Training, Testing, and the Bias-Variance Tradeoff

Kirk Bansak

February 23, 2023

Problem Sets

- Problem Set 3 is now due.
- Problem Set 4 is due next Thursday, March 2, by 12:29PM PT.
- Reminder: You should not use any `R` packages or functionalities beyond what is already loaded when you open up `RStudio`, unless a package is explicitly mentioned.
- The packages automatically loaded by default (and hence allowed) are: `base`, `datasets`, `grDevices`, `graphics`, `methods`, `stats`, `utils`
- You may always use `ggplot2`.

Supervised Learning

We assume some relationship between Y and $X = (X_1, X_2, ..., X_p)$, such that:

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

- f is some fixed but unknown function of $X_1, X_2, ..., X_p$
- ϵ is a random irreducible error term

Supervised Learning

We assume some relationship between Y and $X = (X_1, X_2, ..., X_p)$, such that:

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

- f is some fixed but unknown function of $X_1, X_2, ..., X_p$
- ϵ is a random irreducible error term

Supervised Learning

Using observed data (X and Y) to estimate f with $\hat{f}$

Supervised Learning

We assume some relationship between Y and $X = (X_1, X_2, ..., X_p)$, such that:

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

- f is some fixed but unknown function of $X_1, X_2, ..., X_p$
- ϵ is a random irreducible error term

Supervised Learning

Using observed data (X and Y) to estimate f with $\hat{f}$

Ultimate Goal: Build a $\hat{f}$ that is as close as possible to f

Training vs. Testing

1. Designate a set of n data points, which include both the *output* (response) and *input* (predictor) variables, called training data.

Training vs. Testing

1. Designate a set of n data points, which include both the *output* (response) and *input* (predictor) variables, called training data.
2. Select a method for estimating f.

Training vs. Testing

1. Designate a set of n data points, which include both the *output* (response) and *input* (predictor) variables, called training data.
2. Select a method for estimating f.
3. Use training data $\{(y_1, \mathbf{x}_1), (y_2, \mathbf{x}_2), ..., (y_n, \mathbf{x}_n)\}$ to train (a.k.a. fit, estimate, build) $\hat{f}$, which will be our prediction function.

Training vs. Testing

1. Designate a set of n data points, which include both the *output* (response) and *input* (predictor) variables, called training data.
2. Select a method for estimating f.
3. Use training data $\{(y_1, \mathbf{x}_1), (y_2, \mathbf{x}_2), ..., (y_n, \mathbf{x}_n)\}$ to train (a.k.a. fit, estimate, build) $\hat{f}$, which will be our prediction function.
4. Use $\hat{f}$ to predict values for Y on test data observations $(y_0, \mathbf{x}_0)$ previously unseen by the model.

Training vs. Testing

1. Designate a set of n data points, which include both the *output* (response) and *input* (predictor) variables, called training data.
2. Select a method for estimating f.
3. Use training data $\{(y_1, \mathbf{x}_1), (y_2, \mathbf{x}_2), ..., (y_n, \mathbf{x}_n)\}$ to train (a.k.a. fit, estimate, build) $\hat{f}$, which will be our prediction function.
4. Use $\hat{f}$ to predict values for Y on test data observations $(y_0, \mathbf{x}_0)$ previously unseen by the model.
5. Compare predicted response value $(\hat{y}_0)$ with true response value (y_0) for observations in the test data to evaluate performance.

Training vs. Testing

1. Designate a set of n data points, which include both the *output* (response) and *input* (predictor) variables, called training data.
2. Select a method for estimating f.
3. Use training data $\{(y_1, \mathbf{x}_1), (y_2, \mathbf{x}_2), ..., (y_n, \mathbf{x}_n)\}$ to train (a.k.a. fit, estimate, build) $\hat{f}$, which will be our prediction function.
   - Note: The training process often involves additional subsplits on the training data for exploratory performance assessments that allow us to 'tune' the model. More on this to come next week!
4. Use $\hat{f}$ to predict values for Y on test data observations $(y_0, \mathbf{x}_0)$ previously unseen by the model.
5. Compare predicted response value $(\hat{y}_0)$ with true response value (y_0) for observations in the test data to evaluate performance.

Purpose of Test Set

- **Estimating Real-World Performance**: Allows us to gauge how well our prediction function will perform 'in the wild'.

- **Making Decisions**: To make the final decision on what machine learning method/model to deploy.

Performance Metrics

For Regression problems, the most common performance metric is Mean Squared Error (MSE).

$$\text{Training MSE}: \quad \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(\mathbf{x}_i))^2$$

$$\text{Test MSE}: \quad Ave\left((y_0 - \hat{f}(\mathbf{x}_0))^2\right)$$

Performance Metrics

For Regression problems, the most common performance metric is Mean Squared Error (MSE).

$$\text{Training MSE}: \quad \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(\mathbf{x}_i))^2$$

$$\text{Test MSE}: \quad Ave\left((y_0 - \hat{f}(\mathbf{x}_0))^2\right)$$

For Classification problems, a common performance metric is the Classification Error Rate (1 - Classification Accuracy).

$$\text{Training Error Rate}: \quad \frac{1}{n}\sum_{i=1}^{n}\mathrm{I}(y_i \neq \hat{y}_i)$$

$$\text{Test Error Rate}: \quad Ave\left(\mathrm{I}(y_0 \neq \hat{y}_0)\right)$$

Performance Metrics

For Regression problems, the most common performance metric is Mean Squared Error (MSE).

$$\text{Training MSE}: \quad \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(\mathbf{x}_i))^2$$

$$\text{Test MSE}: \quad Ave\left((y_0 - \hat{f}(\mathbf{x}_0))^2\right)$$

For Classification problems, a common performance metric is the Classification Error Rate (1 - Classification Accuracy).

$$\text{Training Error Rate}: \quad \frac{1}{n}\sum_{i=1}^{n}\mathrm{I}(y_i \neq \hat{y}_i)$$

$$\text{Test Error Rate}: \quad Ave\left(\mathrm{I}(y_0 \neq \hat{y}_0)\right)$$

But recall the issues with focusing exclusively on classification accuracy. We will discuss other classification performance metrics later.

Training vs. Test Set Separation

- How to separate training and test data?
  - **As default practice, the split of your data into training and test sets should be performed randomly.**

Training vs. Test Set Separation

- How to separate training and test data?
  - **As default practice, the split of your data into training and test sets should be performed randomly.**

    Unless...

    More closely mimicking the ultimate deployment context requires a different split strategy.

Training vs. Test Set Separation

- How to separate training and test data?
  - **As default practice, the split of your data into training and test sets should be performed randomly.**

    Unless...

    More closely mimicking the ultimate deployment context requires a different split strategy.

    - e.g. Contexts with a systematic temporal dimension
      $\rightsquigarrow$ temporal training-test split
      (for instance, predicting tomorrow's stock prices with today's stock behavior)

Training vs. Test Set Separation

- How to separate training and test data?
  - **As default practice, the split of your data into training and test sets should be performed randomly.**

    Unless...

    More closely mimicking the ultimate deployment context requires a different split strategy.
    - e.g. Contexts with a systematic temporal dimension
      $\rightsquigarrow$ temporal training-test split
      (for instance, predicting tomorrow's stock prices with today's stock behavior)

- What proportion of data should be in the training vs. test set?
  - **Common advice: 80/20 or 70/30 (training/test)**

Training vs. Test Set Separation

- How to separate training and test data?
  - **As default practice, the split of your data into training and test sets should be performed randomly.**

    Unless...

    More closely mimicking the ultimate deployment context requires a different split strategy.
    - e.g. Contexts with a systematic temporal dimension
      $\rightsquigarrow$ temporal training-test split
      (for instance, predicting tomorrow's stock prices with today's stock behavior)

- What proportion of data should be in the training vs. test set?
  - **Common advice: 80/20 or 70/30 (training/test)**
  - But it depends, because there's a tradeoff:
    - More data in training set allows for building a better model.
    - Too little data in test set leads to less reliable (noisy) error estimates.

To R...

Understanding Supervised Learning Performance

Method Flexibility

Flexibility is a key dimension that varies across different supervised machine learning methods and models.

Method Flexibility

Flexibility is a key dimension that varies across different supervised machine learning methods and models.

- Flexibility of the method itself (e.g. linear regression vs. random forests)

Method Flexibility

Flexibility is a key dimension that varies across different supervised machine learning methods and models.

- Flexibility of the method itself (e.g. linear regression vs. random forests)
- Complexity of the model specifications for a given method
  - e.g. In linear regression: the number of variables included, inclusion of polynomials, etc.
  - e.g. In other machine learning methods: various method-specific settings

Model Flexibility and Training Error

As model flexibility increases, how do you expect **training error** will change? Why?

Model Flexibility and Training Error

As model flexibility increases, how do you expect **training error** will change? Why?

As model flexibility increases, training error will typically decrease monotonically.

This occurs because the actual model estimation/fitting procedures are designed to minimize error on the training set, and greater model flexibility gives the procedure more latitude to do exactly that.

Model Flexibility and Test Error

As model flexibility increases, how do you expect **test error** will change? Why?

Model Flexibility and Test Error

As model flexibility increases, how do you expect **test error** will change? Why?

As model flexibility increases, test error will typically decrease for a while and then increase.

This occurs because as model flexibility increases, more signal can be detected, so test error will decrease at first, but eventually, the estimation process begins to pick up more noise in the training data than signal, so test error will then begin to increase.

Relationship between Flexibility and Performance

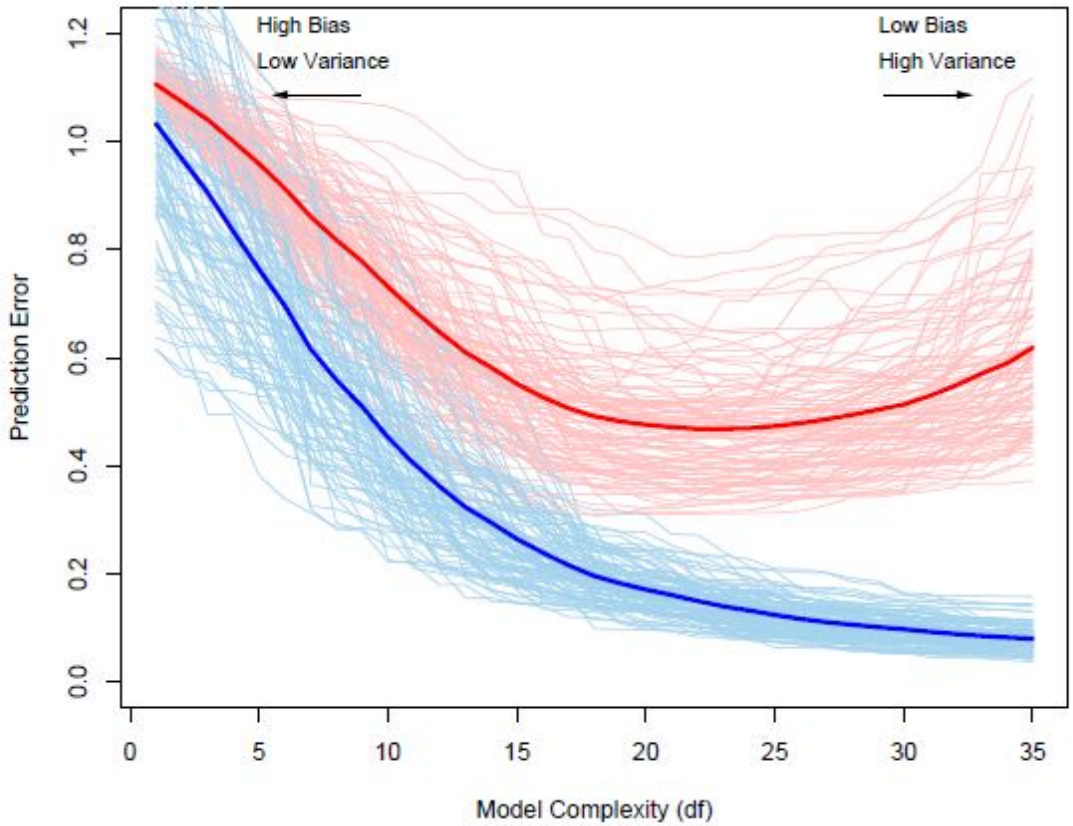

FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for* 100 *training sets of size* 50 *each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$.

Relationship between Flexibility and Performance

Greater flexibility can allow the prediction function to more tightly fit complex patterns in the training data (i.e. data used in the actual estimation process), but can lead to overfitting!

Relationship between Flexibility and Performance

Greater flexibility can allow the prediction function to more tightly fit complex patterns in the training data (i.e. data used in the actual estimation process), but can lead to overfitting!

- Some of the modeling flexibility captures systematic features of the data-generating process (signal)
  $\rightsquigarrow$ Reduces error in both training and test set

Relationship between Flexibility and Performance

Greater flexibility can allow the prediction function to more tightly fit complex patterns in the training data (i.e. data used in the actual estimation process), but can lead to overfitting!

- Some of the modeling flexibility captures systematic features of the data-generating process (signal)
  $\rightsquigarrow$ Reduces error in both training and test set
- Excessive modeling flexibility captures idiosyncratic patterns (noise) in the training set
  $\rightsquigarrow$ Reduces error in training set but increases error in test set

Relationship between Flexibility and Performance

Greater flexibility can allow the prediction function to more tightly fit complex patterns in the training data (i.e. data used in the actual estimation process), but can lead to overfitting!

- Some of the modeling flexibility captures systematic features of the data-generating process (signal)
  $\rightsquigarrow$ Reduces error in both training and test set
- Excessive modeling flexibility captures idiosyncratic patterns (noise) in the training set
  $\rightsquigarrow$ Reduces error in training set but increases error in test set

The optimal amount of flexibility (and optimal type of flexibility inherent in each method) depends on the data.

No one method or model dominates all others!

Relationship between Flexibility and Performance

Greater flexibility can allow the prediction function to more tightly fit complex patterns in the training data (i.e. data used in the actual estimation process), but can lead to overfitting!

- Some of the modeling flexibility captures systematic features of the data-generating process (signal)
  $\rightsquigarrow$ Reduces error in both training and test set
- Excessive modeling flexibility captures idiosyncratic patterns (noise) in the training set
  $\rightsquigarrow$ Reduces error in training set but increases error in test set

The optimal amount of flexibility (and optimal type of flexibility inherent in each method) depends on the data.

No one method or model dominates all others!

This is why a test set is so important!

To R...

Test Error Behavior

What accounts for the U-Shape of Test Error?

Bias-Variance Decomposition

$$MSE = E\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right]$$

Mean-squared error of a predictor (model fit) $\hat{f}$ represents the expected squared distance of its prediction from the truth

Bias-Variance Decomposition

$$MSE = E\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right]$$

Mean-squared error of a predictor (model fit) $\hat{f}$ represents the expected squared distance of its prediction from the truth

(i.e. if we were able to re-train the model over and over using new random samples of training data, and then predict onto test data, what would the average squared error of our predictions be)

Bias-Variance Decomposition

$$MSE = E\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right]$$

Mean-squared error of a predictor (model fit) $\hat{f}$ represents the expected squared distance of its prediction from the truth

(i.e. if we were able to re-train the model over and over using new random samples of training data, and then predict onto test data, what would the average squared error of our predictions be)

Hence, it is a concise summary measure of how good the predictor/model fitting procedure is.

We can decompose the MSE into sub-components to understand the underlying phenomena that lead to good fit.

Bias-Variance Decomposition

$$E\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right] = Var(\hat{f}(\mathbf{x}_0)) + [Bias(\hat{f}(\mathbf{x}_0)]^2 + Var(\epsilon)$$

Expected Test Mean Squared Error

=

Variance of the Prediction Function

+

(Squared) Bias of the Prediction Function

+

Variance of the Random Error

Bias-Variance Decomposition

$$E\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right] = Var(\hat{f}(\mathbf{x}_0)) + [Bias(\hat{f}(\mathbf{x}_0)]^2 + Var(\epsilon)$$

Bias-Variance Decomposition

$$E\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right] = Var(\hat{f}(\mathbf{x}_0)) + [Bias(\hat{f}(\mathbf{x}_0)]^2 + Var(\epsilon)$$

Expected Test MSE: Think of this as the average test MSE that we would obtain if we repeatedly estimated test MSE using a large number of different training sets, and tested each at the values $\mathbf{x}_0$ in the test set.

Bias-Variance Decomposition

$$E\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right] = Var(\hat{f}(\mathbf{x}_0)) + [Bias(\hat{f}(\mathbf{x}_0)]^2 + Var(\epsilon)$$

Expected Test MSE: Think of this as the average test MSE that we would obtain if we repeatedly estimated test MSE using a large number of different training sets, and tested each at the values $\mathbf{x}_0$ in the test set.

Variance of the Prediction Function: Think of this as the amount by which $\hat{f}$ would change (wiggle around) if we estimated it using different training data sets.

Bias-Variance Decomposition

$$E\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right] = Var(\hat{f}(\mathbf{x}_0)) + [Bias(\hat{f}(\mathbf{x}_0)]^2 + Var(\epsilon)$$

Expected Test MSE: Think of this as the average test MSE that we would obtain if we repeatedly estimated test MSE using a large number of different training sets, and tested each at the values $\mathbf{x}_0$ in the test set.

Variance of the Prediction Function: Think of this as the amount by which $\hat{f}$ would change (wiggle around) if we estimated it using different training data sets.

Bias of the Prediction Function: Think of this as the tendency of $\hat{f}$ to make predictions that are systematically too low or too high, error that is introduced by approximating a complicated real-life problem by a much simpler model.

Bias-Variance Decomposition

$$E\left[\left(y_0 - \hat{f}(\mathbf{x}_0)\right)^2\right] = Var(\hat{f}(\mathbf{x}_0)) + [Bias(\hat{f}(\mathbf{x}_0)]^2 + Var(\epsilon)$$

Expected Test MSE: Think of this as the average test MSE that we would obtain if we repeatedly estimated test MSE using a large number of different training sets, and tested each at the values $\mathbf{x}_0$ in the test set.

Variance of the Prediction Function: Think of this as the amount by which $\hat{f}$ would change (wiggle around) if we estimated it using different training data sets.

Bias of the Prediction Function: Think of this as the tendency of $\hat{f}$ to make predictions that are systematically too low or too high, error that is introduced by approximating a complicated real-life problem by a much simpler model.

Variance of the Random Error: Think of this as the irreducible error in the problem, the noise that cannot be explained and will always add error to our predictions no matter how good a job we do and how much data we have.

Bias-Variance Trade-off

Applying different methods and models to a supervised learning problem will result in different degrees of bias and variance, and there is a tradeoff between the two quantities.

Bias-Variance Trade-off

Applying different methods and models to a supervised learning problem will result in different degrees of bias and variance, and there is a tradeoff between the two quantities.

General Patterns:

Bias-Variance Trade-off

Applying different methods and models to a supervised learning problem will result in different degrees of bias and variance, and there is a tradeoff between the two quantities.

General Patterns:

- As method/model **flexibility increases, bias decreases**.

Bias-Variance Trade-off

Applying different methods and models to a supervised learning problem will result in different degrees of bias and variance, and there is a tradeoff between the two quantities.

General Patterns:

- As method/model **flexibility increases, bias decreases**.
- As method/model **flexibility increases, variance increases**.

Bias-Variance Trade-off

Applying different methods and models to a supervised learning problem will result in different degrees of bias and variance, and there is a tradeoff between the two quantities.

General Patterns:

- As method/model **flexibility increases, bias decreases**.
- As method/model **flexibility increases, variance increases**.
- Bias tends to initially decrease faster than the variance increases, leading expected test MSE to initially decline.

Bias-Variance Trade-off

Applying different methods and models to a supervised learning problem will result in different degrees of bias and variance, and there is a tradeoff between the two quantities.

General Patterns:

- As method/model **flexibility increases, bias decreases**.
- As method/model **flexibility increases, variance increases**.
- Bias tends to initially decrease faster than the variance increases, leading expected test MSE to initially decline.
- At some point, increasing flexibility has limited impact on bias but starts to significantly increase variance, leading test MSE to increase.

Bias-Variance Trade-off

Applying different methods and models to a supervised learning problem will result in different degrees of bias and variance, and there is a tradeoff between the two quantities.

General Patterns:

- As method/model **flexibility increases, bias decreases**.
- As method/model **flexibility increases, variance increases**.
- Bias tends to initially decrease faster than the variance increases, leading expected test MSE to initially decline.
- At some point, increasing flexibility has limited impact on bias but starts to significantly increase variance, leading test MSE to increase.

The relative rate of change of the variance and bias determines whether the test MSE increases or decreases.

Bias-Variance Trade-off Depicted

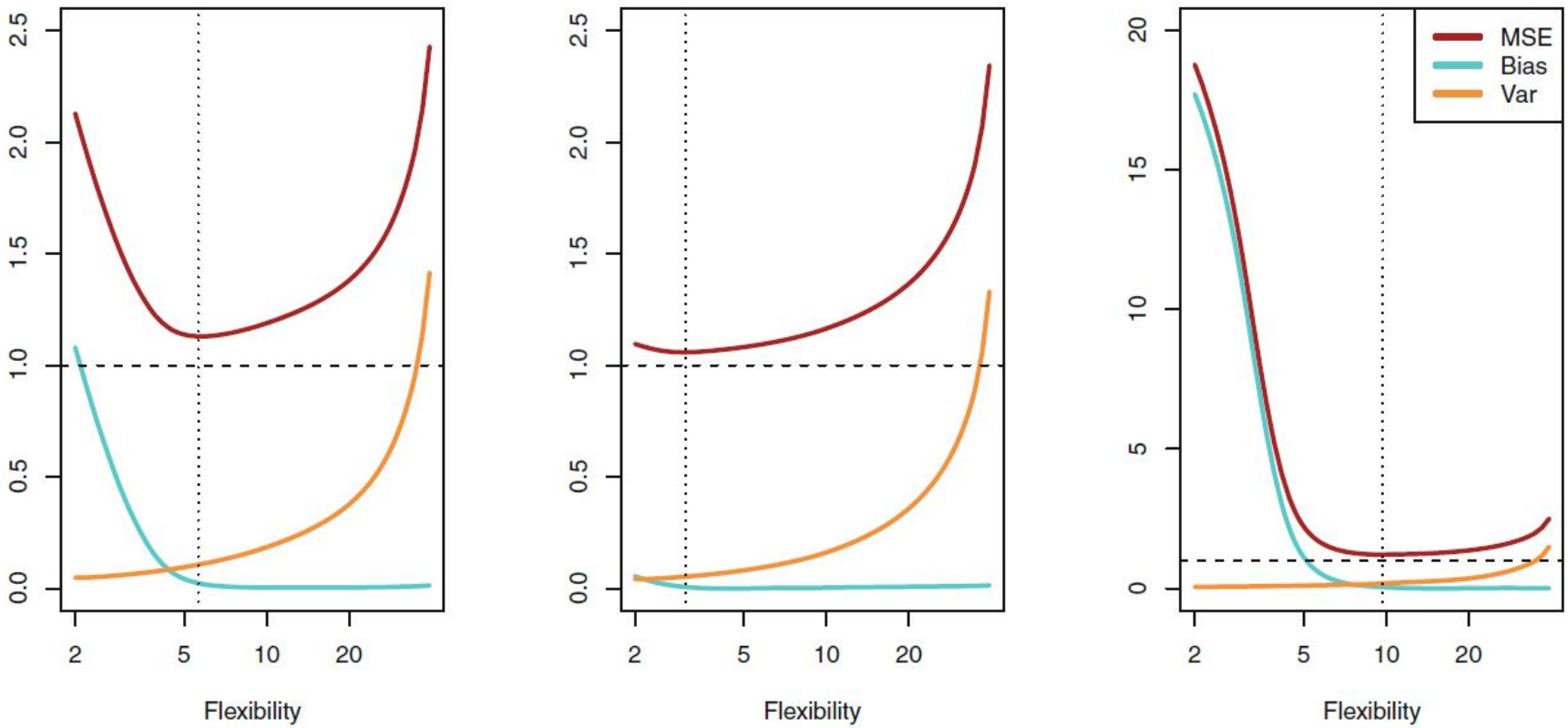

FIGURE 2.12. *Squared bias (blue curve), variance (orange curve), Var(ϵ) (dashed line), and test MSE (red curve) for the three data sets in Figures 2.9–2.11. The vertical dotted line indicates the flexibility level corresponding to the smallest test MSE.*

To R...

In Practice

- In a real-life situation in which f is unknown, it is generally not possible to explicitly compute the bias or variance for a statistical learning method.

- However, one should always keep the bias-variance tradeoff in mind, as it provides an underlying framework for diagnosing and improving one's models.

- Conceptual understanding of this key tradeoff
  $+$ Toolkit of different methods and modeling skills
  $+$ Solid test error assessment practices
  $\rightsquigarrow$ Effectiveness in executing supervised machine learning