# Lecture 7: Regression Part 2

*Leap into the 21st century*

**ADDO AI**

2018

**Instructor: Dr. Syed Waqar ul Qounain Jaffry**
**Assistant Professor PUCIT**

# Theory of Generalization

# Agenda

- Prerequisite Self Check
- Generalization
- Recap of Overfitting
- Bias-Variance Decomposition
- Expectation
- Expectation Math
- Polynomial Curve fitting
- Expectation and Bias Variance Decomposition

- Variance, Bias, Noise
- Guess the best model?
- Bias-Variance tradeoff
- K-cross validation
- Data Splitting
- Hyperparameters Tuning
- Coefficient of Determination $R^2$
- Jupyter Notebook

# Generalization

- Refers to model's ability to handle unseen data

- To best describe the unseen data

- In other words, model should not "Overfit"

- Overfitting can be expressed using another concept
  - Bias-variance tradeoff

# Recap

- True function $y = f(x) + \epsilon$

- Our approximate function: $\hat{y} = \hat{w}_0 + \hat{w}_1 . x$

- Error at data-tuple $(x_i, y_i) : e_i = actual - predicted = y_i - \hat{y}_i$

- Goal: $argmin_{w_0, w_1} \sum_{i=1}^{n} e_i^2 = argmin_{w_0, w_1} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

# Recap

- Again, taking example from last lecture

- M refers to the no. of polynomial degree

- *x-axis* represents input/independent variables

- *y-axis* represents dependent/output variable

- The problem presented in the figure is called polynomial curve fitting
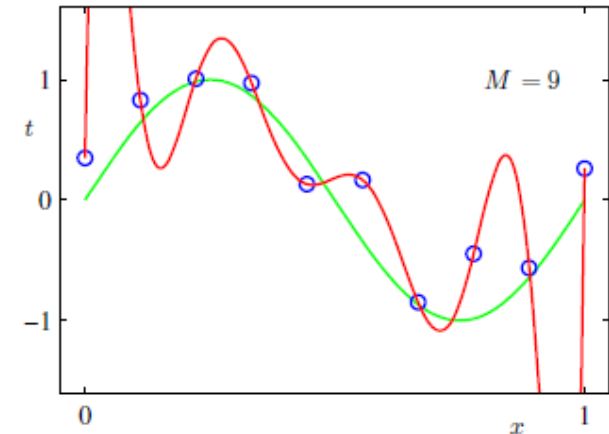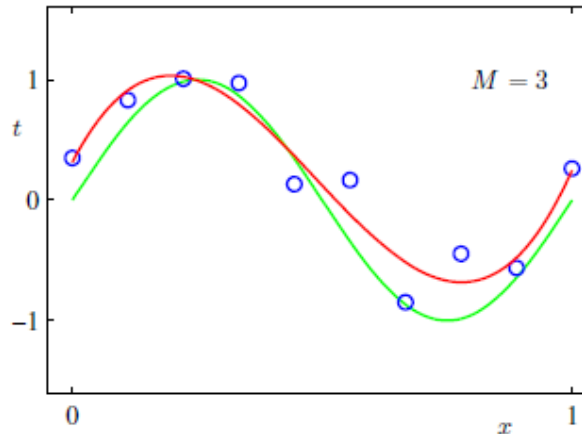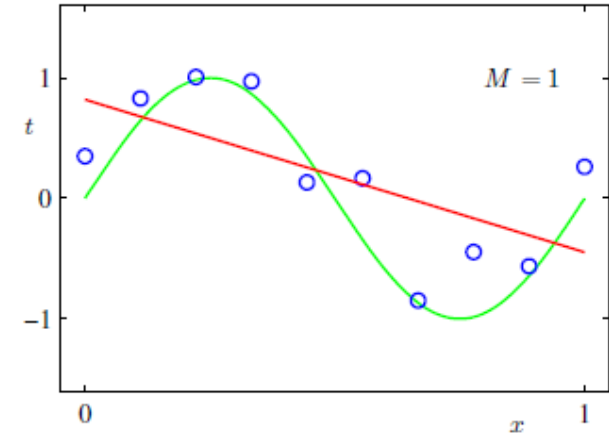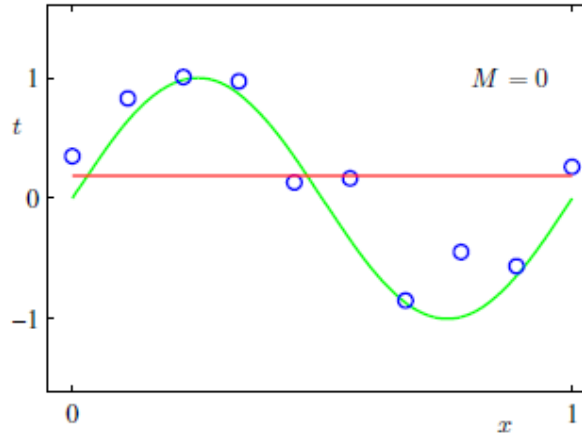
# Polynomial Curve Fitting

- So far, we've discussed linear regression
- If we are to model a non-linear relationship using regression
  - Kernel Trick
- Lets assume that underlying function is of sin i.e. $y = Sin(2\pi x) + \epsilon$
- $\epsilon$ is noise, so assume, noise follows normal distribution with zero-mean and some standard deviation σ.

# Polynomial Curve Fitting

| x | 2πx | Sin(2πx) | ε | y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0.16 | 0.16 |
| 0.1 | 0.63 | 0.59 | 0.11 | 0.7 |
| 0.2 | 1.26 | 0.95 | -0.06 | 0.89 |
| 0.3 | 1.88 | 0.95 | -0.11 | 0.84 |
| 0.4 | 2.51 | 0.59 | -0.08 | 0.51 |
| 0.5 | 3.14 | 0 | -0.14 | -0.14 |
| 0.6 | 3.77 | -0.59 | 0.04 | -0.55 |
| 0.7 | 4.4 | -0.95 | 0.17 | -0.78 |
| 0.8 | 5.03 | -0.95 | -0.04 | -0.99 |
| 0.9 | 5.65 | -0.59 | 0.04 | -0.55 |
| 1 | 6.28 | 0 | 0.12 | 0.12 |
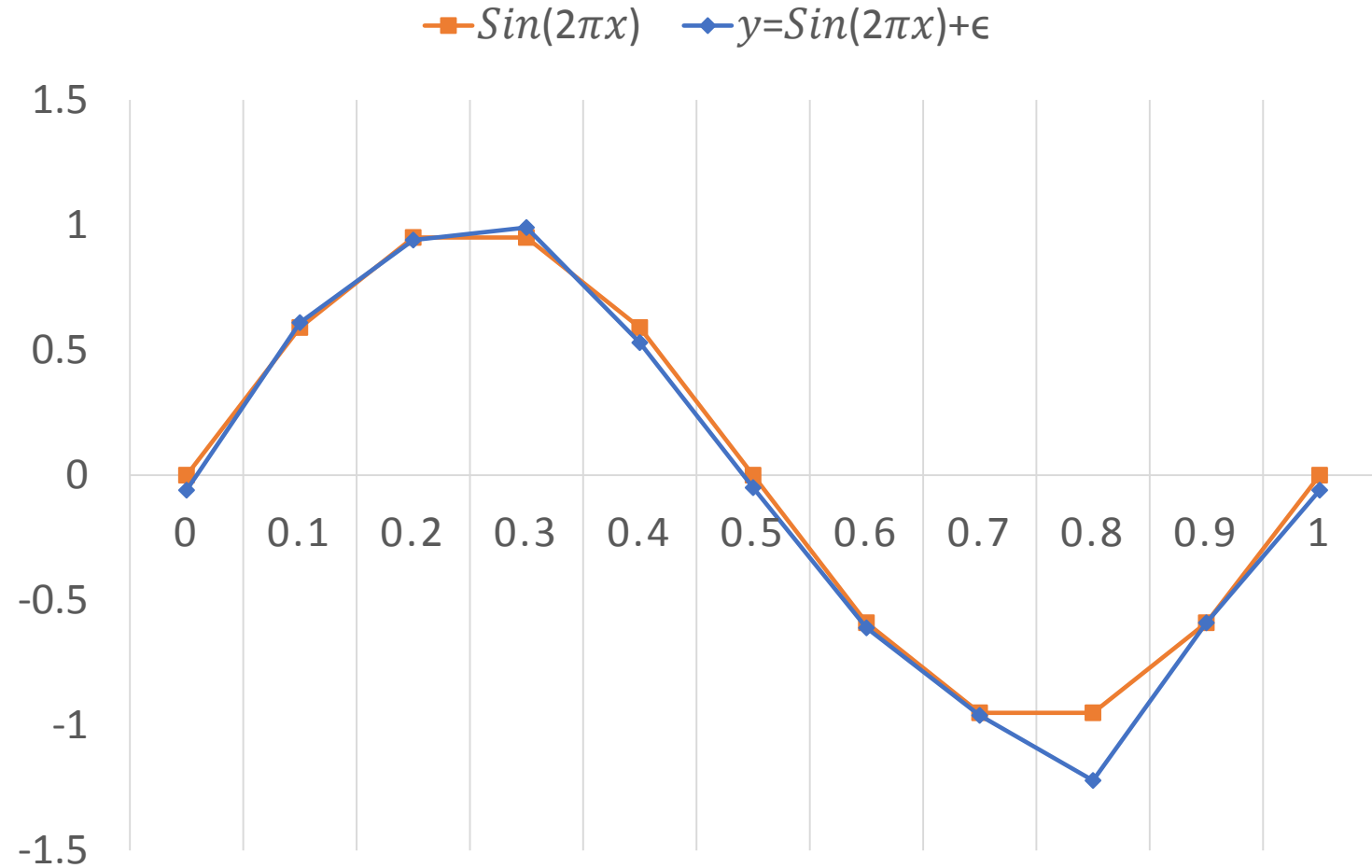
# Polynomial Curve Fitting

- Regression function learnt so far:

  $\hat{y} = \hat{w}_0 + \hat{w}_1 . x$ (Simple linear regression)

  $\hat{y} = \hat{w}_0 + \hat{w}_1 . x + \hat{w}_2 . x_2 + \ldots + \hat{w}_M . x_M$ (Multiple linear regression)

- In case of multiple linear regression, we have m-dimensional input data i.e. input features are M in total

- In current case of polynomial curve fitting:

  - Input is 1-Dimensional
  - Transform into Polynomial
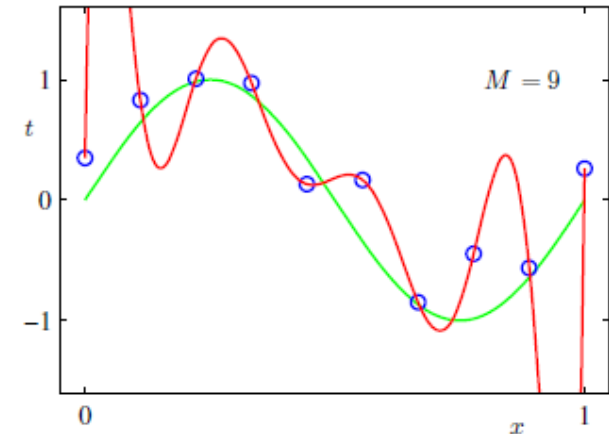
# Polynomial Curve Fitting

- Lets transform our function:
  $$\hat{y} = \hat{w}_0 + \hat{w}_1.x + \hat{w}_2.x^2 + \hat{w}_3.x^3 + \ldots + \hat{w}_M.x^M$$
- Here, we are using only x but has transformed existing model into a non-linear model
- Primary thing to note is that this function is:
  - Polynomial in terms of input variable
  - Linear in terms of parameters
- This function now has capability to model non-linear relationships using regression
- Varying value of M can result in various models

# Polynomial Curve Fitting

- This figure contains four models with varied M values

- As error used is still SSR, hence
  - $argmin_{\overrightarrow{w}} \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

- Recall that previously:
  - $\frac{\partial E}{\partial \hat{w}_i} = -2 \sum_{n=1}^{N}(y_n - \hat{y}_n)(x_n)$

- It now becomes:
  - $\frac{\partial E}{\partial \hat{w}_i} = -2 \sum_{n=1}^{N}(y_n - \hat{y}_n)(x_n)^i$
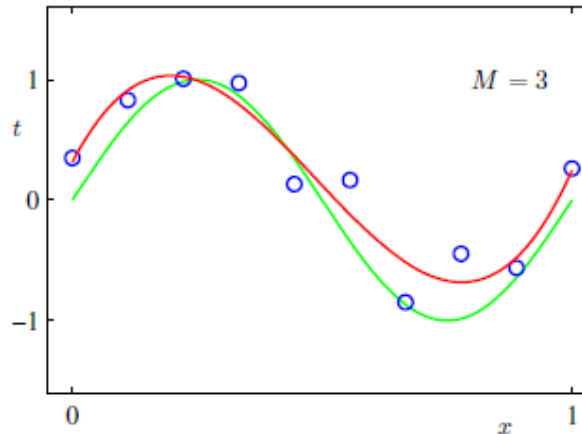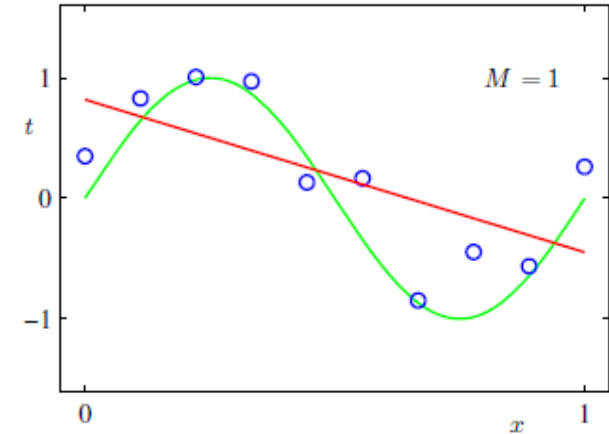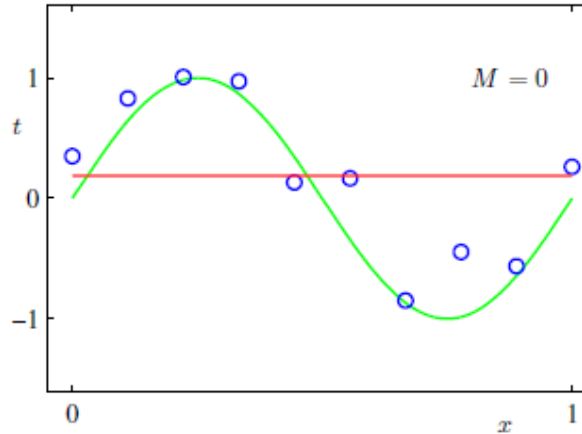
# Polynomial Curve Fitting

- $E = \sum_{n=1}^{N}(y_n - \hat{y}_n)^2$

- $E = \sum_{n=1}^{N}(y_i - (\hat{w}_0 + \hat{w}_1.x_n + \hat{w}_2.x_n^2 + \hat{w}_3.x_n^3 + ... + \hat{w}_M.x_n^M))^2$

$x_n^0 = 1$ for $\hat{w}_0$, Hence, expression can be summarized as:

$E = \sum_{n=1}^{N}\left(y_n - \sum_{j=0}^{M}\hat{w}_j x_n^j\right)^2$

$\frac{\partial E}{\partial \hat{w}_0} = \frac{\partial}{\partial \hat{w}_0}\sum_{n=1}^{N}\left(y_n - \sum_{j=0}^{M}\hat{w}_j x_n^j\right)^2$

$\frac{\partial E}{\partial \hat{w}_0} = -2\sum_{n=1}^{N}\left(y_n - \sum_{j=0}^{M}\hat{w}_j x_n^j\right)\frac{\partial E}{\partial \hat{w}_0}\left(\sum_{j=0}^{M}\hat{w}_j x_n^j\right)$

$\frac{\partial E}{\partial \hat{w}_0} = -2\sum_{n=1}^{N}\left(y_n - \sum_{j=0}^{M}\hat{w}_j x_n^j\right) * [\frac{\partial E}{\partial \hat{w}_0}(\hat{w}_0 * x_n^0) + \frac{\partial E}{\partial \hat{w}_0}(\hat{w}_1 * x_n^1)$

$+ \frac{\partial E}{\partial \hat{w}_0}(\hat{w}_2 * x_n^2) + ... + \frac{\partial E}{\partial \hat{w}_0}(\hat{w}_i * x_n^i) + \cdots \frac{\partial E}{\partial \hat{w}_0}(\hat{w}_M * x_n^M)]$

# Polynomial Curve Fitting

$$\frac{\partial E}{\partial \hat{w}_0} = -2 \sum_{n=1}^{N} \left( y_n - \sum_{j=0}^{M} \hat{w}_j x_n^j \right) * \left[ \frac{\partial E}{\partial \hat{w}_0} \left( \hat{w}_0 * x_n^0 \right) + 0 + \cdots + \cdots + 0 \right]$$

$$\frac{\partial E}{\partial \hat{w}_0} = -2 \sum_{n=1}^{N} \left( y_n - \sum_{j=0}^{M} \hat{w}_j x_n^j \right) * x_n^0$$

Hence, for any $\hat{w}_i$

$$\frac{\partial E}{\partial \hat{w}_i} = -2 \sum_{n=1}^{N} \left( y_n - \sum_{j=0}^{M} \hat{w}_j x_n^j \right) * x_n^i$$

Equating gradient to zero yields:

$$\frac{\partial E}{\partial \hat{w}_i} = -2 \sum_{n=1}^{N} \left( y_n - \sum_{j=0}^{M} \hat{w}_j x_n^j \right) * x_n^i = 0$$

# Polynomial Curve Fitting

$$\frac{\partial E}{\partial \hat{w}_i} = \sum_{n=1}^{N}\left(y_n - \sum_{j=0}^{M} \hat{w}_j x_n^j\right) * x_n^i = 0$$

$$\frac{\partial E}{\partial \hat{w}_i} = \sum_{n=1}^{N}\left(y_n x_n^i - \sum_{j=0}^{M} \hat{w}_j x_n^j x_n^i\right) = 0$$

$$\frac{\partial E}{\partial \hat{w}_i} = \sum_{n=1}^{N}\left(y_n x_n^i\right) - \sum_{n=1}^{N}\left(\sum_{j=0}^{M} \hat{w}_j x_n^j x_n^i\right) = 0$$

$$\Rightarrow \sum_{n=1}^{N}(y_n x_n^i) = \sum_{n=1}^{N}\left(\sum_{j=0}^{M} \hat{w}_j x_n^j x_n^i\right)$$

$$\Rightarrow \sum_{n=1}^{N}(x_n^i) y_n = \sum_{j=0}^{M}\left(\sum_{n=1}^{N} \hat{w}_j x_n^{i+j}\right)$$

$$\Rightarrow \sum_{n=1}^{N}(x_n^i) y_n = \sum_{j=0}^{M} \hat{w}_j \left(\sum_{n=1}^{N} x_n^{i+j}\right)$$

# Polynomial Curve Fitting

$$\Rightarrow \sum_{n=1}^{N}(x_n^i)y_n = \sum_{j=0}^{M}\left(\sum_{n=1}^{N}x_n^{i+j}\right)\hat{w}_j$$

$$If\ A_{ij} = \sum_{n=1}^{N}x_n^{i+j}; T_i = \sum_{n=1}^{N}(x_n^i)y_n$$

$$\Rightarrow T_i = \sum_{j=0}^{M}A_{ij}\hat{w}_j$$

- Solution to these equations provide optimal value for $\hat{w}_i$

$$T_i = \sum_{j=0}^{M}A_{ij}\hat{w}_j$$

- This is similar to the system of equation form: B = AX

- Hence solving these systems of simultaneous equations provides optimal parameter values during training.
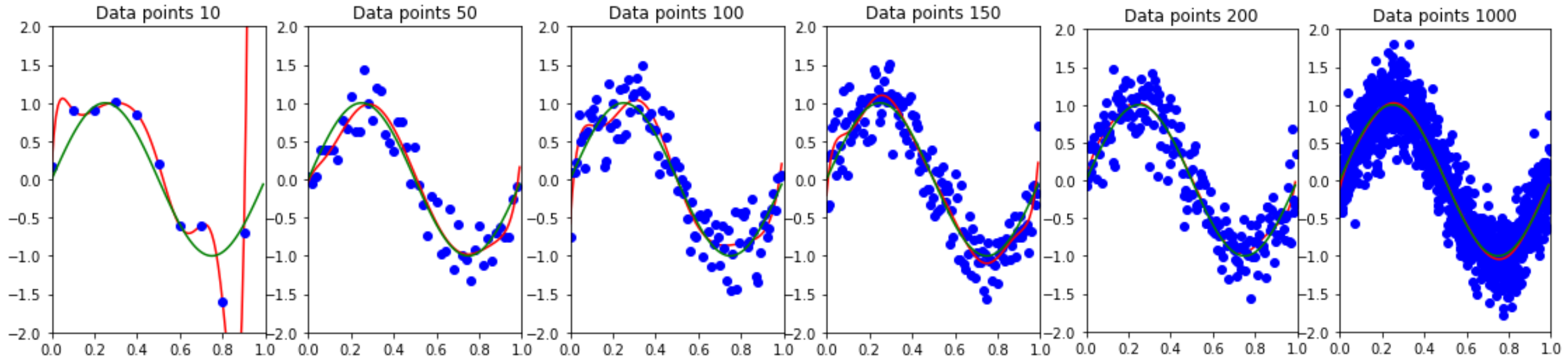
# Polynomial Curve Fitting

$$f(x) = Sin(2\pi x)$$
$$y = f(x) + \epsilon$$
$$\hat{y} = \sum_{j=0}^{M} \hat{w}_j x_n^j$$

- Solution system discussed just now can result in overfitting
- Overfitting can be avoided if training dataset is huge
- These figures show model performance with degree of 9 using variety of datasets.
- With increase in training data points, overfitting is greatly reduced.

# Evaluation of Model Generalization

- There are various ways to evaluate the GOOD FIT of any model

- Some of these include
  - Bias-Variance Tradeoff
  - Error Graphs
  - Coefficient of Determination $R^2$

- We'll discuss each one by one

# Bias-Variance Decomposition

- To understand the concept of overfitting:
  - Break down the error function
  - Error function studied so far can be broken down:
    - Reducible Error
    - Irreducible Error
  - Reducible error can be further broken down:
    - Bias Error
    - Variance Error

# Bias-Variance Decomposition

- Notation
  - For an unseen data point x'
    - y' will represent the actual value
    - $\hat{y}'$ will represent the predicted value
- Goal:
  - To find the expected performance of our model
  - In other words, expected error on unseen instances

# Expectation

- Expectation is also known as Expected Value.
- It can be defined as weighted average of a function
    - $E(x) = \sum_{i=1}^{n} x_i * p(x_i)$ $\qquad p(x_i) \; shows \; probability \; when \; x = x_i$
- If weight is equally distributed among all
    - Expectation would be equal to standard mean or average.
- Hence, average or mean values are also expected values

# Expectation

- For example, when a six-sided unbiased die is rolled, then:
  - As it is unbiased die, probability of each number is equal
  - Hence $p(x_1) = p(x_2) = p(x_3) = p(x_4) = p(x_5) = p(x_6) = \frac{1}{6}$

- Hence, Expectation becomes:

$$E\ (x) = 1 * p(x_1) + 2 * p(x_2) + 3 * p(x_3) + 4 * p(x_4) + 5 * p(x_5) + 6\ * \ p(x_6)$$

$$E\ (x) = 1 * \frac{1}{6} + 2 * \frac{1}{6} + 3 * \frac{1}{6} + 4 * \frac{1}{6} + 5 * \frac{1}{6} + 6\ * \frac{1}{6}$$

$$E\ (x) = \frac{1}{6}\ * \ (\ 1 + 2 + 3 + 4 + 5 + 6) = \frac{1}{6} * (21) = 3.5$$

# Expectation

- Assume an investor who likes to determine his rate of return on three investments. Assume the investments are proportioned according to following probability distributions: 25% in investment A, 25% in investment B, and 50% in investment C. The rate of return is 5% for investment A, 6% for investment B, and 2% for investment C.

- Find the expected rate of return.

# Expectation

- In this example, x = Rate of Return

- Probability of x is determined here by the proportions of investment performed.

- Hence Expected Rate of Return becomes:

  - $E(x) = Rate\ of\ return\ of\ investment\ A\ * prob(A) +$
    $Rate\ of\ return\ of\ investment\ A\ * prob(B) +$
    $Rate\ of\ return\ of\ investment\ A\ * prob(C)$

  - $E(x) = 5\ * 0.25 + 6\ * 0.25 + 2\ * 0.50 = 3.75$

- Hence, Expected Rate of Return is 3.75%

# Expectation Math

Let Z be a random variable with probability distribution P(Z)

Let $\bar{Z} = E[Z]$ be the average value of Z.

Lemma: $\boldsymbol{E\left[(Z - \bar{Z})^2\right] = E[Z]^2 - \bar{Z}^2}$

**Proof**: $E[(Z - \bar{Z})^2] = E[(Z^2 - 2Z\bar{Z} + \bar{Z}^2)]$

$E[(Z - \bar{Z})^2] = E[Z^2] - 2E[Z]\bar{Z} + \bar{Z}^2$ $\qquad$ <span style="color:red">$E[\bar{Z}] = \bar{Z}$</span>

$E[(Z - \bar{Z})^2] = E[Z^2] - 2\bar{Z}\bar{Z} + \bar{Z}^2$

$E[(Z - \bar{Z})^2] = E[Z^2] - 2\bar{Z}^2 + \bar{Z}^2$

$\boldsymbol{E\left[(Z - \bar{Z})^2\right] = E[Z^2] - \bar{Z}^2}$

$\Rightarrow \boldsymbol{E[Z^2] = E\left[(Z - \bar{Z})^2\right] + \bar{Z}^2}$

# Expectation and Bias Variance Decomposition

- Error Function is:
  - $(y - \hat{y})^2$        where $y = f(x) + \epsilon$ and $\hat{y} = \hat{f}(x)$
- Hence, expectation for a new data point x' and corresponding y' would become:
  - $E[(y' - \hat{y}')^2]$     where $y' = f(x') + \epsilon$ and $\hat{y}' = \hat{f}(x')$
- For a given problem, there could be infinite training samples and distributions to which a data could belong
- Hence, assume, that training sample belongs to some Probability Distribution P
- Then, our goal is to find:
  - $E_P[(y' - \hat{y}')^2]$    # For simplicity, $E_P$ would be written as $E$

# Expectation and Bias Variance Decomposition

$$E_P[(y' - \hat{y}')^2] = E[(y' - \hat{y}')^2]$$

$$= E[(y'^2 + \hat{y}'^2 - 2y'\hat{y}')]$$

$$= E[y'^2] + E[\hat{y}'^2] - E[2y'\hat{y}']$$

Using lemma we derived earlier: $\boldsymbol{E[Z^2] = E[(Z - \overline{Z})^2] + \overline{Z}^2}$

$$= E[(y' - \overline{y}')^2] + \overline{y}'^2 + E[(\hat{y}' - \overline{\hat{y}}')^2] + \overline{\hat{y}}'^2 - 2\,E[y']\,E[\hat{y}']$$

As $y = f(x) + \epsilon$, hence input dependent factor contributes towards mean:

$$= E[(y' - f(x'))^2] + f(x')^2 + E[(\hat{y}' - \overline{\hat{y}}')^2] + \overline{\hat{y}}'^2 - 2\,E[y']\,E[\hat{y}']$$

$$= E[(y' - f(x'))^2] + f(x')^2 + E[(\hat{y}' - \overline{\hat{y}}')^2] + \overline{\hat{y}}'^2 - 2\overline{y}'E[\hat{y}']$$

$$= E[(y' - f(x'))^2] + f(x')^2 + E[(\hat{y}' - \overline{\hat{y}}')^2] + \overline{\hat{y}}'^2 - 2f(x')\,\overline{\hat{y}}'$$

# Expectation and Bias Variance Decomposition

After rearranging the terms:

$$= E[(y' - f(x'))^2] + E[(\hat{y}' - \bar{\hat{y}}')^2] + f(x')^2 + \bar{\hat{y}}'^2 - 2f(x')\,\bar{\hat{y}}'$$

$$= E[(y' - f(x'))^2] + E[(\hat{y}' - \bar{\hat{y}}')^2] + (f(x') - \bar{\hat{y}}')^2$$

- $E(y' - \hat{y}')^2 = E[\epsilon^2] + Variance(\hat{y}') + Bias(\bar{\hat{y}}')^2$

- If we assume irreducible noise to follow a normal Gaussian distribution with zero mean and standard deviation of σ:

$$\boldsymbol{E(y' - \hat{y}')^2 = \sigma^2 + Variance(\hat{y}') + Bias(\bar{\hat{y}}')^2}$$

This equation is bias-variance decomposition of expected error.

# Variance $E[(\hat{y} - \overline{\hat{y}})^2]$

- Describes how much $\hat{y}$ varies from one training set S to another

- Describes how much a single model deviates from the average model over multiple datasets

- It is the average squared difference between any single data-set-dependent estimate of $\hat{y}$ and average value of $\hat{y}$ estimated over all datasets

- Variance captures the degree to which model's predictions vary between multiple iterations.

# Bias $\left(f(x) - \overline{\hat{y}}\right)^2$

- Describes the average error of $\hat{y}$ with respect to the actual function

- Assesses how a model approximates actual function

- It is based on behavior over multiple data sets, hence, difference between average model behavior is taken into account

- It describes how much the average estimator fit over datasets

- **bias** represents that how far a model's predictions are from correctness

# Noise $E\left[\left(y - f(x)\right)^2\right]$

- Describes how much y varies from f(x)

- It captures irreducible error

- For the sake of simplicity:
  - it can be assumed zero
  - It can be modeled using Normal distribution
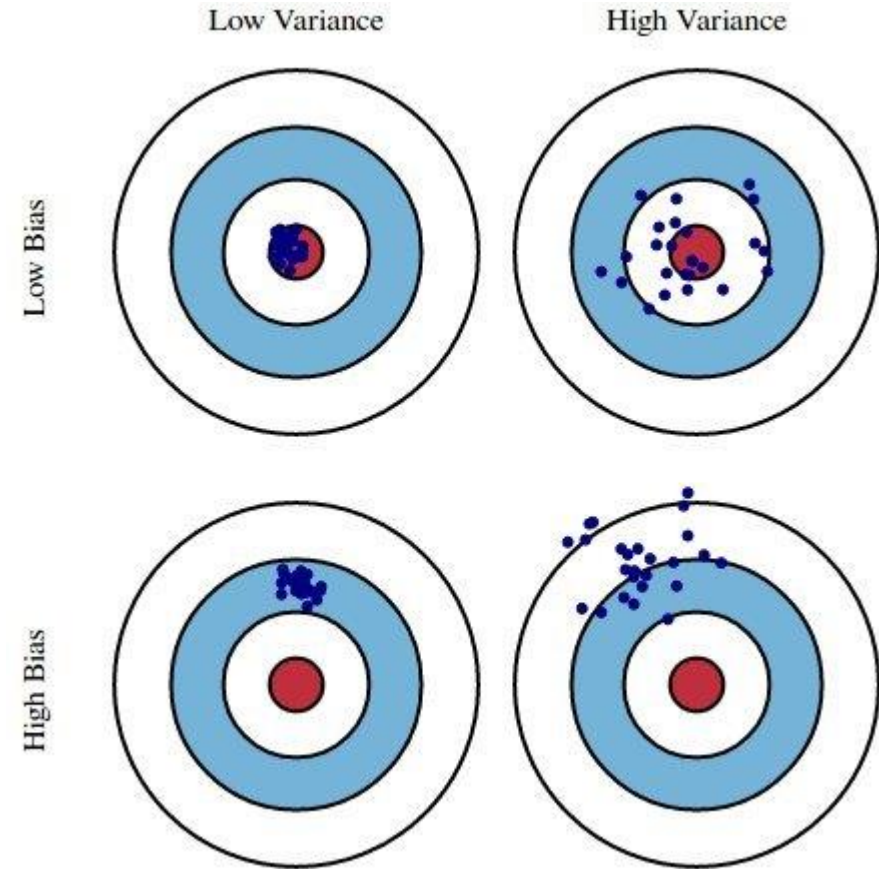
# Guess the best model ?
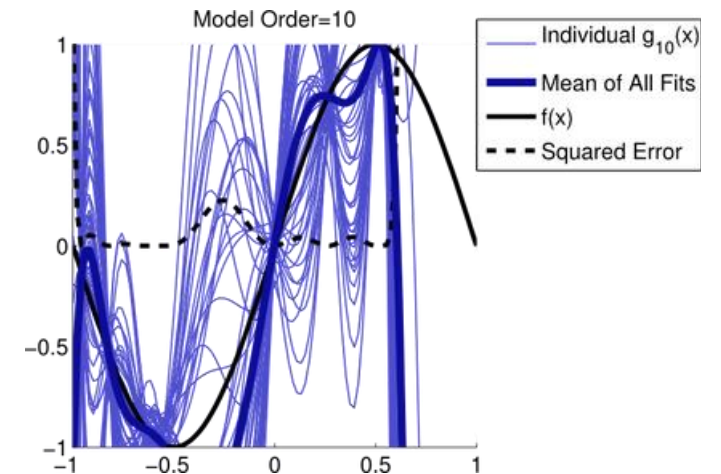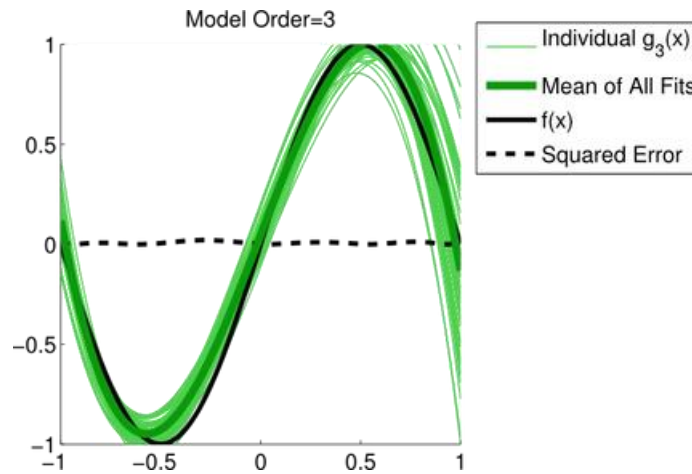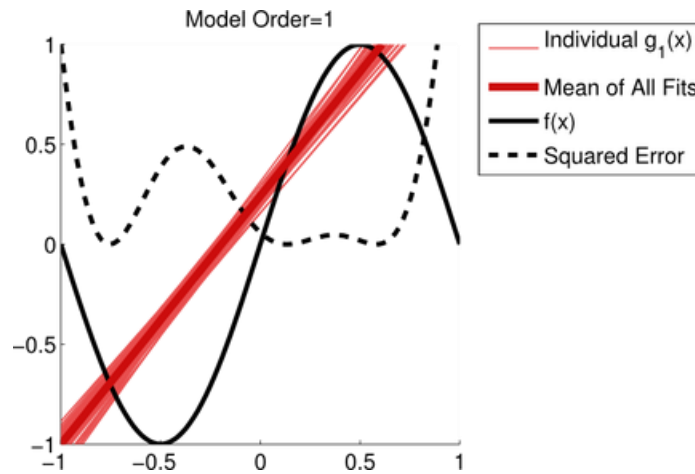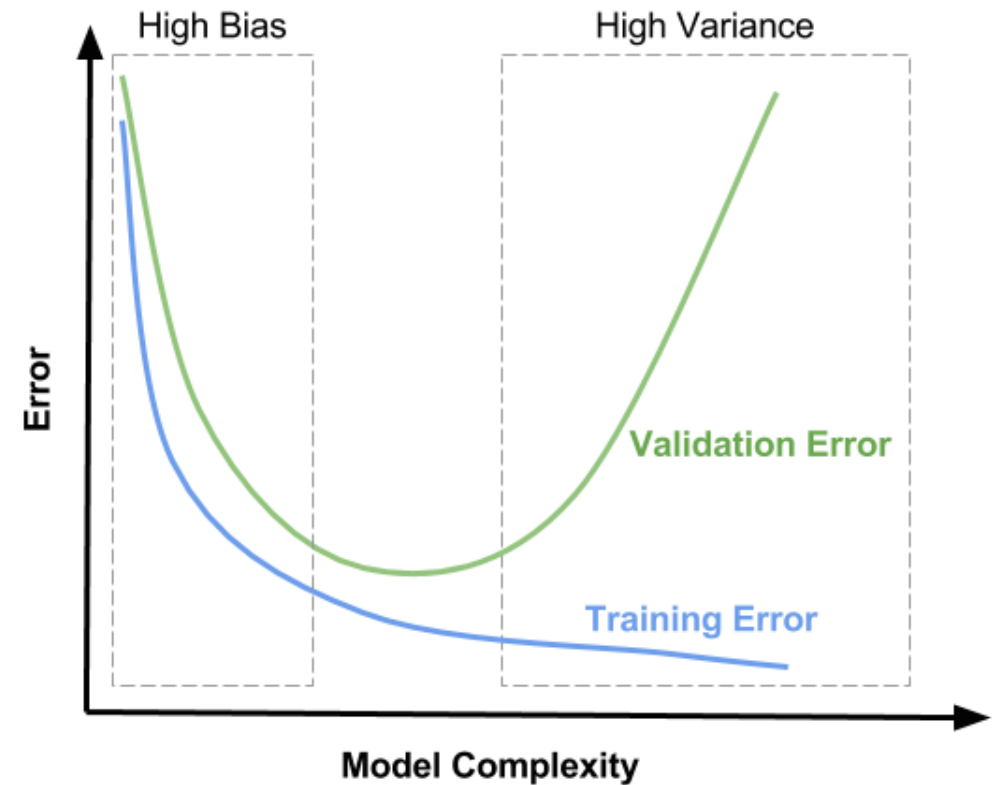


Which model has lowest bias?

Which model has highest Variance ?

# Bias-Variance tradeoff

- If model is too simple
  - High bias
- If model is too complex
  - High Variance
- Goal
  - To have low bias and low variance in model simultaneously
  - Bull's eye represents target function
  - Closer we are to red circle, better the model is

# Bias-Variance tradeoff
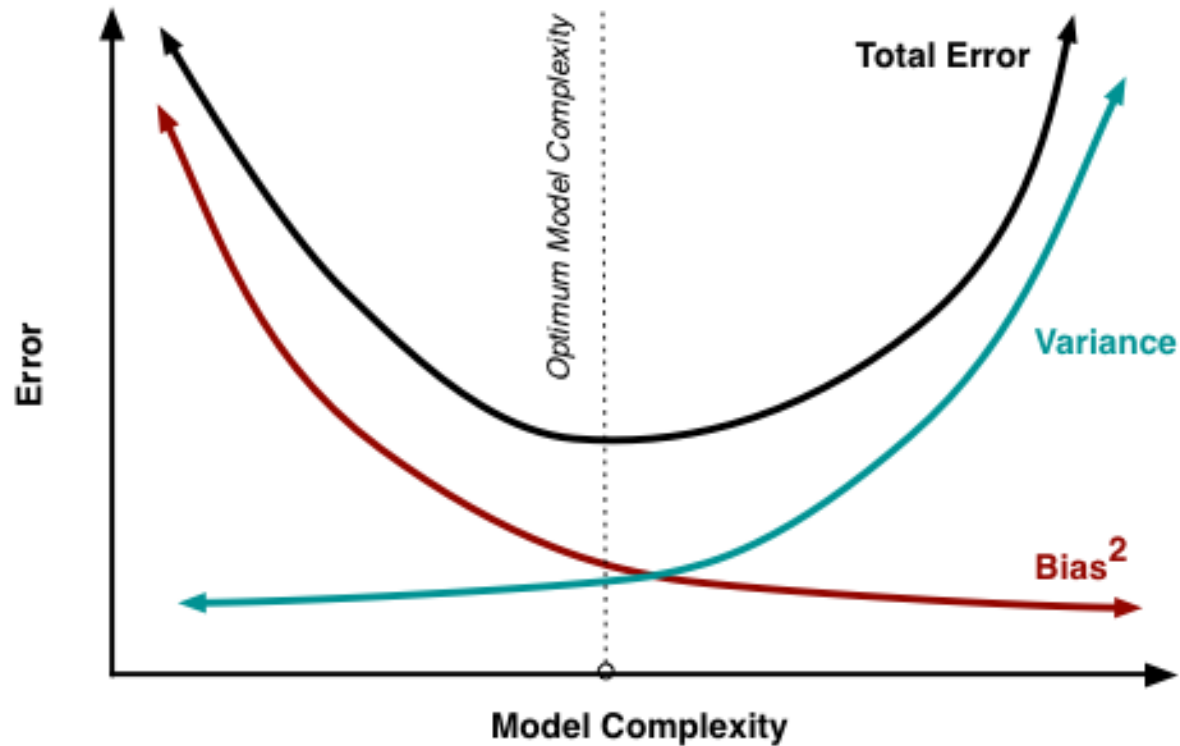
# Bias-Variance tradeoff in practice

- In real life, we don't have infinite amount of training sample populations

- How to compute bias and variance
  - Split the available data
  - Make multiple training and testing datasets
    - For each train-test data prepared: compute $\hat{y}$
  - Using these values, calculate bias and variance

- A common procedure normally used:
  - K-cross validation

# K-cross validation

- A widely used validation technique
- Figure shows 5-cross validation
- Value of K affects the results
  - High Value of K: Low Bias, High Variance
  - Low value of K: High Bias, Low Variance
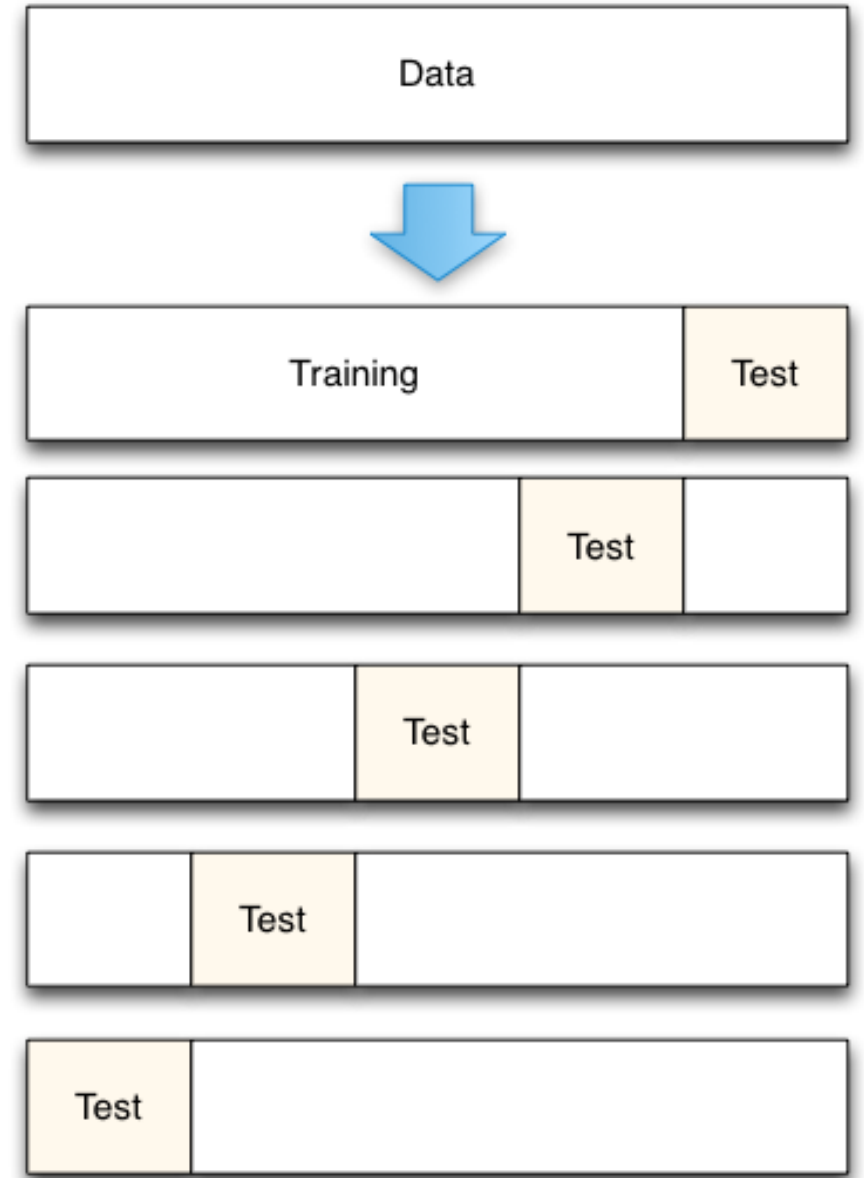
# K-cross validation

- If 5-cross validation is to be performed on this dataset
  - Total data points 10
  - Then total no. of instances per fold will be 2.

- 1$^{st}$ fold contents:
  - Train: [0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9]
  - Test: [0,0.1]

- 2$^{nd}$ fold contents:
  - Train: [0,0.1,0.4,0.5,0.6,0.7,0.8,0.9]
  - Test: [0.2,0.3]

- ....

| x | 2πx | Sin(2πx) | $\epsilon$ | y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0.16 | 0.16 |
| 0.1 | 0.63 | 0.59 | 0.11 | 0.7 |
| 0.2 | 1.26 | 0.95 | -0.06 | 0.89 |
| 0.3 | 1.88 | 0.95 | -0.11 | 0.84 |
| 0.4 | 2.51 | 0.59 | -0.08 | 0.51 |
| 0.5 | 3.14 | 0 | -0.14 | -0.14 |
| 0.6 | 3.77 | -0.59 | 0.04 | -0.55 |
| 0.7 | 4.4 | -0.95 | 0.17 | -0.78 |
| 0.8 | 5.03 | -0.95 | -0.04 | -0.99 |
| 0.9 | 5.65 | -0.59 | 0.04 | -0.55 |

# Data Splitting

- Cross validation is amongst  a widely used method for data split
- Another widely used method of data splitting involves formation of two mutually exclusive datasets
  - One for training
  - Other for testing
- Other techniques
  - Bootstrapping
  - Subsampling
    - With replacement
    - Without replacement (etc.)
- If we are to train our model on training and evaluate on testing, then how to model hyper-parameters?

# Hyperparameters

- Recall the regularized Error-gradient function

$$\frac{\partial E}{\partial \hat{w}_i} = -2 \sum_{i=1}^{n} e_i \ * \ x_i + \lambda w_i$$

- Here $\lambda$ is not a parameter of our predicted function, but it greatly affects the overall error

- Such parameters are known as hyperparameters

- Hyperparameters are set before the training begins
  - Their values are require while model training

# Hyperparameters

- How to model hyperparameters:
  - Instead of having two splits, make three:
    - Training ( to learn the model parameters)
    - Validation (to figure out the value of hyper-parameter)
    - Testing ( to assess the model performance i.e. generalization)
- Usually train-test split of (80%, 20%) is used
  - For Validation: 10-20% records of training set are used

# Hyperparameters

- Firstly:
  - Data will be divided in train and test splits
  - 80% records will be used for training
    - Train set: [0,0.1,0.2,0.3,0.4,0.5,0.6,0.7]
  - Remaining records will be used for testing
    - Test set: [0.8, 0.9]

- Then, training set split will be performed:
  - Train set: [0,0.1,0.2,0.3,0.4,0.5,0.6]
  - Validation set: [0.7]

| x | $2\pi x$ | $Sin(2\pi x)$ | $\epsilon$ | y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0.16 | 0.16 |
| 0.1 | 0.63 | 0.59 | 0.11 | 0.7 |
| 0.2 | 1.26 | 0.95 | -0.06 | 0.89 |
| 0.3 | 1.88 | 0.95 | -0.11 | 0.84 |
| 0.4 | 2.51 | 0.59 | -0.08 | 0.51 |
| 0.5 | 3.14 | 0 | -0.14 | -0.14 |
| 0.6 | 3.77 | -0.59 | 0.04 | -0.55 |
| 0.7 | 4.4 | -0.95 | 0.17 | -0.78 |
| 0.8 | 5.03 | -0.95 | -0.04 | -0.99 |
| 0.9 | 5.65 | -0.59 | 0.04 | -0.55 |

# Hyperparameters

- Consider that the value of $\lambda$ can reside between [0-1] with 0.25 interval
  - Then for $\lambda = 0.0$, model will be trained
    - When training is done; trained model will be validated using validation set
    - Model error will be logged
  - for $\lambda = 0.25$, model will be trained
  - for $\lambda = 0.50$, model will be trained
  - for $\lambda = 0.75$, model will be trained
  - for $\lambda = 1.0$, model will be trained
- As errors are logged against all $\lambda$ values:
  - Value of $\lambda$ that resulted in least error on Validation set will be selected

# Hyperparameters

- Generalization Testing
  - Now, the dataset we reserved earlier for Testing will be used
  - Using best model against validation set
    - Testing error will be calculated
- Widely used Hyperparameters
  - Learning step size
  - Value of regularization Term ($\lambda$)
  - Number of layers in deep networks
  - Number of hidden layer neurons

# Data Splitting and Hyperparameters (Summary)

- For each distinct value of a hyperparameter:
  - Model training is carried out
  - Trained model is evaluated on validation set
- The model's parameter setting as well as respective hyperparameter setting that results in least error are selected
- Using the optimal parameters and hyperparameter values, testing is performed

# Error Graphs

- A widely used approach to see the model performance is to plot various errors

- It plots graph using training, testing and/or validation errors

- Training error is computed using trained model and training records.
  - These training records are already seen by the trained model during training

- Testing error is calculated using trained model and unseen data

- Validation error is calculated using trained model and validation dataset.
  - During model training, this dataset is not used, hence, it is also unseen data

# Error Graphs

- Figure presents a sample plot using various errors
- Best refers to the point, where model's performance is relatively high
- Here Epoch refers to total number of times, complete dataset is exhausted.
- There are multiple ways to stop the training
  - One is to graphically view the error plots
  - Second is to use difference thresholds
  - Third is to hard-code the iterations manually
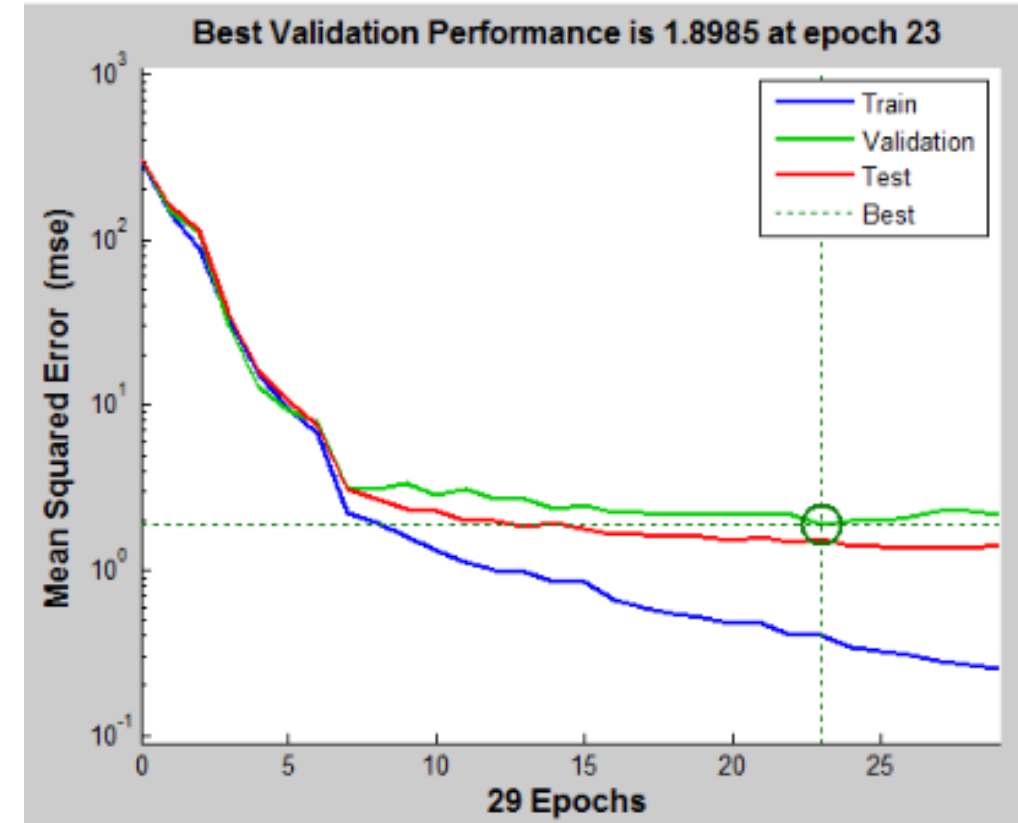
# Coefficient of Determination $R^2$

- In regression, the **$R^2$** coefficient of determination is a statistical measure

- It measure how well the regression predictions approximate the real data points.

- An **$R^2$** of 1 indicates that the regression predictions perfectly fit the data.

- It ranges from [0-1]

- Its formulation is:

$$R^2 = 1 - \frac{SSR}{SSTotal}$$

# Coefficient of Determination $R^2$

- Consider example from last lecture
- For linear regression:
  - $SST = \sum_{i=1}^{n}(y_i - \bar{y})^2 = \sum_{i=1}^{n}(TErr)^2$ where $\bar{y}$ is mean of population
  - $SSR = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(RErr)^2$

| $x$ | $y$ | $\hat{y}$ | $TErr$ | $RErr$ | $TErr^2$ | $RErr^2$ |
|-----|-----|-----------|--------|--------|----------|----------|
| 60 | 3.1 | 3.302 | -0.62 | -0.202 | 0.3844 | 0.040804 |
| 61 | 3.6 | 3.492 | -0.12 | 0.108 | 0.0144 | 0.011664 |
| 62 | 3.8 | 3.682 | 0.08 | 0.118 | 0.0064 | 0.013924 |
| 63 | 4 | 3.872 | 0.28 | 0.128 | 0.0784 | 0.016384 |
| 65 | 4.1 | 4.252 | 0.38 | -0.152 | 0.1444 | 0.023104 |
| | | | | | | |
| Mean(y) = 3.72 | | | SST = 0.628 | | SSR = 0.10588 | |

Hence $R^2$ would become:

$R^2 = 1 - \dfrac{0.10588}{0.628} = 0.831$