Linear Regression and Gradient Descent

Satyanarayana Gajjarapu
AI24BTECH11009
Department of Artificial Intelligence
`ai24btech11009@iith.ac.in`

1 Univariate linear regression

A univariate linear function is a straight line with input x and output y having the form $y = w_1x + w_0$, where w_0 and w_1 are real-valued coefficients. A vector **w** is defined as $\langle w_0, w_1 \rangle$ and linear function as $h_\mathbf{w}(x) = w_1x + w_0$. The task of finding the $h_\mathbf{w}$ that best fits the data is called **linear regression**. To fit a line to the data, it is required to find values of weights $\langle w_0, w_1 \rangle$ that minimize the empirical loss. It is traditional to use the squared error loss function, L_2.

$$Loss(h_\mathbf{w}) = \sum_{j=1}^{N} L_2(y_j, h_\mathbf{w}(x_j)) = \sum_{j=1}^{N}(y_j - h_\mathbf{w}(x_j))^2 = \sum_{j=1}^{N}(y_j - (w_1x_j + w_0))^2$$

The $Loss(h_\mathbf{w})$ is minimized when partial derivatives with respect to w_0 and w_1 are equated to zero. Solving the obtained 2 equations, a unique solution of w_0 and w_1 is obtained.

$$w_1 = \frac{N(\sum x_jy_j) - (\sum x_j)(\sum y_j)}{N(\sum x_j^2) - (\sum x_j)^2}; \qquad w_0 = (\sum y_j - w_1(\sum x_j))/N$$

The space defined by all possible settings of the weights is called **weight space**. For univariate linear regression, the weight space is 2 - dimensional defined by w_0 and w_1. So the graph of loss as a function of w_0 and w_1 is a 3D plot.

2 Gradient Descent

An algorithm called **hill climbing** is used to search through a continuous weight space by incrementally modifying the parameters. But the term **gradient descent** is used because the aim is minimizing loss, not maximizing gain. The algorithm is as follows:

$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} Loss(\mathbf{w})$$

The parameter α is step size which is usually called the **learning rate**. For univariate regression, the loss is quadratic, so the partial derivative will be linear.

$$\begin{aligned}\frac{\partial}{\partial w_i} Loss(\mathbf{w}) &= \frac{\partial}{\partial w_i}(y - h_\mathbf{w}(x))^2 = 2(y - h_\mathbf{w}(x)) \times \frac{\partial}{\partial w_i}(y - h_\mathbf{w}(x)) \\ &= 2(y - h_\mathbf{w}(x)) \times \frac{\partial}{\partial w_i}(y - (w_1x + w_0))\end{aligned}$$

Applying this to both w_0 and w_1,

$$\frac{\partial}{\partial w_0} Loss(\mathbf{w}) = -2(y - h_\mathbf{w}); \qquad \frac{\partial}{\partial w_1} Loss(\mathbf{w}) = -2(y - h_\mathbf{w}) \times x$$

Preprint. Under review.

Using these 2 equations and unspecified learning rate α to get the learning rule for the weights,

$$w_0 \leftarrow w_0 + \alpha(y - h_{\mathbf{w}}(x)); \qquad w_1 \leftarrow w_1 + \alpha(y - h_{\mathbf{w}}(x)) \times x$$

Examples for different types of gradient descent algorithms are **batch gradient descent** (deterministic gradient descent), **stochastic gradient descent** (online gradient descent) or **SGD** and **minibatch gradient descent**. A step that covers all the training examples is called an **epoch**.

3 Multivariable linear regression

This can be extended to **multivariable linear regression** in which $\mathbf{x}_j$ is an n - element vector. The term w_0 is defined as always equal to 1. h can simply written as the dot product of the weights and the input vector. The hypothesis space consists functions of form

$$h_{\mathbf{w}}(\mathbf{x}_j) = w_0 + w_1 x_{j,1} + \cdots + w_n x_{j,n} = w_0 + \sum_i w_i x_{j,i}$$

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

The updated equation for each weight w_i is

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

Let **y** be the vector of outputs for the training examples, and **X** be the **data matrix**. The vector of predicted outputs is $\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$ and the squared error loss over all the training data is

$$L(\mathbf{w}) = ||\hat{\mathbf{y}} - \mathbf{y}||^2 = ||\mathbf{X}\mathbf{w} - \mathbf{y}||^2$$

After setting gradient to 0 and rearranging, minimum loss weight is obtained as

$$\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

This equation is called **normal equation** and $(\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$ is called **pseudoinverse** of data matrix.

4 Cauchy and the Gradient Method

Cauchy is motivated by astronomic calculations which are normally very voluminous. In those days to solve a multivariable equation, one reduces the equation to a single variable by successive eliminations. Consider a function $u = f(x, y, z, \cdots)$ and $X = f'_x, Y = f'_y, Z = f'_z, \cdots$ are the derivatives. Aim is to find the values of $x, y, z, \cdots$ satisfying the equation $u = 0$. Let $\alpha, \beta, \gamma, \cdots$ be small increments given to the particular values $x, y, z, \cdots$

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

Taking θ>0, we obtain approximately

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

If θ increases and the function $f(x, y, z, \cdots)$ is continuous, the value of u will decrease until it vanishes. Cauchy does not seem to believe that the method always finds a solution, but he hoped it does.