Intro ML Homework 1

Jaskin Kabir
Student Id: 801186717
GitHub:
https://github.com/jaskinkabir/Intro_ML/tree/main/HM1

September 2024

1 Problem 1

1. **Model Results**
   The models found by this technique were

   - $y_1 = -1.96X_1 + 5.725$
   - $y_2 = 0.564X_2 + 0.72$
   - $y_3 = -0.486X_3 + 2.784$

   Where y_i represents the model trained solely on the explanatory variable X_i

2. **Plots:**

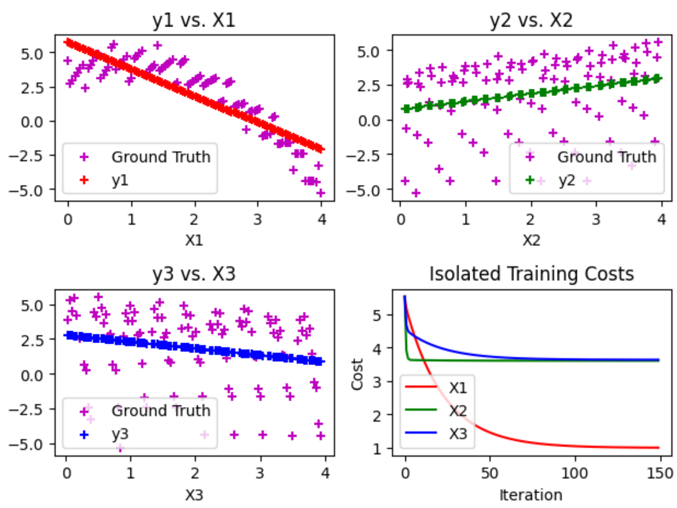

Figure 1: Individually Trained Models and Costs

3. **Cost Analysis**
   The lowest cost for explaining the output was X_1, whose cost was only 0.99 compared to 3.6 and 3.63 for X_2 and X_3. This means that X_1 has the most linear relationship to Y.
   This is consistent with the plots shown in Figure 1. Variables X_2 and X_3 both seem to have a very nonlinear relationship with Y, and seem to be equally nonlinear. In contrast, Y shows a clear curve downward as X_1 increases, albeit this curve isn't quite linear either. This is reflected in the equally high cost values for the models trained on X_2 and X_3 compared to the low cost of X_1's model.

4. **Impact of Learning Rate and Iteration Count**

- *Learning Rate*
  Different learning rates between 0.01 and 0.1 were explored. At $\alpha = 0.01$, the gradient descent algorithm had too little an effect on the θ values to adequately train the model. The final cost value for X_1's model jumps to 3.2 compared to the cost of 0.99 with $\alpha = 0.1$.
  At $\alpha = 0.1$, the final cost is lower across all three regressions, as the gradient of the cost function has more of an impact on the θ values used for the next iteration.

- *Iteration Count*
  With each iteration, the model is refined more and more. Thus, like with the learning rate, increasing the number of iterations also improves the performance of the models. However, increasing the iteration count always reaches a point of diminishing returns, which in this report is defined as the point where increasing the iteration count by 50 only leads to an improvement in cost of less than 0.01.
  This point is reached around 150 iterations for X_1, 5 iterations for X_2 and 50 iterations for X_3. These numbers were taken from experimentation where the learning rate was fixed at 0.1 The number of iterations required to achieve a certain loss is dependent on the learning rate, as a model that learns more slowly will require more iterations to improve itself.

2 Problem 2

1. **Model Results**
   The best fitting model found using this method was $y = -0.203X_3 + 0.603X_2 - 1.944X_1 + 4.89$. Its final cost value was found to be 0.748

2. **Plots:**

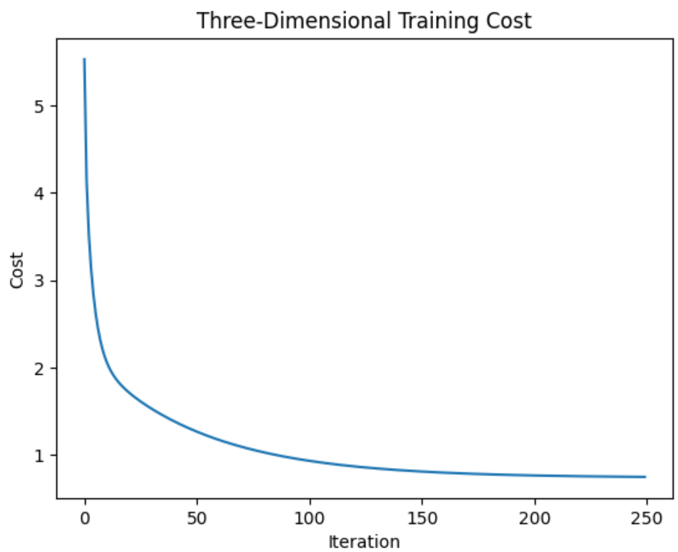

Figure 2: Three-Dimensional Regression

3. **Impact of Learning Rate and Iteration Count**.

   - *Learning Rate*
     Just like in the previous problem, increasing the learning rate will lead to a better fitting model, and it will also require fewer iterations to reach that best fitting model. In this case, 0.1 was found to be the best value for the learning rate.
     If the learning rate were to be increased past a certain point, the 'ping-pong' effect would cause the gradient descent algorithm to launch the cost towards infinity with each iteration. However, using too low a learning rate would require too many iterations to reach a usable model. This is why it is important to find the highest usable alpha value before this divergence occurs
   - *Iteration Count*
     Once again increasing the number of iterations leads to a better fitting model, but after a certain number of iterations, the improvement becomes very minimal.

In this case, that point of diminishing returns was found to be around 250 iterations using the same criteria used in problem 1.
Finding this point of diminishing returns is important because training is a costly computation that should be minimized to cut time, costs, and energy usage

4. **Predictions**
The trained model can now be used to make predictions by placing the input values $X_1 - X_3$ into a row vector X and performing the matrix multiplication $Y(X) = \begin{bmatrix} 1 & X \end{bmatrix} \times \theta$, where θ is a column vector containing the coefficients found by the training process. Here the number 1 must be placed in the first position of the row vector to be multiplied by θ so that the constant coefficient θ_1 is added to the final result. For example, predicting the value of Y for the input values $\begin{bmatrix} 2 & 0 & 4 \end{bmatrix}$ would require the computation:

$$Y(\begin{bmatrix} 2 & 0 & 4 \end{bmatrix}) = \begin{bmatrix} 1 & 2 & 0 & 4 \end{bmatrix} \times \begin{bmatrix} 4.89 \\ -1.944 \\ 0.603 \\ -0.203 \end{bmatrix}$$

The model predicts the following values:

$$Y(\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}) = 3.345$$

$$Y(\begin{bmatrix} 2 & 0 & 4 \end{bmatrix}) = 0.189$$

$$Y(\begin{bmatrix} 3 & 2 & 1 \end{bmatrix}) = 0.060$$