

In the first half of the exercise, you will implement regularized linear regression to predict the amount of water owing out of a dam using the change of water level in a reservoir. In the next half, you will go through some diagnostics of debugging learning algorithms and examine the effects of bias v.s. variance.

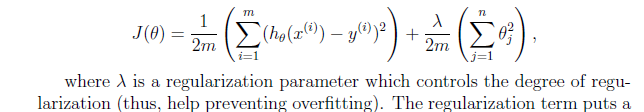
The provided script, ex5.m, will help you step through this exercise.

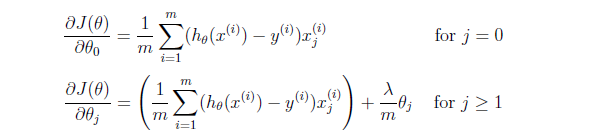
This dataset is divided into three parts:

• A training set that your model will learn on: X, y

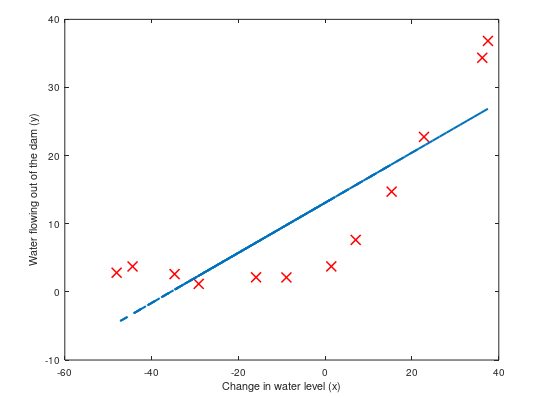
• A cross validation set for determining the regularization parameter**: Xval, yval**

• A test set for evaluating performance. These are unseen" examples which your model did not see during training: Xtest, ytest





* Cost at theta = [1 ; 1]: 303.993192
* Gradient at theta = [1 ; 1]: [-15.303016; 598.250744]
* Iteration 2 | Cost: 2.237391e+01



**Learning Curve**

**Once your cost function and gradient are working correctly**, the next part of ex5.m will run the code in trainLinearReg.m to compute the optimal values 4of Theta . This training function uses fmincg to optimize the cost function.In this part, we set regularization parameter lambda to zero.

function [theta] = **trainLinearReg**(X, y, lambda)

% Initialize Theta

initial\_theta = zeros(size(X, 2), 1);

% Create "short hand" for the cost function to be minimized

costFunction = @(t) linearRegCostFunction(X, y, t, lambda);

% Now, costFunction is a function that takes in only one argument

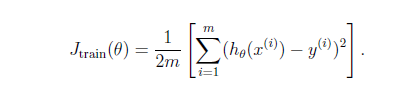
options = optimset('MaxIter', 200, 'GradObj', 'on');

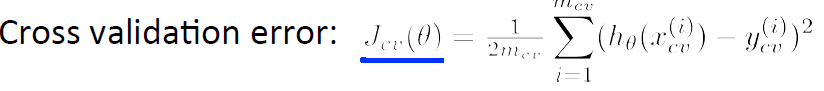
% Minimize using fmincg

theta = fmincg(costFunction, initial\_theta, options);

Usuage

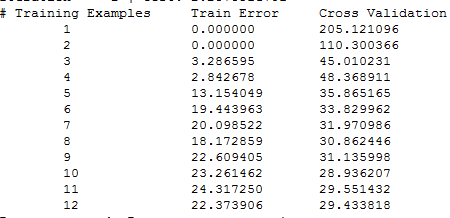
***theta = trainLinearReg(Xtrain,ytrain,lambda);***

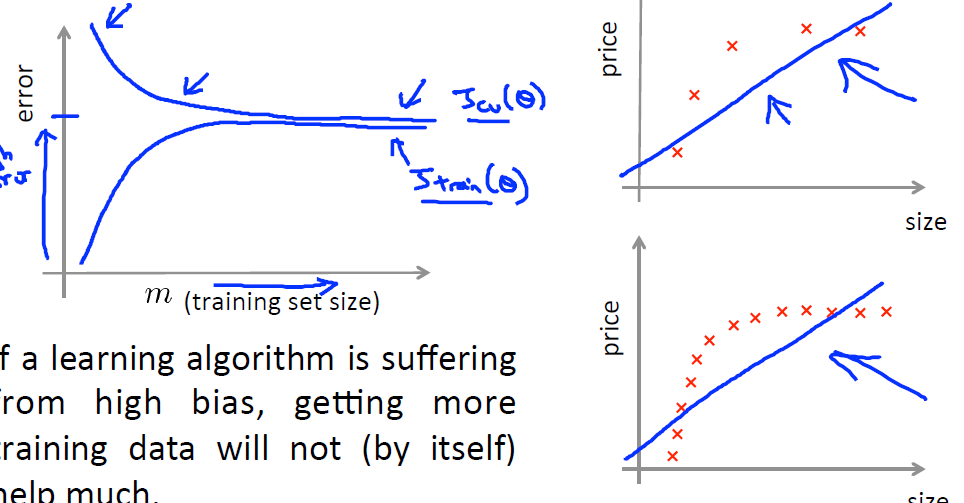
 Already Done Used*linearRegCostFunction*



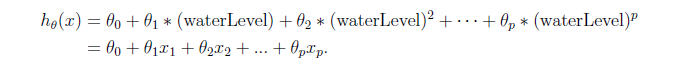
*error\_train(i) = linearRegCostFunction(Xtrain,ytrain,theta,0);*

*error\_val(i) = linearRegCostFunction(Xval,yval,theta,0);*





polyFeatures.m



when a training set X of size m x 1 is passed into the function, the function should return a m x p matrix X\_poly,where

* column 1 holds the original values of X,
* column 2 holds the values ofX.^2, (Square)
* column 3 holds the values of X.^3, and so on .(Cube)

**The Last .m File**

* Even though we have polynomial terms in our feature vector, we are still solving a linear regression optimization problem. The polynomial terms have simply turned into features that we can use for linear regression. We are using the same cost function and gradient that you wrote for the earlier part of this exercise.

%% =========== Part 6: Feature Mapping for Polynomial Regression =============

% One solution to this is to use polynomial regression. You should now

% complete polyFeatures to map each example into its powers

p = 8;

% Map X onto Polynomial Features and Normalize X\_poly

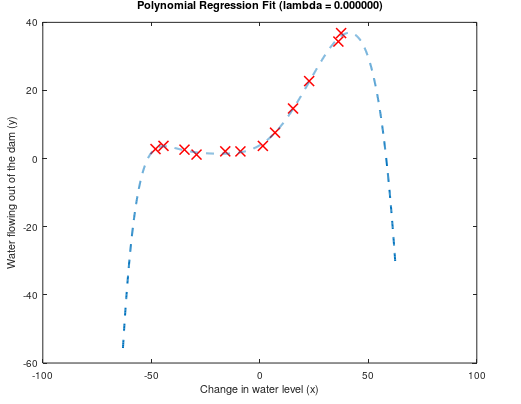
X\_poly = polyFeatures(X, p);

[X\_poly, mu, sigma] = featureNormalize(X\_poly); % Normalize

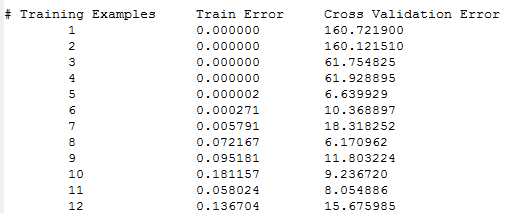
X\_poly = [ones(m, 1), X\_poly]; % Add Ones

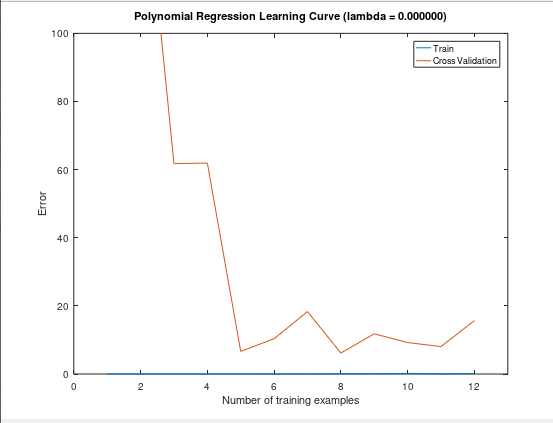
1. For this part of the exercise, you will be using a polynomial of degree 8.
2. It turns out that if we run the training directly on the projected data, will not work well as the features would be badly scaled (e.g., an example with x = 40 will now have a feature x8 = 408 = 6:5 x 10^12). Therefore, you will need to use **feature normalization**
3. .**Before learning the parameters Theta for the polynomial regression**, ex5.m will first call featureNormalize and normalize the features of the training set, storing the mu, sigma parameters separately.
4. We have already implemented this function for you and it is the same function from the rst exercise.
5. **After learning the parameters Theta** , you should see two plots (Figure 4,5)

From Figure 4

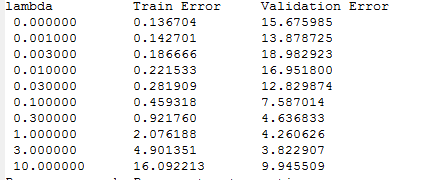


* **Figure 5)**





1. To better understand the problems with the unregularized (lamda = 0) model,you can see that the learning curve (Figure 5) shows the same effect where the low training error is low, but the cross validation error is high.
2. There is a gap between the training and cross validation errors, indicating a highvariance problem.



* One way to combat the over\_ftting (high-variance) problem is to add
* regularization to the model. In the next section, you will get to try di\_erent

parameters to see how regularization can lead to a better model.

|  |
| --- |
| Its Extra Refer the pdf |
|  |
| REVIEW QAUTION FOR THE SECOND PART OF WEEK 6 : |
| Recall, precision , F Score , Error Analysis |

