Assignment 3 - Learning and Basic Architectures
Winter 2024

Atieh Armin
Student ID : 14658308

1 Theory

1. For the function $J = (x_1w_1 - 5x_2w_2 - 2)^2$, where $w = [w_1, w_2]^T$ are our weights to learn:

   (a) What are the partial gradients, $\frac{\partial J}{\partial w_1}$ and $\frac{\partial J}{\partial w_2}$? Show work to support your answer (6pts).
   Solution: We can use chain rule to calculate the gradients.
   If we consider $(x_1w_1 - 5x_2w_2 - 2)$ as h, we will have $J = (h)^2$.
   For $\frac{\partial J}{\partial w_1}$:
   $\frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial h} * \frac{\partial h}{\partial w_1}$
   $\frac{\partial J}{\partial h} = \frac{\partial (h^2)}{\partial h} = 2h = 2(x_1w_1 - 5x_2w_2 - 2)$
   $\frac{\partial h}{\partial w_1} = \frac{\partial (x_1w_1 - 5x_2w_2 - 2)}{\partial w_1} = x_1$
   $=> \frac{\partial J}{\partial w_1} = 2(x_1w_1 - 5x_2w_2 - 2) * x_1$
   For $\frac{\partial J}{\partial w_2}$:
   $\frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial h} * \frac{\partial h}{\partial w_2}$
   $\frac{\partial J}{\partial h} = \frac{\partial (h^2)}{\partial h} = 2h = 2(x_1w_1 - 5x_2w_2 - 2)$
   $\frac{\partial h}{\partial w_2} = \frac{\partial (x_1w_1 - 5x_2w_2 - 2)}{\partial w_2} = -5x_2$
   $=> \frac{\partial J}{\partial w_2} = 2(x_1w_1 - 5x_2w_2 - 2) * (-5x_2)$

   (b) What are the value of the partial gradients, given current values of $w = [0, 0]^T, x = [1, 1]$ (4pts)?
   Solution: We can just put $w = [0, 0]^T, x = [1, 1]$ in the equations above!
   For $\frac{\partial J}{\partial w_1}$:
   $\frac{\partial J}{\partial w_1} = 2(x_1w_1 - 5x_2w_2 - 2) * x_1 = 2(1 * 0 - 5 * 1 * 0 - 2) * 1 = 2(-2) * 1 = -4$
   For $\frac{\partial J}{\partial w_2}$:
   $\frac{\partial J}{\partial w_2} = 2(x_1w_1 - 5x_2w_2 - 2) * (-5x_2) = 2(1 * 0 - 5 * 1 * 0 - 2) * (-5 * 1) = 2(-2) * (-5) = 20$

2. Given the objective function $J = \frac{1}{4}(x_1w_1)^4 - \frac{4}{3}(x_1w_1)^3 + \frac{3}{2}(x_1w_1)^2$:

   (a) What is the gradient $\frac{\partial J}{\partial w_1}$ (2pts)?
   We can use chain rule to calculate the gradients.
   If we consider (x_1w_1) as h, we will have $J = \frac{1}{4}(h)^4 - \frac{4}{3}(h)^3 + \frac{3}{2}(h)^2$.
   $\frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial h} * \frac{\partial h}{\partial w_1}$

$\frac{\partial J}{\partial h} = \frac{(\frac{1}{4}(h)^4 - \frac{4}{3}(h)^3 + \frac{3}{2}(h)^2)}{\partial h} = (h)^3 + 4(h)^2 + 3(h) = (x_1 w_1)^3 + 4(x_1 w_1)^2 + 3(x_1 w_1)$

$\frac{\partial h}{\partial w_1} = \frac{\partial(x_1 w_1)}{\partial w_1} = x_1$

$=> \frac{\partial J}{\partial w_1} = ((x_1 w_1)^3 + 4(x_1 w_1)^2 + 3(x_1 w_1)) * x_1$

(b) What are the locations of the extrema points for this objective function J if $x_1 = 1$? Recall that to find these you take the derivative of the objective function with respect to the unknown, set that equal to zero and solve for said unknown (in this case, w_1). (5pts)

If $x_1 = 1$, $\frac{\partial J}{\partial w_1} = ((w_1)^3 + 4(w_1)^2 + 3(w_1))$.

We can find the extrema points if we put $\frac{\partial J}{\partial w_1} = 0$.

$=> \frac{\partial J}{\partial w_1} = ((w_1)^3 + 4(w_1)^2 + 3(w_1)) = 0$

$=> (w_1)(w_1 - 3)(w_1 - 1) = 0$

=> The exterma points are $w_1 = 0, w_1 = 1$, and $w_1 = 3$

(c) What does J evaluate to at each of your extrema points, again when $x_1 = 1$ (3pts)?

If $x_1 = 1$, $J = \frac{1}{4}(w_1)^4 - \frac{4}{3}(w_1)^3 + \frac{3}{2}(w_1)^2$.

$w_1 = 0 => J = 0$

$w_1 = 1 => J = \frac{1}{4} - \frac{4}{3} + \frac{3}{2} = \frac{3}{12} - \frac{16}{12} + \frac{18}{12} = \frac{3-16+18}{12} = \frac{5}{12} \approx 0.417$

$w_1 = 3 => J = \frac{1}{4}(3)^4 - \frac{4}{3}(3)^3 + \frac{3}{2}(3)^2 = \frac{1}{4}(81) - \frac{4}{3}(27) + \frac{3}{2}(9) = \frac{81}{4} - (4 * 9) + \frac{27}{2} = \frac{81-36*4+27*2}{4} = \frac{81-144+54}{4} = \frac{135-144}{4} = \frac{-9}{4} = -2.25$

2 Visualizing Gradient Descent

In this section we want to visualize the gradient descent process for the following function (which was part of one of the theory questions):

$$J = (x_1 w_1 - 5x_2 w_2 - 2)^2$$

Note that this is more of a *toy problem* to explore the idea of gradient-based learning than it is a deep learning architecture.

Hyperparameter choices will be as follows:

- Initialize your weights to zero.
- Set the learning rate to $\eta = 0.01$.
- Terminate after 100 epochs.

Using the partial gradients you computed in the theory question, perform gradient descent, using $x = [1, 1]$. After each training epoch, evaluate J so that you can plot w_1 vs w_2, vs J as a 3D line plot. Put this figure in your report.

The 3D Plot:

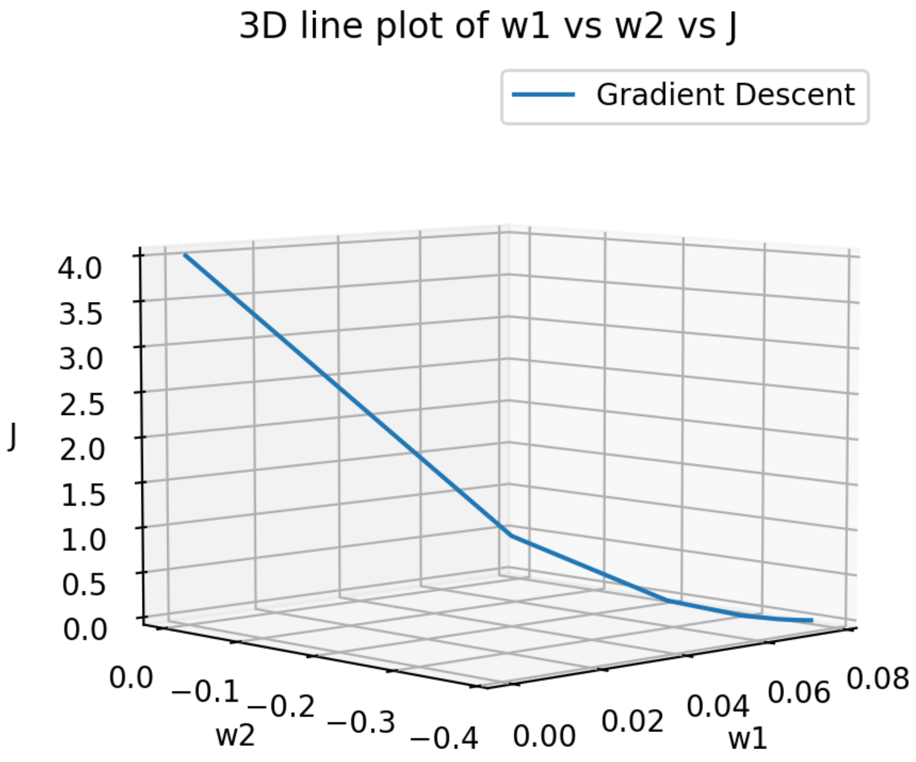

Figure 1: The 3D line plot of w_1 vs w_2, vs J

3 Updating Fully Connected Layer's Weights and Biases

We also need to add an *updateWeights* method for our Fully Connected layer. This method takes a backcoming gradient and a learning rate as parameters, and updates its weights and biases according to the formulas in lecture. The method's prototype should look like:

```
def updateWeights(self,gradIn, eta = 0.0001):
    #TODO
```

4 Linear Regression

In this section you'll use your modules to train a linear regression model for the *medical cost dataset*. The architecture of your linear regression should be as follows:

$$\textit{Input} \rightarrow \text{Fully-Connected} \rightarrow \text{Squared-Error-Objective}$$

Your code should do the following:

1. Read in the dataset to assemble X and Y (recall that our target Y is the *charges* column for this dataset).
2. Shuffle the rows of the dataset (both X and Y, together) and use approximately 2/3 for training and 1/3 for validating.
3. Train, via gradient learning, your linear regression system using the training data. Refer to the pseudocode in the lecture slides on how this training loop should look. Initialize your weights to be random values in the range of $\pm 10^{-4}$ and your learning rate to be $\eta = 10^{-4}$. Terminate the learning process when the absolute change in the mean squared error on the training data is less than 10^{-10} or you pass 100,000 epochs. During training, keep track of the **mean squared error** (MSE) for **both** the training and the validation sets so that we can plot these as a function of the epoch.

In your report provide:

1. Your plots of training and validation MSE vs epoch.

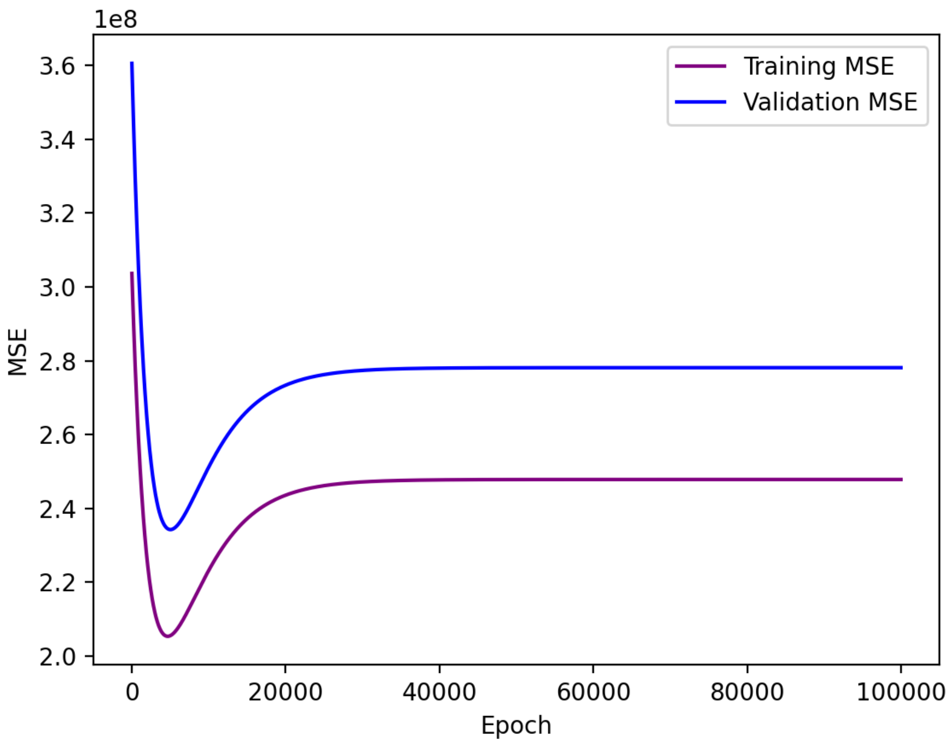

Figure 2: The MSE Plot of Train and Validation Data

2. Your final RMSE for the training and validation data.

   Final RMSE for Train Data: 15742.019977701477

   Final RMSE for Validation Data: 16676.767241001195

3. Your final SMAPE for the training and validation data.

   Final SMAPE for Train Data: 0.1805480098913472

   Final SMAPE for Validation Data: 0.18300108710610172

5 Logistic Regression

Next we'll use a logistic regression model on the *kid creative* dataset to predict if a user will purchase a product. The architecture of this model should be:

$$Input \rightarrow \text{Fully-Connected} \rightarrow \text{Sigmoid-Activation} \rightarrow \text{Log-Loss-Objective}$$

Your code should do the following:

1. Read in the dataset to assemble X and Y (rcall that our target Y is the *Buy* column for this dataset).
2. Shuffle the rows of the dataset (both X and Y, together) and use approximately 2/3 for training and 1/3 for validating.
3. Train, via gradient learning, your logistic regression system using the training data. Initialize your weights to be random values in the range of $\pm 10^{-4}$ and your learning rate to be $\eta = 10^{-4}$. Terminate the learning process when the absolute change in the log loss is less than 10^{-10} or you pass $100,000$ epochs. During training, keep track of the **log loss** for **both** the training and the validation sets so that we can plot these as a function of the epoch.

In your report provide:

1. Your plots of training and validation log loss vs epoch.

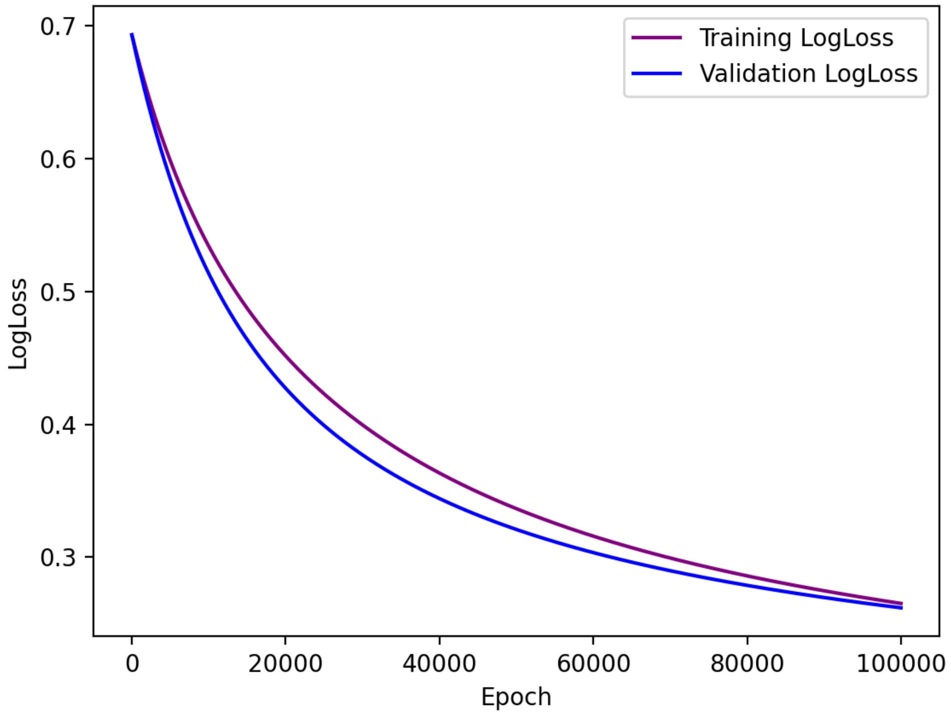

Figure 3: The LogLoss Plot of Train and Validation Data

2. Assigning an observation to class 1 if the model outputs a value greater than 0.5, report the training and validation accuracy.

   Train Accuracy: 0.9285714285714286

   Validation Accuracy: 0.9066666666666666

Submission

For your submission, upload to Blackboard a single zip file containing:

1. PDF Writeup
2. Source Code
3. readme.txt file

The readme.txt file should contain information on how to run your code to reproduce results for each part of the assignment.

The PDF document should contain the following:

1. Part 1: Your solutions to the theory question
2. Part 2: Your plot.
3. Parts 3-4: Nothing.
4. Part 5: Your plot and requested statistics.
5. Part 6: Your plot and requested accuracies.