CSE/ECE-471: Statistical Methods in AI **Spring 2020**

Bias-Variance Tradeoff and Model Selection

Prepared by: Alok(20171103), Jeevesh(2019201058), Sandeep(2019900076)

1 Bias Variance Trade-off

Whenever we discuss model prediction, it's important to understand prediction errors (bias and variance). There is a tradeoff between a model's ability to minimize bias and variance. Gaining a proper understanding of these errors would help us not only to build accurate models but also to avoid the mistake of overfitting and underfitting.

1.1 Bias

What is Bias?

Bias is the difference between the average prediction of our model and the correct value which we are trying to predict. Model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to high error on training and test data.

1.2 Variance

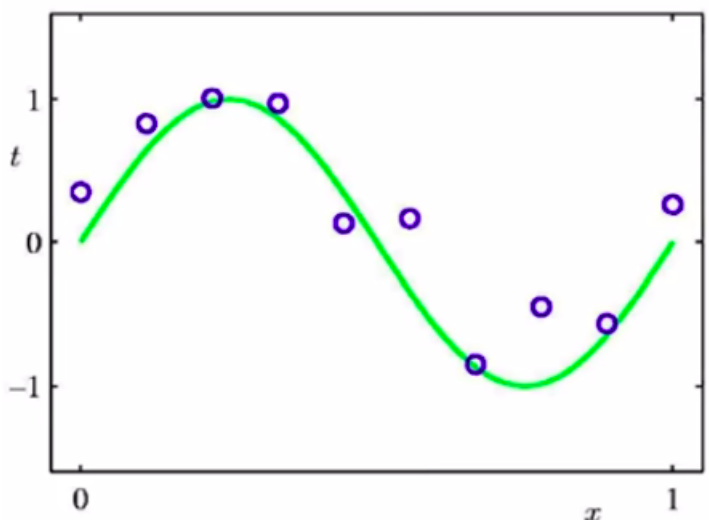
What is Variance?

Variance is the variability of model prediction for a given data point or a value which tells us spread of our data. Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before. As a result, such models perform very well on training data but has high error rates on test data.

1.3 Example

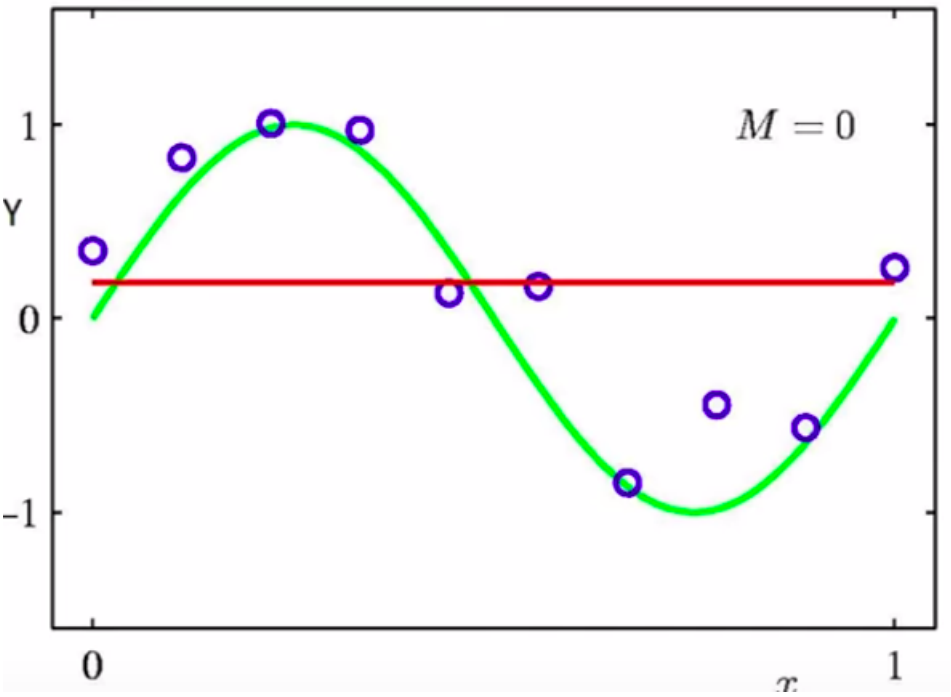
1.3.1 Step 1

We have data points and our objective is to fit the curve so that the error between ground truth and predicted values should be minimum



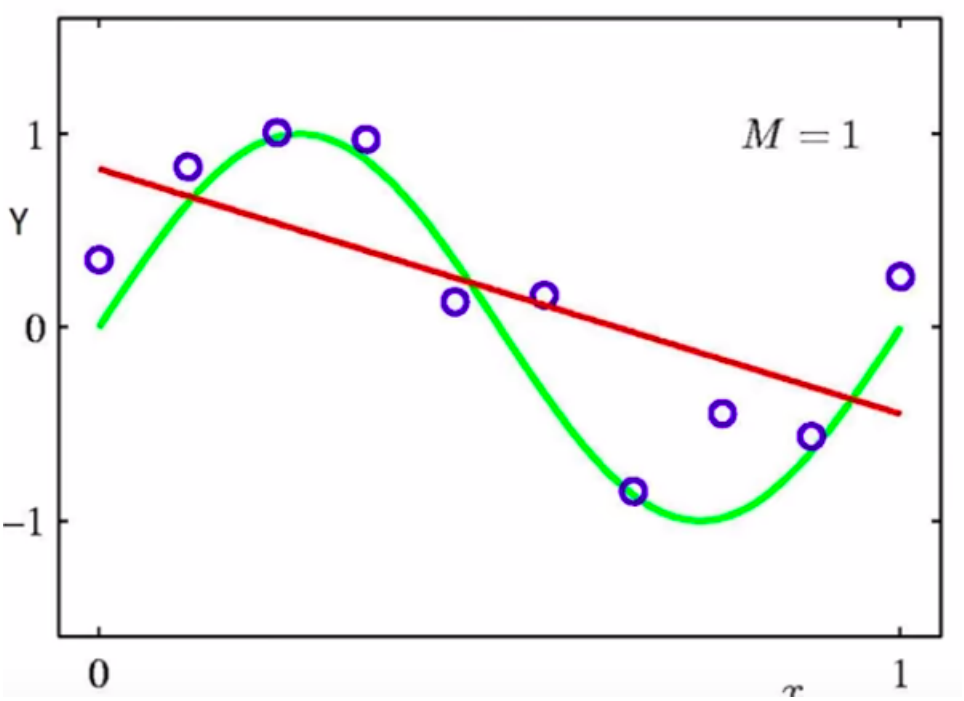
1.3.2 Step 2

We we put the curve corresponding to degree 0, we see that we get the straight line it under-fits the data, By under-fitting we means we are expecting the data points corresponds to the line. Here we get more training Error.



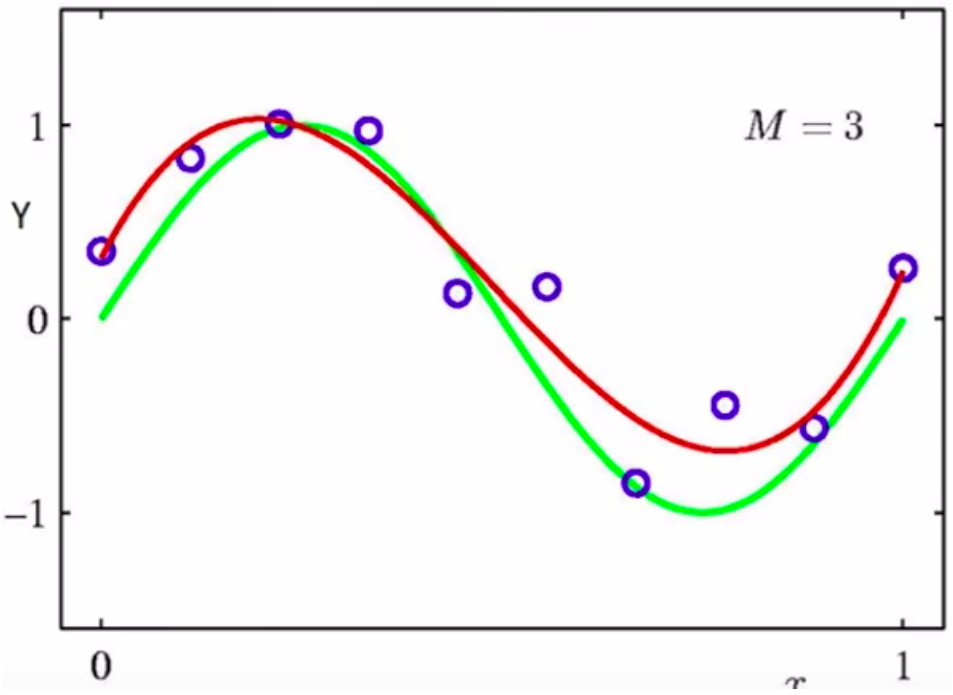
1.3.3 Step 3

Now, We have seen for polynomial of degree 0 we are getting more training error,now me insert to curve higher degree polynomial (let say $p = 1$)



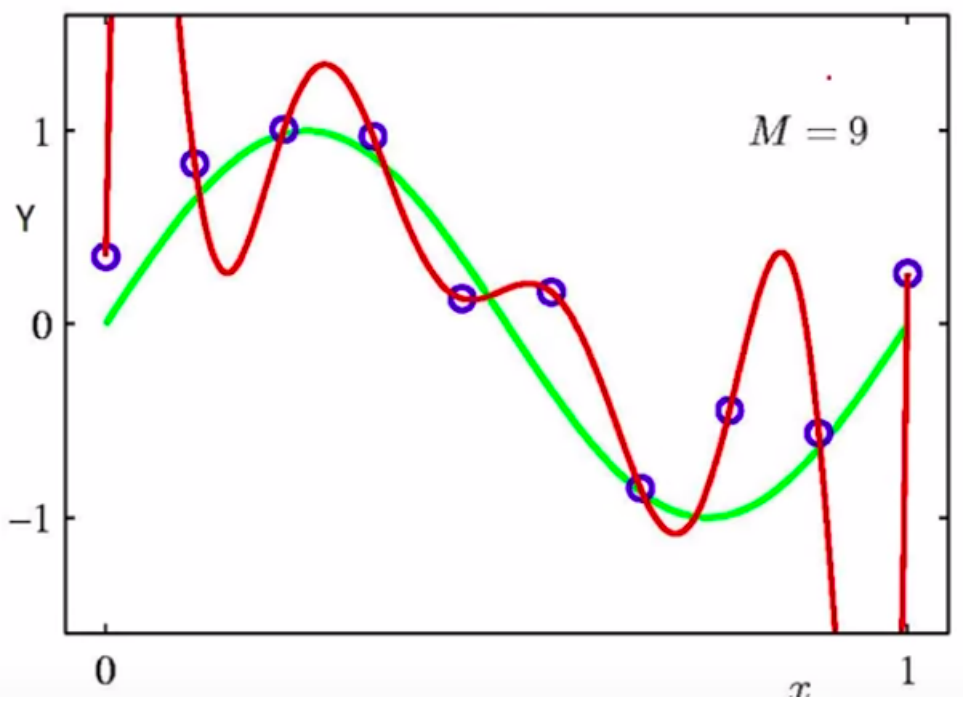
1.3.4 Step 4

We have seen that our curve give less error corresponds to predicted value, we try to increase the order of polynomial till we find optimal degree of polynomial let($p = 3$)



1.3.5 Step 5

Try for degree of polynomial $p = 9$.



For degree 9, we realize that our curve get more specific to the training points and there might be the case we get high testing error, we term this scenario as overfitting.

1.4 Error

Equation of curve of degree p

$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + \cdots + w_p x^p$$

We have $n + m$ points

Training Data(for n points)

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$$

Testing Data (m points)

$$(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}), \ldots, (x_{n+m}, y_{n+m})$$

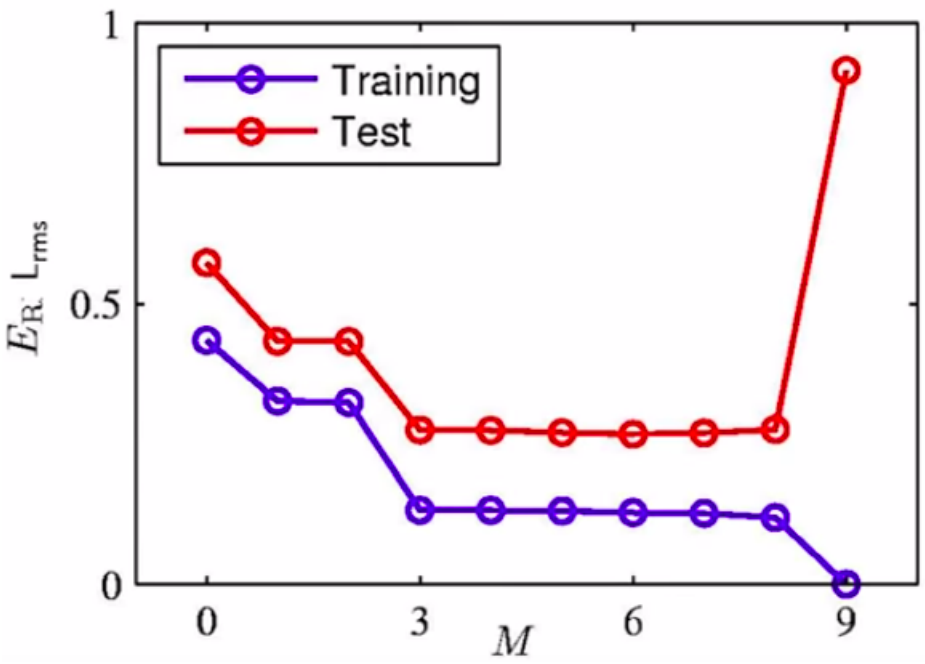
Training Error
Our objective is to minimize the training error

$$TrainingError = \frac{1}{2}\sum_{i=1}^{n}(w_0 + w_1 x_i + w_2 x_i^2 + w_3 x_i^3 + ... + w_p x_i^p - y_i)^2$$

Test Error
Once we get minimum training error, we need to check testing error with corresponds weights which we get in training

$$TestError = \frac{1}{2}\sum_{i=n+1}^{n+m}(w_0 + w_1 x_i + w_2 x_i^2 + w_3 x_i^3 + ... + w_p x_i^p - y_i)^2$$

1.4.1 Error in Training and Testing data



1.4.2 Training Error

Initially, once we start training, we get high error and we increase the degree of polynomial and model fits the data point, if we set the curve corresponding to a higher order, then we get minimum training error.

1.4.3 Testing Error

For very low p, the model is very simple, and so can't capture the full complexities of the data. It **"underfits"** the data. This is called **bias**.
For very high p, the model is complex, and so tends to **"overfit"** to spurious properties of the data. This is called **variance**.

1.5 BootStrapping

The bootstrap is a widely applicable and extremely powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator or statistical learning method.

Importantly, samples are constructed by drawing observations from a large data sample one at a time and returning them to the data sample after they have been chosen. This allows a given observation to be included in a given small sample more than once. This approach to sampling is called **sampling with replacement**.

The process for building one sample can be summarized as follows:

1. Choose the size of the sample.
2. While the size of the sample is less than the chosen size
   (a) Randomly select an observation from the dataset
   (b) Add it to the sample

1.6 Formalizing Bias and Variance

Given the Data Set D

$$D = (x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$$

$D \in \{D_1, D_2, ...D_k\}$
where D_i is the Data set we get from Bootstrapping.

And model built from data set,

$$f(x; D)$$

We can evaluate the effectiveness of the model using MSE:

$$MSE = E_{p(x,y,D)}[(y - f(x; D))^2]$$

with constant $|D| = N$.

Formulation of MSE

$$MSE_x = E_{D|x}[(y - f(x; D))^2]$$

it can be split into three components

$$Error(x) = Bias^2 + variance + Noise$$

Bias : Difference between average model prediction (across data sets) and the target.

$$Bias = (E_D[f(x; D)] - E[y|x])$$

Variance : Model Prediction of the same x across different datasets.

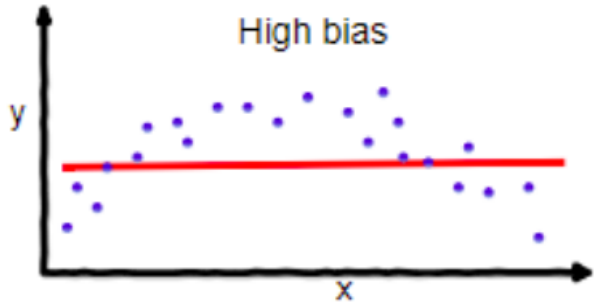
$$Variance = E_D[(f(x; D) - E_D[f(x; D)])^2]$$

Noise : Inherent Noise in our Datasets.

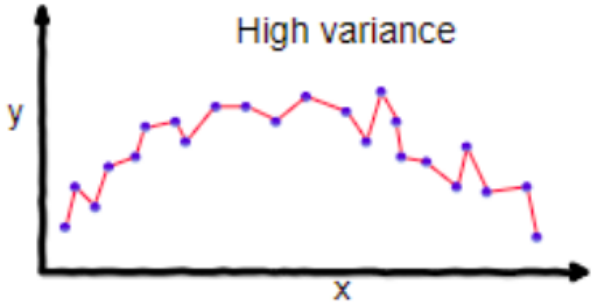
$$Noise = E[(y - E[y|x])^2]$$

$$MSE_x = (E_D[f(x;D)] - E[y|x])^2 + E_D[(f(x;D) - E_D[f(x;D)])^2] + E[(y - E[y|x])^2]$$

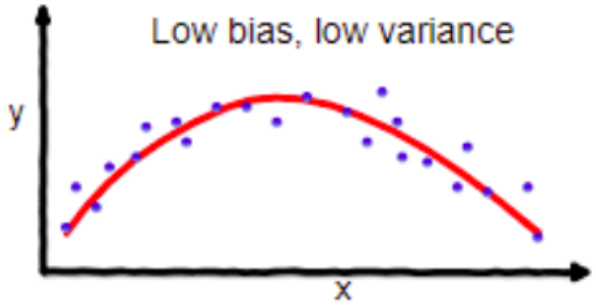
1.7 Bias/Variance is a way to understand Underfitting and Overfitting



underfitting



overfitting



Good balance

2 Cross-Validation and Model Selection

2.1 Motivation

1. Validation techniques are motivated by two fundamental problems in pattern recognition: model selection and performance estimation.
2. We can't afford to get new test data each time.
3. We must never train on test data.
4. We also want to use as much training material as possible (because ML systems trained on more data are almost always better).
5. We can achieve this by using every little bit of training data for testing – under the right kind of conditions.
6. By cleverly iterating the test and training split around.

2.2 Split the dataset into two groups

1. Training set: used to train the classifier.
2. Test set: used to estimate the error rate of the trained classifier.

2.3 The above method has two basic drawbacks

1. In problems where we have a sparse dataset, we may not be able to afford the "luxury" of setting aside a portion of the dataset for testing.
2. Since it is a single train-and-test experiment, the holdout estimate of error rate will be misleading if we happen to get an "unfortunate" split.

2.4 Three-way Data Split

1. If model selection and true error estimates are to be computed simultaneously, the data needs to be divided into three disjoint sets.

2. **Training set**: A set of examples used for learning: to fit the parameters of the classifier. In the MLP case, we would use the training set to find the "optimal" weights with the back-prop rule.

3. **Validation set**: A set of examples used to tune the parameters of a classifier. In the MLP case, we would use the validation set to find the "optimal" number of hidden units or determine a stopping point for the back-propagation algorithm.

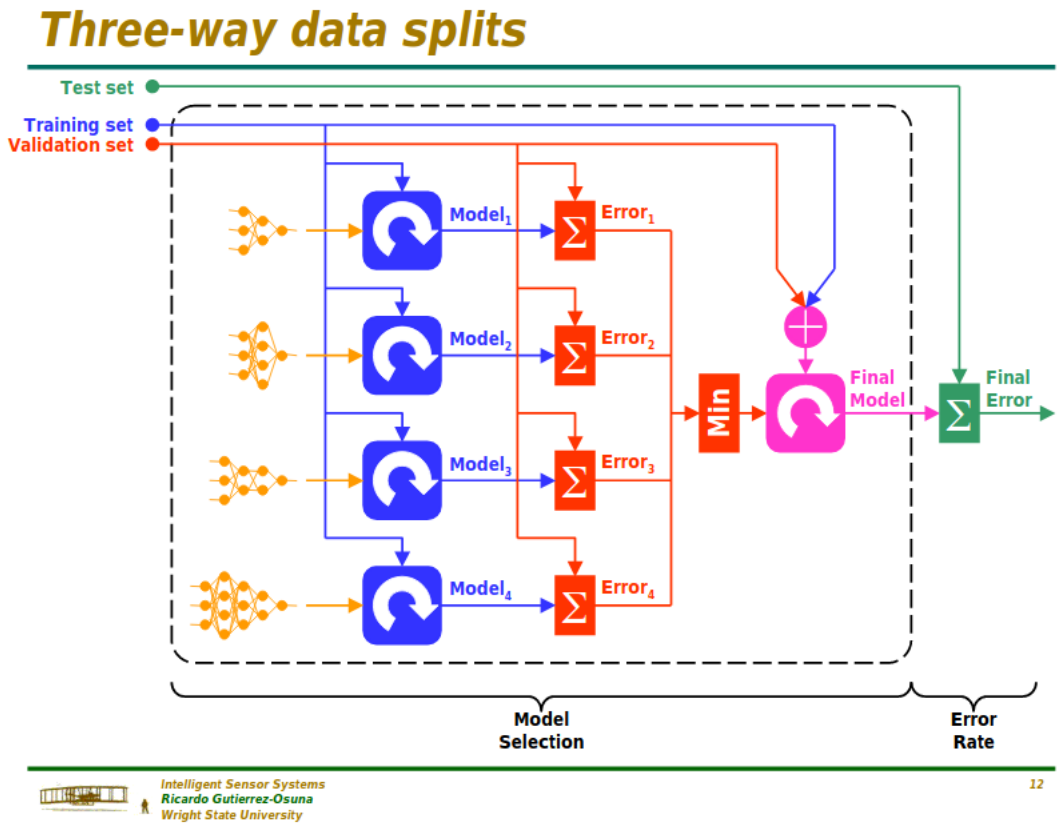
4. **Test set**: A set of examples used only to assess the performance of a fully-trained classifier. In the MLP case, we would use the test to estimate the error rate after we have chosen the final model (MLP size and actual weights). After assessing the final model with the test set,

5. YOU MUST NOT further tune the model.

2.5 Why separate test and validation sets?

1. The error rate estimate of the final model on validation data will be biased (smaller than the true error rate) since the validation set is used to select the final model.

2. After assessing the final model with the test set, YOU MUST NOT tune the model any further



2.6 Leave-One-Out Cross-Validation (LOOCV)

LOOCV does not create two subsets of comparable size. Instead, it does the following:

1. A single observation (x_1, y_1) is used for the validation set.

2. The remaining observations $\{(x_2, y_2), \ldots, (x_n, y_n)\}$ make up the training set.

3. The statistical learning method is fit on the $n-1$ training observations, and a prediction is made for the excluded observation, using its value x_1.
4. Since (x_1, y_1) was not used to fit the model, $MSE_1 = (y_1 - \hat{y}_1)^2$ provides an approximately unbiased estimate for the test error. This isn't enough though, because it's only for one single observation.
5. We repeat the procedure by selecting (x_2, y_2) on the validation set.
6. We train the model on the $n-1$ observations that are leftover.
7. We compute the MSE_2 as we did before, now using (x_2, y_2).
8. We now repeat this process approach times to produce n squared errors, $MSE_1, \ldots, MSE_n$.
9. The LOOCV estimate for the test MSE is the average of these test error estimates:

$$CV(n) = \frac{1}{n} \sum_{i=1}^{n} MSE_i$$

Leave-One-Out Cross-Validation (LOOCV)

Figure 3: Toy data set with *n* observations. (The same as the validation approach.)

Leave-One-Out Cross-Validation (LOOCV)

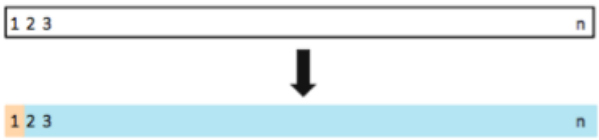


Figure 4: A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige).

Leave-One-Out Cross-Validation (LOOCV)

Figure 5: A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige).

Leave-One-Out Cross-Validation (LOOCV)

Figure 6: A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige).

Leave-One-Out Cross-Validation (LOOCV)

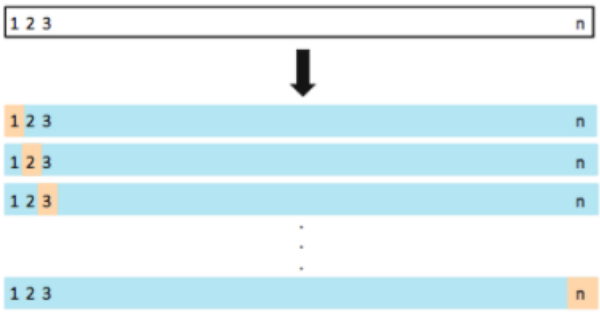


Figure 7: A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige). The test error is then estimated by averaging the n resulting MSE's. The first training set contains all but observation 1, the second training set contains all but observation 2, and so forth.

2.7 LOOCV versus the Validation Method

LOOCV has a couple of major advantages over the validation set approach.

1. It has far less bias
   (a) **LOOCV**: Repeatedly fit the statistical learning method using training sets that contain $n-1$ observations, there are almost as many as are in the entire data set.
   (b) This contrasts with the validation method, where the training set is about half the size of the original data set.
   (c) LOOCV approach tends not to overestimate the test error rate as much as the validation set approach does.
2. Performing LOOCV multiple times produces similar results. This is not typically true for the validation method.

   Remark: LOOCV has the potential to be expensive to implement since the model has to be fit n times. This can be very time consuming if n is large, and if each individual model is slow to fit.

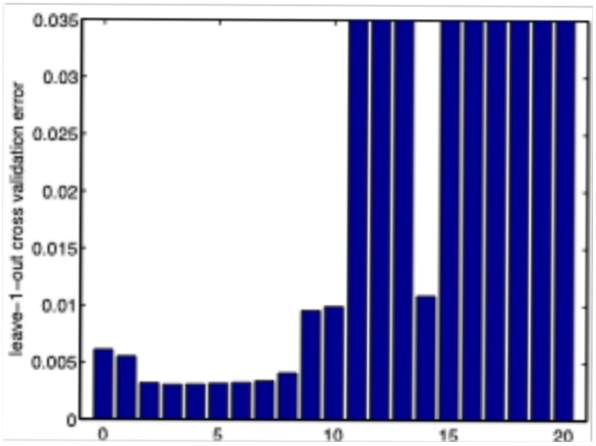
3 K-Fold Cross Validation

We saw, Leave-one-Out Cross Validation, it generalizes really well, but it is very expensive to perform when we have a lot of data, which brings us to K-fold cross validation.

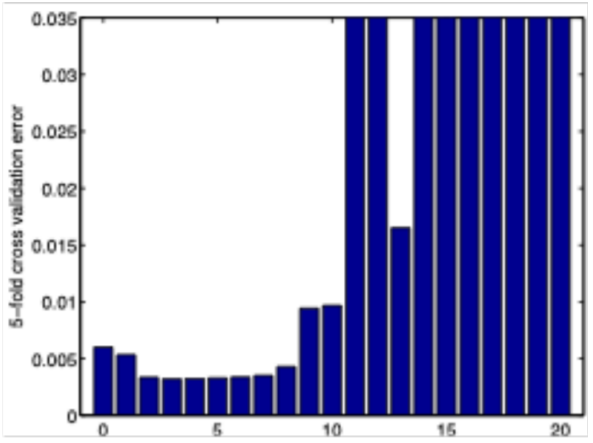
3.1 Process

1. First, we fix the hyper-parameters of our model (p in case of polynomial fitting)
2. Next we break the training dataset D in to K chunks, lets call them $T_1, T_2, \ldots, T_K$.
3. For $i = 1, 2, \ldots, K$:
   (a) Set T_i to be T_{val} and rest $K-1$ chunks to be T_{fit}.
   (b) Train your model on T_{fit}. Then Evaluate how well it performs on T_{val}.
4. Pick the model that has best average evaluation score.
5. Train the model on the whole training dataset D.
6. Evaluate on the Test dataset for the final Performance score.

3.2 Example



(a) Leave-One Out CV Error



(b) 5-Fold CV error

3.3 Comparison

	Downside	Upside
Test-set	May give unreliable estimate of future performance	Cheap
Leave-One Out	Expensive	Doesn't waste data
10-fold	Wastes 10% of the data, 10 times more expensive as test-set	Only wastes 10% of the data, only 10 times more expensive instead of N times
3-fold	Wastes more data than 10-fold	Slightly better than Test-set
N-fold	Identical to Leave-One out	

3.4 Improving Cross-Validation

Repeated cross-validation, generalizes the model, lowers the variance, which in turn improves our performance.

3.5 Final Example/Question

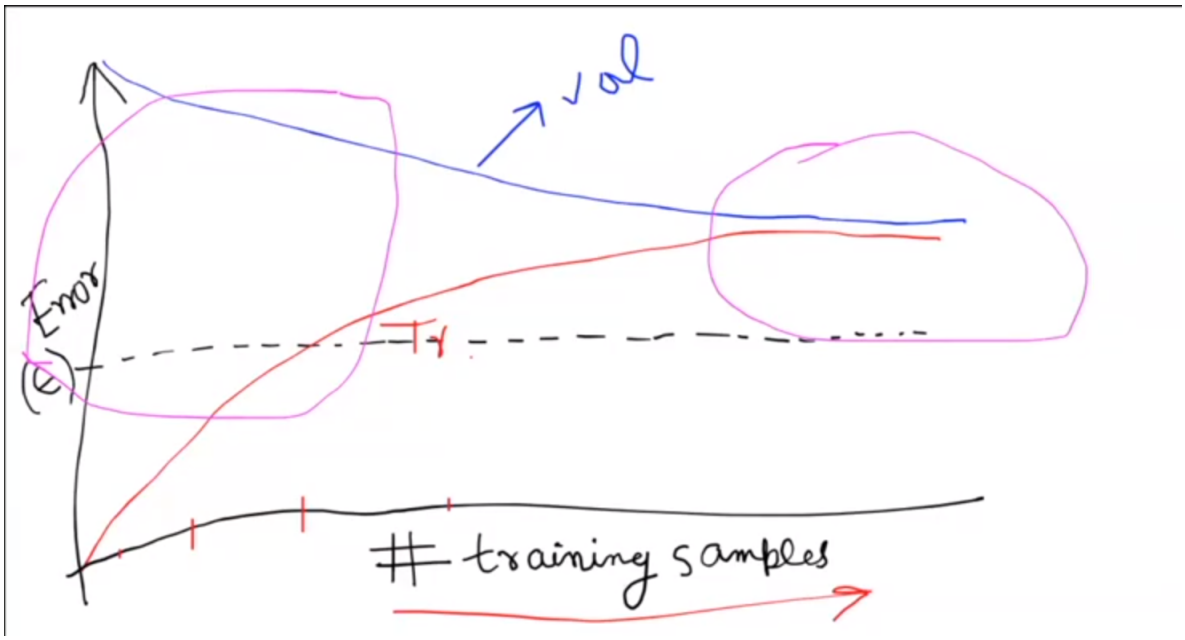


Figure 2: Error vs Number of Samples

3.5.1 Question

Let us consider two sections of the above graph, *Left Side* and *Right Side*(Marked with magenta circles). Classify either sides into:

1. Bias Issue
2. Variance Issue

3.5.2 Solution

The *Right side* of the graph is the Bias Issue. Because we have almost similar training and validation error and both are high which means that our model is not performing well on training data itself, i.e. model is too simple. Hence It is a bias issue.

The *Left Side* of the graph is variance issue, since we have low training error and high validation error, which means our model is overfitting, i.e. our model is too complex. Hence variancce issue.

References

[1] https://www.cs.cmu.edu/ wcohen/10-601/bias-variance.pdf