Machine learning - Stanford - Week # 1

Trung C. Nguyen
email nguyencanhtrung@me.com

January 13, 2017

Contents

1 Linear Regression

1.1 Notation

Notation	Meaning
m	number of training example
x's	"input" variable/ "input"features
y's	"output" variable/ "target" variable
(x,y)	one training example
$(x^{(i)}, y^{(i)})$	training example i^{th}

Table 1: Notation table applied for whole course

Give an example: We have a training set of housing prices (Portland, OR)

Size in $feet^2$ (x)	Price($) in 1000's (y)
2140	460
1416	232
1534	315
852	178
...	...

$x^{(1)} = 2014$; $x^{(2)} = 1416$

1.2 Model Representation

To describe the supervised learning problem slightly more formally, our goal is, given a training set, to learn a function $h : X \to Y$ so that h(x) is a "good" predictor for the corresponding value of y. For historical reasons, this function h is called a hypothesis. Seen pictorially, the process is therefore like figure 1:
Another word, h is a function that maps from x's to y's When the target vari-

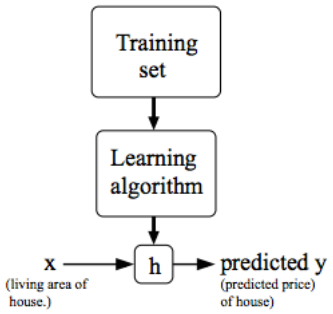

Figure 1: Process of machine learning

able that we're trying to predict is continuous, such as in our housing example, we call the learning problem a regression problem. When y can take on only a small number of discrete values (such as if, given the living area, we wanted to predict if a dwelling is a house or an apartment, say), we call it a classification problem.

If h is a function of one variable we called the model is **Univariate linear regression**

1.3 Cost function

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 \cdot x$

$\theta_i's$: Parameters of the model

Cost function measures the accuracy of our hypothesis function. It is a function of parameters

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

This function is otherwise called the "Squared error function", or "Mean squared error". The mean is halved ($\frac{1}{2}$) as a convenience for the computation of the gradient descent, as the derivative term of the square function will cancel out the $\frac{1}{2}$ term. Now problem turns out of choosing value of θ_0 and θ_1. With each combination (θ_0, θ_1), we have a hypothesis (h) function.

Explain:
For each combination of (θ_0, θ_1), we have a value for hypothesis function. To find a closest $h(x)$ to actual data set means we have to find a combination of (θ_0, θ_1), such as $(h(x) - y)$ is minimum.

Lets consider whole training set, we have to find a combination of (θ_0, θ_1), such as the average of all difference: $\frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$ is minimum. Square the difference to cover the case hypothesis value is smaller than actual one.

If we call the difference above is cost function, the problem becomes find combination of (θ_0, θ_1) such as the cost function is smallest =¿ Minimization problem

1.3.1 Contour plot

A contour plot is a graph that contains many contour lines. A contour line of a two variable function has a constant value at all points of the same line. An example of such a graph is the one to the right below. Taking any color and going along the 'circle', one would expect to get the same value of the cost function. For example, the three green points found on the green line above have the same value for $J(\theta_0, \theta_1)$ and as a result, they are found along the same line.

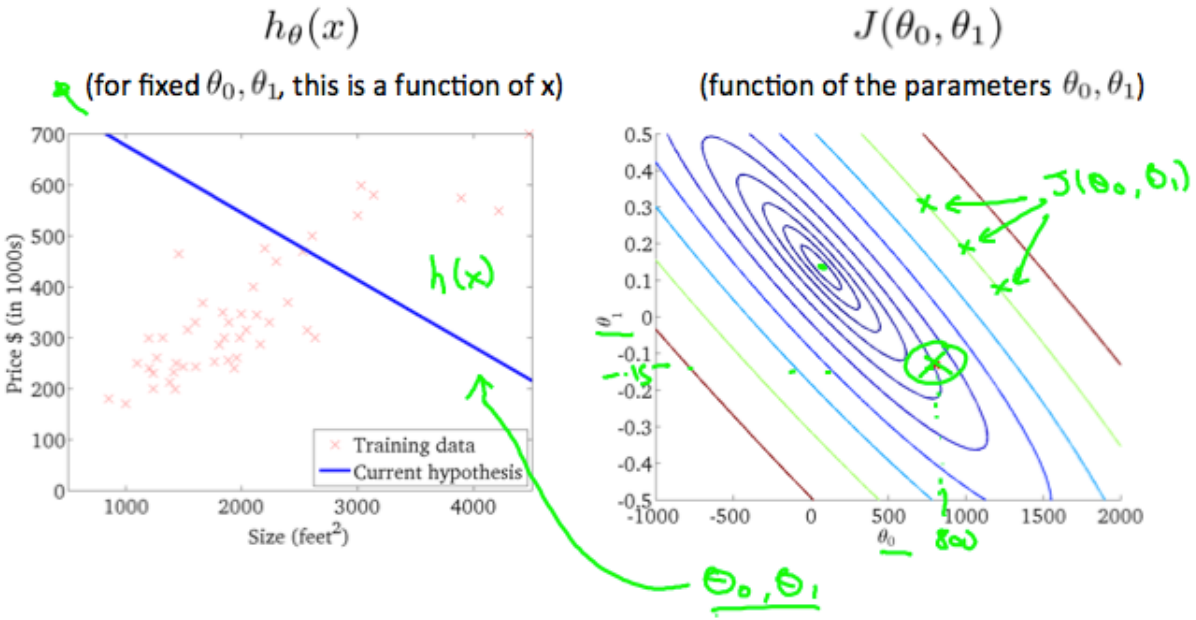

Figure 2: Contour plot

2 Algorithm gradient descent

Here is the gradient descent algorithm:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \tag{1}$$

where j=0,1 represents the feature index number.

Some important notes:

- One should simultaneously update the parameters $\theta_1, \theta_2, ..., \theta_n$. Updating a specific parameter prior to calculating another one on the $j^{(th)}$ iteration would yield to a wrong implementation
- Taking the derivative (the tangential line to a function) of our cost function $J(\theta_0, \theta_1)$. The slope of the tangent is the derivative at that point and it will give us a direction to move towards
- The size of each step is determined by the parameter α, which is called the learning rate. A smaller α would result in a smaller step and a larger α results in a larger step
- We make steps down the cost function in the direction with the steepest descent
- Depending on where one starts on the graph, one could end up at different points.

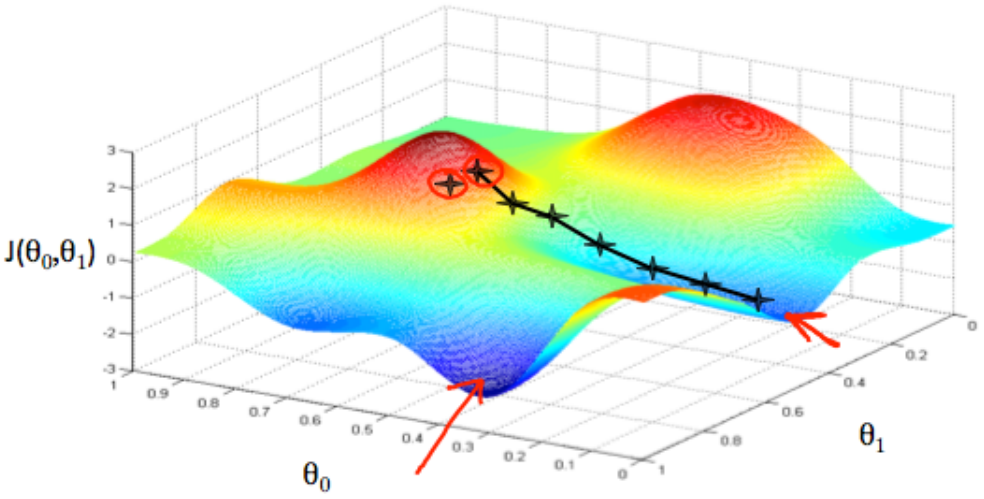

Figure 3: Minimize cost function using gradient descent

Combining all equations we have

$$\begin{aligned}
&\text{repeat until convergence: } \{ \\
&\qquad \theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i) \\
&\qquad \theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x_i) - y_i) x_i) \\
&\qquad \}
\end{aligned} \tag{2}$$

So, this is simply gradient descent on the original cost function J. This method looks at every example in the entire training set on every step, and is called **batch gradient descent**.

Note that, while gradient descent can be susceptible to local minima in general, the optimization problem we have posed here for linear regression has only one global, and no other local, optima; thus gradient descent always converges (assuming the learning rate α is not too large) to the global minimum. Indeed, J is a convex quadratic function. Here is an example of gradient descent as it is run to minimize a quadratic function.