# Machine Learning Stanford

Dylan Bourgeois

March 22, 2014

**Abstract**

Notes from Coursera class.

## 1 Introduction

Introduction to Machine Learning.

Supervised : give the right answer. Regression predicts real value output.

## 2 Model

### 2.1 Model Representations

Training set = data set. **m** = number of training examples.

Training set → Learning algorithm → $h(x) = y$.

**h** is the *hypothesis*, a linear function that outputs results for given entry x.

$$h(x) = \theta_0 + \theta_1.x$$

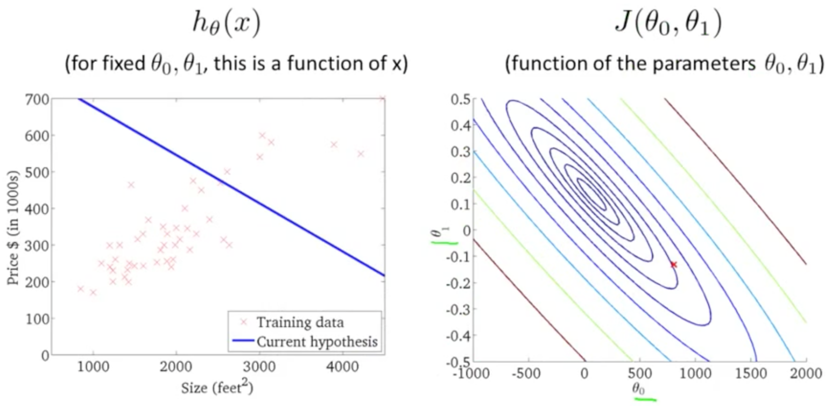
Linear regression with one variable == Univariate linear regression.

### 2.2 Cost Function

We must minimize the cost function (also called *squared error function*), which is function of $\theta_0$ and $\theta_1$.

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

## 2.3 Cost Function Intuition




For linear regression, the contour plot of the cost function is a series of concentruc ellipses. The center of these ellipses is the minimum of this function, so is the point for the desired $(\theta_0, \theta_1)$ tuple.

## 2.4 Gradient Descent

This method is valid for $n$ parameters, so for $J(\theta_0, \ldots, \theta_n)$. Start with a given tuple $(\theta_0, \theta_1)$, and update the value **simultaneously** until covergence :

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta_0, \theta_1)}{\partial \theta_j} \quad for j = 0, 1; \quad \alpha = learning\, rate$$

We must assign the new values **simultaneously** :

```
until Gradient Descent converges do
begin
{
    tmp0 = theta0 - alpha * d(J)/dtheta0;
    tmp1 = theta1 - alpha * d(J)/dtheta1;
    theta0 = tmp0;
    theta1 = tmp1;
}
end;
```

## 2.5 Gradient Descent Intuition

If $\alpha$ is small, gradient descent can be slow. If $\alpha$ is too large, gradient descent can overshoot the minimum. It may fail to converge or even diverge.

If your initial tuple $(\theta_0, \theta_1)$ is already at a local minimum, the gradient descent algorithm will stay at value $\theta_1$.

As we approach a local minimum the gradient descent algorithm will automatically take smaller steps.

## 2.6 Gradient descent for linear regression

Also called catch gradient descent algorithm: on each step you are looking at all the training examples. We need to calculate the derivative of $J(\theta_0, \theta_1)$ for

$$j = 0 \quad \frac{1}{m}\sum_{1}^{m} h(x_i) - y_i$$

$$j = 1 \quad \frac{1}{m}\sum_{1}^{m} x_i(h(xi) - yi)$$

For linear regression, $J(\theta_0, \theta_1)$ is a convex function, which has only one global minimum : no other local minimum than the global minimum.