University of Pittsburgh

ECE 2195: Special Topics – Computers Machine Learning

Models and Trade-offs

Mai Abdelhakim, PhD

ECE Department

Swanson School of Engineering

University of Pittsburgh

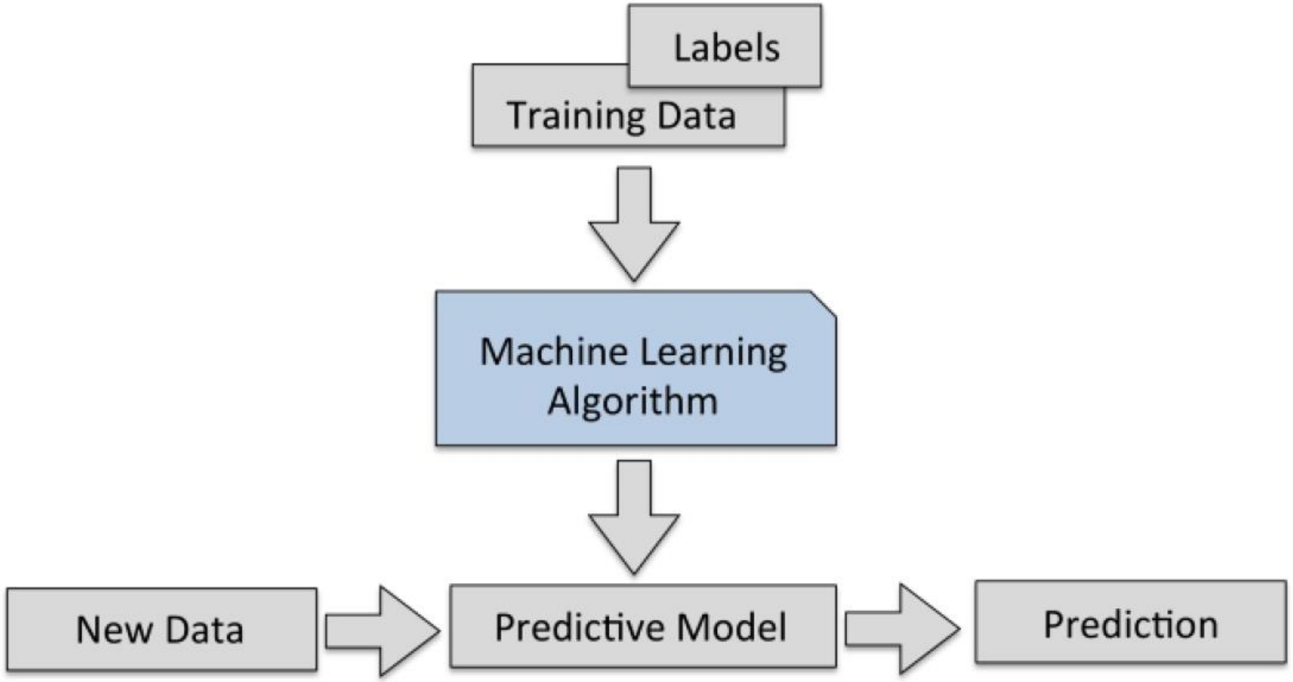
maia@pitt.edu

Recall from Last Meeting

Supervised Learning: predict target values from labeled data

- **Regression**: Target values (Y) are continuous/quantitative
- **Classification**: Target values (Y) are discrete/finite/qualitative

Labels can be obtained manually. Some tools available, e.g. Amazon Mechanical Turk (workers provide labels)



Sabastian Raschka, Python Machine Learning

This Meeting

- Models
- Performance evaluation
  - Training and test
  - Accuracy
- Tradeoffs
  - Flexibility/complexity versus accuracy
  - Bias-variance trade-off
- K-Nearest Neighbor Classifier

Comment: If you need computational resource for project, please contact center for research computing https://crc.pitt.edu/

Supervised Learning
How to make predictions?

- Can we predict the output using the features?

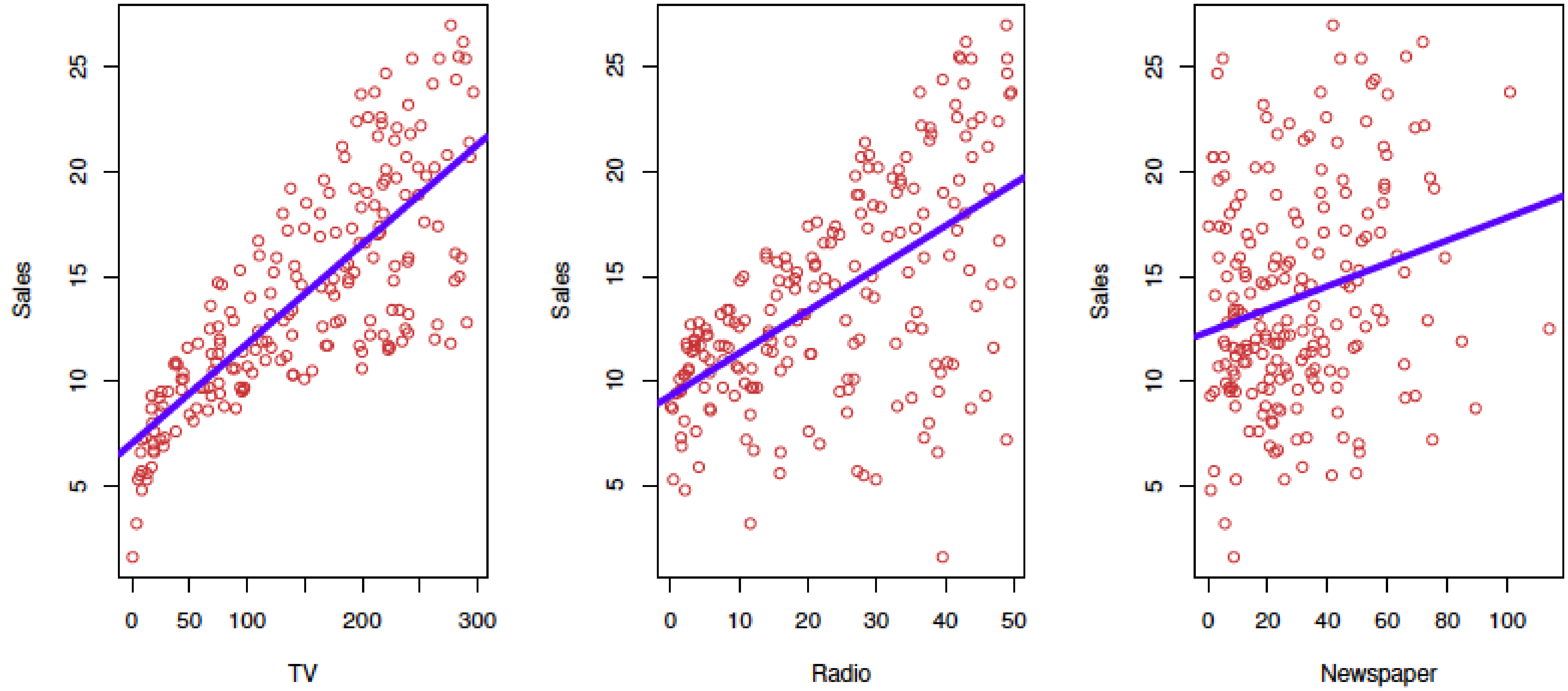
- It is helpful to have a **model**: $Y \approx f(x)$
  - Y label
  - X features

- Example:
  - Assume you are a marketing consultant, and want to suggest marketing plan for next year that results in **high product sales**
    - Given TV, Radio, newspaper **advertising budget** as input features

Advertising Example

- Advertising dataset: sales function of TV, radio, newspaper budgets?

$$\text{sales} \approx f(\text{TV}, \text{radio}, \text{newspaper})$$



Sales in thousands of units, vs TV, Radio, Newspaper in thousands of dollars for 200 markets

Why do we need a model? Why estimate f?

- **Predictions**: Make predictions for new inputs/features
- **Inference**: understand the way Y is affected by each features
  - Which feature has stronger **impact on the response**?
    - Example: Not interested in predicting house price, but want to know the impact of river view on a house value
  - Is relation **positive or negative**
  - Is the relationship **linear or more complicated**

From Advertising Example

- **How accurately can we predict** future sales given advertising budget?

- **Is there a relationship** between advertising budget and sales?
  - If there isn’t, then maybe we don’t need advertising

- **How strong** is the relationship between advertising budget and sales?

- **Which media** contribute to sales?

- **Is there synergy** among the advertising media?
  - Spend $50, 000 on television and $50, 000 on radio advertising results in more sales than allocating $100, 000 to either television or radio individually?

Model

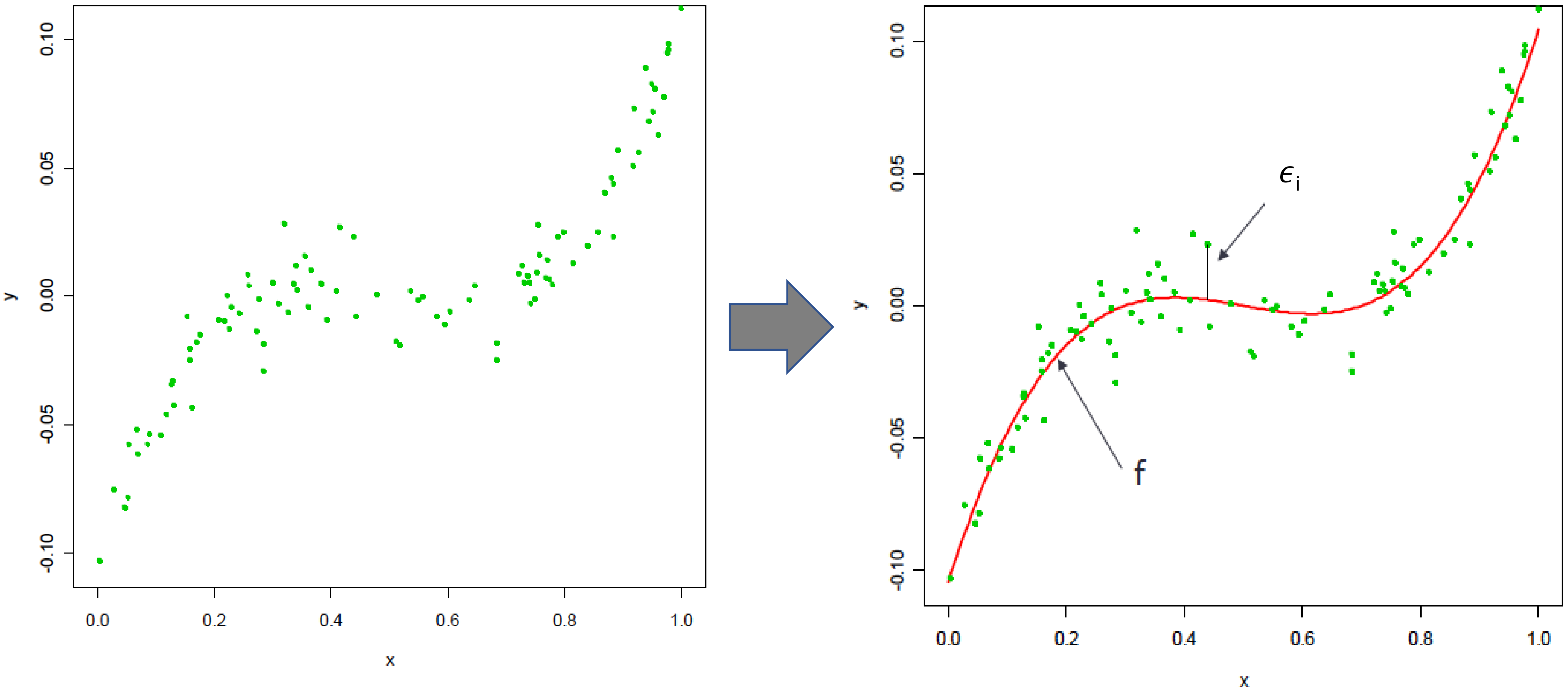
- Y: output
  - Target/response/label that we wish to predict. In previous example, Y= sales
- Features/predictors (X): vector of p features
  - E.g. TV, radio, newspaper budgets
  - Let TV be X_1, Radio be X_2, newspaper be X_3 .. Then $X = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix}$
- We can write

$$Y = f(X) + \epsilon$$

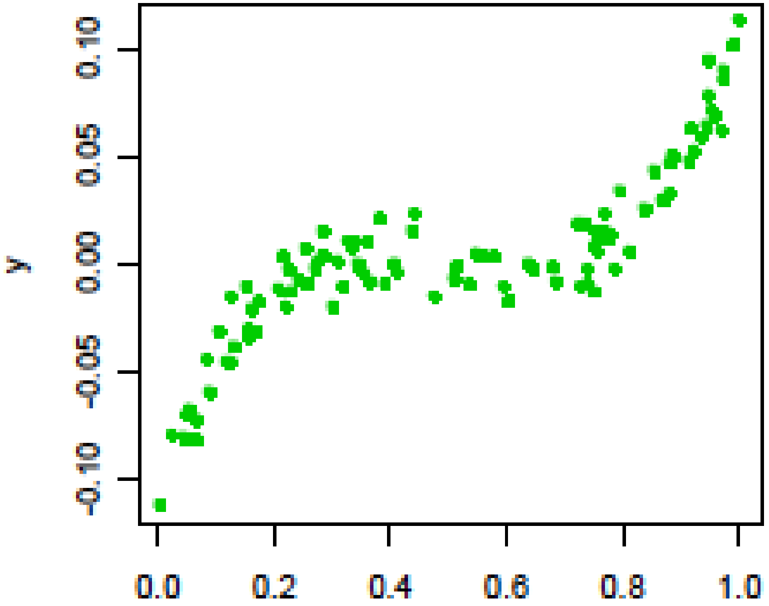
  - ∈ is a random measurements error or due to the distribution of Y at each,
  - f is the unknown function
- **With good estimate of f, we will be able to make predictions.**

Example

- $Y = f(X) + \epsilon$

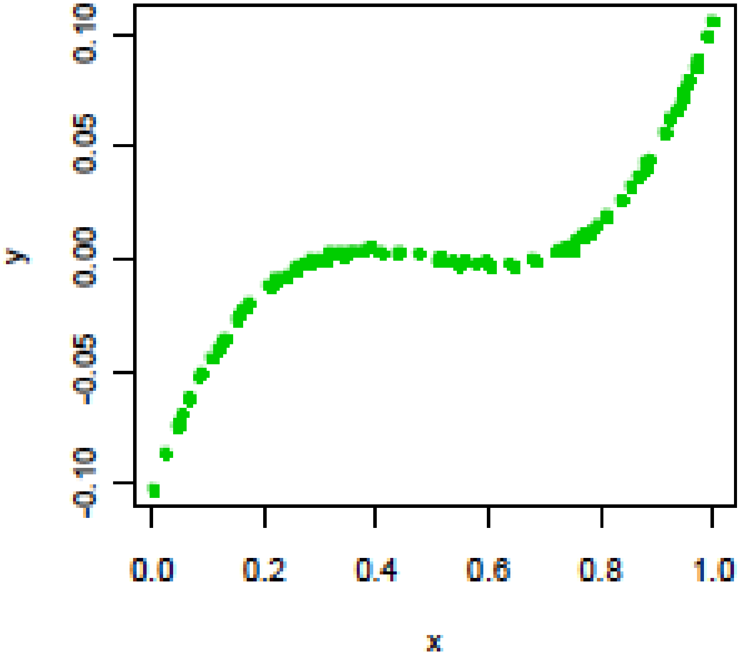


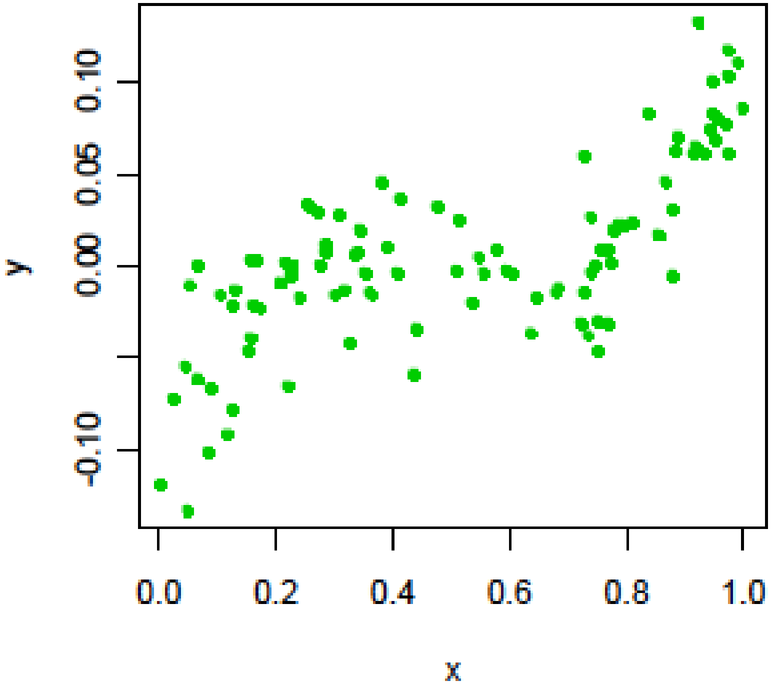
- The variance of the error affects the accuracy of estimating *f*



Var(ϵ) = 10^{-4}

Variance error = Var(ϵ) = 10^{-6}





Var(ϵ) = $9\text{x}10^{-4}$

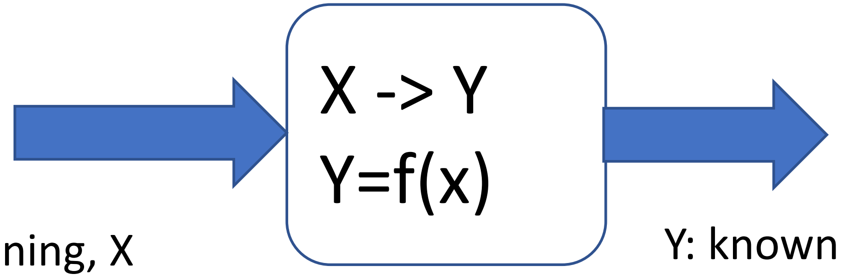
How to estimate f?

- We use data to estimate (“**learn**”) *f*.

Training data: $\{(x_1, y_1),\ldots, (x_n, y_n)\}$

Training Phase: the model learns, i.e., estimate function *f*

- Two approaches to estimate *f*
  - **Parametric approach**
    - Assumes a functional form
  - **Non-parametric approach**



X -> Y

Y=f(x)

Training, X

Y: known in supervised learning, and unknown in unsupervised learning

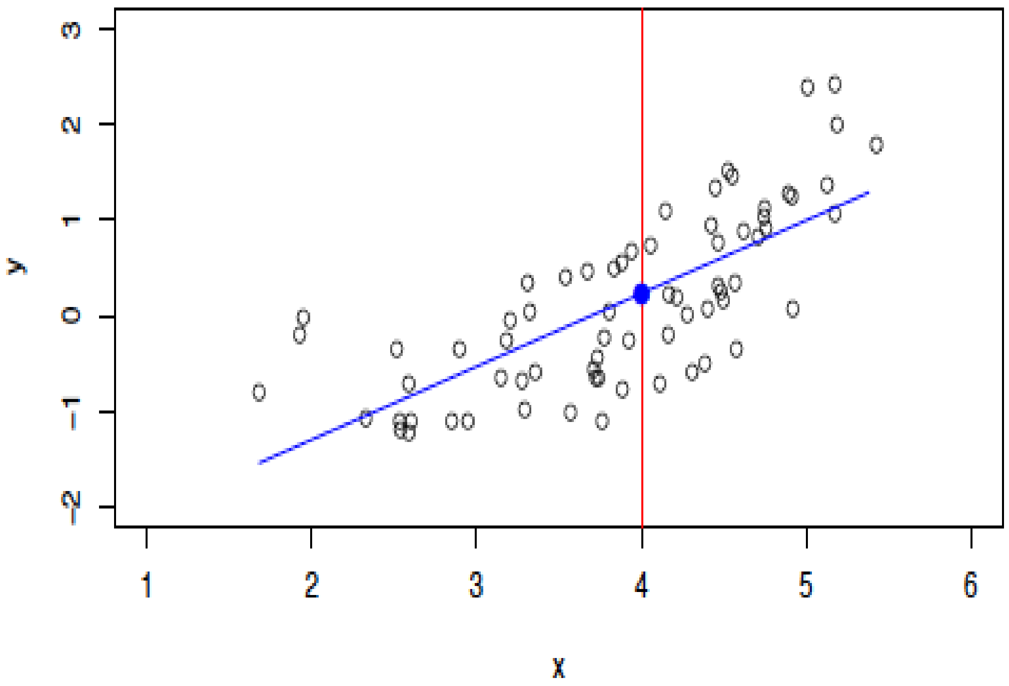
How to estimate f? - Parametric Approach

- First, **assume function form**
  - Example: linear regression, $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots . + \beta_P X_P$

- Second, use training to fit the model (get the parameters of the assumed function)
  - Find $\beta_0, \beta_1, \ldots., \beta_P$
  - Common approach for linear regression is ordinary least square (discussed later)

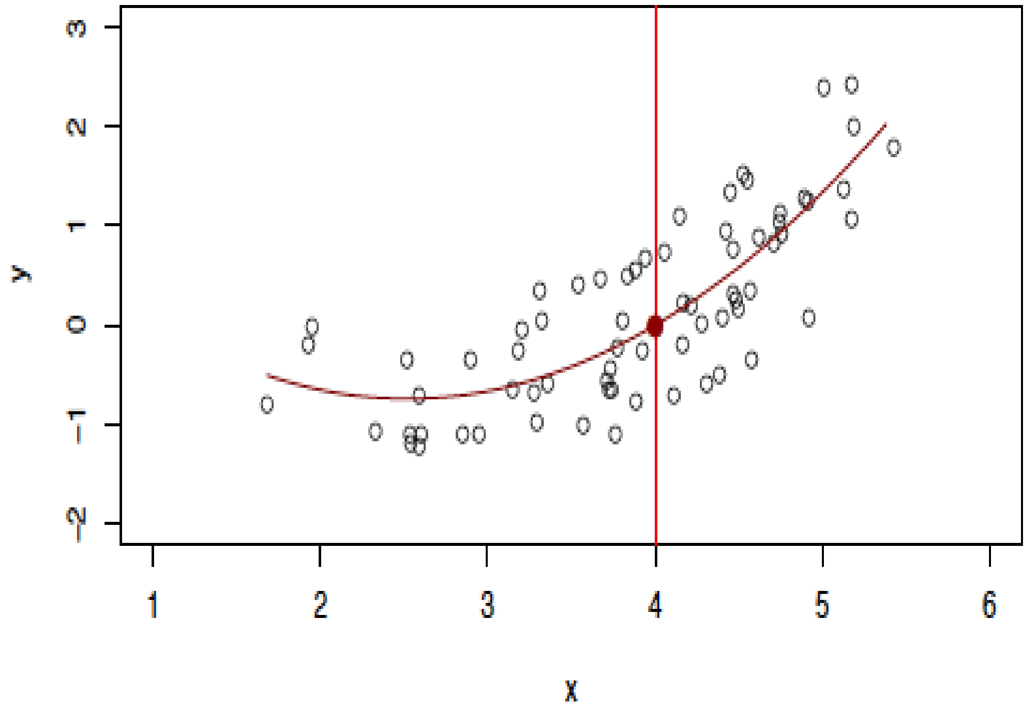
- Examples:
  - income$= \beta_0 + \beta_1 Education + \beta_2 Seniority$
  - $Sales = \beta_0 + \beta_1 TV + \beta_2 Radio + \beta_3 Newspaper$

Parametric Approach – Simple functions



Linear function

$$\hat{f}_L(X) = \hat{\beta}_0 + \hat{\beta}_1 X$$



Quadratic function

$$\hat{f}_Q(X) = \hat{\beta}_0 + \hat{\beta}_1 X + \hat{\beta}_2 X^2$$

If the assumed model is wrong, then the accuracy of the algorithm will be low

Note

- Parametric approaches have finite number of parameters – independent of the training data!

How to estimate f? – Non-Parametric Approach

- Seek to estimate model *f* as close as possible to the data points
  - The form of the model (f) depends on the data set
  - Still have hyper-parameters to control complexity

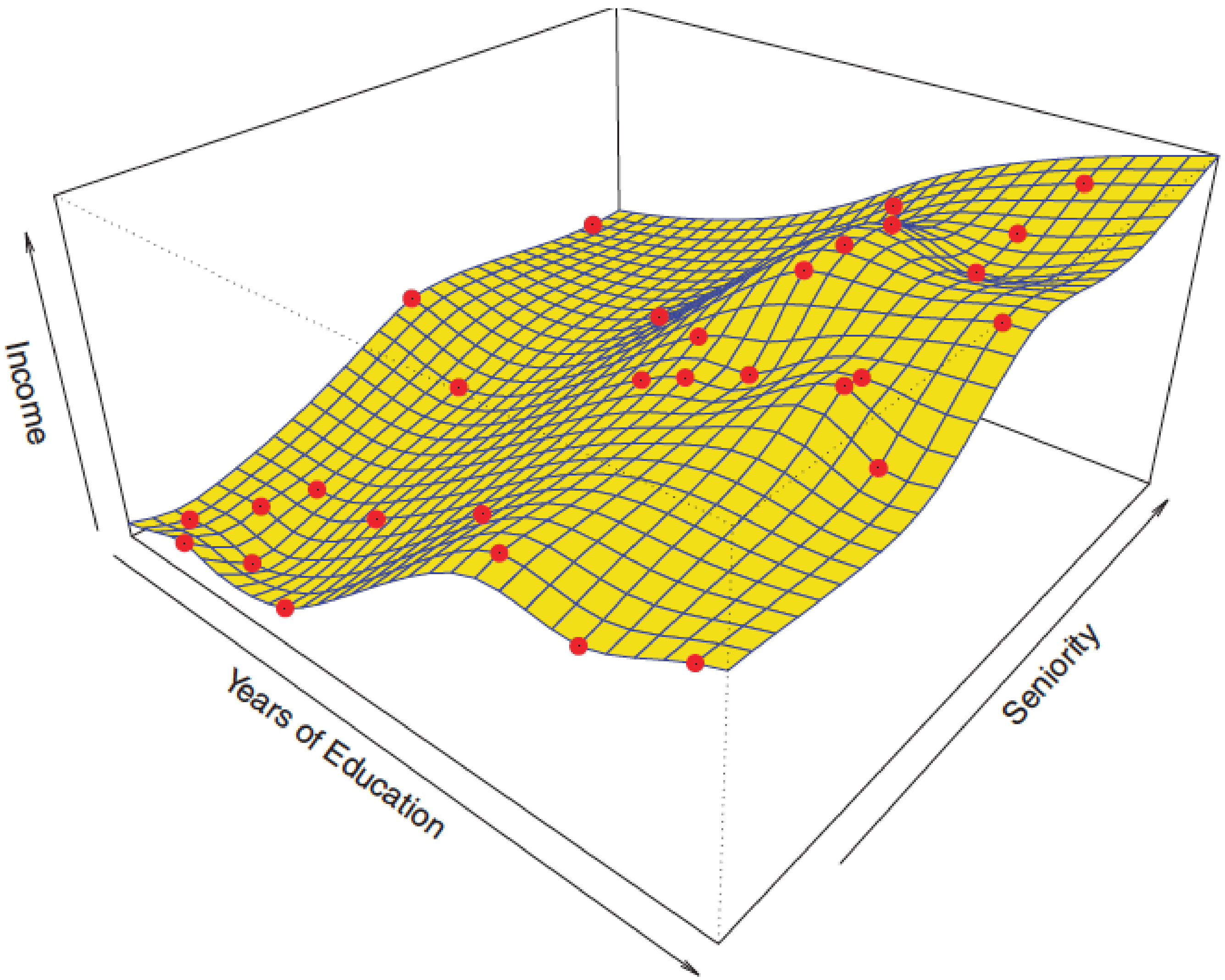
- Advantage: Could work well for a wider range of possible shapes of *f*

- Disadvantage:
  - Needs large number of data points
  - Difficult to understand relationship between output and features

- Example: K-Nearest Neighbors

More Flexible/Complex Models

- More flexible models is used to estimate *f*
- Fit to simulated/training data very well
- May not work well on new data!
  - Why?!

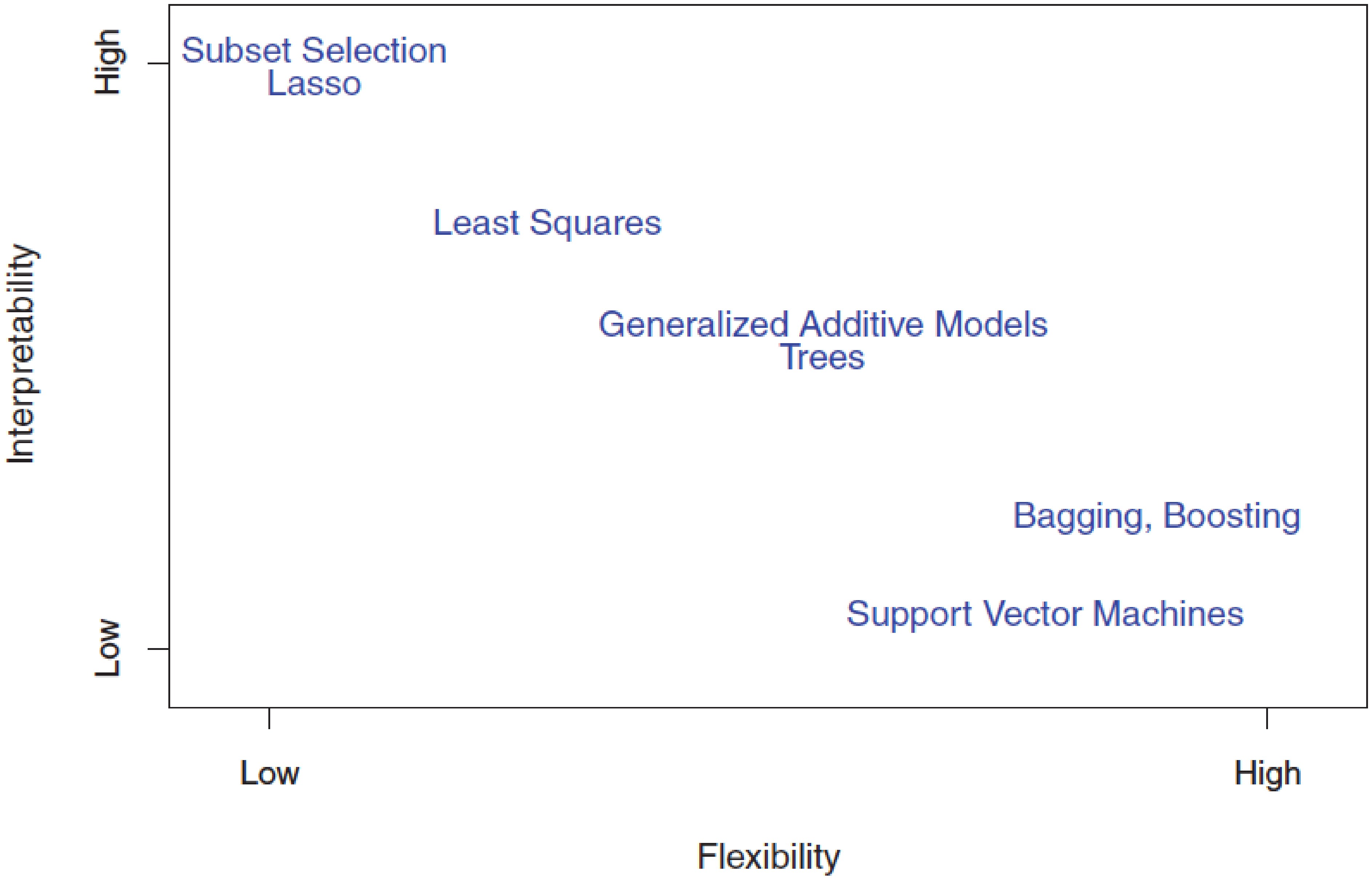


Trade-off: Model Flexibility vs Model Interpretability

- **Less flexible models, less complex, more restrictive:** Produce relatively small range of shapes to estimate the function *f*
  - *Simple and **easy to interpret,*** e.g. *Linear regression*
  - *Helps in inference*

- **More flexible models**, **more complex**: generate wide rang of shapes for *f*
  - ***Harder to interpret,** e.g. SVM, neural networks*

- **More flexible** model would be better for **prediction**
  - But this is not always the case! Overfitting should be avoided
  - It is possible that simpler models lead to higher accuracy

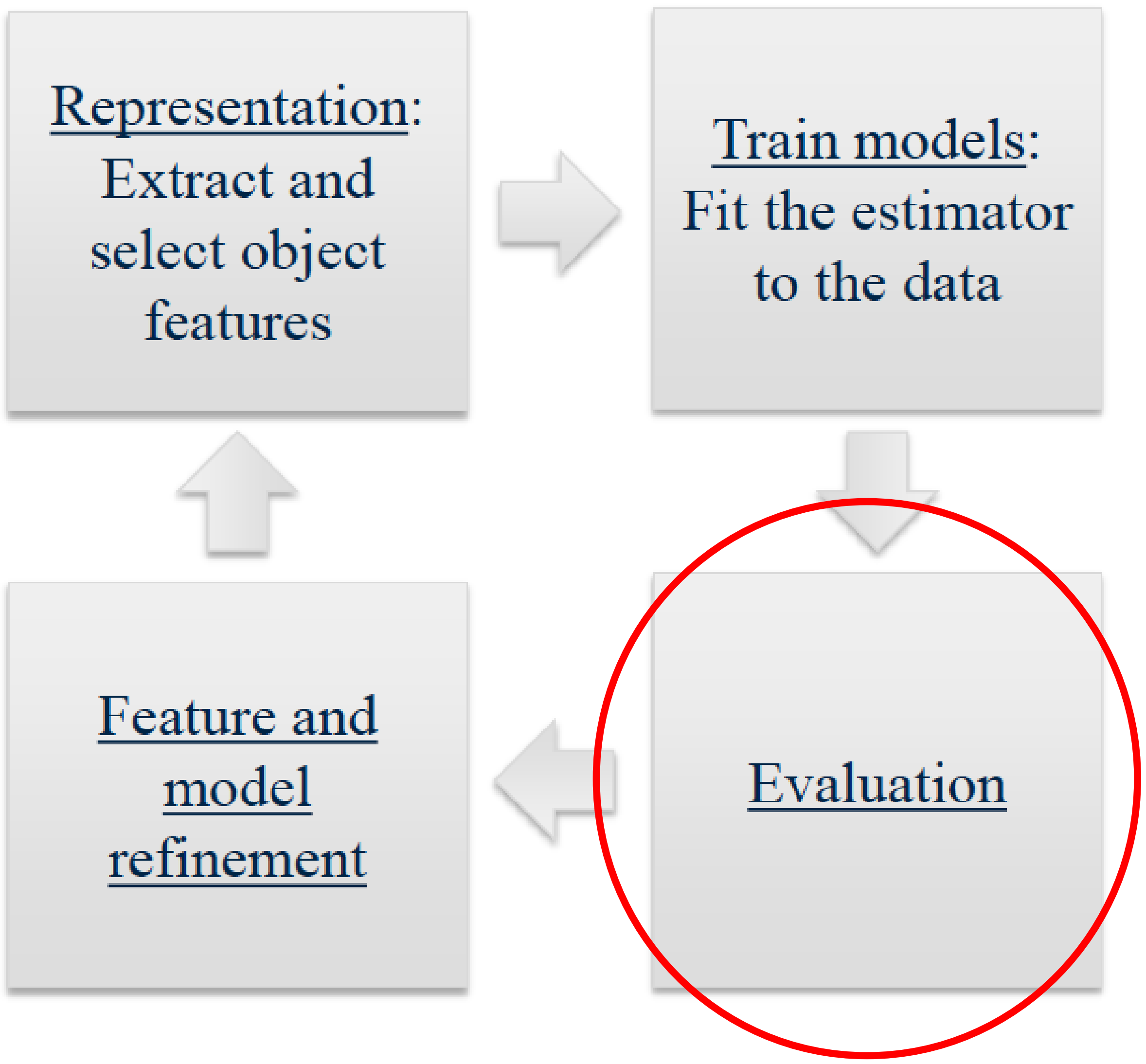


Or Complexity

Assessing the Model Accuracy

How does model flexibility (complexity) affect the accuracy?

Machine Learning Process



Representation: Extract and select object features
Train models: Fit the estimator to the data
Evaluation
Feature and model refinement

Recall Variables

- Outcome label **Y**
- Input **features** vector of length P **(P features)**
- We have **n training data** instances: also called **observations**, **data points**
  - For supervised learning, training data contains input feature and output label pairs:

    $((x_1; y_1), (x_2; y_2), \ldots, (x_n; y_n) = \{x_i; yi\}_{i=1}^{n}$

    where features of training example i is $x_i = \begin{pmatrix} x_{i,1} \\ \cdot \\ x_{i,p} \end{pmatrix}$

Assessing the Model Accuracy
Supervised Learning - Regression Setting

- **Quality of a** Fit: How good the predictions match the actual data?

- In **regression setting, a common measure is mean squared error (MSE)**

- True model $is\ y = f(x) + \epsilon$, and model estimates *f(x)* as $\hat{f}(x)$

- Then **training MSE** is

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

where $\hat{f}(x_i)$ is the prediction of label of the *i*th training sample (y_i).. Also referred to as $\hat{y}_i$

- However, we are interested in how well the model works when it is applied to previously unseen data
  - Evaluating accuracy on training data is not very helpful!

- Therefore, we evaluate **MSE on test data sets**
  - **Test data** set has observations that were not used to train the learning model
  - No guarantee that model with low training MSE will have low test MSE

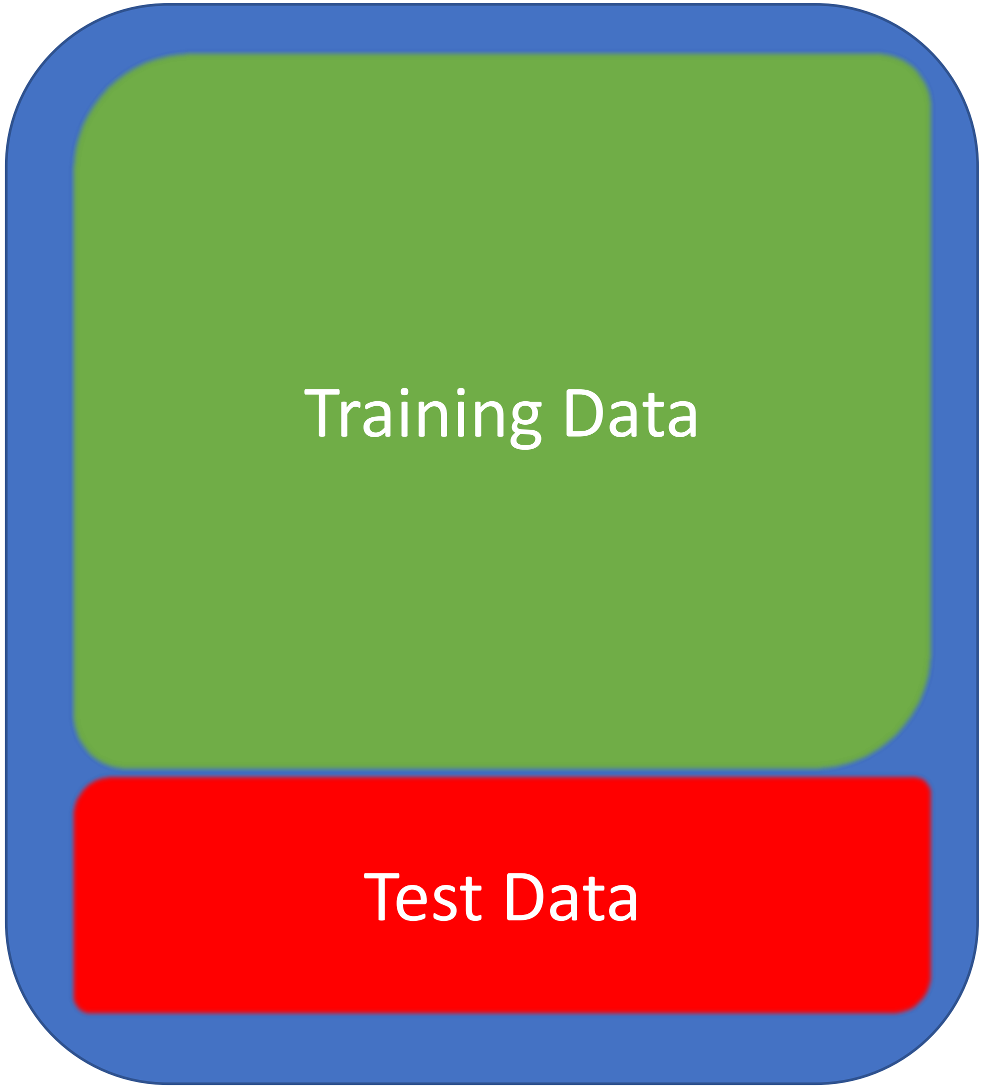
- When (x_0, y_0) is previously unseen by the model, then **test MSE** is

$$average\left(y_0 - \hat{f}(x_0)\right)^2$$

  - This is the average square prediction error for the test observations

Training and Test Sets

- Measure **the model ability to generalize well**, i.e., performs well on new data
  - We **can not use the training data for testing**
- Typical approach: split the available data (features-response pairs) into two parts:
  - **PART I - Training set**: for **building/fitting** the machine learning algorithm
  - **PART II - Test set**: for **testing the accuracy** of the model
- Typically, split data:75% training, 25% test
- **Find the model that has lowest test MSE**



Overfitting and Underfitting – Complexity vs. Accuracy

Two thing we need to avoid:

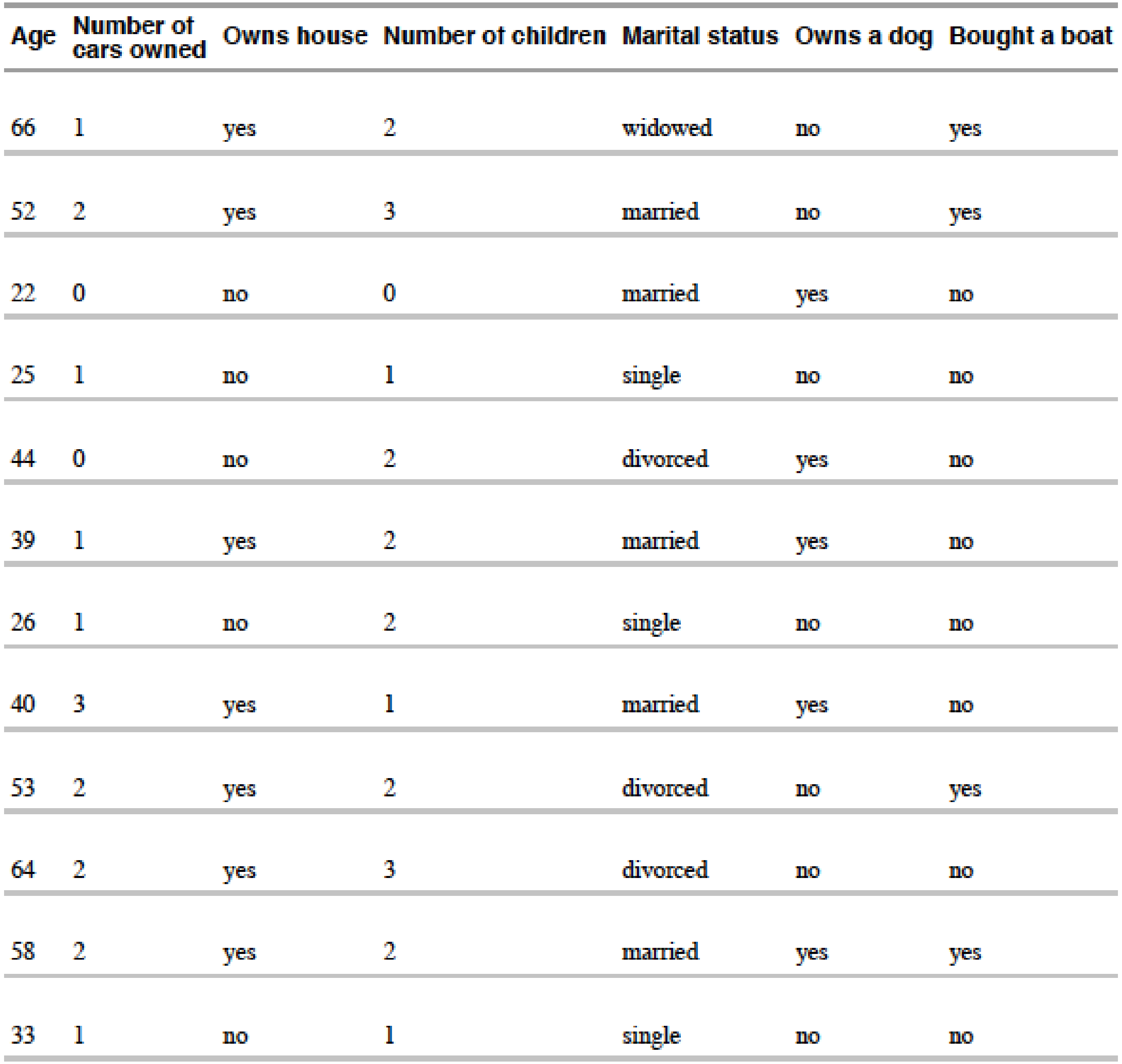
- **Overfitting**: Building a model that is too **complex**, fits training data very well, but **fail to generalize** to new data

- **Underfitting**: build **simple model - unable to capture variability in data**

    - train MSE & test MSE?

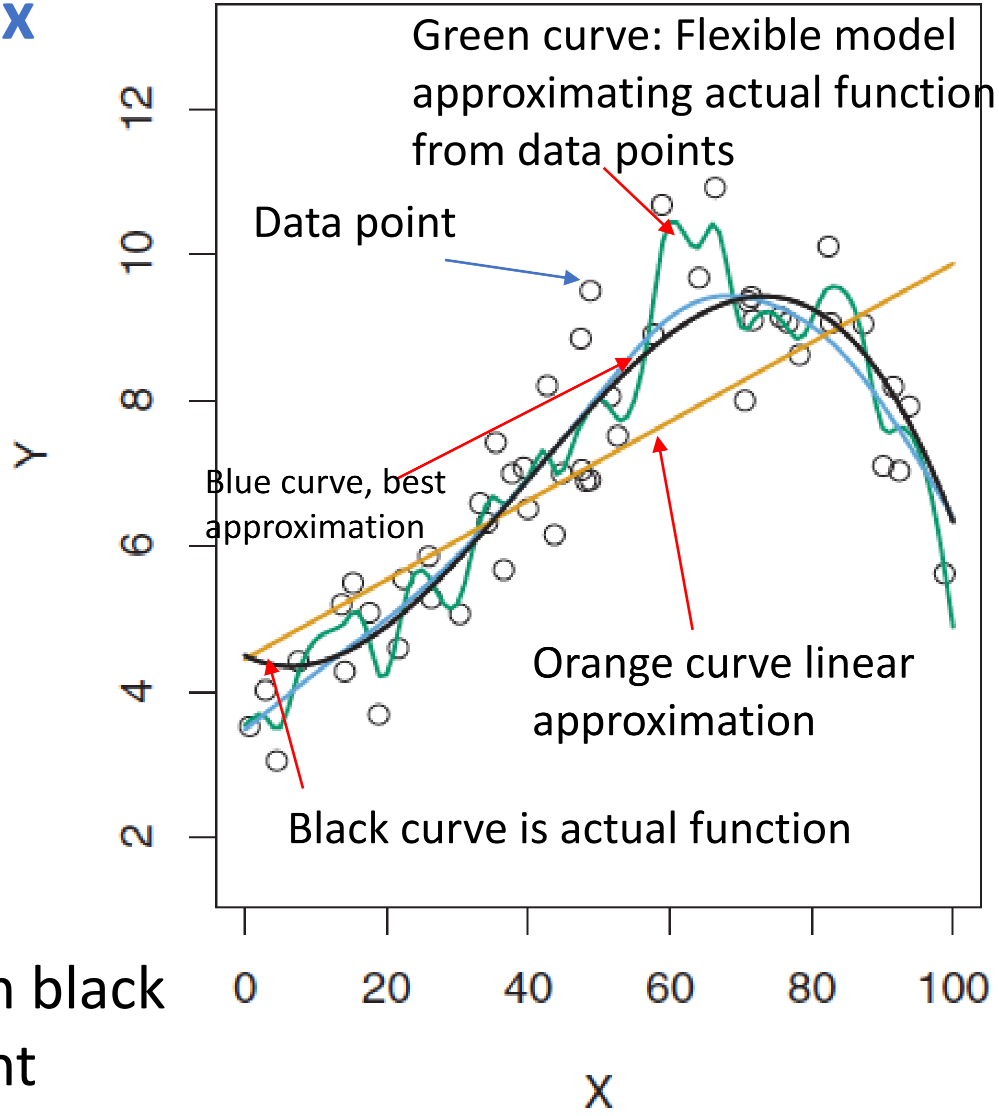
Example

- Predict whether a customer will buy a boat to send promotions
- Given records of previous buyers
- Complex rules that work well on training: Buy a boat if
  - Age 66, 52, 53, 58,..
- **Complex** rules supported by little data results in **Overfitting**
- Over **simplified** rules result in **underfitting**
  - E.g. who has a house, buys a boat

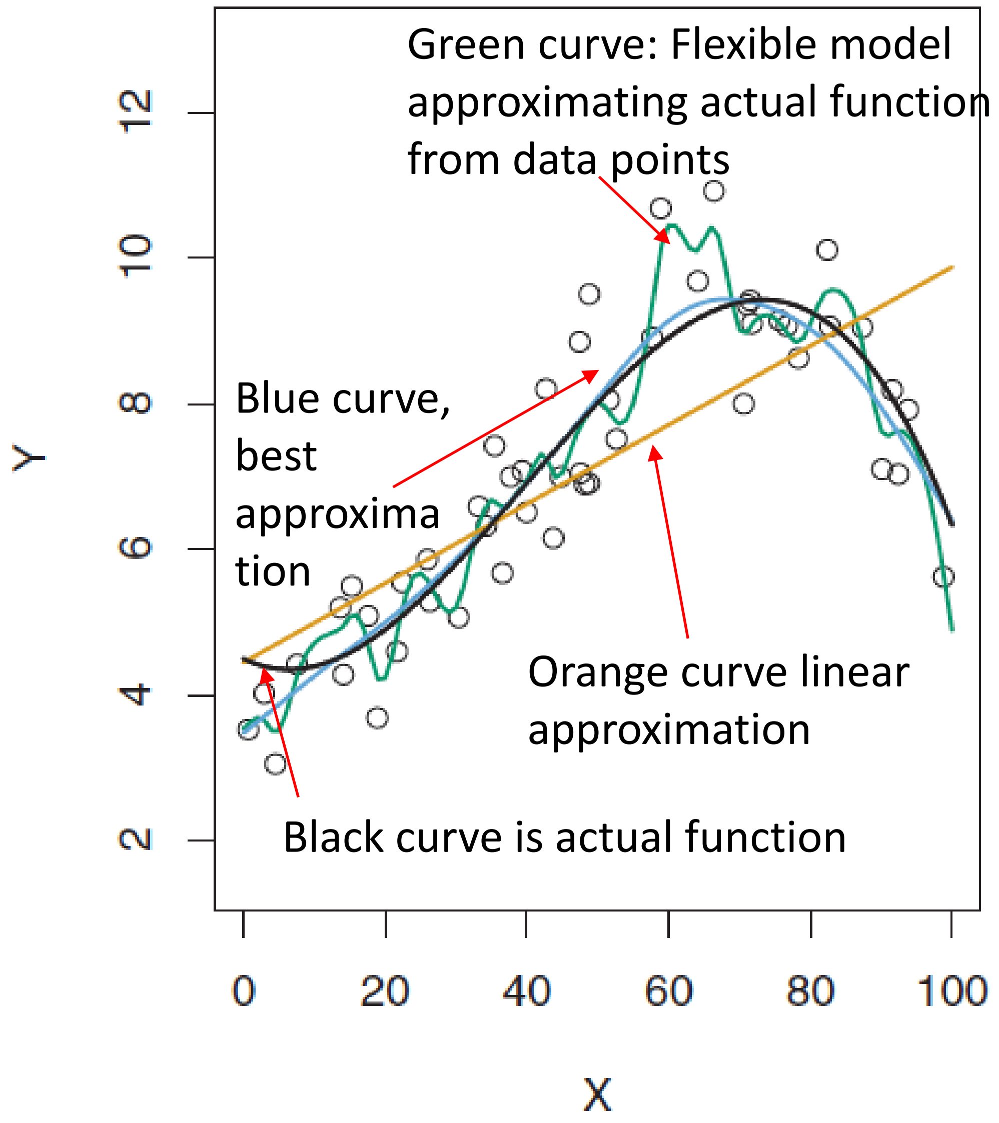


Age	Number of cars owned	Owns house	Number of children	Marital status	Owns a dog	Bought a boat
66	1	yes	2	widowed	no	yes
52	2	yes	3	married	no	yes
22	0	no	0	married	yes	no
25	1	no	1	single	no	no
44	0	no	2	divorced	yes	no
39	1	yes	2	married	yes	no
26	1	no	2	single	no	no
40	3	yes	1	married	yes	no
53	2	yes	2	divorced	no	yes
64	2	yes	3	divorced	no	no
58	2	yes	2	married	yes	yes
33	1	no	1	single	no	no

- Less complex (flexible) model or more complex (flexible) model works well?
    - **Depending on application and the data**
    - **Find right model for your application/data**
    - **Simple models may not capture the variability** in the data
    - **Complex models may not generalize**



Example in fig.: Data point drawn from black curve, and three models (with different flexibilities) are used to fit curve

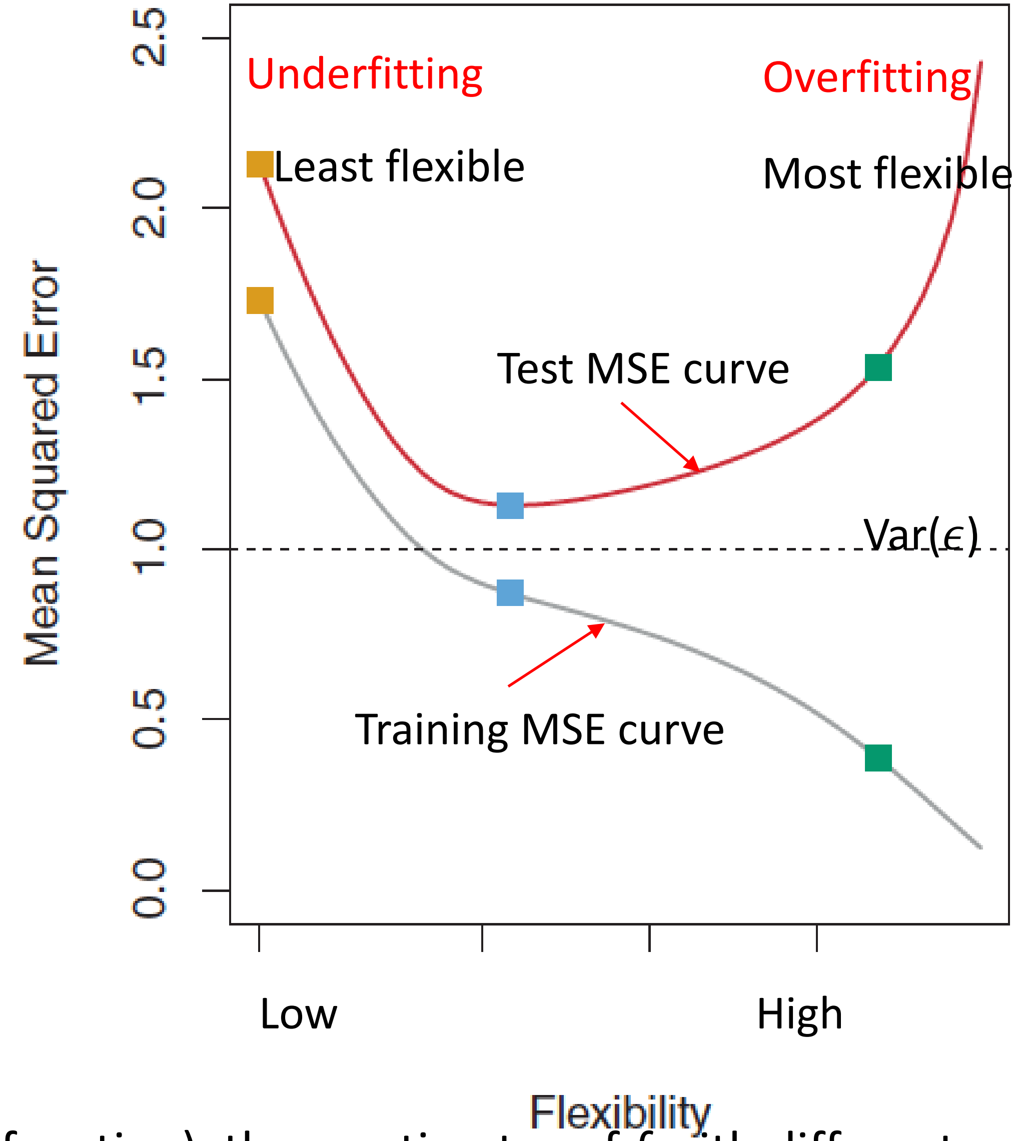


Green curve: Flexible model approximating actual function from data points

Blue curve, best approximation

Orange curve linear approximation

Black curve is actual function



Underfitting

Overfitting

Least flexible

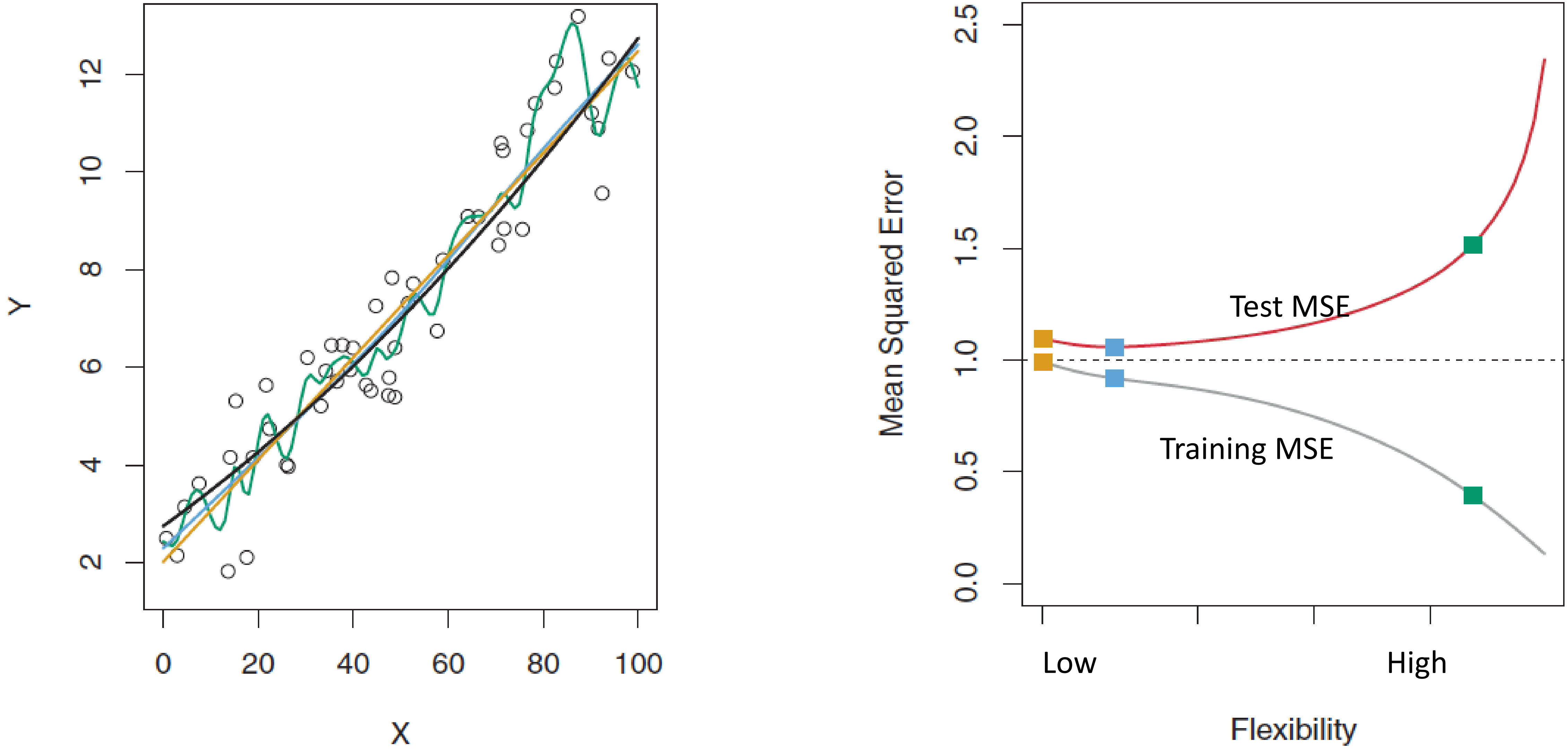
Most flexible

Test MSE curve

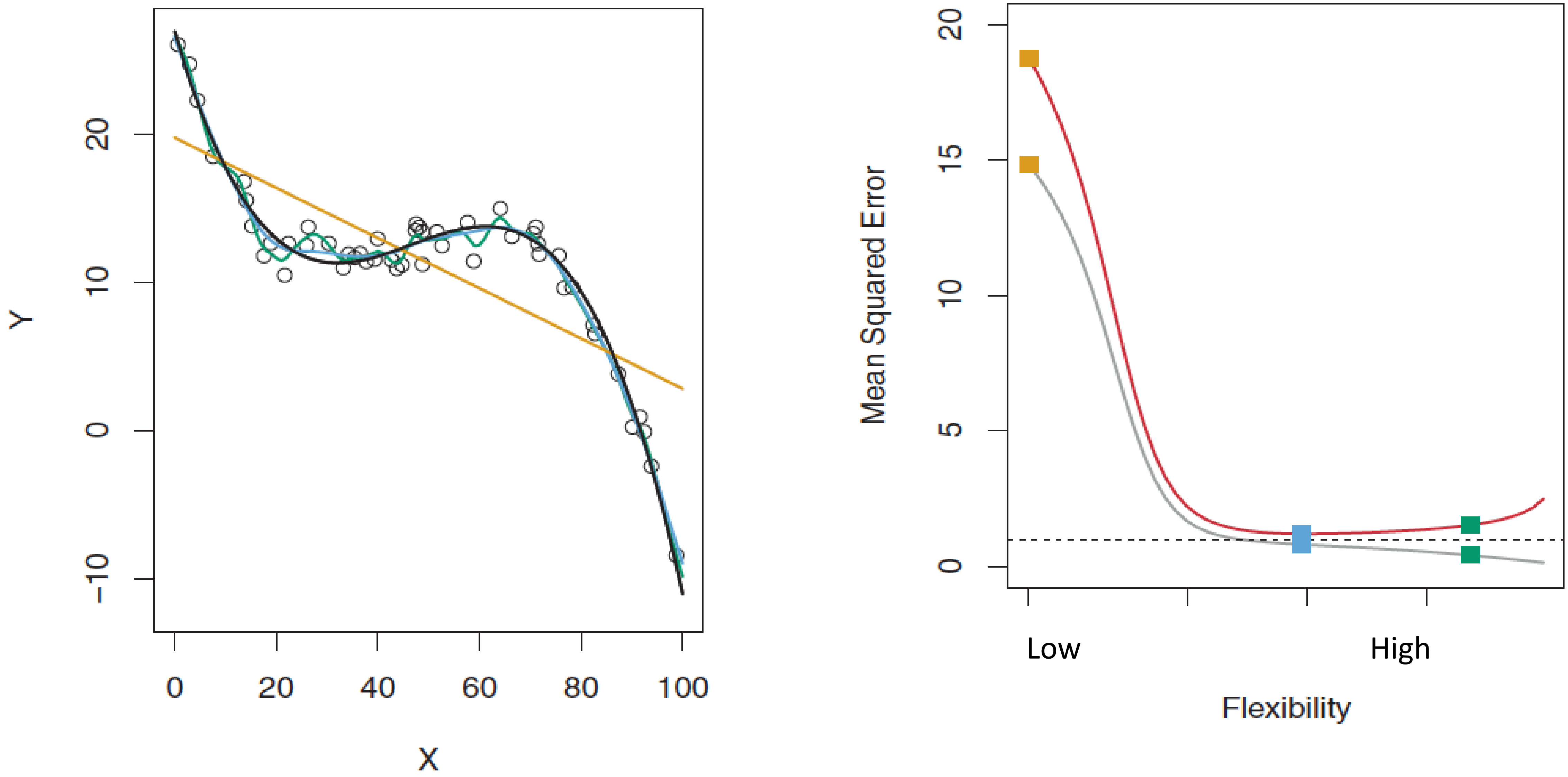
Training MSE curve

In this example, actual *f* in black (data points drawn from this function), three estimates of *f* with different flexibility: the linear regression fit (orange), and two more flexible models (blue and green)

The most flexible fit (green) is wiggly, which fits the training data very well, but has high test MSE



In this example, the linear model (least flexible) fits data very well, more flexibility results in more test MSE



The actual function is wiggly, so more flexible models do better.

Bias-Variance Tradeoff

- The U-shape of the test MSE is a result of the **bias-variance** tradeoff
  - Two **competing properties** of statistical learning methods

- Test MSE is composed of sum of quantities: variance of $\hat{f}(x_0)$, bias of $\hat{f}(x_0)$, and variance of error term ϵ

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon)$$

Expected value of MSE

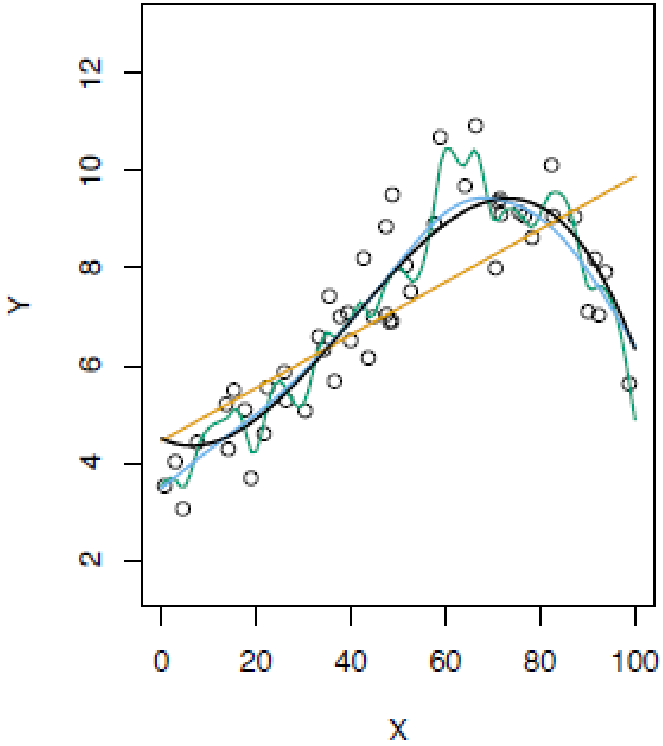
$$E[\hat{f}(x_0)] - f(x_0)$$

  - The expected value of MSE: the average MSE if we repeatedly estimate *f* using large number of training sets and test each at x_0; in the end , average result over all test data

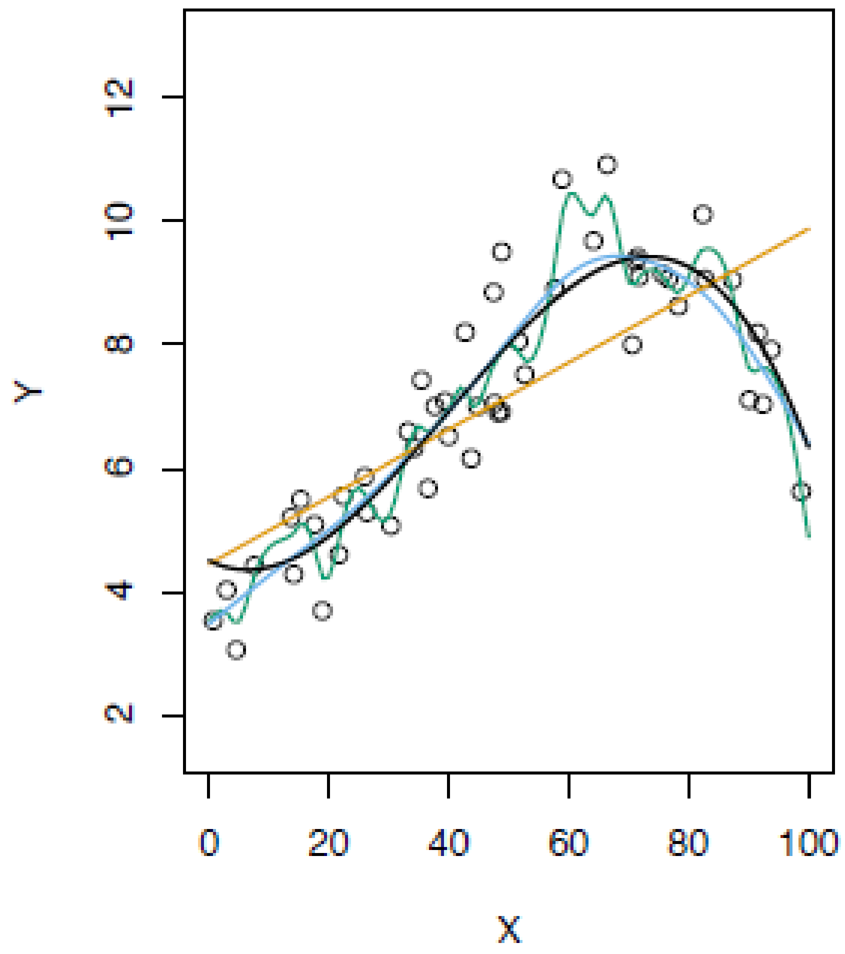
Bias-Variance Tradeoff

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon)$$

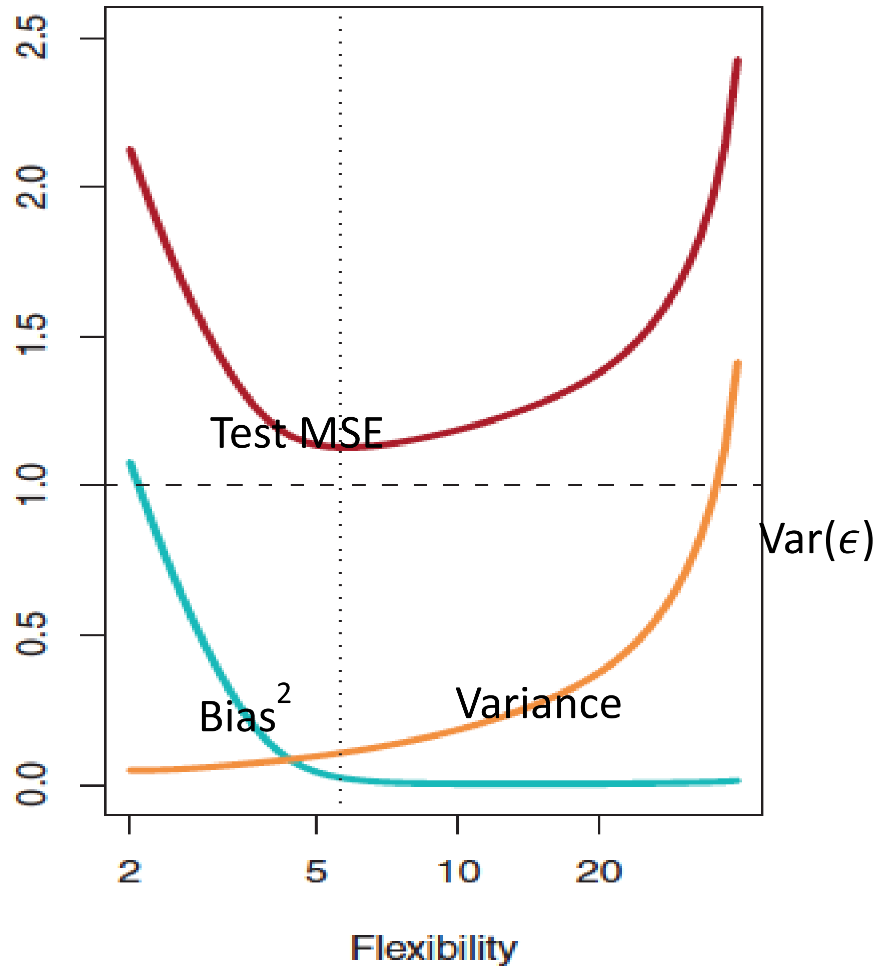
- Bias-variance trade off:
  - Variance: amount by which $\hat{f}$ changes if we made the estimation by different training set
    - High variance means small changes in training data leads to large changes in estimation $\hat{f}$
    - More flexible models could be of high variance
      - E.g. Green curve in the figure has high variance, while orange curve has low variance
  - Bias: Errors from approximating real-life problems by a simpler model
    - How far the estimated model is from the actual function



Test MSE, Bias and Variance



Test MSE = Bias + model variance + error variance



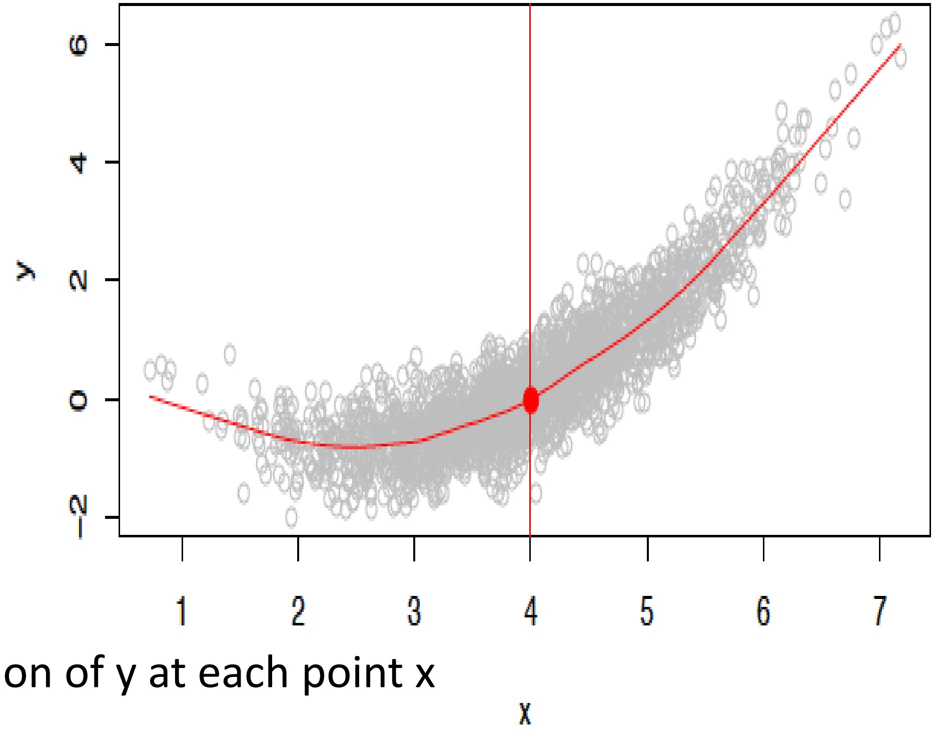
- Best model has low variance and low bias ➔ low test MSE
- **Flexible models (complex models) have low bias but high variance**
- **Less flexible models have high bias and low variance**

Reducible and Irreducible Error

- Test MSE

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \underbrace{\mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2}_{\text{Reducible error}} + \underbrace{\mathrm{Var}(\epsilon)}_{\text{Irreducible}}$$

- Reducible error: can be improved by better estimate of *f(x)*
- Irreducible error: cannot be reduced with better estimate of *f(x)*



ϵ= Y - f(x) is the irreducible error

Measurement errors or due to distribution of y at each point x

Key takeaways

- Prediction and Inference objectives
- Tradeoffs between accuracy and complexity
    - Overfitting vs Underfitting
    - Bias-variance tradeoffs