October 9, 2017

Lecture 1: Linear Regression

Lecturer: Andrew Ng *Scribe: Mithlesh Kumar*

This is based on a Stanford lecture for the course CS229 delivered by Andrew Ng.

1. Introduction

Let us start by defining some notations first. We will use $x^{(i)}$ and $y^{(i)}$ to denote i^{th} input variable and target variable respectively. So, the pair $(x^{(i)}, y^{(i)})$ denotes the i^{th} training example. Suppose we are given m training examples $(x^{(i)}, y^{(i)})_{i=1,2,3,..,m}$. Each training example consists of n features.

When the target variable we are going to predict is a continuous real value, the learning problem is called **Regression**. It is an example of supervised learning problem. In this lecture, we will develop linear model for regression problem and use **Gradient Descent** to learn parameters of the model. Later, we will also derive a closed form solution for the parameters.

2. Linear Regression

In linear regression, we develop a linear hypothesis function $h_\theta(x)$ which approximates target variable y.

$$y \approx h_\theta(x) = \theta_0 + \theta_1 x_1 + ... + \theta_n x_n$$

Here $\theta^{'s}$ are the parameters of the model. When the context of parameters is clear, we can drop θ in $h_\theta(x)$. To simplify the notation, we define $x_0 = 1$, so

$$h(x) = \sum_{i=1}^{n} \theta_i x_i = \theta^T x$$

Job of our learning problem is to learn these parameters. On what basis, we sholud pick these parameters? Most obvious method is to choose parameters which make $h(x)$ as close to y as possible for the training examples provided to us. we define **cost function** which measures how close the $h(x)^{'s}$ are to the corresponding $y^{'s}$.

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

Thus, the objective of regression is to find

$$\arg\min_\theta J(\theta)$$

3. Gradient Descent

We want to choose θ which minimizes $J(\theta)$. A general strategy would be to start with some random θ and keep changing θ to reduce our objective function $J(\theta)$. More specifically, Gradient descent starts with some

initialization of θ and repeatedly performs the following update until convergence:

$$\theta_i \coloneqq \theta_i - \alpha \frac{\partial J(\theta)}{\partial \theta_i} \qquad \forall i \in [0, 1, 2, 3, .., n]$$

Here, α is the learning rate which is a hyperparameter and we will have to tune it. In order to implement this algorithm, we will first have to calculate the partial derivative.

$$\begin{aligned} \frac{\partial J(\theta)}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j} \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \\ &= \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \end{aligned}$$

Putting the value of partial derivative, we get following update rule:

$$\theta_j \coloneqq \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \qquad \forall j \in [1, 2, 3, .., n]$$

This method looks at all training examples for every iteration of the loop. One problem with the gradient descent is that it is susceptible to local optimum. But we need not worry about this drawback here because the optimization problem posed here for linear regression has only one global optimum and only one local optimum. So, gradient descent converges to global optimum in case of linear regression.

4. Closed Form Solution

Another method to find parameters is to set derivative of objective function to zero and obtain required parameters without resorting to iterative algorithm. Before doing this, we quickly introduce notations for matrix.

Given a training set, we define the design matrix X to be $m \times n$ matrix as

$$X = \begin{bmatrix} (x^{(1)})^T \\ (x^{(2)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix}$$

Similarly, let Y be the m dimensional vector containing the target labels as

$$Y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

Writing $J(\theta)$ in matrix representation, we get

$$J(\theta) = \frac{1}{2} (X\theta - Y)^T (X\theta - Y)$$

Finally, we take derivative of $J(\theta)$ with respect to θ and set it to zero.

$$\nabla_\theta J(\theta) = X^T (X\theta - Y) = 0$$

Thus, the value of θ which minimizes $J(\theta)$ is given in closed form by

$$\theta = (X^T X)^{-1} X^T Y$$

References

[1] source of video lecture:
`https://www.youtube.com/watch?v=5u4G23_OohI&index=2&list=PLA89DCFA6ADACE599`

[2] Andrew Ng lecture notes
`http://cs229.stanford.edu/notes/cs229-notes1.pdf`