being said this issue can be solved by switching off the sigmoid gradient in this case.

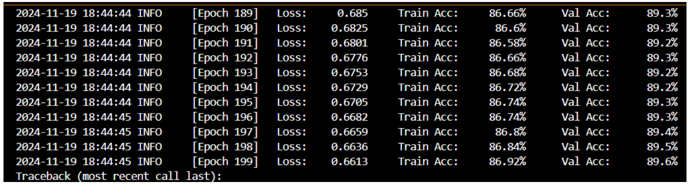
c) If (3) outperformed (1), why might that be? Consider the derivative of the sigmoid and ReLU functions.
Sigmoid functions exist between 0 and 1 this leads to inputs with large positive or negative values to get close to 0. This leads to smaller gradients during backpropagation slowing down learning within each step and every iteration, another words its stopping effective updates to the learning algorithm. On the other hand ReLU has derivative of 1 for the positive values and 0 for the negative values meaning that during pack proportion is allows for stronger gradients. This leads to better learning.

Question 6)
Default)



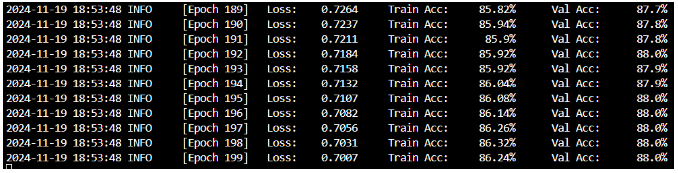
```
2024-11-19 18:43:44 INFO     [Epoch 189]   Loss:      0.74     Train Acc:     85.32%     Val Acc:      87.3%
2024-11-19 18:43:44 INFO     [Epoch 190]   Loss:    0.7372     Train Acc:     85.34%     Val Acc:      87.2%
2024-11-19 18:43:44 INFO     [Epoch 191]   Loss:    0.7344     Train Acc:     85.42%     Val Acc:      87.2%
2024-11-19 18:43:44 INFO     [Epoch 192]   Loss:    0.7315     Train Acc:     85.36%     Val Acc:      87.2%
2024-11-19 18:43:44 INFO     [Epoch 193]   Loss:    0.7288     Train Acc:     85.38%     Val Acc:      87.2%
2024-11-19 18:43:44 INFO     [Epoch 194]   Loss:     0.726     Train Acc:     85.46%     Val Acc:      87.2%
2024-11-19 18:43:44 INFO     [Epoch 195]   Loss:    0.7233     Train Acc:      85.5%     Val Acc:      87.3%
2024-11-19 18:43:44 INFO     [Epoch 196]   Loss:    0.7206     Train Acc:     85.56%     Val Acc:      87.3%
2024-11-19 18:43:44 INFO     [Epoch 197]   Loss:     0.718     Train Acc:     85.62%     Val Acc:      87.3%
2024-11-19 18:43:44 INFO     [Epoch 198]   Loss:    0.7154     Train Acc:      85.7%     Val Acc:      87.3%
2024-11-19 18:43:44 INFO     [Epoch 199]   Loss:    0.7127     Train Acc:     85.58%     Val Acc:      87.3%
```

Seed 1) Seed = 5



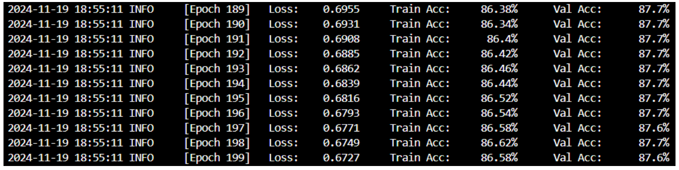
```
2024-11-19 18:44:44 INFO     [Epoch 189]   Loss:     0.685     Train Acc:     86.66%     Val Acc:      89.3%
2024-11-19 18:44:44 INFO     [Epoch 190]   Loss:    0.6825     Train Acc:      86.6%     Val Acc:      89.3%
2024-11-19 18:44:44 INFO     [Epoch 191]   Loss:    0.6801     Train Acc:     86.58%     Val Acc:      89.2%
2024-11-19 18:44:44 INFO     [Epoch 192]   Loss:    0.6776     Train Acc:     86.66%     Val Acc:      89.3%
2024-11-19 18:44:44 INFO     [Epoch 193]   Loss:    0.6753     Train Acc:     86.68%     Val Acc:      89.2%
2024-11-19 18:44:44 INFO     [Epoch 194]   Loss:    0.6729     Train Acc:     86.72%     Val Acc:      89.2%
2024-11-19 18:44:44 INFO     [Epoch 195]   Loss:    0.6705     Train Acc:     86.74%     Val Acc:      89.3%
2024-11-19 18:44:45 INFO     [Epoch 196]   Loss:    0.6682     Train Acc:     86.74%     Val Acc:      89.3%
2024-11-19 18:44:45 INFO     [Epoch 197]   Loss:    0.6659     Train Acc:      86.8%     Val Acc:      89.4%
2024-11-19 18:44:45 INFO     [Epoch 198]   Loss:    0.6636     Train Acc:     86.84%     Val Acc:      89.5%
2024-11-19 18:44:45 INFO     [Epoch 199]   Loss:    0.6613     Train Acc:     86.92%     Val Acc:      89.6%
Traceback (most recent call last):
```

Seed2) Seed = 50



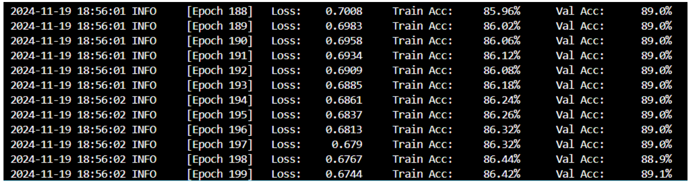
```
2024-11-19 18:53:48 INFO     [Epoch 189]   Loss:    0.7264     Train Acc:     85.82%     Val Acc:      87.7%
2024-11-19 18:53:48 INFO     [Epoch 190]   Loss:    0.7237     Train Acc:     85.94%     Val Acc:      87.8%
2024-11-19 18:53:48 INFO     [Epoch 191]   Loss:    0.7211     Train Acc:      85.9%     Val Acc:      87.8%
2024-11-19 18:53:48 INFO     [Epoch 192]   Loss:    0.7184     Train Acc:     85.92%     Val Acc:      88.0%
2024-11-19 18:53:48 INFO     [Epoch 193]   Loss:    0.7158     Train Acc:     85.92%     Val Acc:      87.9%
2024-11-19 18:53:48 INFO     [Epoch 194]   Loss:    0.7132     Train Acc:     86.04%     Val Acc:      87.9%
2024-11-19 18:53:48 INFO     [Epoch 195]   Loss:    0.7107     Train Acc:     86.08%     Val Acc:      88.0%
2024-11-19 18:53:48 INFO     [Epoch 196]   Loss:    0.7082     Train Acc:     86.14%     Val Acc:      88.0%
2024-11-19 18:53:48 INFO     [Epoch 197]   Loss:    0.7056     Train Acc:     86.26%     Val Acc:      88.0%
2024-11-19 18:53:48 INFO     [Epoch 198]   Loss:    0.7031     Train Acc:     86.32%     Val Acc:      88.0%
2024-11-19 18:53:48 INFO     [Epoch 199]   Loss:    0.7007     Train Acc:     86.24%     Val Acc:      88.0%
```

Seed 3) Seed = 100



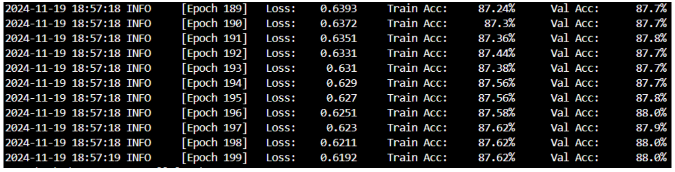
```
2024-11-19 18:55:11 INFO     [Epoch 189]   Loss:    0.6955     Train Acc:     86.38%     Val Acc:      87.7%
2024-11-19 18:55:11 INFO     [Epoch 190]   Loss:    0.6931     Train Acc:     86.34%     Val Acc:      87.7%
2024-11-19 18:55:11 INFO     [Epoch 191]   Loss:    0.6908     Train Acc:      86.4%     Val Acc:      87.7%
2024-11-19 18:55:11 INFO     [Epoch 192]   Loss:    0.6885     Train Acc:     86.42%     Val Acc:      87.7%
2024-11-19 18:55:11 INFO     [Epoch 193]   Loss:    0.6862     Train Acc:     86.46%     Val Acc:      87.7%
2024-11-19 18:55:11 INFO     [Epoch 194]   Loss:    0.6839     Train Acc:     86.44%     Val Acc:      87.7%
2024-11-19 18:55:11 INFO     [Epoch 195]   Loss:    0.6816     Train Acc:     86.52%     Val Acc:      87.7%
2024-11-19 18:55:11 INFO     [Epoch 196]   Loss:    0.6793     Train Acc:     86.54%     Val Acc:      87.7%
2024-11-19 18:55:11 INFO     [Epoch 197]   Loss:    0.6771     Train Acc:     86.58%     Val Acc:      87.6%
2024-11-19 18:55:11 INFO     [Epoch 198]   Loss:    0.6749     Train Acc:     86.62%     Val Acc:      87.7%
2024-11-19 18:55:11 INFO     [Epoch 199]   Loss:    0.6727     Train Acc:     86.58%     Val Acc:      87.6%
```

Seed 4) Seed = 1000



```
2024-11-19 18:56:01 INFO     [Epoch 188]   Loss:    0.7008     Train Acc:     85.96%     Val Acc:      89.0%
2024-11-19 18:56:01 INFO     [Epoch 189]   Loss:    0.6983     Train Acc:     86.02%     Val Acc:      89.0%
2024-11-19 18:56:01 INFO     [Epoch 190]   Loss:    0.6958     Train Acc:     86.06%     Val Acc:      89.0%
2024-11-19 18:56:01 INFO     [Epoch 191]   Loss:    0.6934     Train Acc:     86.12%     Val Acc:      89.0%
2024-11-19 18:56:01 INFO     [Epoch 192]   Loss:    0.6909     Train Acc:     86.08%     Val Acc:      89.0%
2024-11-19 18:56:01 INFO     [Epoch 193]   Loss:    0.6885     Train Acc:     86.18%     Val Acc:      89.0%
2024-11-19 18:56:02 INFO     [Epoch 194]   Loss:    0.6861     Train Acc:     86.24%     Val Acc:      89.0%
2024-11-19 18:56:02 INFO     [Epoch 195]   Loss:    0.6837     Train Acc:     86.26%     Val Acc:      89.0%
2024-11-19 18:56:02 INFO     [Epoch 196]   Loss:    0.6813     Train Acc:     86.32%     Val Acc:      89.0%
2024-11-19 18:56:02 INFO     [Epoch 197]   Loss:     0.679     Train Acc:     86.32%     Val Acc:      89.0%
2024-11-19 18:56:02 INFO     [Epoch 198]   Loss:    0.6767     Train Acc:     86.44%     Val Acc:      88.9%
2024-11-19 18:56:02 INFO     [Epoch 199]   Loss:    0.6744     Train Acc:     86.42%     Val Acc:      89.1%
```

Seed 5) Seed = 15123

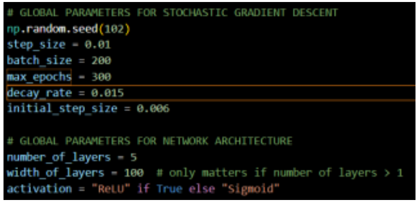


```
2024-11-19 18:57:18 INFO     [Epoch 189]   Loss:    0.6393     Train Acc:     87.24%     Val Acc:      87.7%
2024-11-19 18:57:18 INFO     [Epoch 190]   Loss:    0.6372     Train Acc:      87.3%     Val Acc:      87.7%
2024-11-19 18:57:18 INFO     [Epoch 191]   Loss:    0.6351     Train Acc:     87.36%     Val Acc:      87.8%
2024-11-19 18:57:18 INFO     [Epoch 192]   Loss:    0.6331     Train Acc:     87.44%     Val Acc:      87.7%
2024-11-19 18:57:18 INFO     [Epoch 193]   Loss:     0.631     Train Acc:     87.38%     Val Acc:      87.7%
2024-11-19 18:57:18 INFO     [Epoch 194]   Loss:     0.629     Train Acc:     87.56%     Val Acc:      87.7%
2024-11-19 18:57:18 INFO     [Epoch 195]   Loss:     0.627     Train Acc:     87.56%     Val Acc:      87.8%
2024-11-19 18:57:18 INFO     [Epoch 196]   Loss:    0.6251     Train Acc:     87.58%     Val Acc:      88.0%
2024-11-19 18:57:18 INFO     [Epoch 197]   Loss:     0.623     Train Acc:     87.62%     Val Acc:      87.9%
2024-11-19 18:57:18 INFO     [Epoch 198]   Loss:    0.6211     Train Acc:     87.62%     Val Acc:      88.0%
2024-11-19 18:57:19 INFO     [Epoch 199]   Loss:    0.6192     Train Acc:     87.62%     Val Acc:      88.0%
```

Using the default hyperparameters, set the random seed to 5 different values and report the validation accuracies you observe after training. What impact does this randomness have on the certainty of your conclusions in the previous questions?

The seed had no real impact on the validation accuracies within the 5 differing numbers. This means that that data is converting to a similar solution regardless of the initial conditions. This fact leads me to the idea that in the previous problem my ideas still have not changed.

Question 7)



```
# GLOBAL PARAMETERS FOR STOCHASTIC GRADIENT DESCENT
np.random.seed(102)
step_size = 0.01
batch_size = 200
max_epochs = 300
decay_rate = 0.015
initial_step_size = 0.006

# GLOBAL PARAMETERS FOR NETWORK ARCHITECTURE
number_of_layers = 5
width_of_layers = 100  # only matters if number of layers > 1
activation = "ReLU" if True else "Sigmoid"
```

For the last question I changed a few things I am using 5 layers and 100 width on the layers. I am also using ReLU along with a batch size of 200 and a max_epochs of 300. Most notably I am running a decay function within the step size with an initial step size of 0.006 and a exponential decay of 0.015 calculated by

```
step_size = initial_step_size * np.exp(-decay_rate * i)
```

Where the variable I is the current epoch

Debriefing

1) Approximately how many hours did you spend on this assignment?
   8 Hours
2) Would you rate it as easy moderate or difficult

moderate

3) Did you work on it mostly alone or did you discuss the problem with others?
   Mostly alone - first problem I asked the Prof where my magic math was happening
4) How deeply do you feel you understand the material it covers
   I think I understand 85% of the material covered
5) Any other comments?
   This assignment was quite fun.