CS434 - HW 3

Analyzing Naïve Bayes

1. Prove Bernoulli Naïve Bayes has a Linear Decision Boundary

Posteriors:

$$P(y = 1|x_1,\dots,x_d) = \theta_1^{z_1}(1-\theta_1)^{1-x}\dots\theta_d^{z_d}(1-\theta_d)^{1-z_d} * \theta_0$$

$$P(y = 0|x_1,\dots,x_d) = \theta_1^{z_1}(1-\theta_1)^{1-x}\dots\theta_d^{z_d}(1-\theta_d)^{1-z_d} * (1-\theta_0)$$

Substitute:

$$\frac{\theta_1^{z_1}(1-\theta_1)^{1-x}\dots\theta_d^{z_d}(1-\theta_d)^{1-z_d}*\theta_0}{\theta_1^{z_1}(1-\theta_1)^{1-x}\dots\theta_d^{z_d}(1-\theta_d)^{1-z_d}*(1-\theta_0)} > 1$$

Cancel Out:

$$\prod_{i=1}^{d}\frac{\theta_i^{x_i}(1-\theta_i)^{1-x_i}}{\theta_i^{x_i}(1-\theta_i)^{1-x_i}}\cdot\frac{\theta_0}{1-\theta_0} > 1$$

Simplified:

$$log(\frac{\theta_0}{1-\theta_0}) + \sum_{i=1}^{d} x_i log\frac{\theta_i}{1-\theta_i} > 0$$

$$b = log\frac{\theta_0}{1-\theta_0}$$

$$w_i = log\frac{\theta_i}{1-\theta_i}$$

$$b + \sum_{i=1}^{d} x_i w_i > 0$$

2. Duplicate Features in Naïve Bayes

$$P(y = 1|X_1 = x_1, X_2 = x_2) = P(X_1 = x_1|y = 1)P(X_2 = x_2|y = 1)P(y = 1)$$
$$P(y = 0|X_1 = x_1, X_2 = x_2) = P(X_1 = x_1|y = 0)P(X_2 = x_2|y = 0)P(y = 0)$$
$$P(y = 1) = P(y = 0)$$
$$P(X_2 = x_2|y = 1) = P(X_1 = x_1|y = 1)$$
$$P(X_2 = x_2|y = 0) = P(X_1 = x_1|y = 0)$$

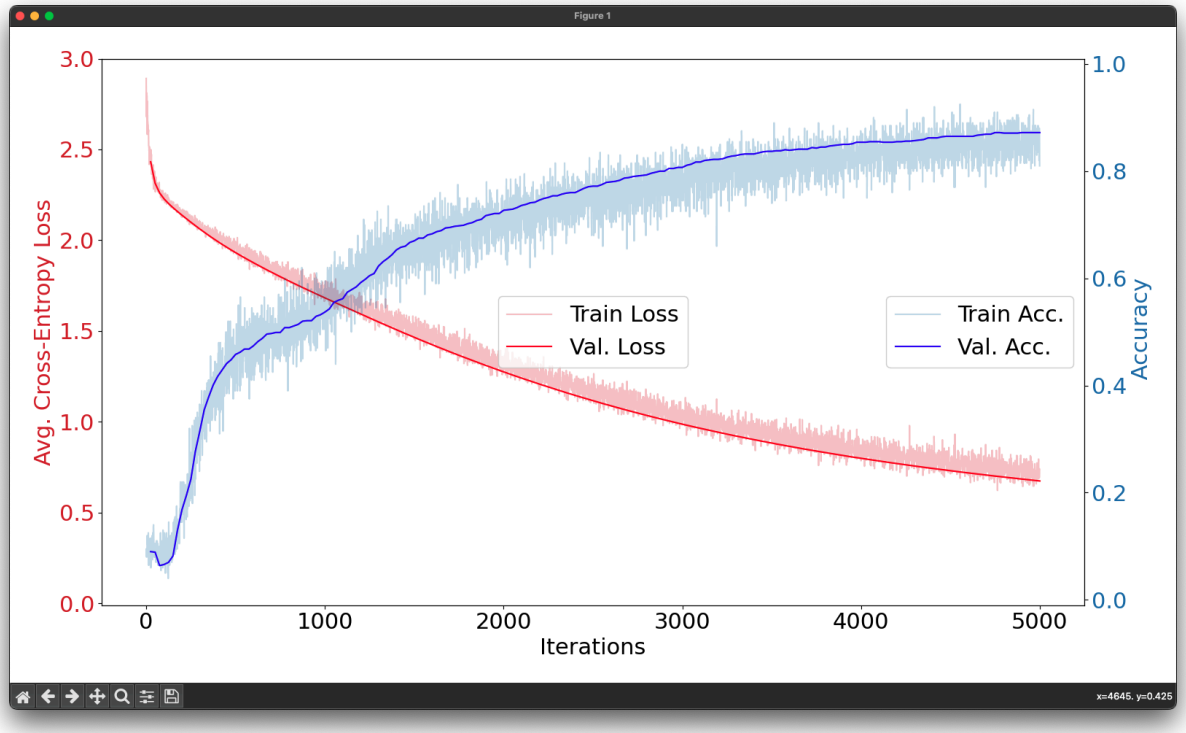
Thus

$$\frac{P(y=1|X_1=x_1,X_2=x_2)}{P(y=0|X_1=x_1,X_2=x_2)} = \frac{P(X_1=x_1|y=1)^2}{P(X_1=x_1|y=0)^2}$$

$$\frac{P(X_1=x_1|y=1)^2}{P(X_1=x_1|y=0)^2} > \frac{P(X_1=x_1|y=1)}{P(X_1=x_1|y=0)}$$

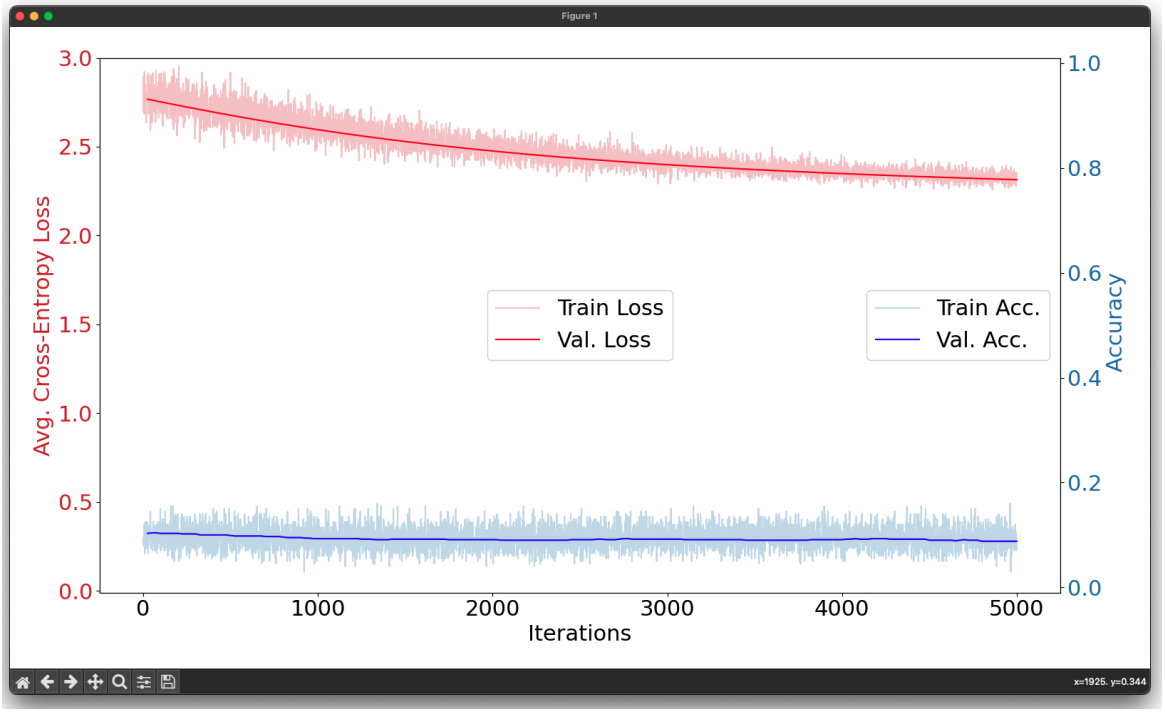
Implementing a Neural Network For Digit Identification

3. Implementing the Backward Pass for a Linear Layer
   *See code
   Resulting Graph:



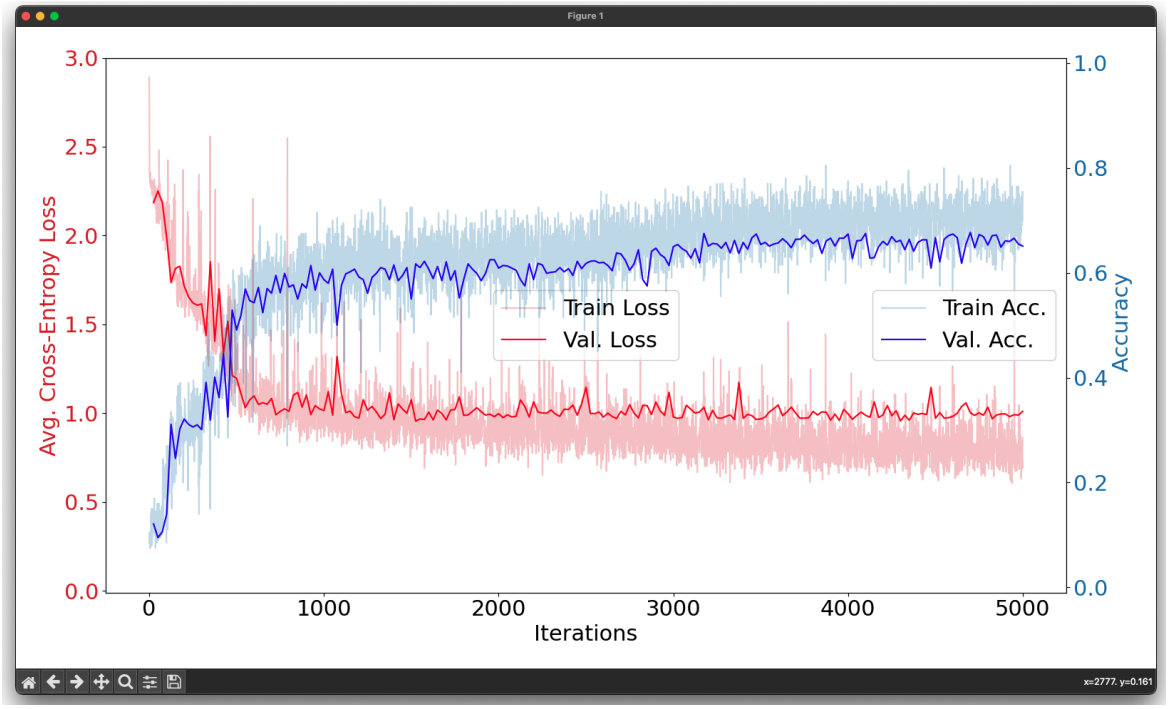
4. Learning Rate

Step size = 0.0001



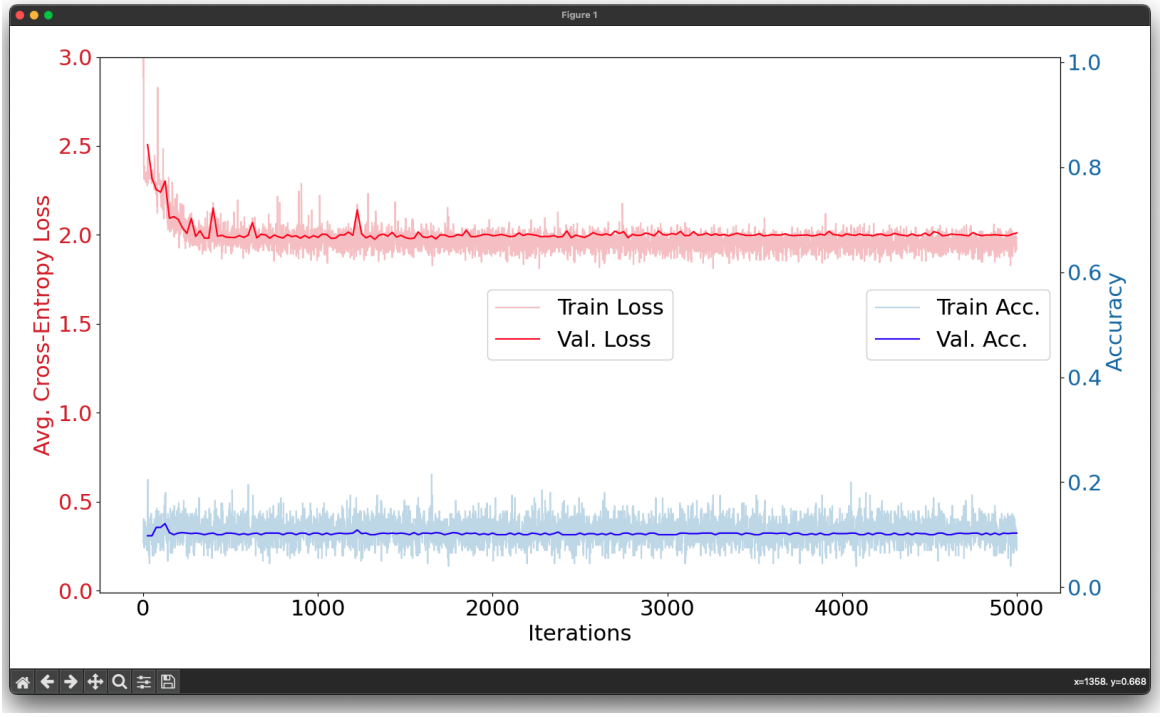
With a step size of 0.0001, we can see that the curve is much more smooth for both lines, meaning there is less variation between each iteration. Something else that we can see is that the cross-entropy loss no longer has a greater decrease in the first few iterations. Now, the cross-entropy loss very slowly decreases and evens out at around 2.4 (compared to the 0.5 with the default parameters). We also see that the accuracy never truly improves, rather staying at around 10% accurate the whole time.

Step size = 5



With a step size of 5, we see a very similar overall shape as the default parameters, but now with a lot more drastic changes between each iteration. Both lines are no longer fairly smooth. Accuracy and loss both take hits, now of around a 20% loss and 0.15 gain respectively.

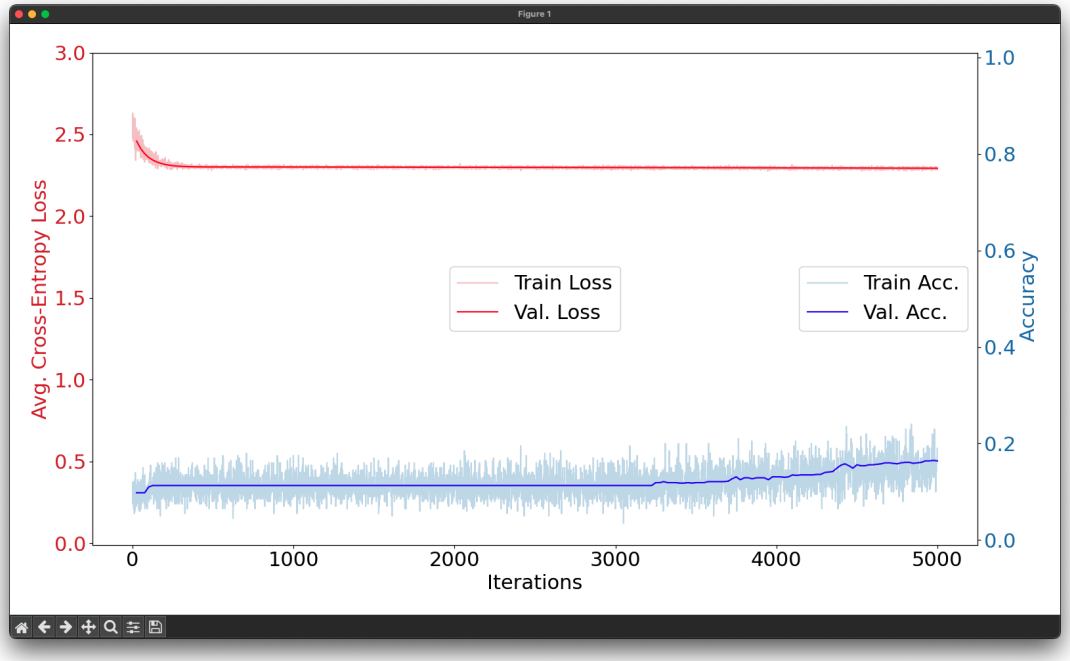
Step size = 10

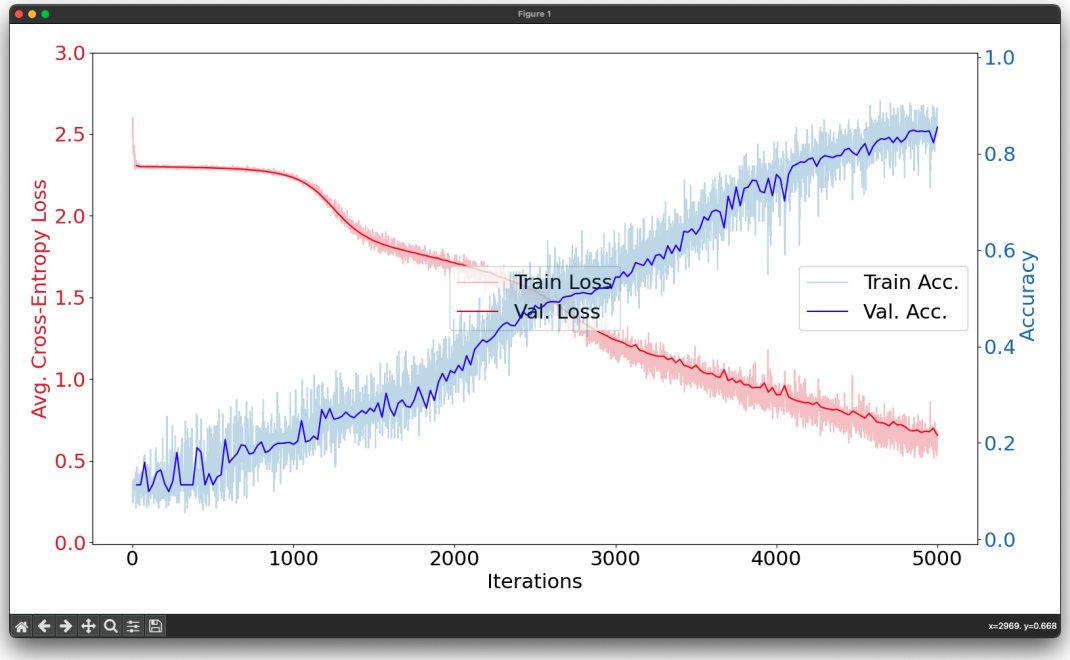


For a step size of 10, we see that it looks fairly similar to that of a 0.0001 step size. Both lines

are fairly smooth (though the cross-entropy loss has more variation in some places, particularly the beginning). We also see that both lines never really improve that much. The accuracy never improves, while the loss sees a slight improvement at the beginning, but quickly stays stable at around 2.
b) If the max epochs were increased, we could expect more accurate results for our training and validation data. For step size 0.0001 and 10, this might not do much as it is going to be inaccurate anyways. For step size 5, we might also expect overfitting to occur.

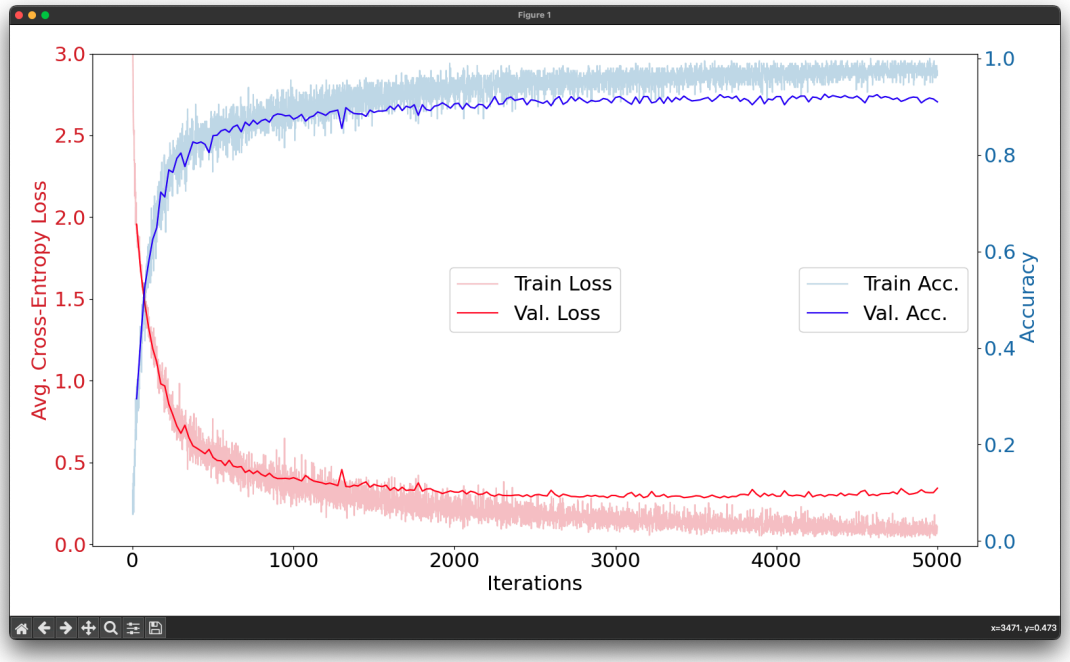
5. ReLU's and Vanishing Gradients

5-Layer Sigmoid
 Compared to the default parameters, we see that the lines are still fairly smooth, but so are the curves (or lack thereof). Cross-entropy loss hardly decreases except at the start, while the accuracy only increases very slightly at the start and a small amount a little over half way through.
5-Layer Sigmoid with 0.1 Step Size



Here, we see that the accuracy increases in a slightly more linear fashion, while also consistently being much less smooth. We also see that cross-entropy starts smooth and then slowly becomes more jagged. There is also a large dip at around the 1000th iteration.

5-Layer ReLU



When we switch to ReLU, we see that there is a much more drastic change at the start for both

lines. The lines both eventually even out at their limits with these settings. These produce the most accurate results, even compared to the default values, with the least loss as well. The lines are not as smooth as the default values, but much less jagged compared to the previous settings.
b) The observed improvement due to learning rate in (2) compared to (1) might result from (1) getting stuck at a local minimum and not being able to improve further, as a more drastic change was needed for improvement. With (2), the step size was large enough so that it was able to bypass this and continue improving.
c) The reason that the (3) outperform (1) might be because with Sigmoid, you are using increasingly smaller derivatives. With ReLU, your derivative is always 0 or 1. With ReLU, it's either true or false. Sigmoid is not, so it not only takes much longer to compute, but also will need to deal with vanishing gradients.

6. Measuring Randomness

Seed	Validation Accuracy
Default: 102	87.2%
10	88.4%
88	88.3%
399	86.9%
997	89.3%
273	88.2%

With the five accuracies, they all vary but only by slight amounts, though the range is 2.4%. The inclusion of randomness does make me a little less certain of my answer, though not too much so.

7. Kaggle Submission
   See Kaggle

Debriefing

1. How many hours were spent on this assignment?
   ~4.5
2. Rating?
   4/10

3. Alone or with others?
   Alone.
4. Understanding?
   Pretty good understanding.
5. Comments
   No.