Machine Learning Week-2

ramay7

February 15, 2017

Contents

1 Multivariate Linera Regression 1

2 Computing Parameters Analytically(Normal Equation) 2

3 The End 2

1 Multivariate Linera Regression

The multivariable form of the hypothesis function accommodating these multiple features is as follows:

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

In addition, if we set $x_0^{(i)} = 1$, we can get:

$$h_\theta(X) = X\theta$$

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

The Gradient Descent is: **repeat until convergence:{**

$$\theta_j := \theta_j - \alpha\frac{1}{m}\frac{\partial}{\partial\theta_j}J(\theta) \tag{1}$$

$$:= \theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \tag{2}$$

for j = 0, 1, 2,...,n

}

We can speed up gradient descent by having each of our input values in roughly the same range, such as:

$$-1 \le x_{(i)} \le 1 \ or \ -0.5 \le x \le 0.5$$

Two techniques to help with this are **feature scaling** and **mean normalization**.

$$x_i := \frac{x_i - \mu_i}{s_i}$$

Where μ_i is the **average** of all the value for feature(i) and s_i is the range of values (max - min), or s_i is the standard deviation.

By the way, in the ex1.pdf, there is a introduction to 'standard deviation'. The standard deviation is a way of measuring how much variation there is in the range of values of a particular feature (most data points will lie within ± 2 standard deviations of the mean); this is an alternative to taking the range of values (max - min). In MATLAB, we can use the "std" function to compute the standard devision.

We can **combine** multiple features into one. For example, we can combine x_1 and x_2 into a new feature x_3 by taking $x_1 \dot{x}_2$. We can **change the behavior or curve** of our hypothesis function by making it a quadratic, cubic, or square root function (or any other form).

2 Computing Parameters Analytically(Normal Equation)

It is no doubt that gradient descent gives one way of minizing J. Let's discuss a second way of doing so, this time performing the minimization explicitly and without resorting to an iterative algorithm. The normal equation formula is given below:

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

There is no need to do feature scaling with the normal equation.

The following is a comparison of gradient descent and the normal equation:

Gradient Descent	Normal Equation
Need to choose alpha	No need to choose aplha
Needs many iterations	No need to iterate
O(kn^2)	O(n^3), need to calculate inverse of $X^T X$
Works well when n is large	Slow if n is very large

When implementing the normal equation in MATLAB, we want to use the 'pinv' function rather than 'inv'. The 'pinv' function will give you a value of θ even if $X^T X$ is not invertible.

if $X^T X$ is **noninvertible**, the common causes might be having:

- Redundant features, where two features are very closely related (i.e. they are lineraly dependent)
- Too many features (e.g. m $\leq$ n). In this case, delete some features or use "regularization" .

Solutions to the above problems include deleting a feature that is lineraly dependent with another or deleting one or more features when there are too many features.

3 The End

The most important harvest for me is finishing Exercise 1 completely, including the optional exercises. By this, I have got a little familiar with MATLAB. At last, I have got my first machine learning model for predicting the prices of houses roughly. You can get the source code in my week-2 file.