Machine Learning Week-1

ramay7

February 12, 2017

Contents

1 Introduction

The popular definition of Machine Learning is :" A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if **its performance at tasks in T, as measured by P, improves with Experience E.**"

In general, any machine learning problem can be assigned to one of two broad classification: **Supervised learning and Unsupervised learning**.

In supervised learning, we are given a data set and **already** know what our correct output should look like, having the idea that there is a relationship between the input and the output. Supervised learning problems are categorized into **"regression" and "classification" problems**. In a regression problem, we are trying to map input variables to some **continuous function**. In a classification problem, we are instead trying to predict results in a **discrete output**.

Unsupervised learning allows us to approach problems with **little or no idea** what our results should look like. We can derive structure frome data where we don't necessarily know the effect of the varaibles. We can derive this structure by clustering the data **based on relationships among the variables** in the data. With unsupervised learning there is no feedback based on the prediction results.

2 Model and Cost Function

To describe the supervised learning problem slightly more formally, our goal is, given a training set, to learn a function h: X $\rightarrow$ 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**.

Specificly, we can get a linear regression hypothesis:

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

We can measure the accuracy of our hypothesis function by using a **cost funciton**. This takes an average difference (actually a fancier version of an average) of all the results of the hypothesis with inputs from x's and the actual output y's.

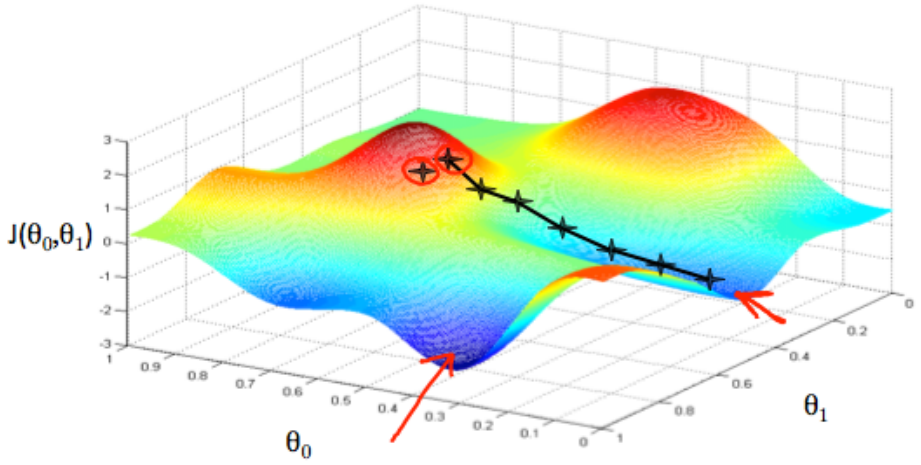
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

This function is otherwise called the "Squared error function", or "Mean squared error".

3 Parameter Learning

Now we need to estimate the parameters in the hypothesis function. That's where **gradient descent** comes in.

We put θ_0 on the x axis and θ_1 on the y axis, with the cost function on the vertical z axis. The points on our graph will be the result of the cost function using our hypothesis with those specific theta parameters. The graph below depicts such a setup.



We will know that we have succeeded when our cost function is at the very bottom of the pits in our graph. I.e. when its value is the minimum. The red arrows show the minimum points in the graph.

The way we do this 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. We make steps down the cost function in the direction with the steepest descent. The size of each step is determined by the parameter α, which is called the **learning rate**.

The gradient descent algorithm is:

repeat until convergence:{

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

where j=0,1 represents the feature index number

}

At each iteration j, one should **simultaneously** update the paramerers $\theta_1, \theta_2, \cdots, \theta_n$. That means updating all parameters together after calculating all parameters at one iteration.

If α is too small, gradient descent can be too slow. And on the other hand, if α is too large, gradient descent can overshoot the minimum, it may fail to converge, or even diverge.

3.1 Gradient Descent For Linear Regression

We can substitute our actual cost function and our actual hypothesis function and modify the equation to:

repeat until convergence: {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i) \tag{1}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x_i) - y_i)x_i) \tag{2}$$

}

where m is the size of the training set, θ_0 a constant that will be changing simultaneously with θ_1 and x_i, y_i are values of the given training set(data).

We can easily prove these two equations with the definition of linear regression hypothesis and gradient descent.

Note that, while gradient descent can be susceptible to local minima in general, the optimization problem we have posed here for linear gregression 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.

In other words, there are some problems who may have local optima. And evenly, we may at the worse condition (the top point on the contour plots) at first. So we need to test our θ_0, θ_1 and α many times.

4 Linear Algebra Review

For a matrix B with m rows and o colums multiplied by a matrix A with n rows and m columns, we can get a matrix with n rows and o columns.

In a general way, for two matrices A and B, $A \times B \neq B \times A$. However, $A \times E = E \times A$, where A is a square matrix and E is a unit matrix.

It is also necessary to know the concept of Inverse and Transpose.