ieeta instituto de engenharia electrónica e telemática de aveiro

universidade de aveiro

Departamento de Eletrónica, Telecomunicações e Informática

Machine Learning

Lecture 6: Model selection and validation – Bias vs. Variance

Petia Georgieva
(petia@ua.pt)

Deciding what to do next ?

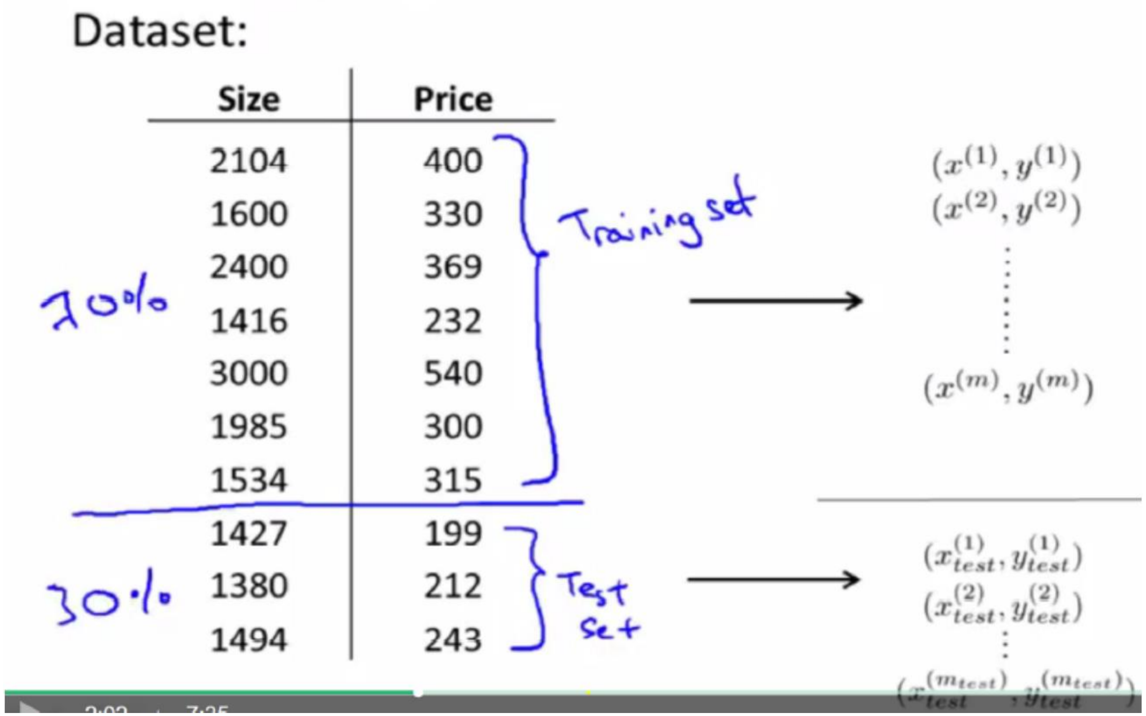
Suppose you have trained a ML model on some data. However, when you test your hypothesis on a new set of data, you find that it makes unacceptably large errors in its prediction . What should you do ?

-- Get more training examples ?
-- Try smaller sets of features ?
-- Try getting additional features ?
-- Try adding polynomial features ?
-- Try decreasing/ increasing the regularization parameter lambda ?

Machine learning diagnostics:

You need to run tests to gain insight what isn't working with the learning algorithm and how to improve its performance.
Diagnostics can take time to implement, but can be a very good use of your time.

Training/Testing subsets (*<u>one model</u>*)



- **Learn model parameters Theta from training data**

  (minimize cost function J)

- Compute the test error (MSE !!!)

$$E_{test}(\theta) = \frac{1}{2m_{test}}\left[\sum_{i=1}^{m_{test}}\left(h_\theta\left(x_{test}^{(i)}\right) - y_{test}^{(i)}\right)^2\right]$$

or

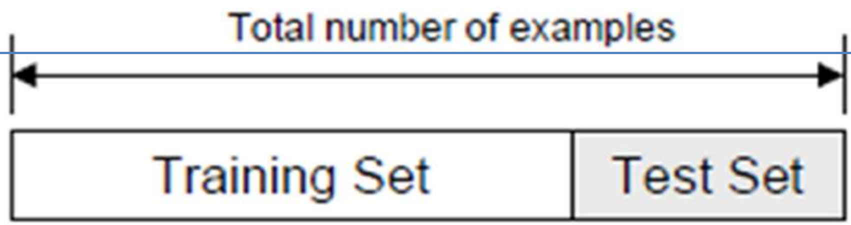
- **Compute misclassification error** (for classification problems)

  (# of correctly classified test examples / # of all test examples)

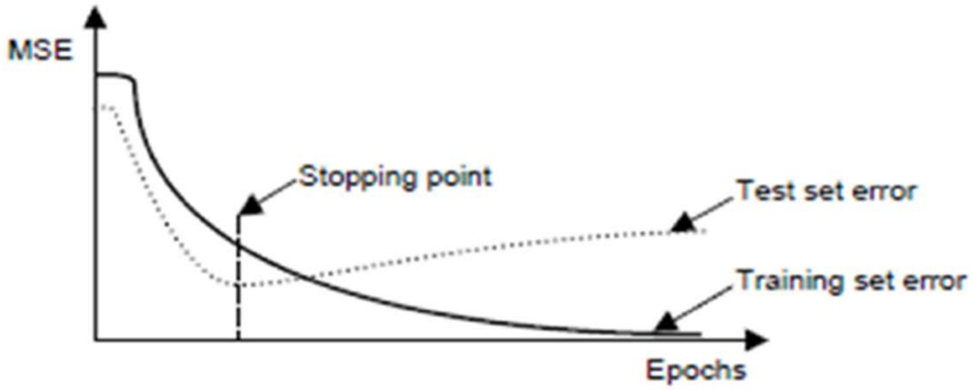
Holdout method (train and test subsets)

1. **Split data into two sets:**

- **Training set : used to train the model**
- **Test set : used to test the trained model**



Mean Squared Error (MSE) is not the same as the cost function!!!



COST (LOSS) FUNCTIONS

Training data MSE

- **Regularized Linear Regression Cost Function**

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

Ridge Regression

- **Regularized Logistic Regression Cost Function**

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

- **Regularized SVM Cost Function**

$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)}cost_1(\theta^T x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{i=1}^{n}\theta_j^2$$

Model selection (choose the best hypothesis)

1. $h_\theta(x) = \theta_0 + \theta_1 x$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_3 x^3$

$\vdots$

10. $h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10}$

Given many models (for example with different polynomial degrees or differrent algorithms – LogReg, NN, SVM, etc.).

In order to choose the best model, devide dataset in 3 sets : training, cross validation (CV) and test sets.

Training/Validation/Test subsets

Dataset:

	Size	Price	
	2104	400	
	1600	330	
60%	2400	369	Training set
	1416	232	
	3000	540	
	1985	300	
20%	1534	315	Cross validation set (CV)
	1427	199	
20%	1380	212	test set
	1494	243	

$$(x^{(1)}, y^{(1)})$$
$$(x^{(2)}, y^{(2)})$$
$$\vdots$$
$$(x^{(m)}, y^{(m)})$$

$$(x_{cv}^{(1)}, y_{cv}^{(1)})$$
$$(x_{cv}^{(2)}, y_{cv}^{(2)})$$
$$\vdots$$
$$(x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$$

$$(x_{test}^{(1)}, y_{test}^{(1)})$$
$$(x_{test}^{(2)}, y_{test}^{(2)})$$
$$\vdots$$
$$(x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$$

Model selection (choose the best hypothesis)

Step 1: Optimize parameters Theta (to minimize the cost function *J*) using the same training set for each model. Compute the training error:

Training error:

$$E_{train}(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2\right]$$

Step 2: Test the optimized models from step 1 with the CV set and choose the model with the min CV error:

Cross validation (CV) error:

$$E_{cv}(\theta) = \frac{1}{2m_{cv}}\left[\sum_{i=1}^{m_{cv}}\left(h_\theta\left(x_{cv}^{(i)}\right) - y_{cv}^{(i)}\right)^2\right]$$

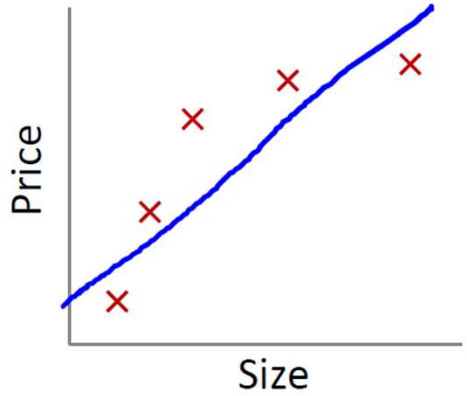
Step 3: Retrain the best model from step 2 with both train and CV sets starting from the parameters got at step 2. Test the retrained model with test set and compute test error (***this is the real model performance !!!***):

Test error:

$$E_{test}(\theta) = \frac{1}{2m_{test}}\left[\sum_{i=1}^{m_{test}}\left(h_\theta\left(x_{test}^{(i)}\right) - y_{test}^{(i)}\right)^2\right]$$

Bias vs. Variance

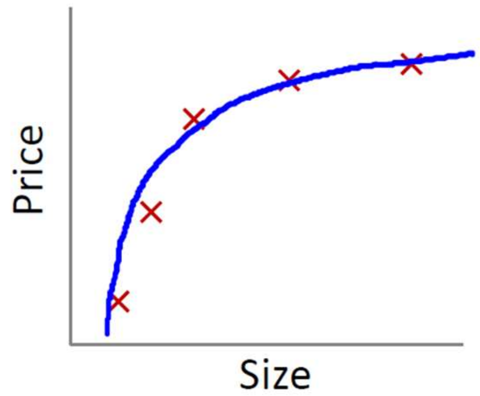
An important concept in machine learning is the bias-variance tradeoff. Models with high bias are not complex enough for the data and tend to under-fit, while models with high variance over-fit the training data.



underfit- high bias

(1st order polinom. model)

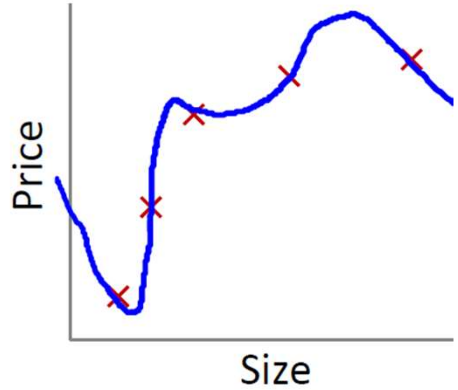
$$h_\theta(x) = \theta_0 + \theta_1 x$$



just right

(3rd order polinom. model)

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$



overfit- high variance

(higher ord. polinom. Model)

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + ... + \theta_{16} x^n$$

Diagnosing Bias vs. Variance

Suppose your learning algorithm is performing less well than you expect (Cross Validation Error or Test Error is high).
Is it a bias problem or a variance problem ?

Bias (underfit) problem:

E_{train} will be high
E_{cv} will be also high

Variance (overfit) problem:

E_{train} will be low
E_{cv} much higher than E_{train}

Model selection (choose the best regularization parameter λ)

For a given model:

Try different values of λ =[0, 0.01, 0.1, 1,....]

Step 1: For each λ, optimize parameters θ using the training set

$$E_{train}(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2\right]$$

Step 2: Test the optimized models from step 1 with the CV set and choose the model with λ that gets min CV error:

$$E_{cv}(\theta) = \frac{1}{2m_{cv}}\left[\sum_{i=1}^{m_{cv}}\left(h_\theta\left(x_{cv}^{(i)}\right) - y_{cv}^{(i)}\right)^2\right]$$

Step 3: Retrain the model with best λ from step 2 with both train and CV sets starting from the parameters θ got at step 2. Test the retrained model with the test set and compute the error:

$$E_{test}(\theta) = \frac{1}{2m_{test}}\left[\sum_{i=1}^{m_{test}}\left(h_\theta\left(x_{test}^{(i)}\right) - y_{test}^{(i)}\right)^2\right]$$

Bias/Variance as a function of the regularization parameter

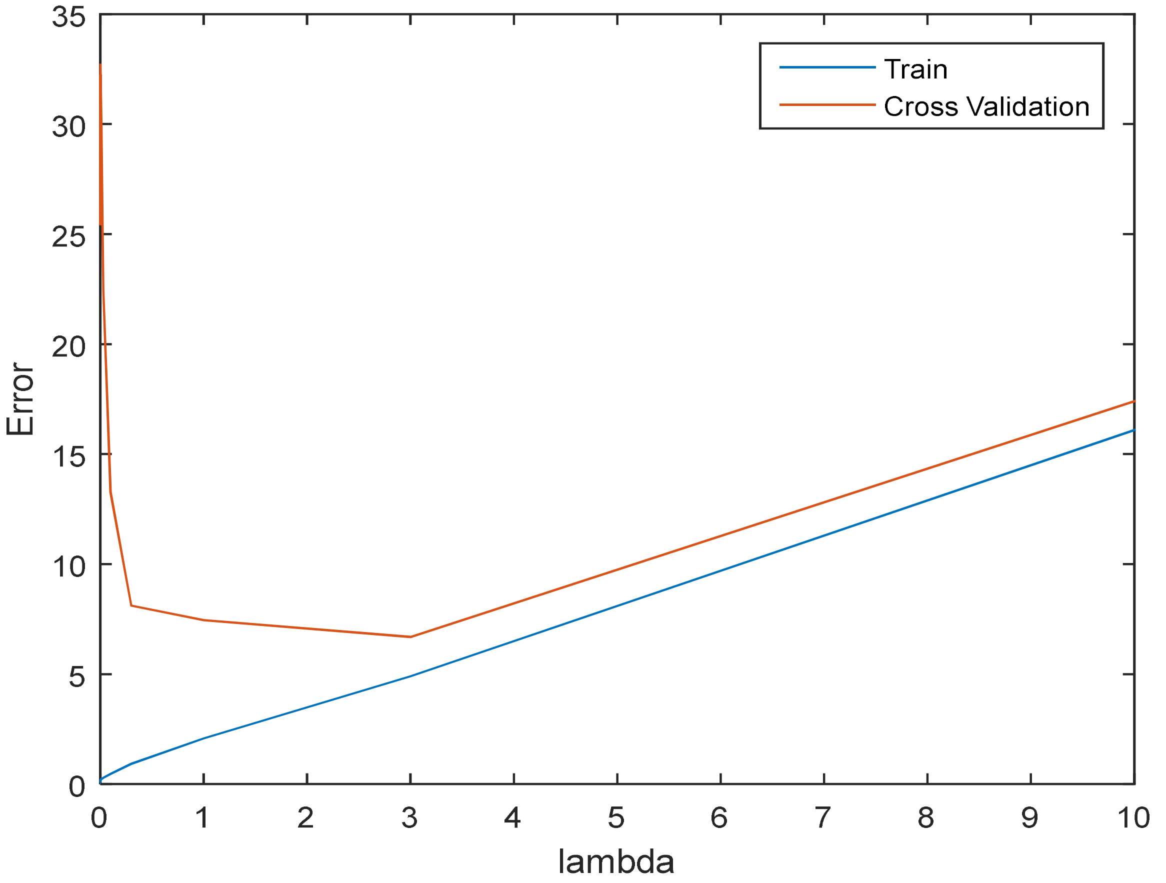
Bias (underfit) problem => too large λ :

E_{train} will be high
E_{cv} *will be also high*

Variance (overfit) problem=> too small λ

E_{train} will be low
E_{cv} *will be much higher than* E_{train}

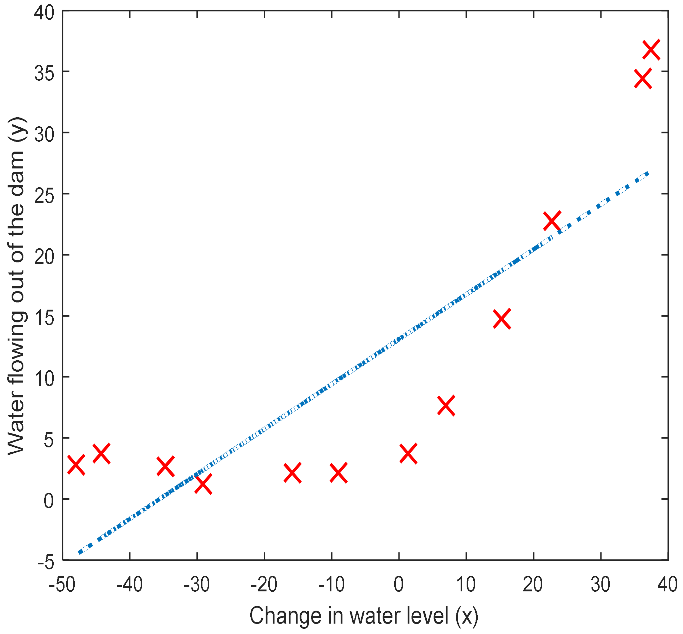
Select λ using CV set

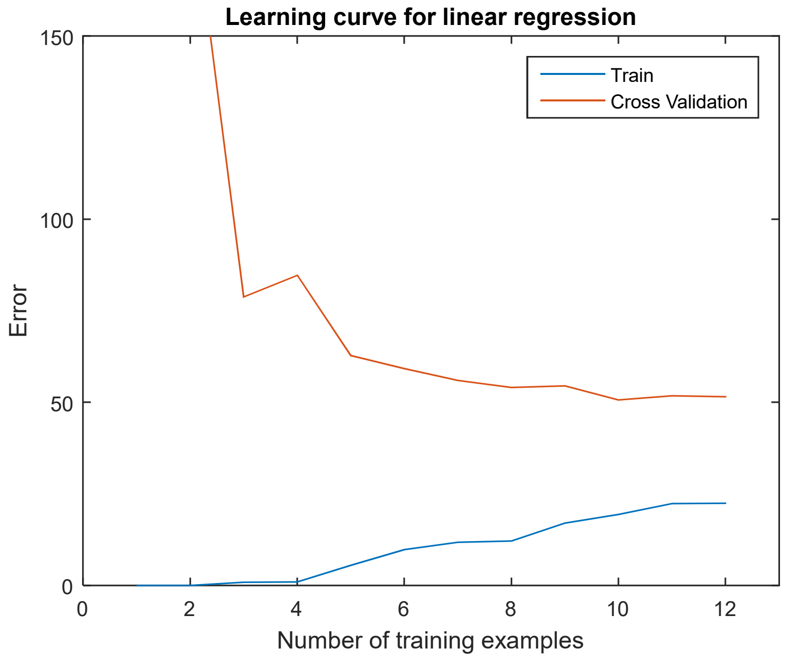


Best λ =3

Learning Curves

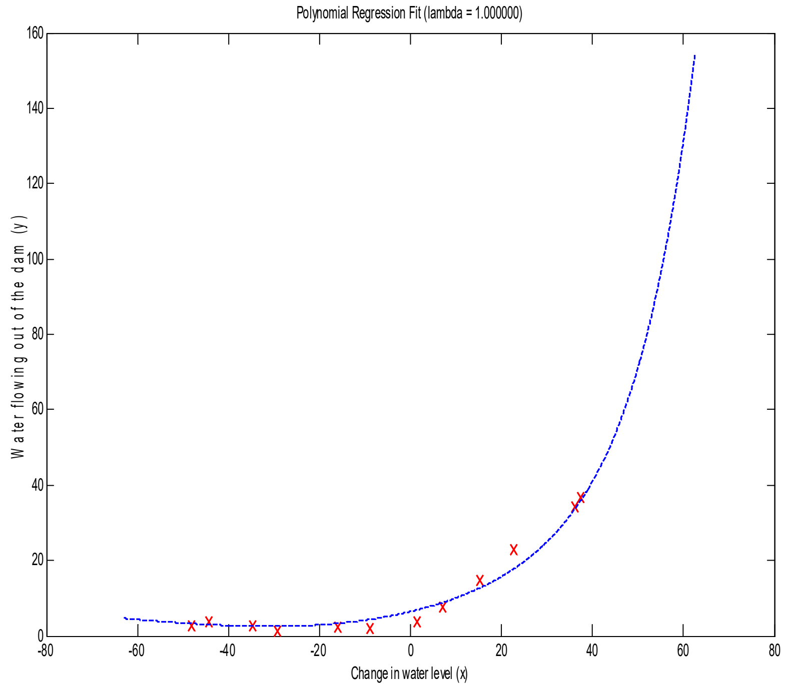
$$h_\theta(x) = \theta_0 + \theta_1 x$$

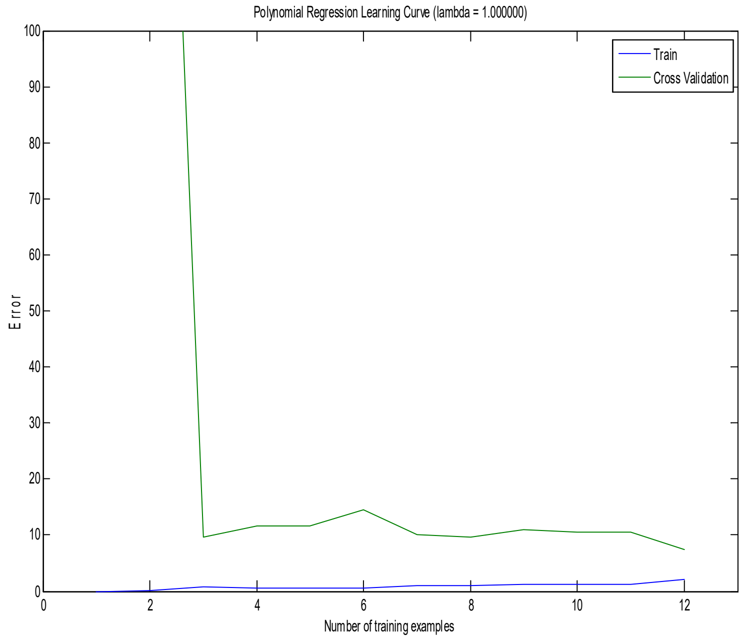




If a learning algorithm is suffering from high bias, getting more training data will not help much

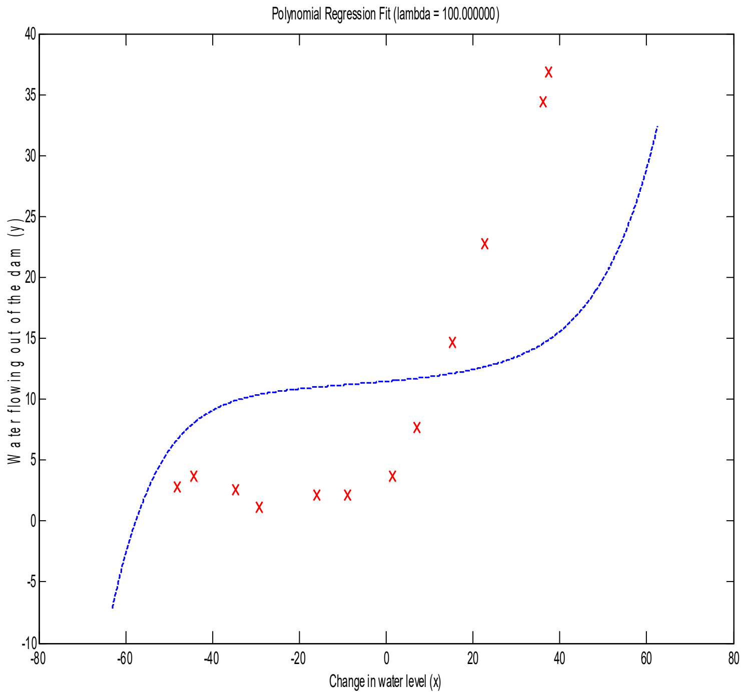
Learning Curves

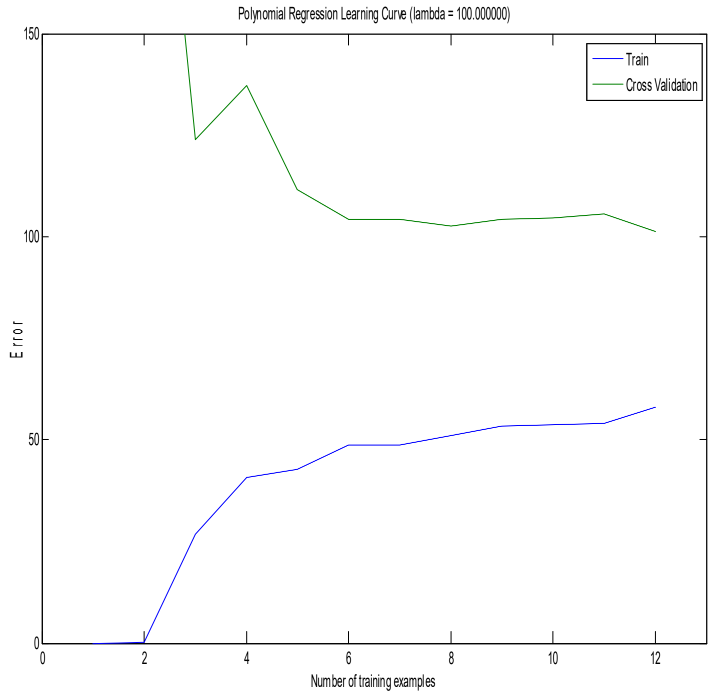




If a learning algorithm is suffering from high variance, getting more training data is likely to help

Regularization and Learning Curves





Polynomial regression, λ =100 **Learning curve, λ =100**

Advice for applying machine learning

Suppose you have learned a data model (hypothesis). However, when you test your hypothesis on a new set of data, you find that it makes unacceptably large errors in its prediction (regression or classification). What should you try next?

-- Get more training examples – fixes high variance

-- Try smaller sets of features – fixes high variance

-- Try getting additional features – fixes high bias

-- Try adding polynomial features - fixes high bias

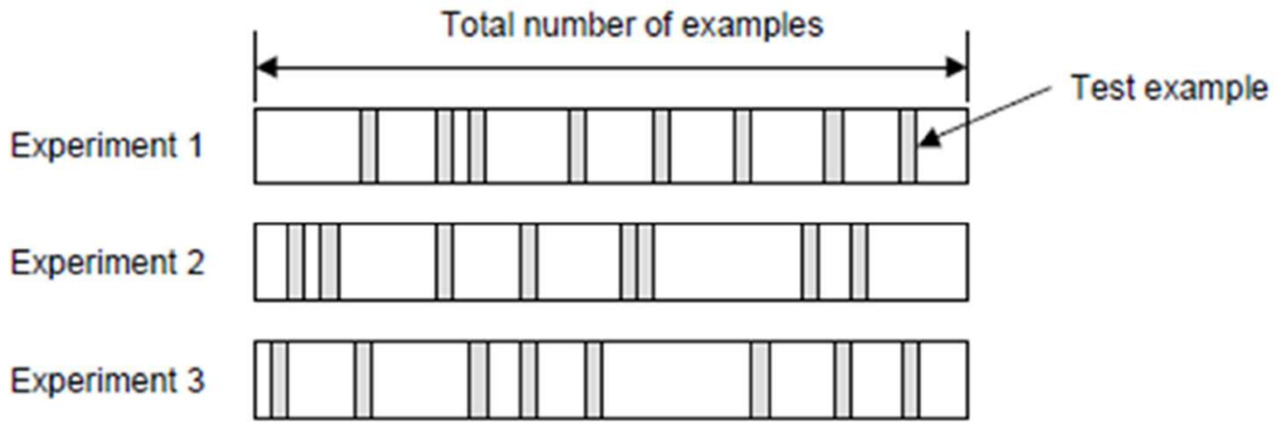
-- Try decreasing λ – fixes high bias

- Try increasing λ – fixes high variance

Cross Validation – Random Subsampling

- Make K training experiments, where each time randomly select some of the examples for training (70%) and the rest for CV without replacement.
- For each data split retrain the model from scratch with the training examples and estimate the error *Ecv* with the CV examples.
- The final validation error is obtained as the average of the CV errors.
- The estimate is significantly better than the holdout method.

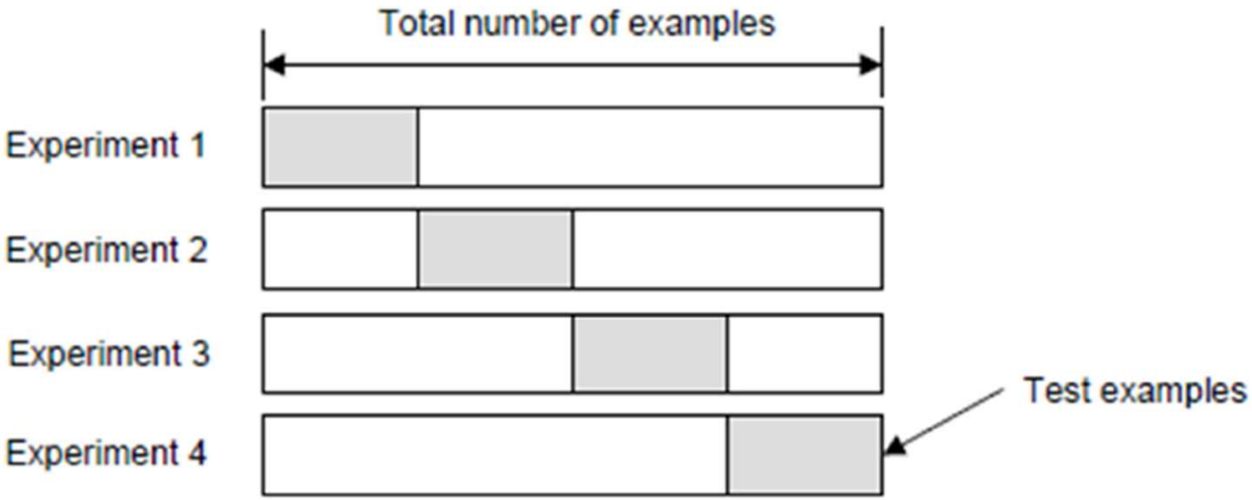
$$E_{cv} = \frac{1}{K}\sum_{i=1}^{K} E_{testi}$$



K –fold Cross Validation

- Devide data into K subsets (K-fold).
- Use K-1 subsets for training and the remaining subset for CV.
- The advantage of K-fold CV is that all examples in the dataset are used for both training and validation.
- As before the final validation error is estimated as the average CV error.

$$E_{cv} = \frac{1}{K} \sum_{i=1}^{K} E_{testi}$$



universidade de aveiro

Leave-one-out Cross Validation

- Leave-one-out is the degenerate case of K-fold CV, where K is chosen as the total number of examples.
- For a dataset with *N* examples, perform m experiments.
- For each experiment use *N-1* examples for training and the remaining example for CV.
- As before the final validation error is estimated as the average error on CV examples.
- Useful for small data sets.

$$E_{cv} = \frac{1}{K}\sum_{i=1}^{K} E_{testi}$$