CSE 643: Artificial Intelligence - Assignment 4

Akshat Gupta

November 23, 2023

1. (a) Consider the linear regression model with five variables, represented in expanded form as:

$$y = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5$$

where w_0 is the bias term, and w_1, w_2, w_3, w_4, w_5 are the weights corresponding to the features x_1, x_2, x_3, x_4, x_5 respectively. Let x_0 be 1 for all samples, then bias can be included as part of the weights such that:

$$\begin{aligned}\mathbf{w} &= [w_0, w_1, w_2, w_3, w_4, w_5] \\ \mathbf{x} &= [1, x_1, x_2, x_3, x_4, x_5] \\ y &= \mathbf{w}^T \cdot \mathbf{x}\end{aligned}$$

Gradients in Expanded Form:

The Mean Squared Error (MSE) is given by:

$$\begin{aligned}L(\mathbf{w}; \mathbf{x}^{(i)}, y^{(i)}) &= \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - \hat{y}^{(i)})^2 \\ L(\mathbf{w}; \mathbf{x}^{(i)}, y^{(i)}) &= \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - \mathbf{w}^T \cdot \mathbf{x}^{(i)})^2\end{aligned}$$

where N is the number of data points, $y^{(i)}$ is the actual output, and $\hat{y}^{(i)}$ is the predicted output. Consider weight w_1 corresponding to input x_1:

$$\begin{aligned}\Rightarrow L(\mathbf{w}; \mathbf{x}^{(i)}, y^{(i)}) &= \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - \mathbf{w}^T \cdot \mathbf{x}^{(i)})^2 \\ \Rightarrow L(\mathbf{w}; \mathbf{x}^{(i)}, y^{(i)}) &= \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - \sum_{j=0}^{5}(w_j \cdot x_j))^2 \\ \Rightarrow \frac{\partial L}{\partial w_j} &= \frac{1}{N}\sum_{i=1}^{N}(\frac{\partial L}{\partial e_i} \cdot \frac{\partial e_i}{\partial \hat{y}^{(i)}} \cdot \frac{\partial \hat{y}^{(i)}}{\partial w_j}) \\ \Rightarrow \frac{\partial L}{\partial w_j} &= \frac{1}{N}\sum_{i=1}^{N}(2(y^{(i)} - \hat{y}^{(i)}) \cdot (-1) \cdot (x_{j,i})) \\ \Rightarrow \frac{\partial L}{\partial w_j} &= -\frac{2}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)x_{j,i}\end{aligned}$$

where $e_i = (y^{(i)} - \hat{y}^{(i)})$ and $\hat{y}^{(i)} = \mathbf{w}^T \cdot \mathbf{x}^{(i)}$

The gradients with respect to each of the weights are:

$$\frac{\partial L}{\partial w_0} = -\frac{2}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)$$
$$\frac{\partial L}{\partial w_1} = -\frac{2}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)x_{1,i}$$
$$\frac{\partial L}{\partial w_2} = -\frac{2}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)x_{2,i}$$
$$\frac{\partial L}{\partial w_3} = -\frac{2}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)x_{3,i}$$
$$\frac{\partial L}{\partial w_4} = -\frac{2}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)x_{4,i}$$
$$\frac{\partial L}{\partial w_5} = -\frac{2}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)x_{5,i}$$

Vector Form:

In vector form, if we define $\mathbf{x}$ as the matrix of input features and y as the vector of actual outputs, the predicted output $\hat{y}$ can be written as $\mathbf{w}^T \cdot \mathbf{x}$, where $\mathbf{w}$ is the vector of weights.
As can be seen from the expanded equation form and the definition of $L(\mathbf{w})$ in vector form, the gradient vector in vector form is given by:

$$\nabla L(\mathbf{w}) = -\frac{2}{N}\mathbf{x}^T(y - \mathbf{x} \cdot \mathbf{w})$$

Gradient Descent Update Rule:

The gradient descent update rule for the weights in vector form is:

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} - \alpha \nabla L(\mathbf{w}_{\text{old}})$$
$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} + \alpha \frac{2}{N}\mathbf{x}^T(y - \mathbf{x} \cdot \mathbf{w})$$

The gradient descent update rule for the weights in expanded equation form is:

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} - \alpha \frac{\partial L}{\partial \mathbf{w}_{\text{old}}}$$
$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} + \alpha \frac{2}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)x_{j,i}$$

where α is the learning rate.

(b) X here is a randomly generated set of data generated using `np.arange()` between -20 to +20 with steps of 0.1. The data is then shuffled to ensure the actual outputs are not generated in cohescence.
The true output Y is generated as: $Y = 23X + 43 + noise$. Noise is modelled as an array of random values of length 400 (since X is of length 400) and is added to the actual output.
The regression model then implements gradient descent as:

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} - \alpha \frac{2}{N}\mathbf{x}^T(\mathbf{x} \cdot \mathbf{w} - y)$$

The weights are initialized using `np.random.rand()` and of dimensionality 2 x 1 (Bias, Weight). The learning rate used for this problem is $lr = 0.005$ and the number of iterations are $num_iters = 100$, as specified.

The weights are updated according to the gradient descent rule mentioned earlier, and the final results obtained are:

$$\begin{aligned} \mathbf{w} &= [23.01747050669566, 30.587406705595512] \\ Weight &= 23.01747050669566 \\ Bias &= 30.587406705595512 \end{aligned}$$

2. Linear Regression

The dataset was imported into the python environment and preprocessed as:

- The only categorical feature i.e. *sex* is one-hot encoded for better performance as past evidence supports that one-hot encoded features tend to fit better to the problem and thus have better performing models.
- The numerical features were all transformed as per the following function:

  $$f(z) : \frac{1}{1 + e^{-\sqrt{z}}}$$

  This allowed for a non-linear transformation thus extracting salient information from the given features.
- This processed data is then concatenated to obtain a full pre-processed dataset.

The R^2 score when trained on the entire dataset is 0.**564090764732424**.

When making the 70-15-15 splits, the model is first raw-trained on the train set. It is then used to evaluate its performance on the validation set, which if satisfactory enough (i.e. ≥ 0.56) the model is then evaluated on the test set and the corresponding R^2 score is stored as part of an array the mean and standard deviation of which are then reported:

mean = 0.5570046682225838, std = 0.026122851555662345