Contents

Debugging a Learning Algorithm

April 18, 2016

Part I

Week 6

0.1 Machine Learning Diagnostic

It is important to run tests to gain insight of what is/isn't working with a learning algorithm.

0.1.1 Evaluating a hypothesis

Because a hypothesis has low training error does not mean the hypothesis is best (could be overfitting). One approach to test an hypothesis, is to split the dataset into 2 sub-sets after shuffling (mixing) the data:

- Training set (≈70% of the entire dataset)
- Test set (≈30%)
- Training/Testing procedure for **linear regression**:
  - Learn parameter θ from training data (minimum training error $J(\theta)$)
  - Compute test set error using θ values obtained the training set:

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

- Training/Testing procedure for **logistic regression**:
  - Learn parameter θ from training data (minimum training error $J(\theta)$)
  - Compute test set error using θ values obtained the training set:

$$J_{test}(\theta) = \frac{-1}{2m_{test}} \sum_{i=1}^{m_{test}} y_{test}^{(i)} \log(h_\theta(x_{test}^{(i)})) + (1 - y_{test}^{(i)}) \log(1 - h_\theta(x_{test}^{(i)})) \tag{2}$$

  - Alternative: misclassification error ("0/1 misclassification")

$$\text{err}(h_\theta(x), y) = \begin{cases} 1 & \text{if } h_\theta(x) \geq 0.5 \text{ and } y = 0 \\ & \text{orif } h_\theta(x) < 0.5 \text{ and } y = 1 \\ 0 & \text{otherwise (hypothesis classify the example correctly} \end{cases}$$

$$\text{testErr} = \frac{1}{m_{test}} \sum_{i=1}^{m_{test}} \text{err}(h_\theta(x_{test}^{(i)}, y_{test}^{(i)}) \tag{3}$$

  This is the fraction of the examples in the **test set** that the hypothesis labeled.

0.1.2 Model selection and training/validation/test

- we want to decide on a model for the hypothesis (d is the degree of polynomial):
  1. $h_\theta(x) = \theta_0 + \theta_1 x$ (d=1)
  2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$ (d=2)
  3. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$ (d=3)
     ....
     ....

4. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + ... + \theta_{10} x^{10}$ (d=10)

One option is to fit all the models to the train data and then compute $J_{test}(\theta^{(d)})$, and select the model giving the lowest test set error.

1. $h_\theta(x) = \theta_0 + \theta_1 x$ (d=1) $\rightarrow \theta^{(1)} \rightarrow J_{test}(\theta^{(1)})$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$ (d=2) $\rightarrow \theta^{(2)} \rightarrow J_{test}(\theta^{(2)})$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$ (d=3) $\rightarrow \theta^{(3)} \rightarrow J_{test}(\theta^{(3)})$
   ....
   ....
4. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + ... + \theta_{10} x^{10}$ (d=10) $\rightarrow \theta^{(10)} \rightarrow J_{test}(\theta^{(10)})$

How well the model generalize to new examples, cannot be determined by selecting the model with the lowest $J_{test}(\theta^{(d)})$. Indeed, if we develop new features by examining the test set, then we may end up choosing features that work well specifically for the test set. So, $J_{test}(\theta)$ is no longer a good estimate of how well we generalize to new examples.

- In fact a more robust approach is to split the dataset into 3 parts:

1. Training set (≈60% of dataset) with training error:

$$J_{train}(\theta) = \frac{1}{2m_{train}} \sum_{i=1}^{m_{train}} (h_\theta(x_{train}^{(i)}) - y_{train}^{(i)})^2 \tag{4}$$

2. Cross validation (CV) or Validation set (≈20%) with cross validation error:

$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2 \tag{5}$$

3. Test set (≈20%) with test error:

$$J_{test}(\theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} (h_\theta(x_{test}^{(i)} - y_{test}^{(i)})^2 \tag{6}$$

When faced with a model selection problem, we will use Cross Validation error:

1. $h_\theta(x) = \theta_0 + \theta_1 x$ (d=1) $\rightarrow \theta^{(1)} \rightarrow J_{cv}(\theta^{(1)})$
2. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$ (d=2) $\rightarrow \theta^{(2)} \rightarrow J_{cv}(\theta^{(2)})$
3. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$ (d=3) $\rightarrow \theta^{(3)} \rightarrow J_{cv}(\theta^{(3)})$
   ....
   ....
4. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + ... + \theta_{10} x^{10}$ (d=10) $\rightarrow \theta^{(10)} \rightarrow J_{cv}(\theta^{(10)})$

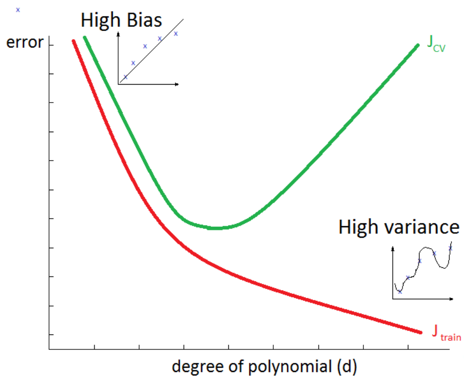
We select the model resulting in the lowest Cross validation error $J_{cv}(\theta^{(d)})$, and use it on the test set to check the generalization of the model.

0.2 Diagnosing Bias vs. Variance

- **Definition**:
  - High Bias = underfit
  - High Variance = overfit

0.2.1 Choice of model

Let's consider the problem of choosing the polynomial degree (d) of the model:



If the algorithm suffers from:

- High Bias (underfit): $J_{train}(\theta) \approx J_{cv}(\theta)$, both high
- Variance (overfit): $J_{train}(\theta)$ is low and $J_{cv}(\theta) >> J_{train}(\theta)$

0.2.2 Regularization (λ) and Bias/Variance

Increasing regularization helps to fix High variance (overfitting).

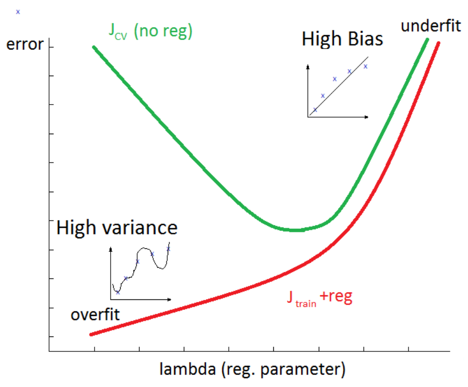
- Model:

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 \tag{7}$$

$$h_\theta(x) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)} - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2 \tag{8}$$

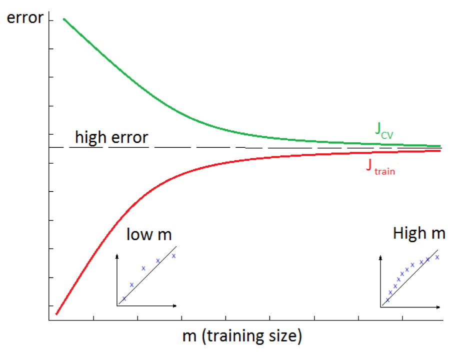
- Choosing the regularization parameter λ:
  - (1) try $\lambda = 0 \Rightarrow \min_\theta J_{test}(\theta^{(1)}) \rightarrow \theta^{(1)} \Rightarrow J_{cv}(\theta^{(1)})$
  - (2) try $\lambda = 0.01 \Rightarrow \min_\theta J_{test}(\theta^{(2)}) \rightarrow \theta^{(2)} \Rightarrow J_{cv}(\theta^{(2)})$
  - (3) try $\lambda = 0.02 \Rightarrow \min_\theta J_{test}(\theta^{(3)}) \rightarrow \theta^{(3)} \Rightarrow J_{cv}(\theta^{(3)})$
  - (4) try $\lambda = 0.04 \Rightarrow \min_\theta J_{test}(\theta^{(4)}) \rightarrow \theta^{(4)} \Rightarrow J_{cv}(\theta^{(4)})$
    ...
    ...
    ...
  - (12) try $\lambda = 10 \Rightarrow \min_\theta J_{test}(\theta^{(10)}) \Rightarrow \theta^{(10)} \rightarrow J_{cv}(\theta^{(10)})$

Pick λ with lowest $J_{cv}(\theta^{(\lambda)})$ and compute $J_{test}(\theta^{(\lambda)})$



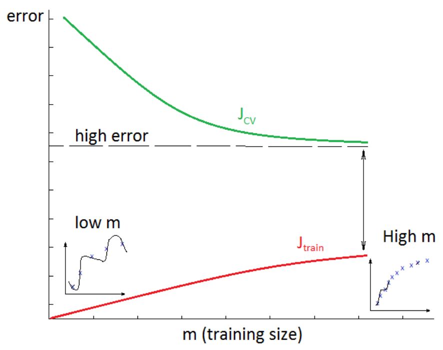
0.2.3 Learning curves: training size

- case of High Bias



If algorithm suffers from High Bias:
1) residual error $J_{train}(\theta)$ and $J_{cv}(\theta)$ is high
2) $J_{train}(\theta) \approx J_{cv}(\theta)$
3) Increasing Nbr of training data will not help by itself

- Case of High Variance (for example: use of a high order polynomial hypothesis)



If algorithm suffers from High variance :
1) residual error of $J_{train}(\theta)$ is low
2) $J_{cv}(\theta) >> J_{train}(\theta)$
3) Increasing Nbr of training data is likely to help

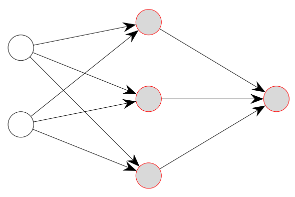
0.3 steps to debug a learning algorithm

Suppose we have implemented regularized linear regression to predict housing prices. However, when you test your hypothesis on a new set of houses, you find that it makes unacceptably large errors in its prediction.

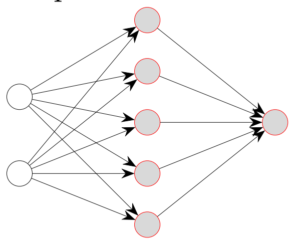
- Get more training examples $\rightarrow$ to fix high variance
- try smaller sets of features $\rightarrow$ to fix high variance
- try getting additional features $\rightarrow$ to fix high bias
- try adding polynomial features $\rightarrow$ to fix high bias
- decrease λ $\rightarrow$ to fix high bias
- increase λ $\rightarrow$ to fix high variance

0.4 Neural Network

small NN with fewer parameters
more prone to underfitting

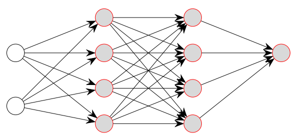


Large Network
more prone to overfitting



more hidden layer units
Use regularization
to adress overfitting

Large Network
more prone to overfitting



more hidden layers
Use regularization
to adress overfitting

Using a larger NN with regularization is often more effective than a small NN. In order to determine what architecture is best (large Nbr of hidden units or larger Nbr of hidden layers), use Train/CV/Test split and evaluate the architecture with CV error ($J_{cv}(\theta)$)

0.5 Error Analysis

When developing an algorithm, it is advised to :

- start with a simple model and check it with cross validation data
- plot learning curves to decide if more data, more features... are likely to help
- Error analysis:
  - manually examine the examples in cross validation set where the algorithm made errors on.
  - classify the errors and prioritize on the error to tackle
    For example, in spams classifier if among the misclassified emails, one finds:
    That suggests that emails 'steal pwds' is the category that degrades the algorithm performance .

pharma	12 examples
replica/fake	4
steal pwds	53
other	31

0.6 Error metrics for skewed classes

- Definition: case of **skewed classes** occur when in the dataset there is much more examples of one class over the class.

For skewed classes, classification accuracy is not a good metric for the performance of the algorithm. It is better to use **Precision/ Recall** parameters:

Predicted class	Actual class	
	1	0
1	True Positive	False Positive
0	False Negative	True Negative

$$P = \frac{\text{\# True Positive}}{\text{\# Predicted Positive}} = \frac{\text{\# True Positive}}{\text{True Positive } + \text{ False Positive}}$$
$$R = \frac{\text{\# True Positive}}{\text{\# actual Positive}} = \frac{\text{\# True Positive}}{\text{True Positive } + \text{ False Negative}} \tag{9}$$

We typically target High recall and High Precision for a performing algorithm.

- In the case of skewed classes, the convention is to label the rare class as $y = 1$
- **Trade off between Precision/Recalls**.
  - Increase the precision (P) by increasing the threshold:
    - Predict 1 if $h_\theta(x) > \cancel{0.5}\quad 0.7$
    - Predict 1 if $h_\theta(x) < \cancel{0.5}\quad 0.7$

    Decrease threshold $\Longrightarrow$ Higher Precision / Lower Recall.
  - Increase the Recall (R) by decreasing the threshold (avoid 'False negative')
    - Predict 1 if $h_\theta(x) \geq \cancel{0.5}\quad 0.3$
    - Predict 1 if $h_\theta(x) < \cancel{0.5}\quad 0.3$

    Decrease threshold $\Longrightarrow$ Higher Recall / Lower Precision.

  More generally, we want to predict 1 if $h_\theta(x) < \textit{threshold}$, depending if we target higher recall or higher precision.

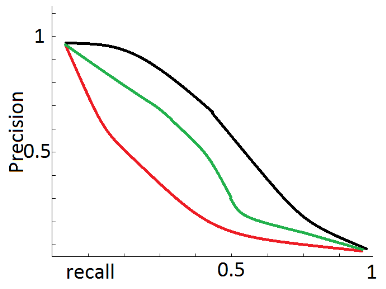


Figure 1: The precision/recall curve can have different shapes

- Optimum Precision/Recall is determined with F_1 score (target = Higher F_1):

$$F_1 = 2\frac{PR}{P+R} \tag{10}$$

There are other expressions for F_1 score.