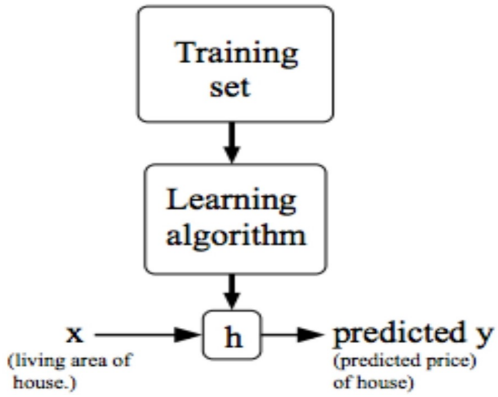
Linear Regression with one variable

In supervised learning, we have the training set (x (i),y (i)). Let's say our dataset has m training examples. Linear regression is a model that assumes a linear relationship between the input variables (x) and the single output variable (y). So the hypothesis function is

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Here θi 's are called the parameters.

A.Model Representation



In supervised learning problems, from a given training set, we want to learn a function h: X -> Y so that h(x) can efficiently predict the corresponding value of y. When the target variable that we are trying to predict is continuous, such as in our housing example, we call it a Regression problem.

B.Cost Function

Cost function is used to measure the accuracy of our hypothesis function. This takes an average difference of all the results of the hypothesis with inputs from x's and the actual y's.

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

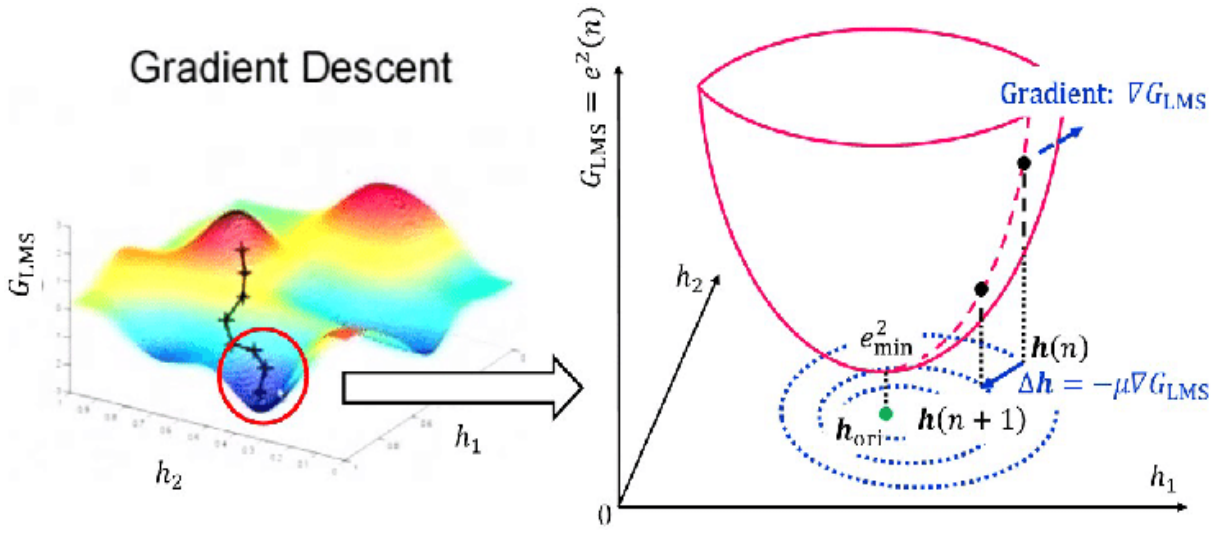
This function is also called “Squared error function”, or “Mean squared error”.

C.Gradient Descent

Gradient Descent is an algorithm to find the parameters(θ) for which the cost function J(θ) is minimized. The idea is to start from any random point and move in the direction of the steepest descent to reach the minima.

The way we do this is by taking the derivative (the tangential line to a function) of our cost function. 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.

Consider the following plot of J(θ) wrt θ_0 & θ_1. Gradient descent algorithm works such that it takes you to the local minima closest to the starting point. Thus if we start from a different point we may reach a different local minima which may not be the global minima.
However, the Cost Function for linear regression with one variable is always a convex function with only one global minima. Thus, gradient descent always converges to the global minimum in this case.



The gradient descent algorithm is repeated until convergence:

$$\theta_j := \theta_j - \alpha((\partial/\partial\theta_j)J(\theta))$$

At each iteration j, one should simultaneously update the parameters. Updating a specific parameter prior to calculating another one would yield to the wrong implementation.

Linear Regression with Multiple Variables

Also known as "multivariate linear regression", the multivariable form of the hypothesis function is given as:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \ldots\ldots + \theta_n x_n$$

By vectorization of our hypothesis, our multivariable hypothesis can be represented as:

$$h_\theta(x) = \begin{bmatrix} \theta_0 & \theta_1 & \ldots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = \theta^T x$$

Gradient Descent for Multiple Variables

The formula for Cost function doesn't change so, The gradient descent equation itself is generally the same form; we just have to repeat it for our 'n' features:

$$\begin{aligned} &\text{repeat until convergence: } \{ \\ &\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_0^{(i)} \\ &\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_1^{(i)} \\ &\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_2^{(i)} \\ &\ldots \\ &\} \end{aligned}$$

Features and Polynomial Regression

We can improve our features and form of our hypothesis function by combining multiple features into one, like we can combine x_1 and x_2 to give a more appropriate feature $x_3 = x_1.x_2$.

Considering our hypothesis function to be linear is not always reliable and thus we can change the curve of our hypothesis function by making it a polynomial function such as a quadratic or cubic equation.

For example, we can add additional features based on x_1, to get the quadratic function $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$ or the cubic function $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$.
In cubic function, $x_2 = x_1^2$ and $x_3 = x_1^3$ are the two newly created features.

Normal Equation

As Gradient descent, the Normal equation is also a method to minimize J. In the "Normal Equation" method, we will minimize J by explicitly taking its derivatives wrt the θ_j's, and setting them to zero. This allows us to find the optimum theta without iteration. The normal equation formula is given as

$$\Theta = (X^T X)^{-1} X^T y$$

Comparison between Gradient descent and Normal equation:-

m **training examples,** n **features.**

Gradient Descent	Normal Equation
• Need to choose α.	• No need to choose α.
• Needs many iterations.	• Don't need to iterate.
• Works well even when n is large.	• Need to compute $(X^T X)^{-1}$
	• Slow if n is very large.