Assignment-10
Linear Regression and Gradient Descent

Harshvardhan Patidar
Department of Artificial Intelligence
Indian Institute of Technology Hyderabad
`ai24btech11015@iith.ac.in`

1 Linear Regression

1.1 Univariate Linear Regression

A univariate linear function relates the input x and output y in a linear relation of the form $y = w_1x + w_0$. The coefficients are considered as weights and hence are named as w. The value of y can be altered by modifying the weights of the terms. For more general case, we will define **w** to be a vector $< w_0, w_1 >$ and define our function as

$$h_w(x) = w_1x + w_0$$

The process of finding the best fitting h_w is called linear regression. To find the best fit function, we will find the value of the weight $< w_0, w_1 >$ which minimizes the loss function. The traditional method is to use the squared-error loss function, denoted as L_2, over the complete training data set.

$$\text{Loss}(h_w) = \sum_{j=1}^{N} L_2(y_j, h_w(x_j)) = \sum_{j=1}^{N} (y_j - h_w(x_j))^2$$

where

$$h_w(x) = w_1x + w_0$$

We will select w^* such as $w^* = \arg\min_w \text{Loss}(h_w)$. The function for Loss($h_w$) can be minimized by making its partial derivative w.r.t w_0 and w_1 zero:

$$\frac{\partial}{\partial w_0} \sum_{j=1}^{N} (y_j - (w_1x_j + w_0))^2 = 0 \quad \text{and} \quad \frac{\partial}{\partial w_1} \sum_{j=1}^{N} (y_j - (w_1x_j + w_0))^2 = 0.$$

These equations have a unique solution given as,

$$w_1 = \frac{N\left(\sum x_jy_j\right) - \left(\sum x_j\right)\left(\sum y_j\right)}{N\sum x_j^2 - \left(\sum x_j\right)^2} \quad \text{and} \quad w_0 = \frac{\sum y_j - w_1\left(\sum x_j\right)}{N}.$$

As in this task, we are trying to find the set of weights which minimize a loss, we can plot the loss function in the 3D space, where w_0 and w_1 are the arguments. We observe that the loss function is convex, which implies that there are no local minimas, but only global minima.

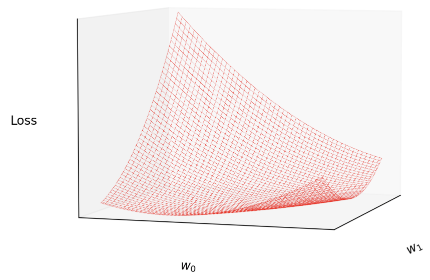

Figure 1: Plot of Loss with w_1 and w_0

1.2 Gradient Descent

This method helps in minimizing the loss function without calculating the derivatives and making them zero. Instead in this method, we try for some (w_0, w_1) surface, compute the approximate gradient at that point, and then move in the direction of the steepest downhill direction on the surface, to approach the minima of the loss function. Here we use a parameter α, called the learning rate. It can be a constant or a dynamic variable depending upon the requirements.

Let's first have a look at the partial derivatives of the loss function with respect to w_1 and w_0

$$\begin{aligned}\frac{\partial \text{Loss}(w)}{\partial w_i} = \frac{\partial (y - h_w(x))^2}{\partial w_i} &= 2(y - h_w(x)) \times \frac{\partial (y - h_w(x))}{\partial w_i} \\ &= 2(y - h_w(x)) \times \frac{\partial (y - (w_1 x + w_0))}{\partial w_i}.\end{aligned}$$

We get from these,

$$\frac{\partial \text{Loss}(w)}{\partial w_0} = -2(y - h_w(x)) \quad \text{and} \quad \frac{\partial \text{Loss}(w)}{\partial w_1} = -2(y - h_w(x)) \times x$$

' So, after for every point (w_0, w_1), we will update them as below for all the training examples

$$w_0 \leftarrow w_0 + \alpha \sum (y_j - h_w(x_j)) \quad \text{and} \quad w_1 \leftarrow w_1 + \alpha \sum (y_j - h_w(x_j)) \times x_j$$

These updates constitute the batch gradient descent learning rule. We call one step, that covers the complete training examples as an epoch. This method is slow as we need to sum the partial derivatives over all the training examples, which might be computationally expensive. To consider this, we use another method called Stochastic Gradient Descent (SGD).

The SGD selects only a few training examples to update the values of the wrights at each step rather than considering the complete training set. If we take N/η examples for consideration, our calculations reduce by a factor of η, and our error increases only by a factor of $\sqrt{\eta}$. The η can be considered as an hyperparameter which needs to be specifically tuned for ever problem.

SGD is laso effective when enw data is coming one-by-one, in the absence of the stationarity assumption. (SGD is also known as Online GradIent descent). SGD works significantly nice in cases where there are multiple local minimas as well.

1.3 Multivariable Linear Regression

‘

We can consider multivariable linear regressions as well, where our input $\mathbf{x}_j$ is an n-element vector. Here,

$$h_w(x_j) = w_0 + w_1 x_{j,1} + \cdots + w_n x_{j,n} = w_0 + \sum_{i=1}^{n} w_i x_{j,i}$$

We can make it beautiful by adding the lingering w_0 in the $\mathbf{w}$ and introducing a new attribute $x_{j,0}$ which has value as 1. Then h is simply the dot product of weights and the input vector.

$$h_{\mathbf{w}}(\mathbf{x}_j) = \mathbf{w} \cdot \mathbf{x}_j = \mathbf{w}^\top \mathbf{x}_j = \sum_{i=1}^{n} w_i x_{j,i}$$

We then select the best vector w^* which minimizes the loss function

$$\mathbf{w}^* = \arg\min \sum L_2(y_j, \mathbf{w} \cdot \mathbf{x}_j)$$

For updating the weights, we can still use the same previous formula, with slight modifications

$$w_i \leftarrow w_i + \alpha \sum (y_j - h_{\mathbf{w}}(\mathbf{x}_j)) \times x_{j,i}$$

The feasible $\mathbf{w}$ can also be found analytically by using linear algebra and vector calculus. The predicted output vector is $\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$, then the squared loss error is

$$\mathbf{L}(\mathbf{w}) = \|\hat{\mathbf{y}} - \mathbf{y}\|_2 = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2$$

To minimize, we make the gradient zero,

$$\begin{aligned} \nabla_{\mathbf{w}} L(\mathbf{w}) &= 2\mathbf{X}^\top(\mathbf{X}\mathbf{w} - \mathbf{y}) = 0 \\ \Rightarrow \mathbf{w}^* &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y} \end{aligned}$$

The expression $(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$ is called the pseudoinverse of the data matrix, and the above equation is called as the normal equation.

2 The Gradient Method

This paper was authored by Louis Augustin Cauchy in 1847. In this paper Cauchy tried to solve the algebraic equations which represented the heavenly bodies and their movements. His motivation was the complex differential equations which required to be solved to get a solution for such problems, which he didn't want to do. So he tried to find the solutions directly, instead of finding the differential equations and thus giving this beautiful method, called the Gradient Method.

His aim was to find those values of the variables $x, y, z, \ldots$ which minimized the function u, taking variables $x, y, z, \ldots$ as input, is always non-negative and is continuous.

$$u = f(x, y, z, \ldots)$$

He tried to indefinitely decrease the function u to its minimum value, by altering the variables, to continuously decrease the value of u. For this, he calculated the derivatives

$$X = f'_x, \quad Y = f'_y, \quad Z = f'_z, \ldots$$

And gave small increment/decrement $\alpha, \beta, \gamma, \ldots$ respectively to $x, y, z, \ldots$

$$f(x + \alpha, y + \beta, z + \gamma, \ldots) = u + X\alpha + Y\beta + Z\gamma + \ldots$$

He replaced the $\alpha, \beta, \gamma, \ldots$ as below by introducing a decrement parameter θ:

$$\alpha = -\theta X, \quad \beta = -\theta Y, \quad \gamma = -\theta Z, \ldots$$

Then u was calculated approximately as

$$f(x - \theta X, y - \theta Y, z - \theta Z, \ldots) = u - \theta(X^2 + Y^2 + Z^2 + \ldots)$$

This way as θ varies, u also decreases and eventually obtains its minimum value.

He mentioned Convergence as, that if the new value of u is not the smallest, an even smaller value can be found by proceeding in a similar manner. He was not fully confident that this might not always find the solution as this was an approximation only and not complete calculation. On the contrary, he was convinced that u will always converge to its minimum value. This was a significant contribution by one of the notable mathematicians of that time which is still used.