Regression Notes

Tevfik Aytekin[1]

[1]Parts of these notes are largely inspired by Andrew Ng's ML course notes.

November 17, 2020

1 Preliminaries

Assume we are given a data set $D = ((x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(m)}, y^{(m)}))$ where $x^{(i)} \in \mathbb{R}^n$ and $y^{(i)} \in \mathbb{R}$.

2 Linear Regression

Hypothesis (model):

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_n x_n \tag{1}$$

where $\theta \in \mathbb{R}^{n+1}$ is the parameter vector and $x_0 = 1$. This model assumes that the output is a linear function of the inputs.

Cost function:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)}))^2 \tag{2}$$

The objective is to find the θ values which minimizes the cost.

2.1 Batch Gradient descent

repeat
  $\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$
until *convergence*;

Algorithm 1: Gradient Descent.

Below is the derivative of the cost function for a data set where there is a single example (x, y).

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2}(y - h_\theta(x))^2 \\
&= 2\frac{1}{2}(y - h_\theta(x))\frac{\partial}{\partial \theta_j}(y - h_\theta(x)) \\
&= (y - h_\theta(x))\frac{\partial}{\partial \theta_j}\left(y - \sum_{i=0}^{n} \theta_i x_i\right) \\
&= -(y - h_\theta(x))x_j
\end{aligned} \tag{3}$$

For m examples:

$$\frac{\partial}{\partial \theta_j} J(\theta) = -\sum_{i=1}^{m}(y^{(i)} - h_\theta(x^{(i)}))x_j \tag{4}$$

So, gradient descent algorithm becomes:

repeat
  $\theta_j := \theta_j + \alpha \sum_{i=1}^{m}(y^{(i)} - h_\theta(x^{(i)}))x_j \quad$ (for every j)
until *convergence*;

Algorithm 2: Gradient Descent.

α is called the learning rate which controls the magnitude of the updates. Note that you need to update θ_j's simultaneously.

2.2 Stochastic Gradient descent

repeat
  shuffle the data
  for $i = 0$ *to* m **do**
    $\theta_j := \theta_j + \alpha(y^{(i)} - h_\theta(x^{(i)}))x_j \quad$ (for every j)
  end
until *convergence*;

Algorithm 3: Stochastic Gradient Descent.

Different from the batch version stochastic gradient ascent update the parameters after seeing every individual example. Stochastic gradient descent achieves faster convergence than the batch version.

2.3 Closed Form Solution

Using vector notation we can write the cost function

$$J(\theta) = \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)}))^2 \tag{5}$$

as follows:

$$(y - X\theta)^T (y - X\theta) \tag{6}$$

In order to find the values of θ which minimizes the cost function we need to set the derivative to zero and solve for θ.

$$\begin{aligned} \nabla(y - X\theta)^T (y - X\theta) &= 0 \\ -2X^T (y - X\theta) &= 0 \\ -2X^T y + 2X^T X\theta &= 0 \\ (X^T X)^{-1} X^T X\theta &= (X^T X)^{-1} X^T y \\ I\theta &= (X^T X)^{-1} X^T y \\ \theta &= (X^T X)^{-1} X^T y \end{aligned} \tag{7}$$

Note that the time complexity of the matrix inverse operation is $O(d)$.

2.4 Regularized Linear Regression

2.4.1 Ridge Regression

Cost function:

$$J(\theta) = \frac{1}{2m} \left[\sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)}))^2 + \lambda \sum_{j=1}^{n} \theta_j^2 \right] \tag{8}$$

Gradient Descent:

repeat

$\theta_0 := \theta_0 + \alpha \frac{1}{m} \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)}) x_0$

$\theta_j := \theta_j + \alpha \left[\frac{1}{m} \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)})) x_j - \frac{\lambda}{m} \theta_j \right] \quad (j = 1, 2, 3, ..., n)$

until *convergence*;

Algorithm 4: Gradient Descent for Ridge Regression.

Closed form solution:

$$\theta = (X^T X + \lambda I)^{-1} X^T y \tag{9}$$