CS 577: Homework 1

Wenxin Jiang
jiang784@purdue.edu

Due Date: 11:59pm, Mar 2, 2023

Q1: Write down the logistic function and loss function you used in Logistic Regression and derive the gradient (4 points)

Logistic function

Sigmoid function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Then calculate

$$z = \mathbf{w} \cdot \mathbf{x} + b$$

And pass the result to the sigmoid function.

Loss function

L1 loss with L1 regularization

$$L1 = \sum_{i=1}^{n} |y_{true} - y_{predicted}| + \lambda \sum_{i=1}^{n} \theta_i$$

Here

$$y_{predicted} = \sigma(z)$$

Gradient

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial w} = x\sigma(1 - \sigma)$$

$$w = w - lr * x\sigma(1 - \sigma)$$

Q2: Describe the architecture of the neural network you implemented using a computation graph. Your answer should include the **activation function** used in each layer (2 Points), the **number of layers and number of neurons in each layer** (2 Points). State the **loss function** used in your neural network implementation (2 Points). Derive the partial derivative associated with each node in the computation graph (4 Points)

Computation Graph: See Figure 1.
Activation function: I used ReLU for the hidden layer and softmax for the output layer.
Number of layers: Two, including one hidden layer and one output layer.
Number of neurons:

Input layer: 3645
Hidden layer: 120
Output layer: 6
Number of neurons in W1: $3645 * 120 = 437400$
Number of neurons in W2: $120 * 6 = 720$
Total number of neurons: $437400 + 720 = 438120$

Loss function: Cross Entropy:

$$cost(\hat{Y}, Y) = -\sum_{i=1}^{n} Y_i \log(\hat{Y}_i)$$

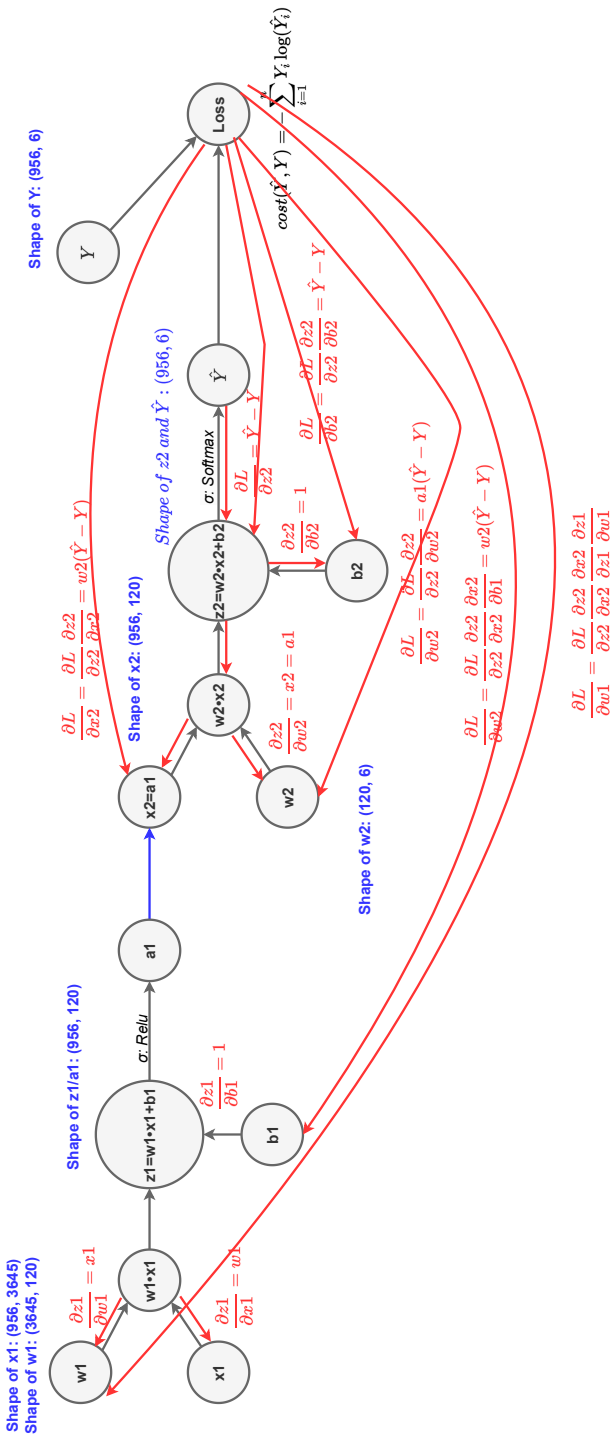

Figure 1: Computation Graph of Neural Network

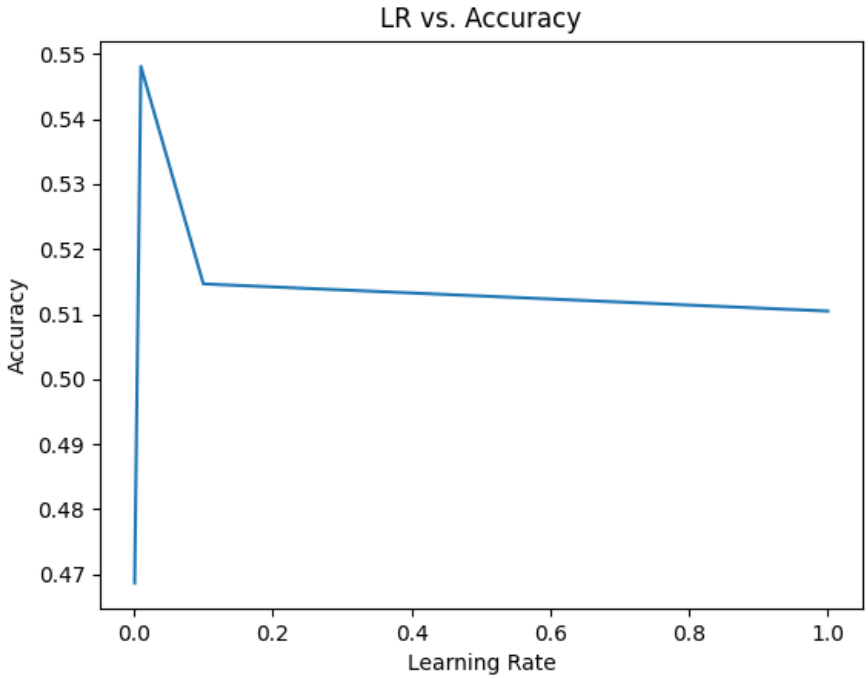

Figure 2: LR: Learning Rate vs. Accuracy

Q3: State one **hyper-parameter** you tuned in case of Logistic Regression and one in case of Neural Network. How did tuning these hyper-parameters change the result? Explain with learning curves. You may have tuned more than one hyper-parameter for each model. In this section explain only one from each model (6 Points).

Logistric Regression I tuned the learning rate and regularization term. Figure 2 shows the accuracy vs. learning rate curve. This figure shows that the best accuracy was achieved at $lr = 0.1$.

I also use k-fold cross validation to tune the training, the results for one hyper-parameter setting is shown in Figure 3:

Neural Network

Similar to Logistic Regression, I also tuned the learning rate for neural network. When I use a larger learning rate, especially when $lr \geq 0.1$, the cross entropy loss came to NaN. The final learning curve was shown in Figure 4.

For neural network, I also tried using 5-fold cross validation. Here is the results for fixed hyper-parameters, as shown in Figure 5.

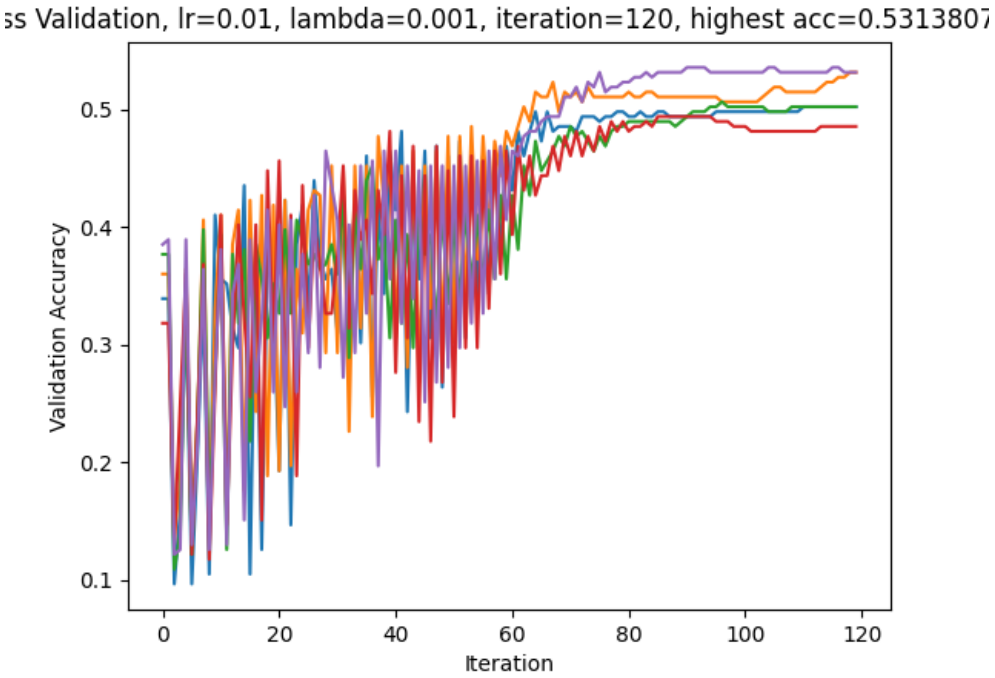

Figure 3: LR: 5-fold Cross Validation, lr=0.01, lambda=0.001, iter=120

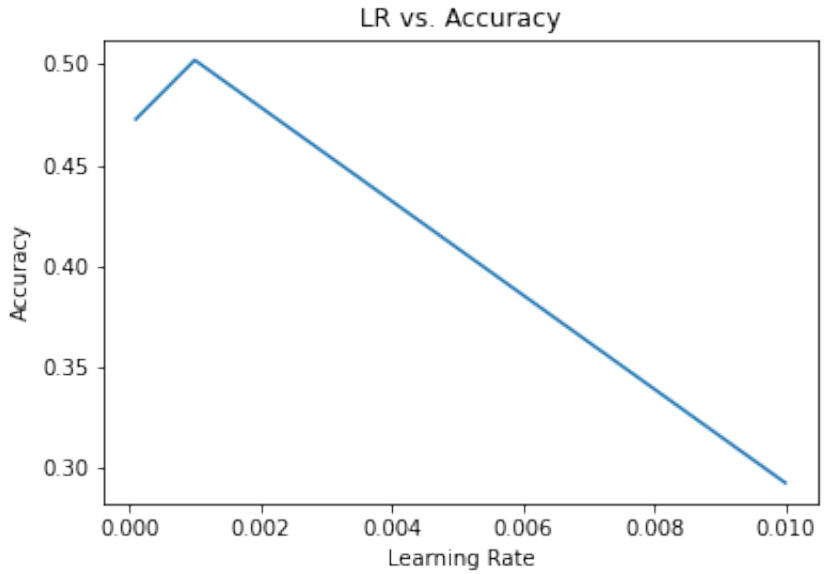

Figure 4: NN: Learning Rate vs. Accuracy

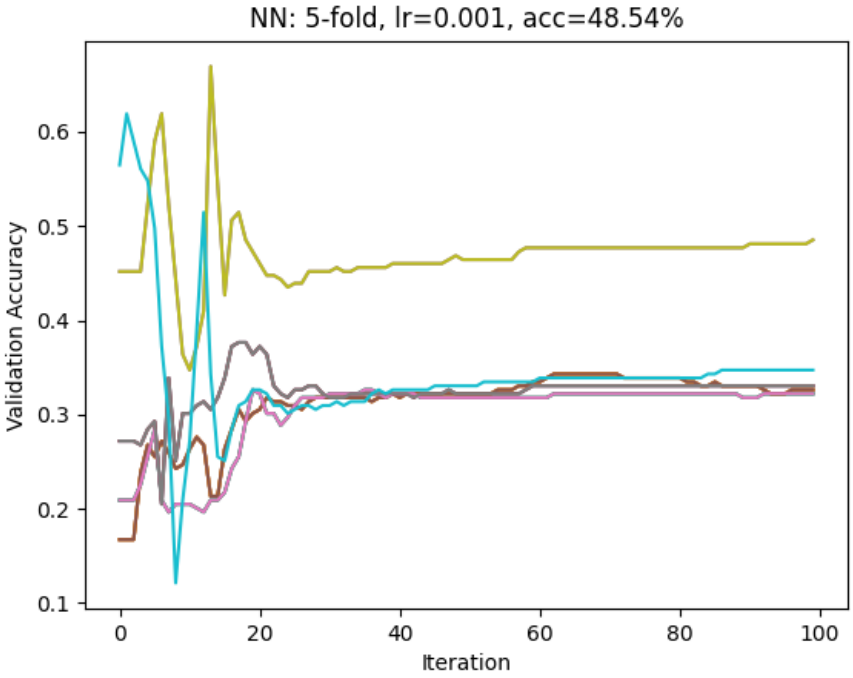

Figure 5: NN: Epochs vs. Accuracy

Reference

1. https://pylessons.com/Neural-network-single-layer-part3
2. https://towardsai.net/p/machine-learning/nothing-but-numpy-understandi