EX 2 Tutorial

Ex 2 Tutorial: Sigmoid

You can get a one-line function for sigmoid(z) if you use only element-wise operators.

- The exp() function is element-wise.
- The addition operator is element-wise.
- Use the element-wise division operator ./

Combine these elements with a few parenthesis, and operate only on the parameter 'z'. The return value 'g' will then be the same size as 'z', regardless of what data 'z' contains.
============
keywords: tutorial sigmoid

Ex2 Tutorial: vectorizing the Cost function

Tom MosherMentorWeek 3 · 2 years ago · Edited

The regularized cost calculation can be vectorized easily. Here is the cost equation from ex2.pdf, page 9.

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

1. The hypothesis is a vector, formed from the sigmoid() of the products of X and θ
   . See the equation on ex2.pdf - Page 4. Be sure your sigmoid() function passes the submit grader before going any further.
2. First focus on the circled portions of the cost equation. Each of these is a vector of size (m x 1). In the steps below we'll distribute the summation operation, as shown in purple, so we end up with two scalars (for the 'red' and 'blue' calculations).
3. The red-circled term is the sum of -y multiplied by the natural log of h. Note that the natural log function is log(). Don't use log10().   Since we want the sum of the products, we can use a vector multiplication. The size of each argument is (m x 1), and we want the vector product to be a scalar, so use a transposition so that (1 x m) times (m x 1) gives a result of (1 x 1), a scalar.
4. The blue-circled term uses the same method, except that the two vectors are (1 - y) and the natural log of (1 - h).
5. Subtract the right-side term from the left-side term

6. Scale the result by 1/m. This is the unregularized cost.
7. Now we have only the regularization term remaining. We want the regularization to exclude the bias feature, so we can set theta(1) to zero. Since we already calculated h, and theta is a local variable, we can modify theta(1) without causing any problems.
8. Now we need to calculate the sum of the squares of theta. Since we've set theta(1) to zero, we can square the entire theta vector. If we vector-multiply theta by itself, we will calculate the sum automatically. So use the same method we used in Steps 3 and 4 to multiply theta by itself with a transposition.
9. Now scale the cost regularization term by (lambda / (2 * m)). Be sure you use enough sets of parenthesis to get the correct result. **Special Note for those whose cost value is too high:** 1/(2*m) and (1/2*m) give drastically different results.
10. Now add your unregularized and regularized cost terms together.

Ex2 Tutorial: vectorizing the gradient calculation

Tom MosherMentorWeek 3 · 2 years ago · Edited

The gradient calculation can be easily vectorized. See this two formulas from ex2.pdf pages 9 and 10.

Note: ignore the

$\boldsymbol{\lambda}$

term in the 2nd equation if you are working on costFunction() - just do Step 1 and Step 2.

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \qquad \text{for } j = 0$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \left(\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j \quad \text{for } j \geq 1$$

Note that if we set

$\boldsymbol{\theta}$

0

to zero (in Step 3 below), the second equation is exactly equal to the first equation. So we can ignore the "j = 0" condition entirely, and just use the second equation.

1. Recall that the hypothesis vector h is the sigmoid() of the product of X and
   θ
   (see ex2.pdf - Page 4). You probably already calculated h for the cost J calculation.
2. The left-side term is the vector product of X and (h - y), scaled by 1/m.

You'll need to transpose and swap the product terms so the result is (m x n)' times (m x 1) giving you a (n x 1) result. This is the unregularized gradient. Note that the vector product also includes the required summation.

3. Then set theta(1) to 0 (if you haven't already).
4. Then calculate the regularized gradient term as theta scaled by (lambda / m).
5. The grad value is the sum of the Step 2 and Step 4 results. Since you forced theta(1) to be zero, the grad(1) term will only be the unregularized value.

============

keywords: ex2 tutorial costfunction tutorial costfunctionreg gradient

Tutorial for ex2: predict()

Tom MosherMentorWeek 3 · 9 months ago · Edited

This is logistic regression, so the hypothesis is the sigmoid of the product of X and theta.

Logistic prediction when there are only two classes uses a threshold of >= 0.5 to represent 1's and < 0.5 to represent a 0.

Here's an example of how to make this conversion in a vectorized manner. Try these commands in your workspace console, and study how they work:

```
v = rand(10,1)     % creates some random values between 0 and 1
v >= 0.5           % performs a logical comparison on each value
```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Inside your predict.m script, you will need to assign the results of this sort of logical comparison to the 'p' variable. You can use "p = " followed by a logical comparison inside a set of parenthesis.
