Coursera Beta

- ShenShen

- Contact Us
- My Contributions
- Log Out

ML:Advice for Applying Machine Learning

From Coursera

Contents

Deciding What to Try Next

Errors in your predictions can be troubleshooted by:

- Getting more training examples // 获取更多的训练数据
- Trying smaller sets of features // 减小特征值
- Trying additional features // 增加特征值
- Trying polynomial features // 对特征值使用多项式
- Increasing or decreasing λ // 增加或减少 λ

Don't just pick one of these avenues at random. We'll explore diagnostic techniques for choosing one of the above solutions in the following sections.

Evaluating a Hypothesis

A hypothesis may have low error for the training examples but still be inaccurate (because of overfitting).

With a given dataset of training examples, we can split up the data into two sets: a training set and a test set.

The new procedure using these two sets is then:

1. Learn Θ and minimize $J(\Theta)$

2. Compute the test set error: $J_{test}(\Theta) = \dfrac{1}{2m_{test}} \sum_{i=1}^{m_{test}} \left(h_\Theta(x_{test}^{(i)}) - y_{test}^{(i)} \right)^2$

3. Misclassification error (aka 0/1 misclassification error):

$$err(h_\Theta(x), y) = \begin{array}{l} 1 \quad \text{if } h_\Theta(x) \geq 0.5 \text{ and } y = 0 \text{ or } h_\Theta(x) < 0.5 \text{ and } y = 1 \\ 0 \qquad\qquad\qquad\qquad\qquad otherwise \end{array}$$

This gives us a binary 0 or 1 error result based on a misclassification.

The average test error for the test set is

$$\text{Test Error} = \dfrac{1}{m_{test}} \sum_{i=1}^{m_{test}} err(h_\Theta(x_{test}^{(i)}), y_{test}^{(i)})$$

This gives us the proportion of the test data that was misclassified.

Model Selection and Train/Validation/Test Sets

- Just because a learning algorithm fits a training set well, that does not mean it is a good hypothesis.
- The error of your hypothesis as measured on the data set with which you trained the parameters will be lower than any other data set.

In order to choose the model of your hypothesis, you can test each degree of polynomial and look at the error result.

Without the Validation Set

1. Optimize the parameters in Θ using the training set for each polynomial degree.
2. Find the polynomial degree d with the least error using the test set.
3. Estimate the generalization error also using the test set with $J_{test}(\Theta^{(d)})$, (d = theta from polynomial with lower error);

In this case, we have trained one variable, d, or the degree of the polynomial, using the test set. This will cause our error value to be greater for any other set of data.

To solve this, we can introduce a third set, the Cross Validation Set, to serve as an intermediate set that we can train d with. Then our test set will give us an accurate, non-optimistic error.

One example way to break down our dataset into the three sets is:

- Training set: 60%
- Cross validation set: 20%
- Test set: 20%

We can now calculate three separate error values for the three different sets.

With the Validation Set

1. Optimize the parameters in Θ using the training set for each polynomial degree.
2. Find the polynomial degree d with the least error using the cross validation set.
3. Estimate the generalization error using the test set with $J_{test}(\Theta^{(d)})$, (d = theta from polynomial with lower error);

This way, the degree of the polynomial d has not been trained using the test set.

Diagnosing Bias vs. Variance

In this section we examine the relationship between the degree of the polynomial d and the underfitting or overfitting of our hypothesis.

- We need to distinguish whether bias or variance is the problem contributing to bad predictions.
- High bias is underfitting and high variance is overfitting. We need to find a golden mean between these two.

The training error will tend to decrease as we increase the degree d of the polynomial.

At the same time, the cross validation error will tend to decrease as we increase d up to a point, and then it will increase as d is increased, forming a convex curve.

High bias (underfitting): both $J_{train}(\Theta)$ and $J_{CV}(\Theta)$ will be high. Also, $J_{CV}(\Theta) \approx J_{train}(\Theta)$.

High variance (overfitting): $J_{train}(\Theta)$ will be low and $J_{CV}(\Theta)$ will be much greater than $J_{train}(\Theta)$.

The is represented in the figure below:
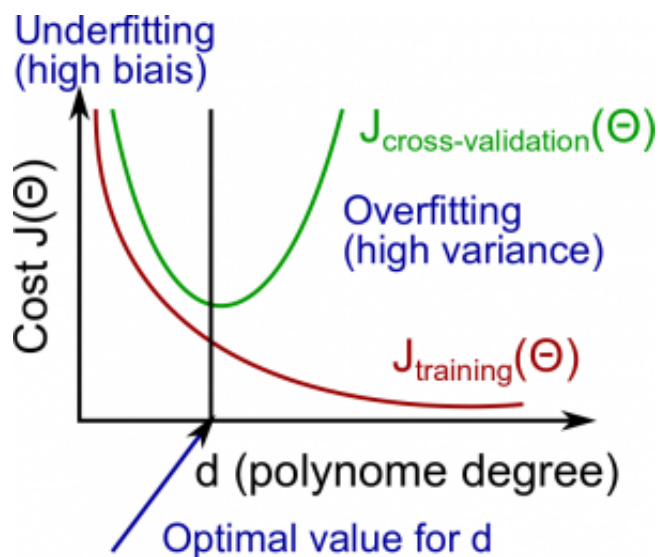
Regularization and Bias/Variance

Instead of looking at the degree d contributing to bias/variance, now we will look at the regularization parameter λ.

    Large λ: High bias (underfitting)

    Intermediate λ: just right

    Small λ: High variance (overfitting)

A large lambda heavily penalizes all the Θ parameters, which greatly simplifies the line of our resulting function, so causes underfitting.
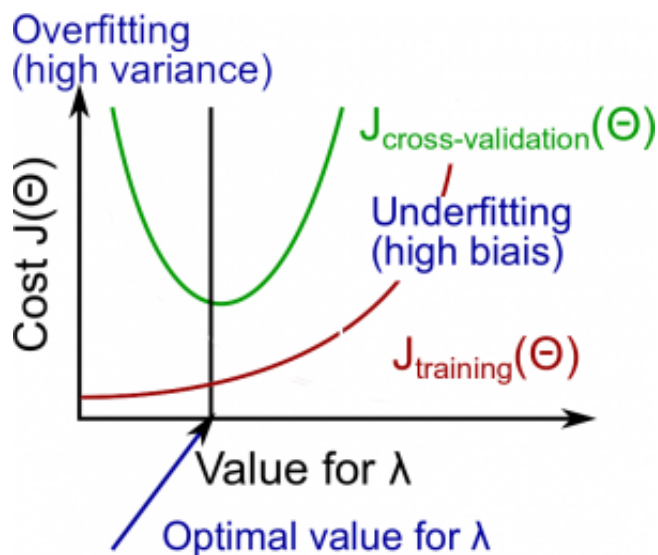
The relationship of λ to the training set and the variance set is as follows:

    Low λ: $J_{train}(\Theta)$ is low and $J_{CV}(\Theta)$ is high (high variance/overfitting).

    Intermediate λ: $J_{train}(\Theta)$ and $J_{CV}(\Theta)$ are somewhat low and $J_{train}(\Theta) \approx J_{CV}(\Theta)$.

    Large λ: both $J_{train}(\Theta)$ and $J_{CV}(\Theta)$ will be high (underfitting/high bias)

The figure below illustrates the relationship between lambda and the hypothesis:

In order to choose the model and the regularization λ, we need:

1. Create a list of lambda (i.e.
   $\lambda \in \{0, 0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.64, 1.28, 2.56, 5.12, 10.24\}$;
2. Select a lambda to compute;
3. Create a model set like degree of the polynomial or others;
4. Select a model to learn Θ;
5. Learn the parameter Θ for the model selected, using $J_{train}(\Theta)$ with λ selected (this will learn Θ for the next step);
6. Compute the train error using the learned Θ (computed with λ) on the $J_{train}(\Theta)$ without regularization or λ = 0;
7. Compute the cross validation error using the learned Θ (computed with λ) on the $J_{CV}(\Theta)$ without regularization or λ = 0;
8. Do this for the entire model set and lambdas, then select the best combo that produces the lowest error on the cross validation set;
9. Now if you need visualize to help you understand your decision, you can plot to the figure like above with: (λ x Cost $J_{train}(\Theta)$) and (λ x Cost $J_{CV}(\Theta)$);
10. Now using the best combo Θ and λ, apply it on $J_{test}(\Theta)$ to see if it have a good generalization of the problem.
11. To help decide the best polynomial degree and λ to use, we can diagnose with the learning curves, that is the next subject.

Learning Curves

Training 3 examples will easily have 0 errors because we can always find a quadratic curve that exactly touches 3 points.

- As the training set gets larger, the error for a quadratic function increases.
- The error value will plateau out after a certain m, or training set size.

With high bias

Low training set size: causes $J_{train}(\Theta)$ to be low and $J_{CV}(\Theta)$ to be high.

Large training set size: causes both $J_{train}(\Theta)$ and $J_{CV}(\Theta)$ to be high with $J_{train}(\Theta) \approx J_{CV}(\Theta)$.

If a learning algorithm is suffering from high bias, getting more training data will not (by itself) help much.

For high variance, we have the following relationships in terms of the training set size:

With high variance

Low training set size: $J_{train}(\Theta)$ will be low and $J_{CV}(\Theta)$ will be high.

Large training set size: $J_{train}(\Theta)$ increases with training set size and $J_{CV}(\Theta)$ continues to decrease without leveling off. Also, $J_{train}(\Theta) < J_{CV}(\Theta)$ but the difference between them remains significant.

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

Deciding What to Do Next Revisited

Our decision process can be broken down as follows:

- Getting more training examples
  - Fixes high variance
- Trying smaller sets of features
  - Fixes high variance
- Adding features
  - Fixes high bias
- Adding polynomial features
  - Fixes high bias
- Decreasing λ
  - Fixes high bias
- Increasing λ
  - Fixes high variance

Diagnosing Neural Networks

- A neural network with fewer parameters is prone to underfitting. It is also computationally cheaper.
- A large neural network with more parameters is prone to overfitting. It is

also computationally expensive. In this case you can use regularization (increase λ) to address the overfitting.

Using a single hidden layer is a good starting default. You can train your neural network on a number of hidden layers using your cross validation set.

Model Selection:

Hi Everyone, I have just compiled a Machine Learning Diagnostics Notes to share it with all of you.

- Choosing M the order of polynomials.

- How can we tell which parameters Θ to leave in the model (known as "model selection")?

  There are several ways to solve this problem:

  1. Get more data (very difficult).
  2. Choose the model which best fits the data without overfitting (very difficult).
  3. Reduce the opportunity for overfitting through regularization.

Bias: approximation error (Difference between expected value and optimal value)

- High Bias = UnderFitting (BU)
- $J_{train}(\Theta)$ and $J_{CV}(\Theta)$ both will be high and $J_{train}(\Theta) \approx J_{CV}(\Theta)$

Variance: estimation error due to finite data

- High Variance = OverFitting (VO)
- $J_{train}(\Theta)$ is low and $J_{CV}(\Theta) \gg J_{train}(\Theta)$

Intuition for the bias-variance trade-off:

- Complex model => sensitive to data => much affected by changes in X => high variance, low bias.
- Simple model => more rigid => does not change as much with changes in X => low variance, high bias.

One of the most important goals in learning: finding a model that is just right in the bias-variance trade-off.

Regularization Effects:

- Small values of λ allow model to become finely tuned to noise leading to large variance => overfitting.
- Large values of λ pull weight parameters to zero leading to large bias => underfitting.

Model Complexity Effects:

- Lower-order polynomials (low model complexity) have high bias and low variance. In this case, the model fits poorly consistently.
- Higher-order polynomials (high model complexity) fit the training data extremely well and the test data extremely poorly. These have low bias on the training data, but very high variance.
- In reality, we would want to choose a model somewhere in between, that can generalize well but also fits the data reasonably well.

A typical rule of thumb when running diagnostics is:

- More training examples fixes high variance but not high bias.
- Fewer features fixes high variance but not high bias.
- Additional features fixes high bias but not high variance.
- The addition of polynomial and interaction features fixes high bias but not high variance.
- When using gradient descent, decreasing lambda can fix high bias and increasing lambda can fix high variance (lambda is the regularization parameter).
- When using neural networks, small neural networks are more prone to under-fitting and big neural networks are prone to over-fitting. Cross-validation of network size is a way to choose alternatives.

I hope this helps.

References:

- https://class.coursera.org/ml/lecture/index
- http://www.cedar.buffalo.edu/~srihari/CSE555/Chap9.Part2.pdf
- http://blog.stephenpurpura.com/post/13052575854/managing-bias-variance-tradeoff-in-machine-learning
- http://www.cedar.buffalo.edu/~srihari/CSE574/Chap3/Bias-Variance.pdf

---

Next: Machine Learning System Design Back to Index: Main

Retrieved from "https://share.coursera.org/wiki/index.php?title=ML:Advice_for_Applying_Machine_Learning&oldid=32428"
Category: ML:Lecture Notes

- This page was last modified on 21 August 2015, at 01:11.
- This page has been accessed 17,832 times.
- Privacy policy
- About Coursera
- Disclaimers