

Unit 2 Review: Tuning Predictive Models

STAT 471

Lecture 1: Model complexity

Lecture 2: Bias-variance trade-off

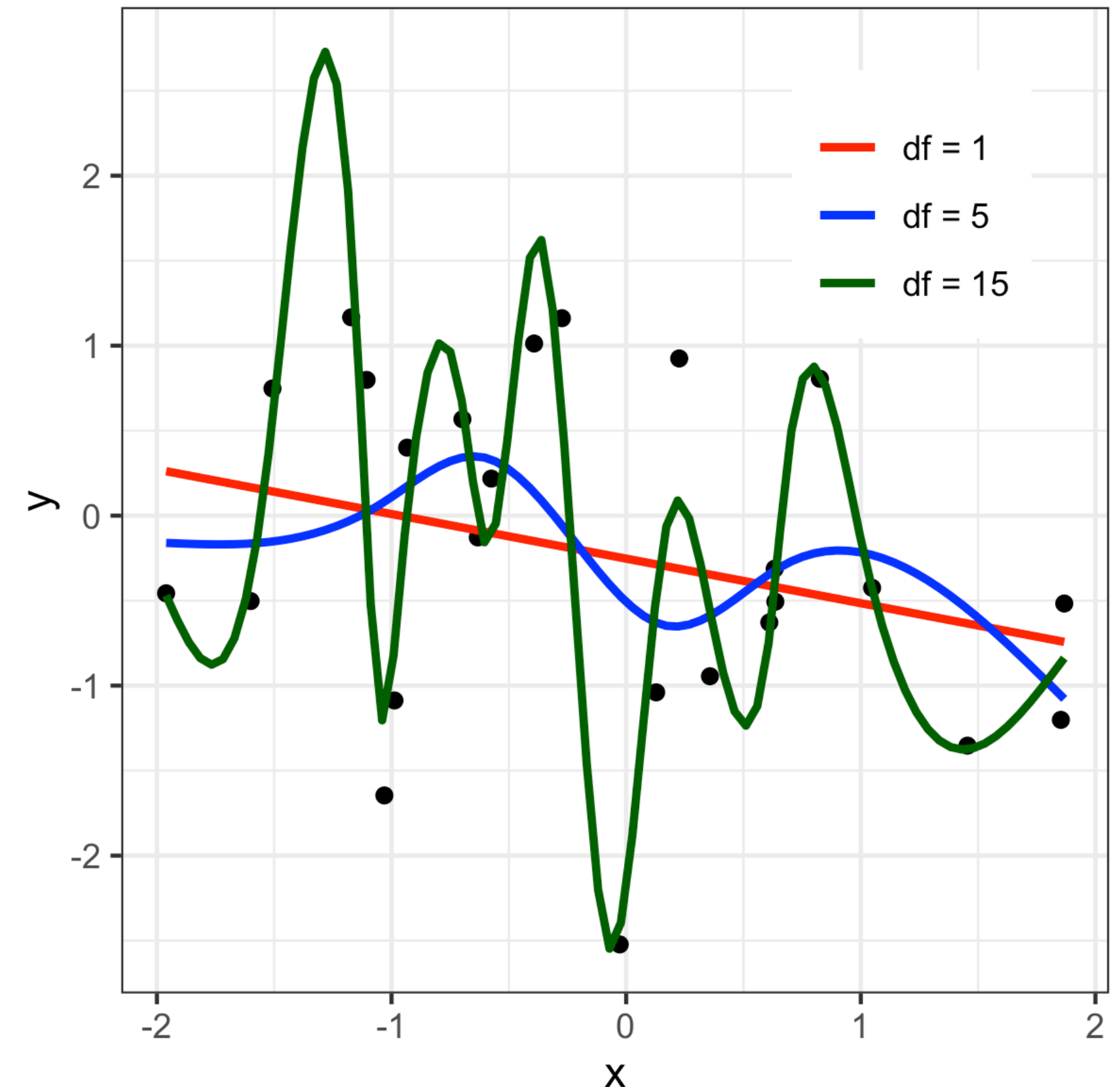
Lecture 3: Cross-validation

Lecture 4: Classification

September 30, 2021

Lecture 1: Model Complexity

- The same data can be fit with models of varying degrees of *flexibility* or *complexity*.
- Example: Smooth curve fits to data (called *splines*) are more flexible if they are more wiggly.
- Model complexity has an important effect on predictive performance:
 - Too flexible → too sensitive to noise in training data
 - Not flexible enough → can't capture the underlying trend



Fitting curves to data

Not all relationships are linear...

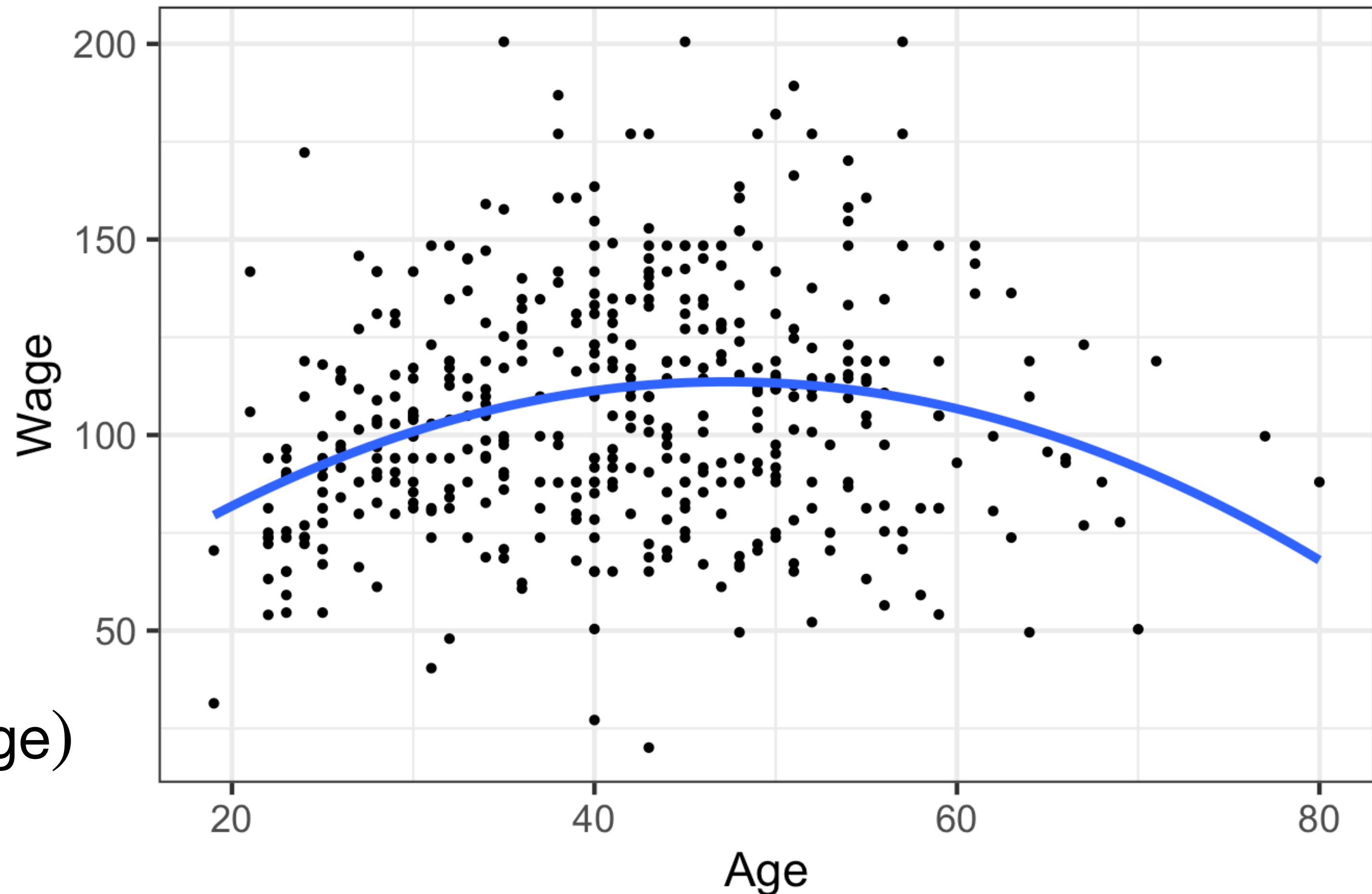
...at least in terms of the original variables.

Consider the *polynomial* model

$$\text{Wage} \approx \beta_0 + \beta_1 \text{Age} + \beta_2 \text{Age}^2.$$

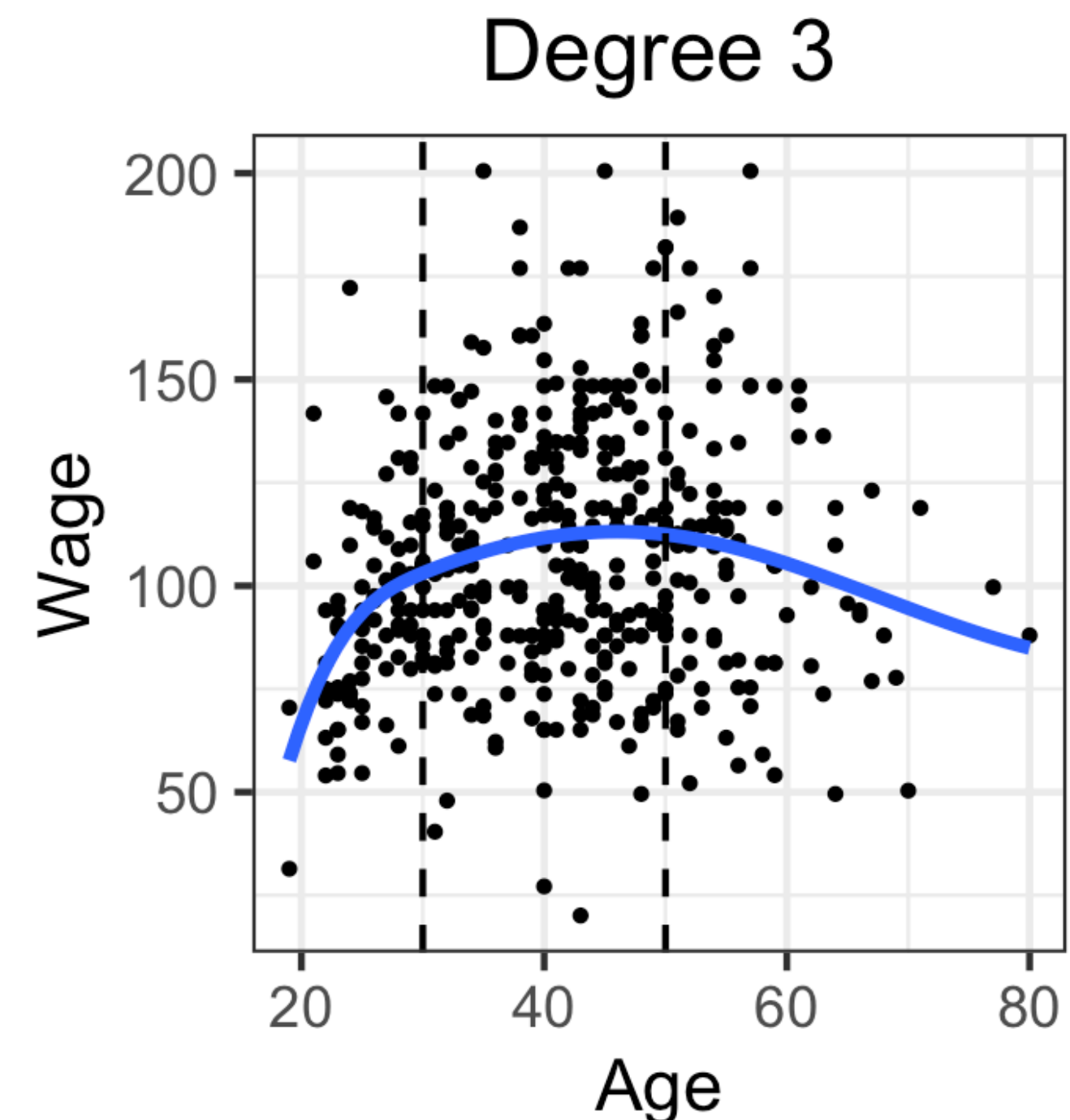
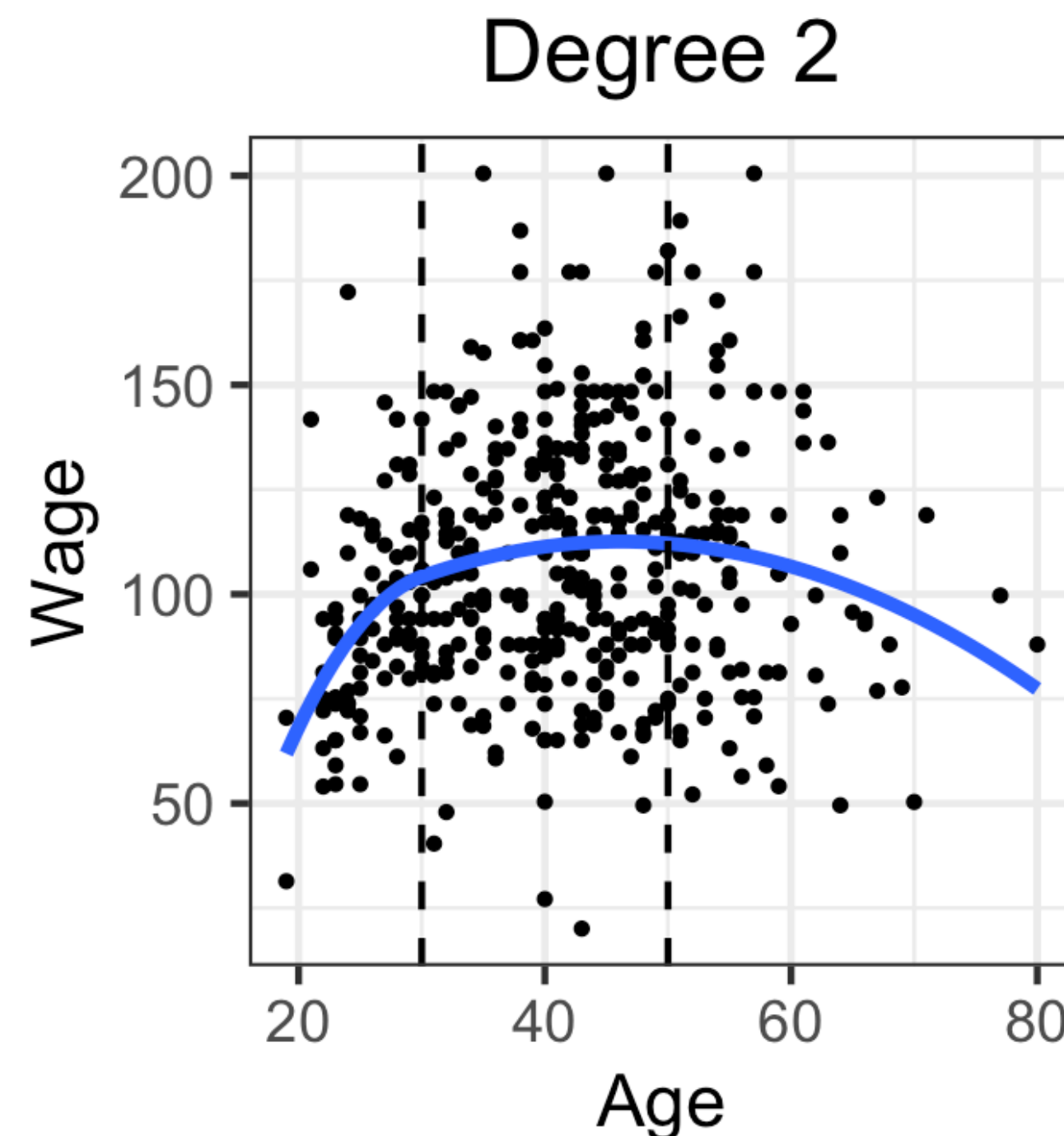
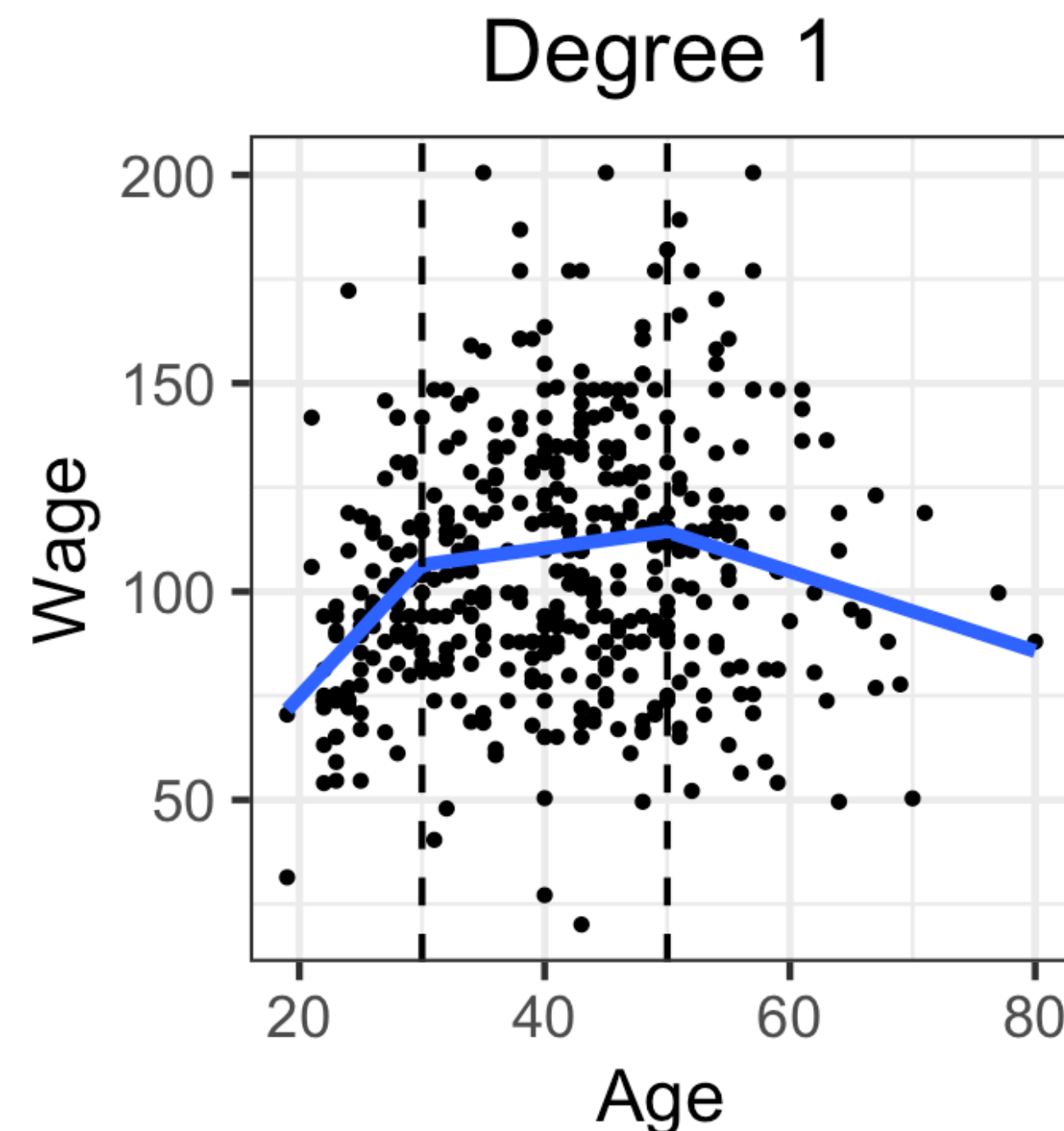
Or more generally:

$$\text{Wage} \approx \beta_1 f_1(\text{Age}) + \cdots + \beta_p f_p(\text{Age})$$



Splines: A fancier curve-fitting method

- Break range of Age into intervals separated by *knots*.
- Fit a polynomial of degree d to the data in each interval.
- “Stitch” the polynomials together to smooth transitions at knots.

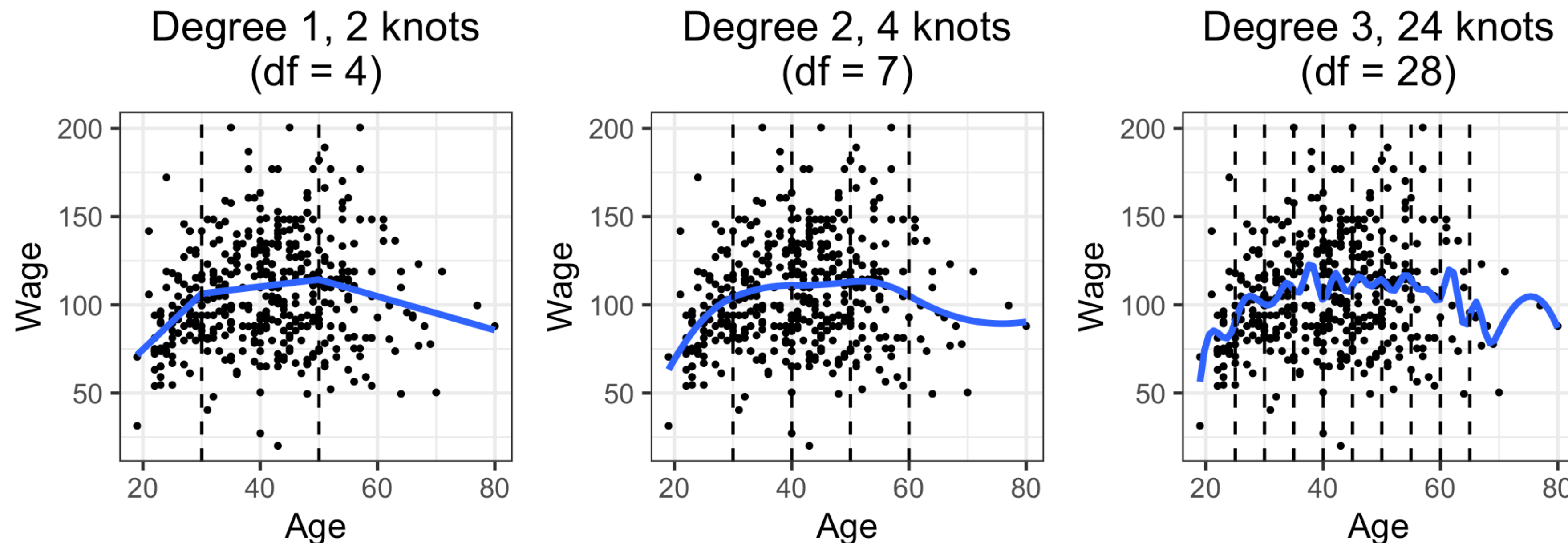


Degrees of freedom

Each spline fit can be represented in terms of a basis representation

$$\text{Wage} \approx \beta_1 f_1(\text{Age}) + \cdots + \beta_p f_p(\text{Age}).$$

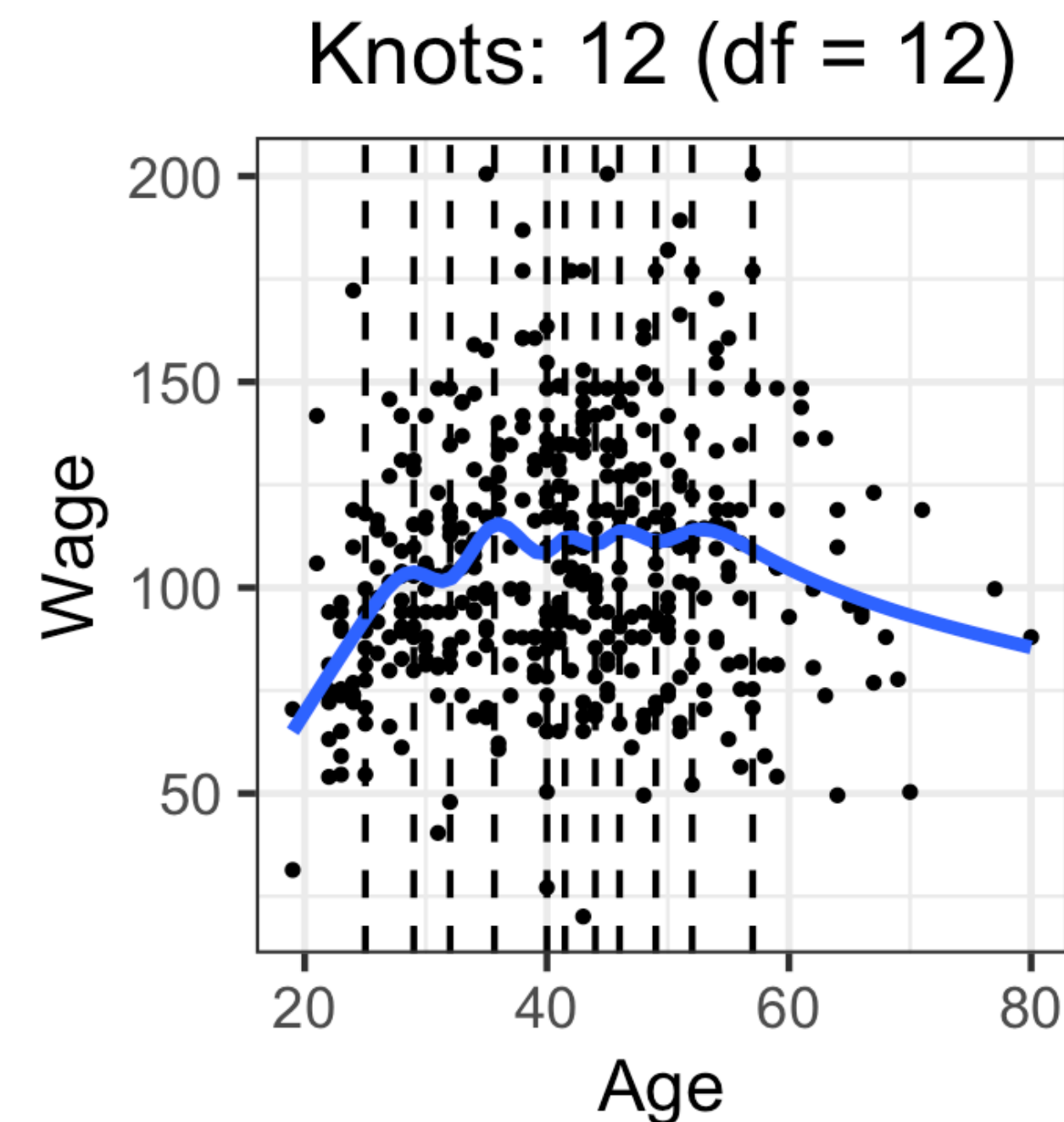
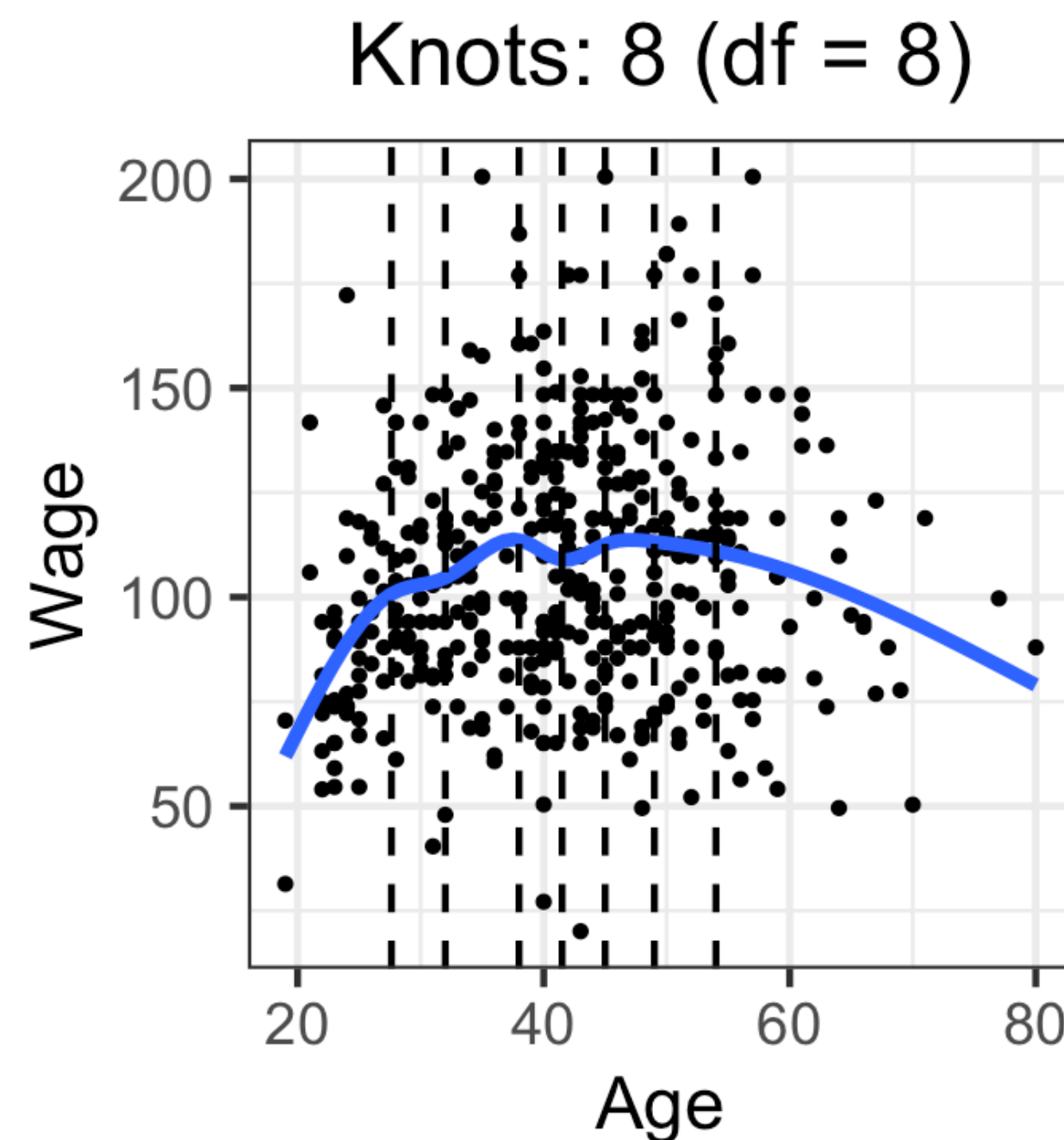
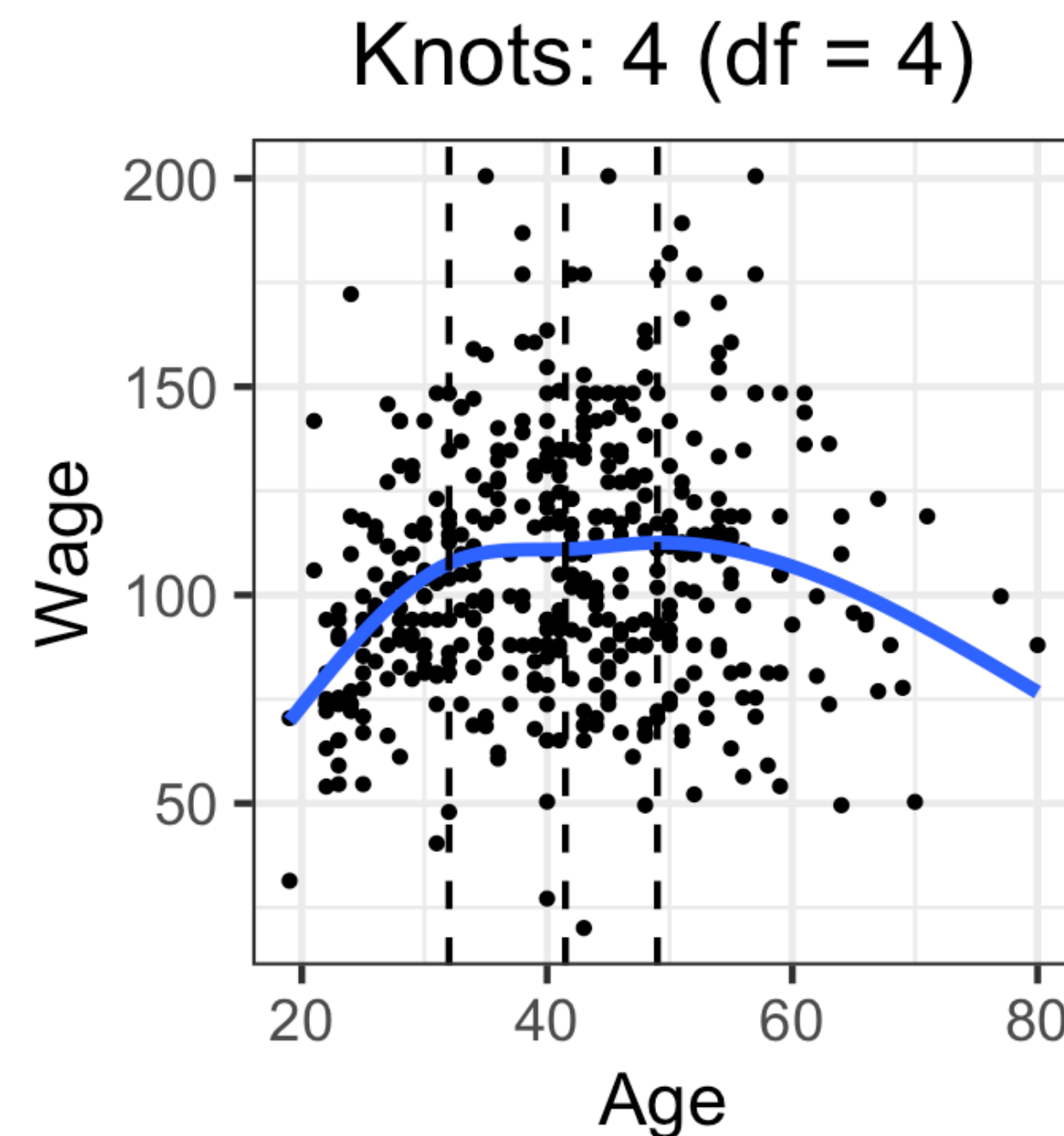
The higher p is, the more freedom the curve has to fit the data. Hence, we define *degrees of freedom (df)* = p .



Natural cubic splines

Like regular splines, except:

- Degree is always cubic
- Fit must be linear as you approach edges of the data
- Knots are chosen automatically at quantiles of the data



Recall: Prediction performance

You have **training data** $(X_1^{\text{train}}, Y_1^{\text{train}}), \dots, (X_n^{\text{train}}, Y_n^{\text{train}})$, based on which you construct a predictive model \hat{f} such that, hopefully, $Y \approx \hat{f}(X)$.

Will deploy \hat{f} on **test data** $X_1^{\text{test}}, \dots, X_N^{\text{test}}$ to guess $\hat{Y}_i^{\text{test}} = \hat{f}(X_i^{\text{test}})$ for each i .

Each X_i^{test} comes with a response Y_i^{test} , unknown to the predictive model.

Prediction quality: extent to which $Y_i^{\text{test}} \approx \hat{f}(X_i^{\text{test}})$, e.g. **mean squared test error**:

$$\text{Test error of } \hat{f} = \frac{1}{N} \sum_{i=1}^N (Y_i^{\text{test}} - \hat{f}(X_i^{\text{test}}))^2.$$

Contrast test error with the training error:

$$\text{Training error of } \hat{f} = \frac{1}{n} \sum_{i=1}^n (Y_i^{\text{train}} - \hat{f}(X_i^{\text{train}}))^2.$$

Model complexity impacts prediction performance

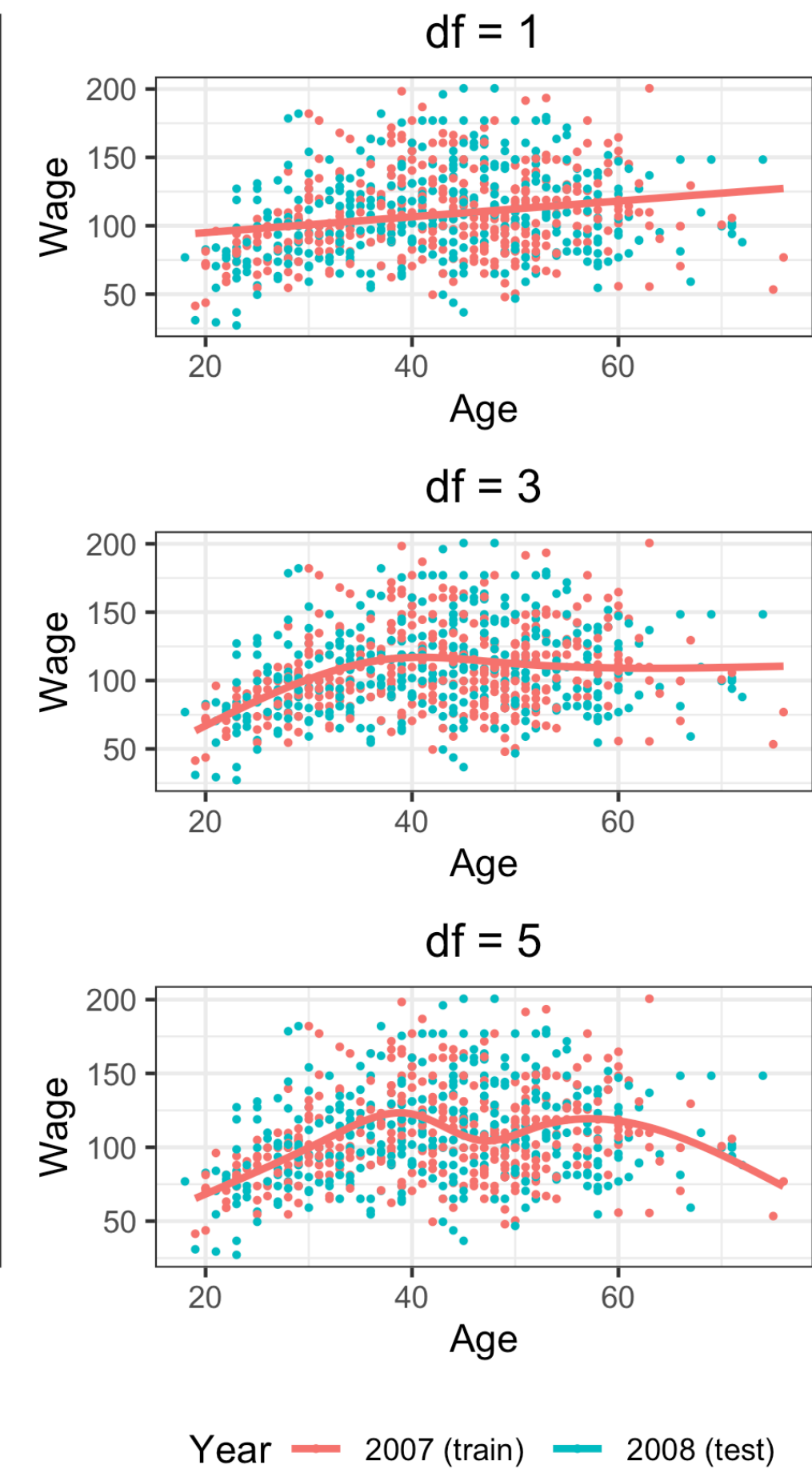
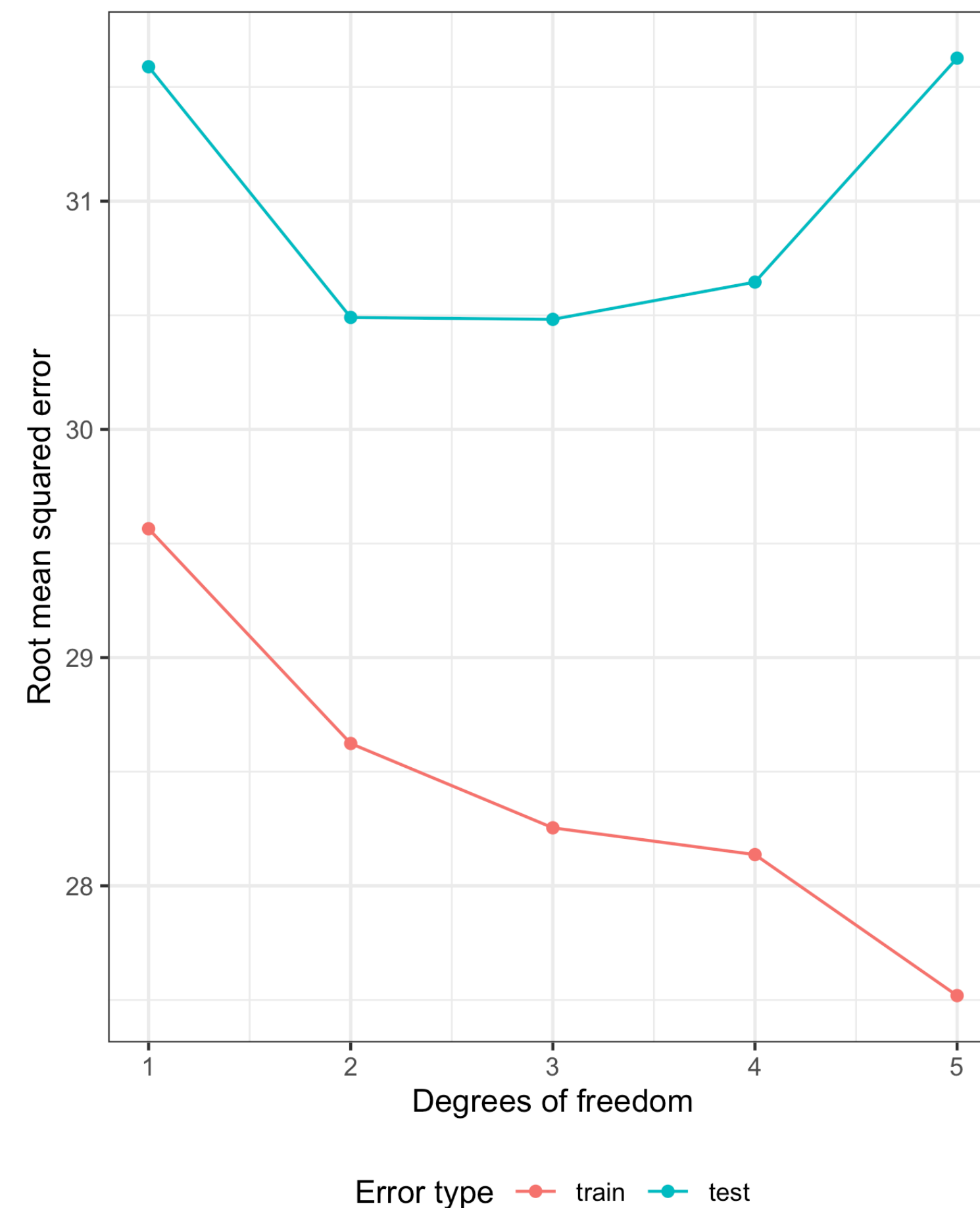
Model complexity: how closely the model \hat{f} fits the training data:

$$Y_i^{\text{train}} = f(X_i^{\text{train}}) + \epsilon_i.$$

During training, \hat{f} picks up on patterns in both f (the signal) and ϵ_i (the noise).

Training error of \hat{f} decreases as we increase model complexity, but **test error** will be high if model complexity is too low or too high.

Training error is an underestimate of the test error, especially as the model complexity increases (**overfitting**).



Lecture 2: Bias-variance tradeoff

What drives model complexity?

Problem parameters

- Sample size
- Noise level
- Fitted model complexity (number of parameters)
- True model complexity

Phenomena

- Model bias: extent to which model unable to capture the truth
- Overfitting: extent to which the fit is sensitive to noise in training data
- Irreducible error: noise in test points that is impossible to predict

Expected test error

Given a fitted \hat{f} and a test set $(X_1^{\text{test}}, Y_1^{\text{test}}), \dots, (X_N^{\text{test}}, Y_N^{\text{test}})$, recall that

$$\text{Test error of } \hat{f} = \frac{1}{N} \sum_{i=1}^N (Y_i^{\text{test}} - \hat{Y}_i^{\text{test}})^2 = \frac{1}{N} \sum_{i=1}^N (Y_i^{\text{test}} - \hat{f}(X_i^{\text{test}}))^2.$$

The test error is a random function of the test set and the training set.

Define the **expected test error (ETE)** of a prediction rule as

$$\text{ETE} = \frac{1}{N} \sum_{i=1}^N \mathbb{E}[(Y_i^{\text{test}} - \hat{f}(X_i^{\text{test}}))^2].$$

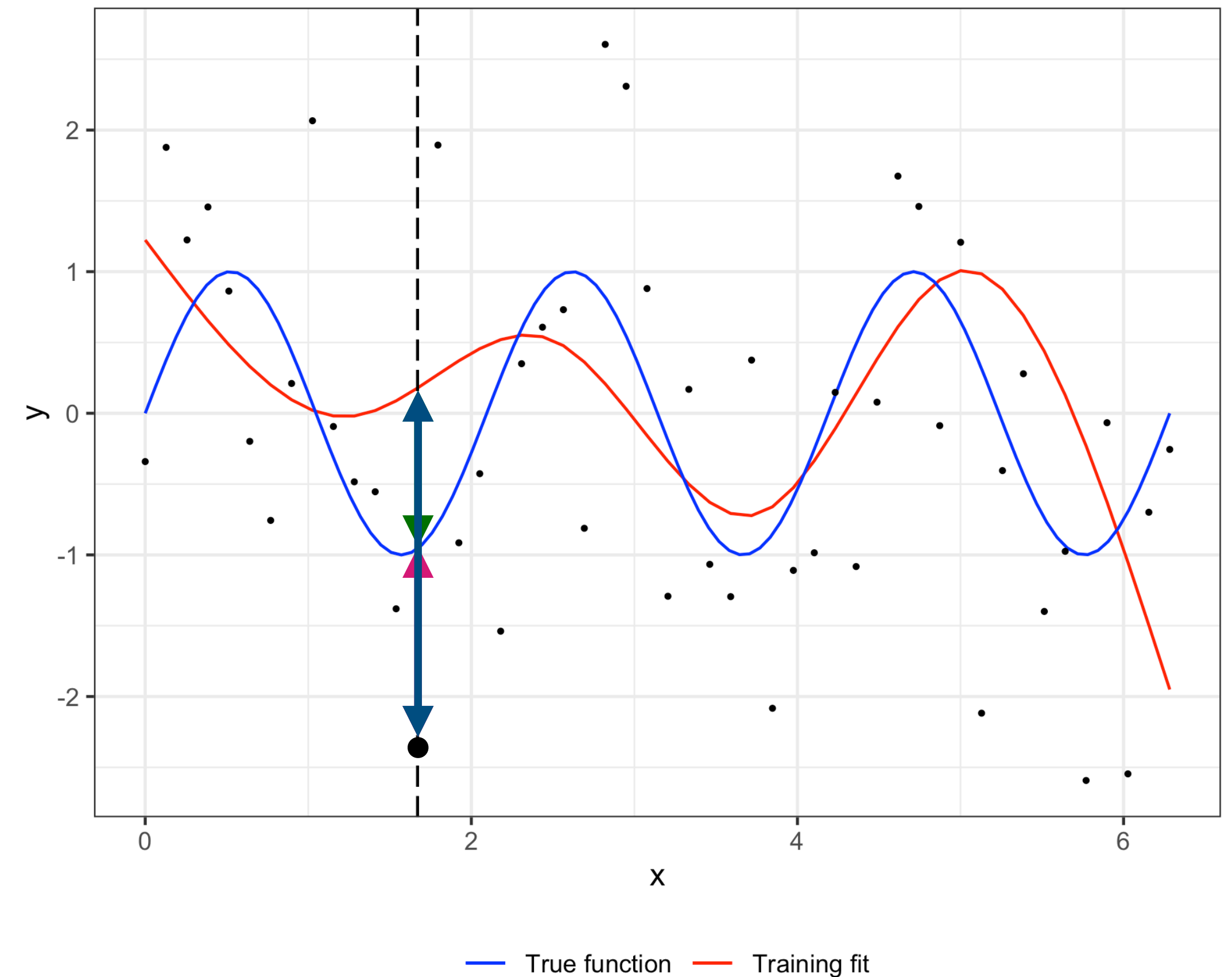
The ETE is easier to understand theoretically.

Dissecting the expected test error

Suppose $Y = f(X) + \epsilon$, $\epsilon \sim N(0, \sigma^2)$.

Then,

$$\begin{aligned} \text{ETE}_i &= \mathbb{E}[(Y_i^{\text{test}} - \hat{f}(X_i^{\text{test}}))^2] \\ &= \mathbb{E}[(f(X_i^{\text{test}}) + \epsilon_i^{\text{test}} - \hat{f}(X_i^{\text{test}}))^2] \\ &= \mathbb{E}[(\underbrace{f(X_i^{\text{test}}) - \hat{f}(X_i^{\text{test}})}_{\text{Bias}})^2] + \underbrace{\sigma^2}_{\text{Variance}} \end{aligned}$$



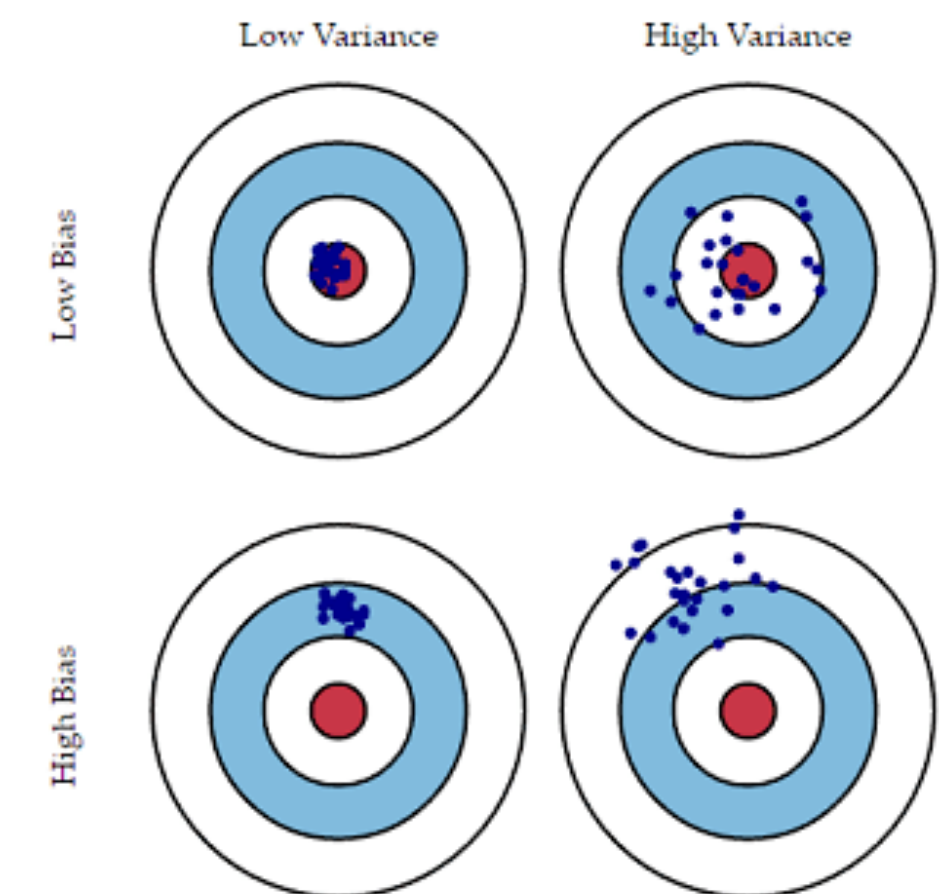
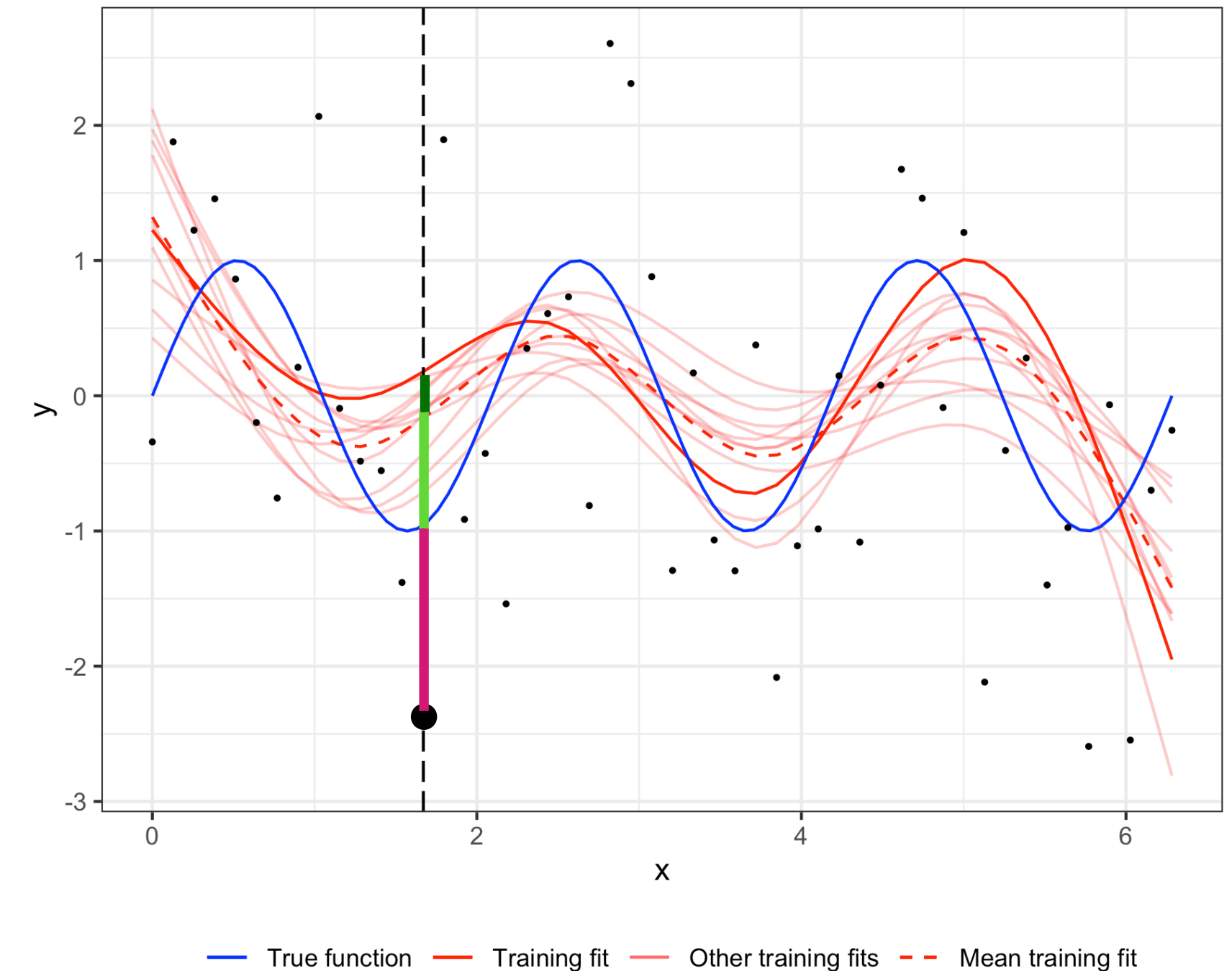
Dissecting the expected test error

$$\begin{aligned}
 \text{ETE}_i &= \mathbb{E}[(f(X_i^{\text{test}}) - \hat{f}(X_i^{\text{test}}))^2] + \sigma^2 \\
 &= (f(X_i^{\text{test}}) - \text{Ave}(\hat{f}(X_i^{\text{test}}))^2 + \text{Var}[\hat{f}(X_i^{\text{test}})] + \sigma^2 \\
 &= \text{Bias}_i^2 + \text{Variance}_i + \text{Irreducible error}_i
 \end{aligned}$$

Averaging over i , we get

$$\begin{aligned}
 \text{ETE} &= \frac{1}{N} \sum_{i=1}^N \mathbb{E}[(Y_i^{\text{test}} - \hat{f}(X_i^{\text{test}}))^2] \\
 &= \frac{1}{N} \sum_{i=1}^N \text{Bias}_i^2 + \frac{1}{N} \sum_{i=1}^N \text{Variance}_i^2 + \sigma^2 \\
 &= \text{"Bias"} + \text{"Variance"} + \text{"Irreducible error"}
 \end{aligned}$$

This is the **bias-variance decomposition**.

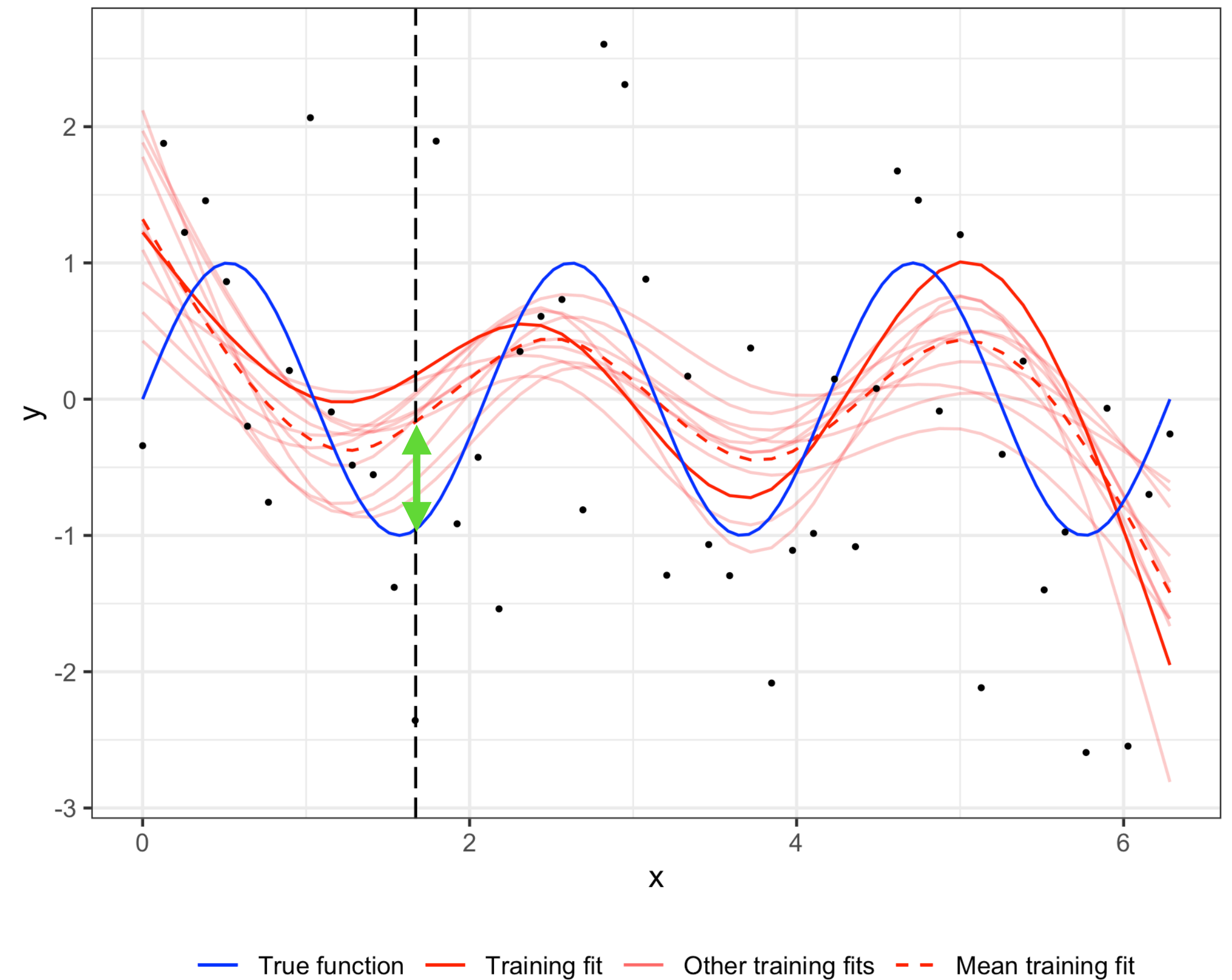


Understanding bias

$$\text{Bias}_i = \text{Ave}(\hat{f}(X_i^{\text{test}})) - f(X_i^{\text{test}})$$

Bias reflects the distance from the average fitted model to the true trend.

Adding model complexity reduces bias.



Understanding variance

$$\text{Variance}_i = \underline{\mathbb{E}[(\hat{f}(X_i^{\text{test}}) - \text{Ave}(\hat{f}(X_i^{\text{test}})))^2]}$$

Variance is the wobbling of the model fit due to the randomness in the training data.

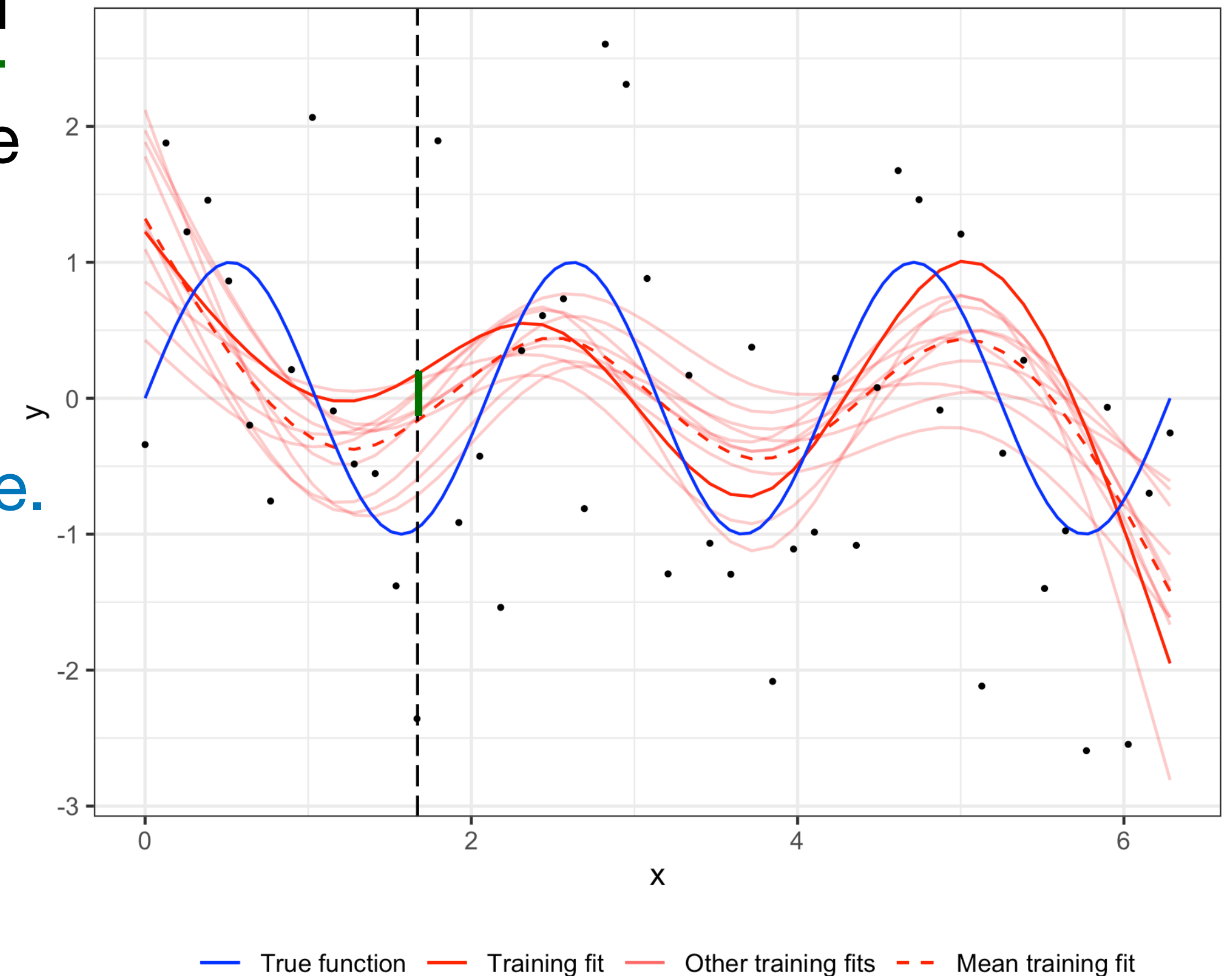
Variance is a consequence of **overfitting**.

Adding model complexity increases variance.

In linear models,

$$\text{Variance} = \frac{1}{n} \sum_{i=1}^n \text{Variance}_i = \frac{\sigma^2 p}{n}$$

(assuming $n = N$ and $X_i^{\text{test}} = X_i^{\text{train}}$)



The bias-variance tradeoff

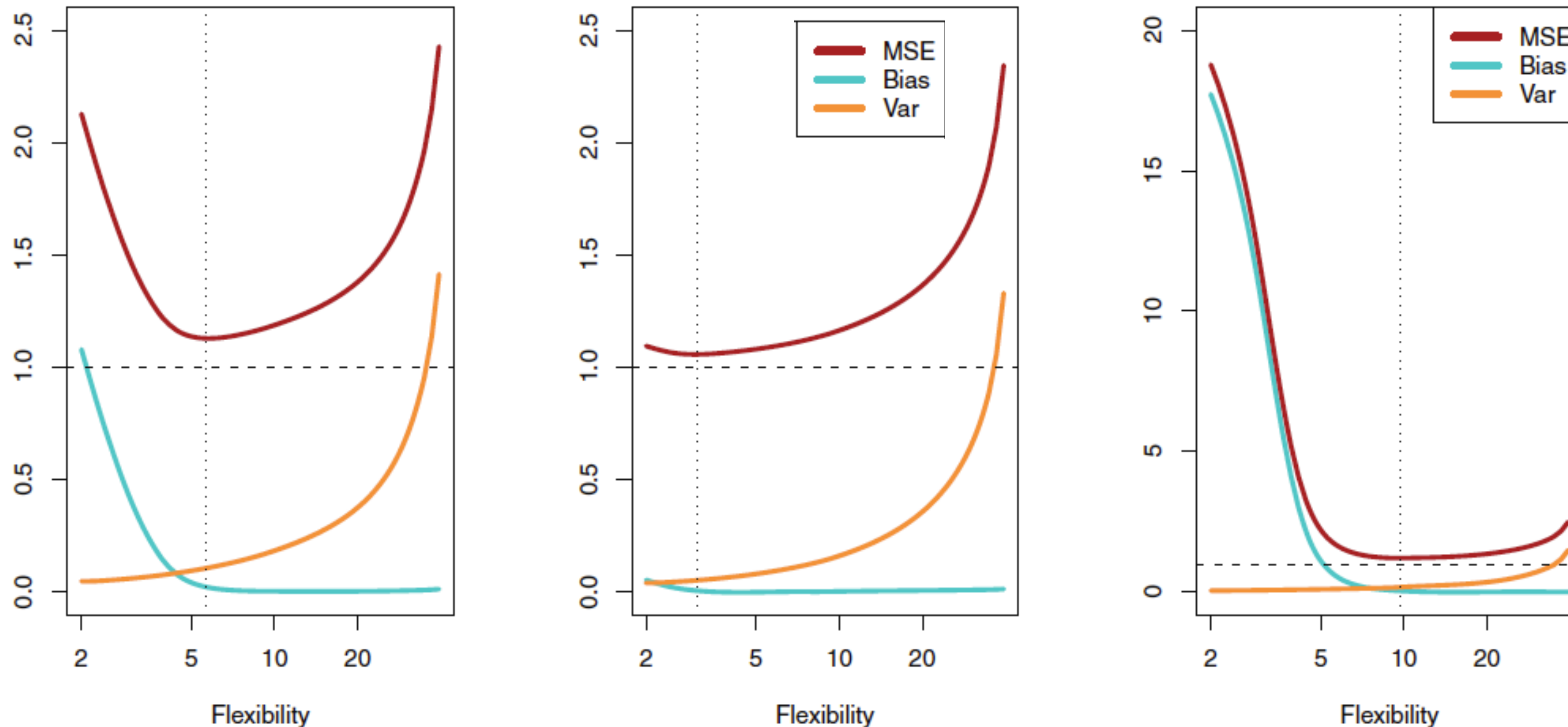
Recall that

$$\text{ETE} = \text{“Bias”} + \text{“Variance”} + \text{“Irreducible error”}$$

- Adding model complexity reduces bias
- Adding model complexity increases variance

When varying model complexity, there is a tradeoff between bias and variance.

Navigating the bias-variance tradeoff



The shapes of these curves differ based on the problem parameters.

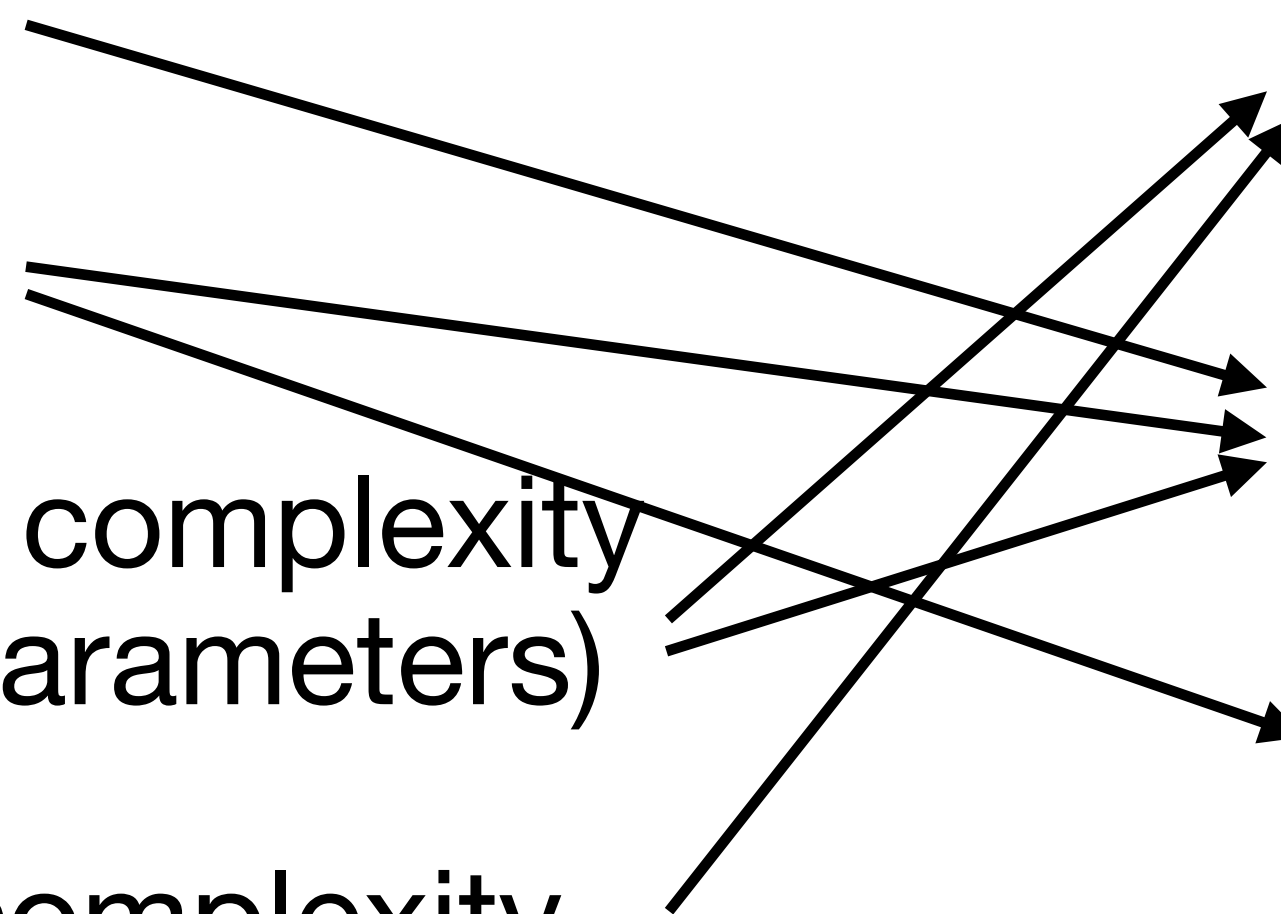
What drives test error?

Problem parameters

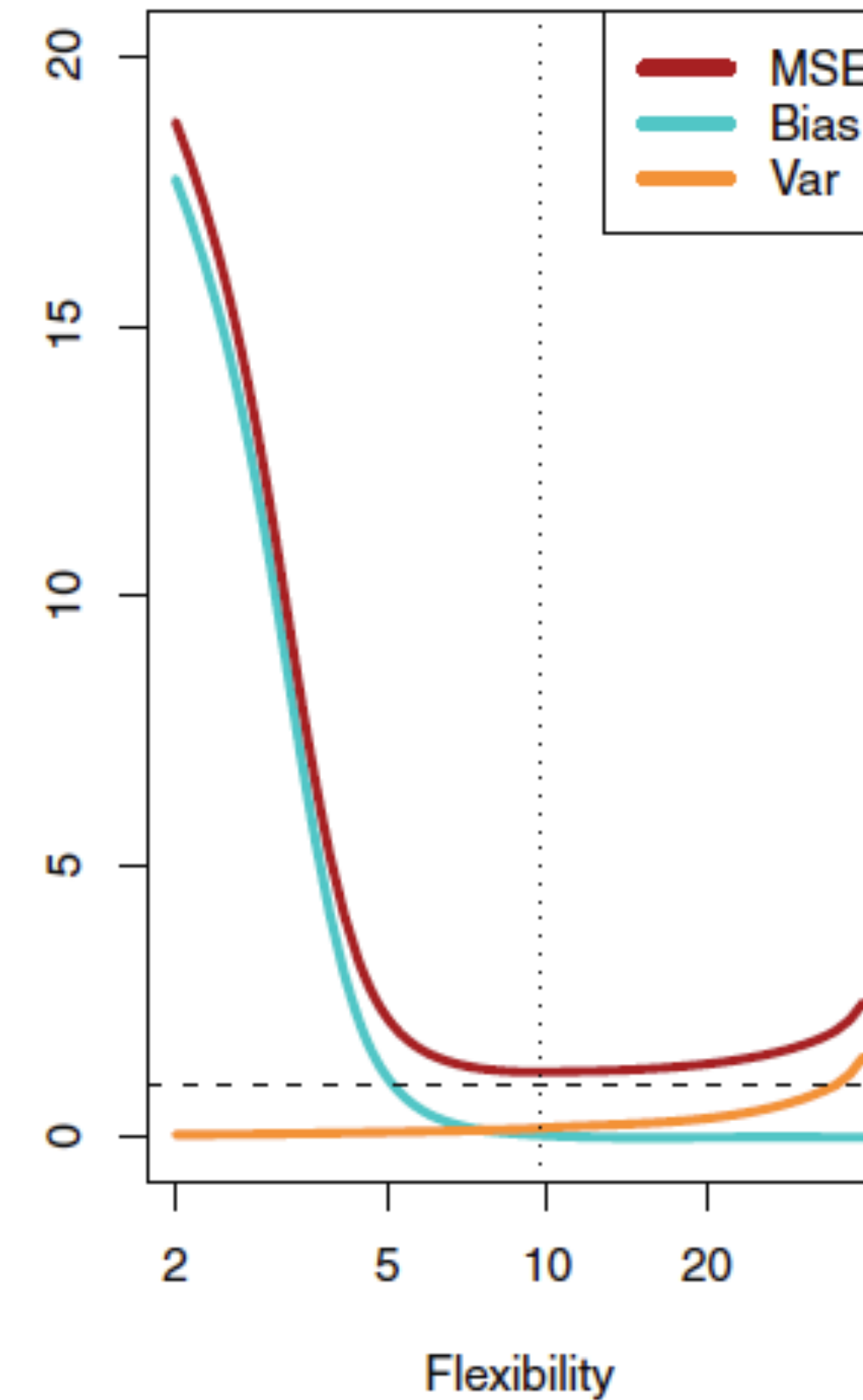
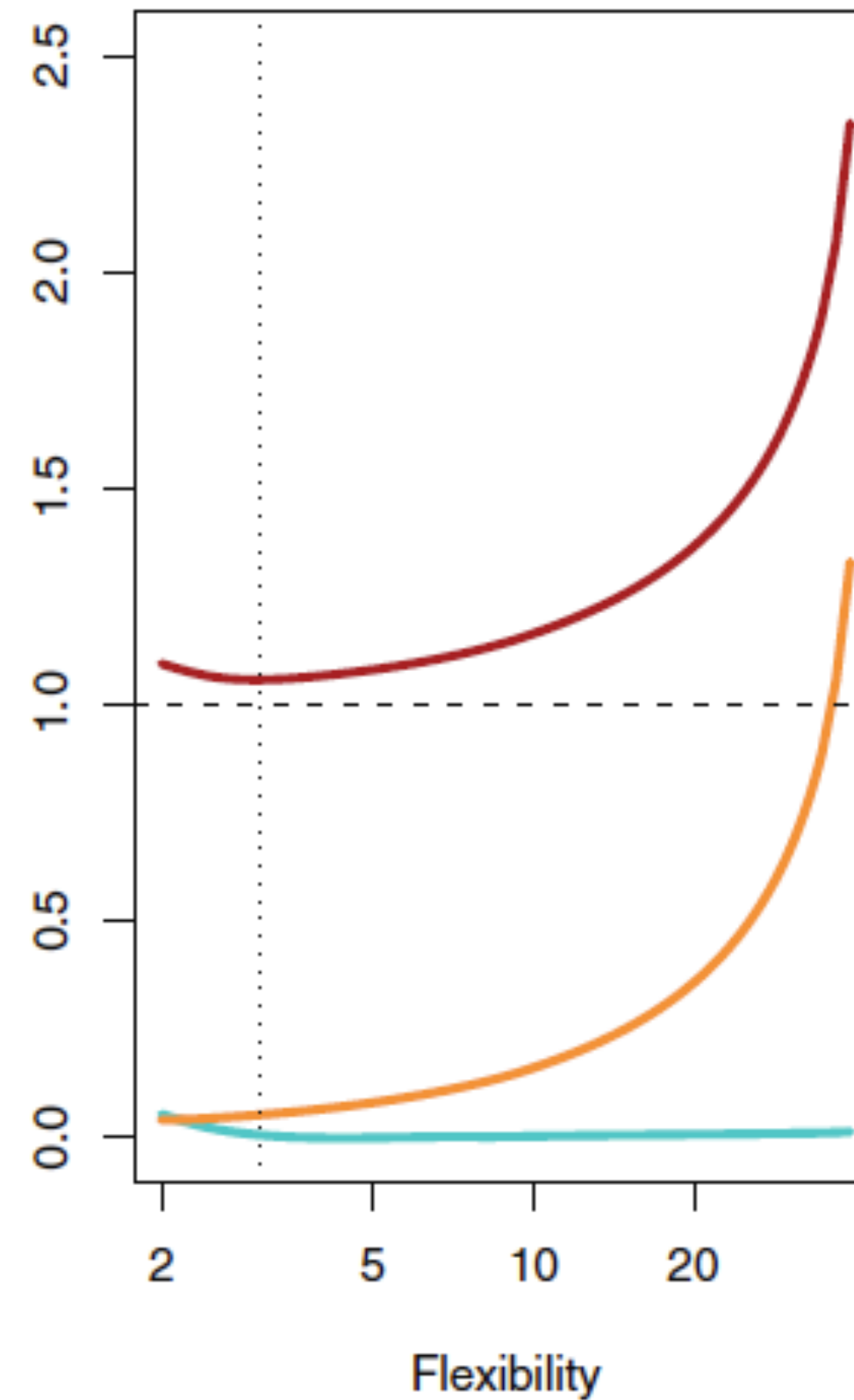
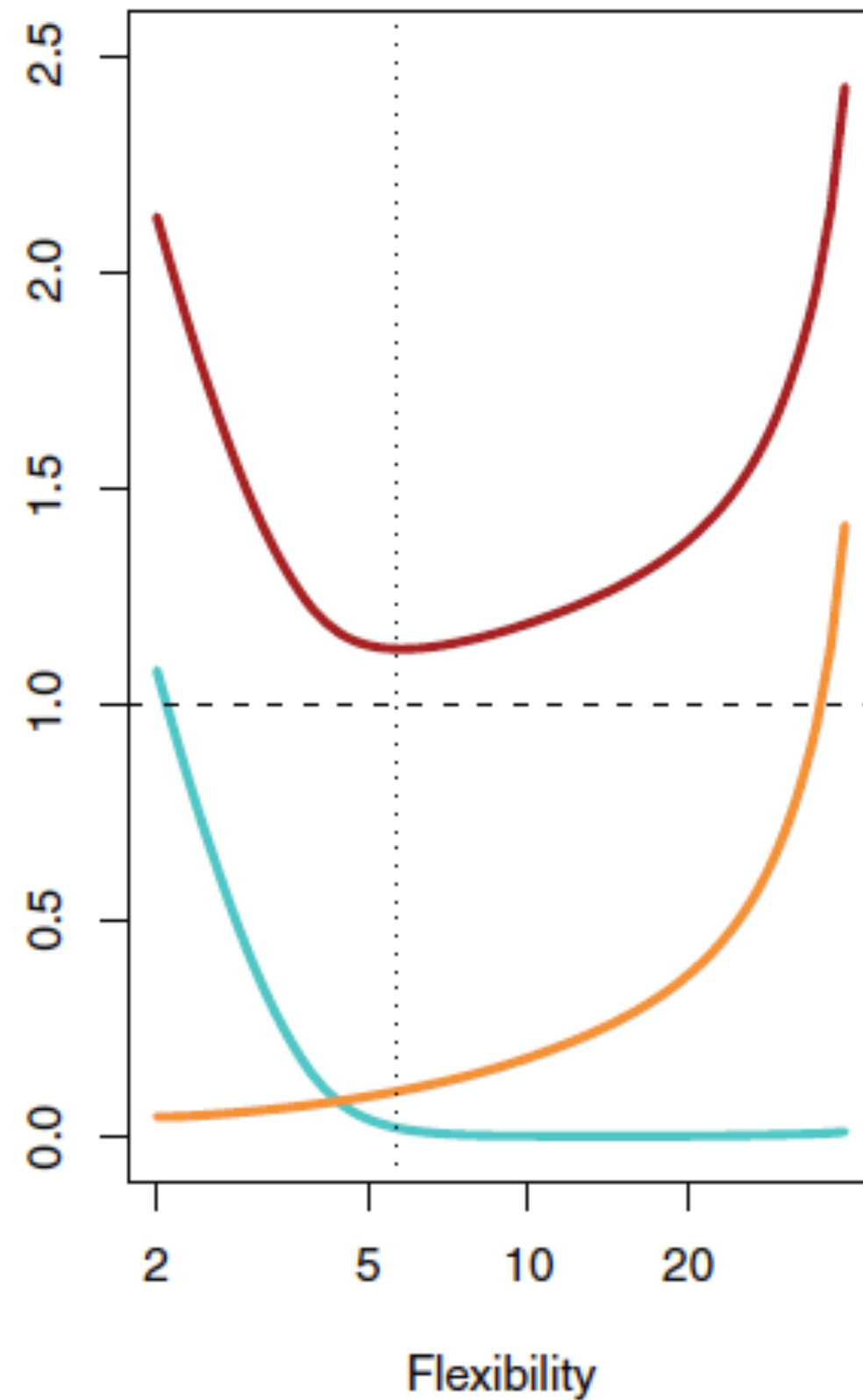
- Sample size
- Noise level
- Fitted model complexity (number of parameters)
- True model complexity

Phenomena

- Model bias: extent to which model unable to capture the truth
- +
• Overfitting: extent to which the fit is sensitive to noise in training data
- +
• Irreducible error: noise in test points that is impossible to predict
= ETE

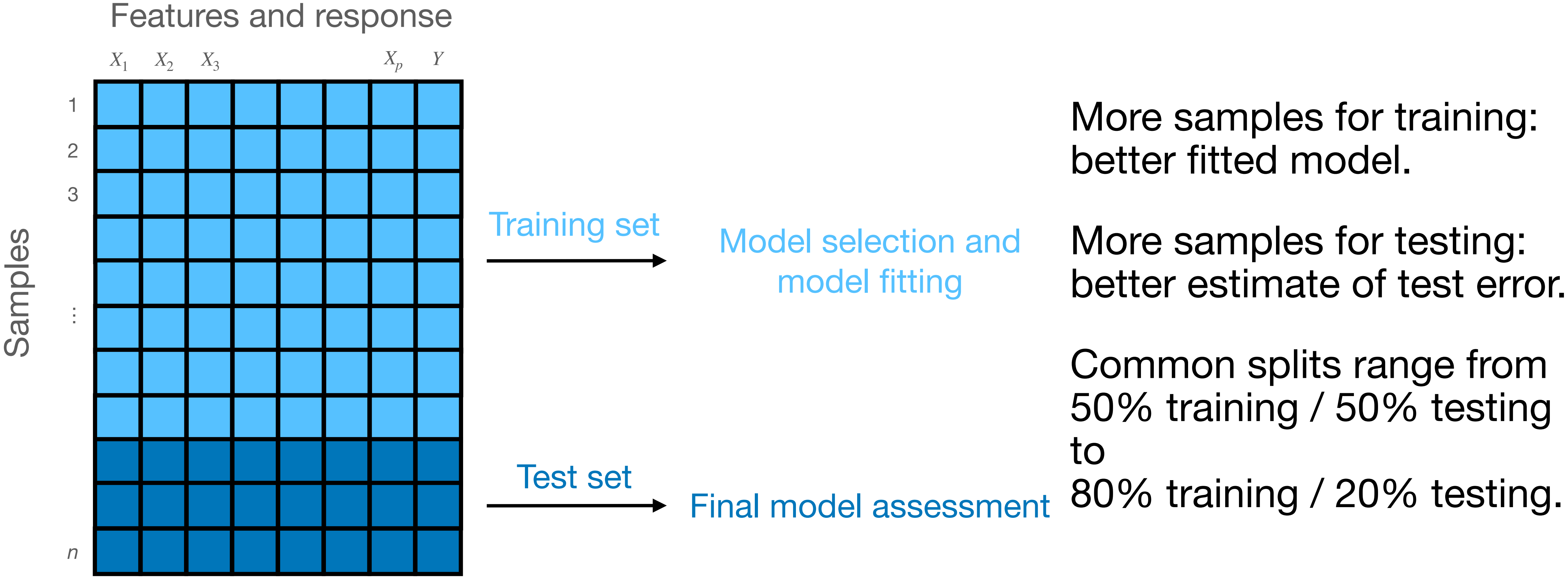


Lecture 3: Cross-validation

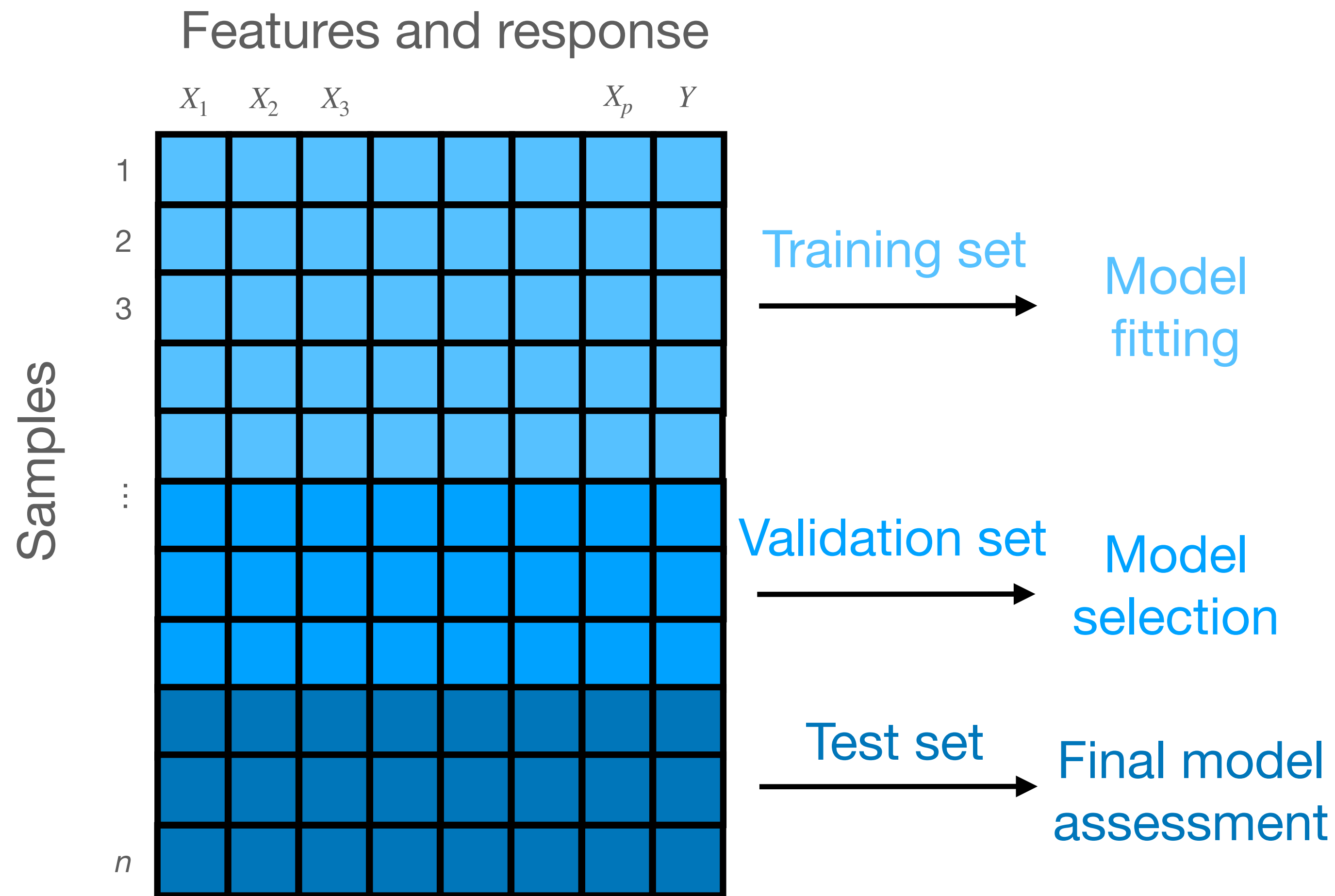


- How do we estimate the test error for model selection?
- How do we estimate the test error for final model assessment?

Estimating test error for final model assessment



Validation set approach for model selection



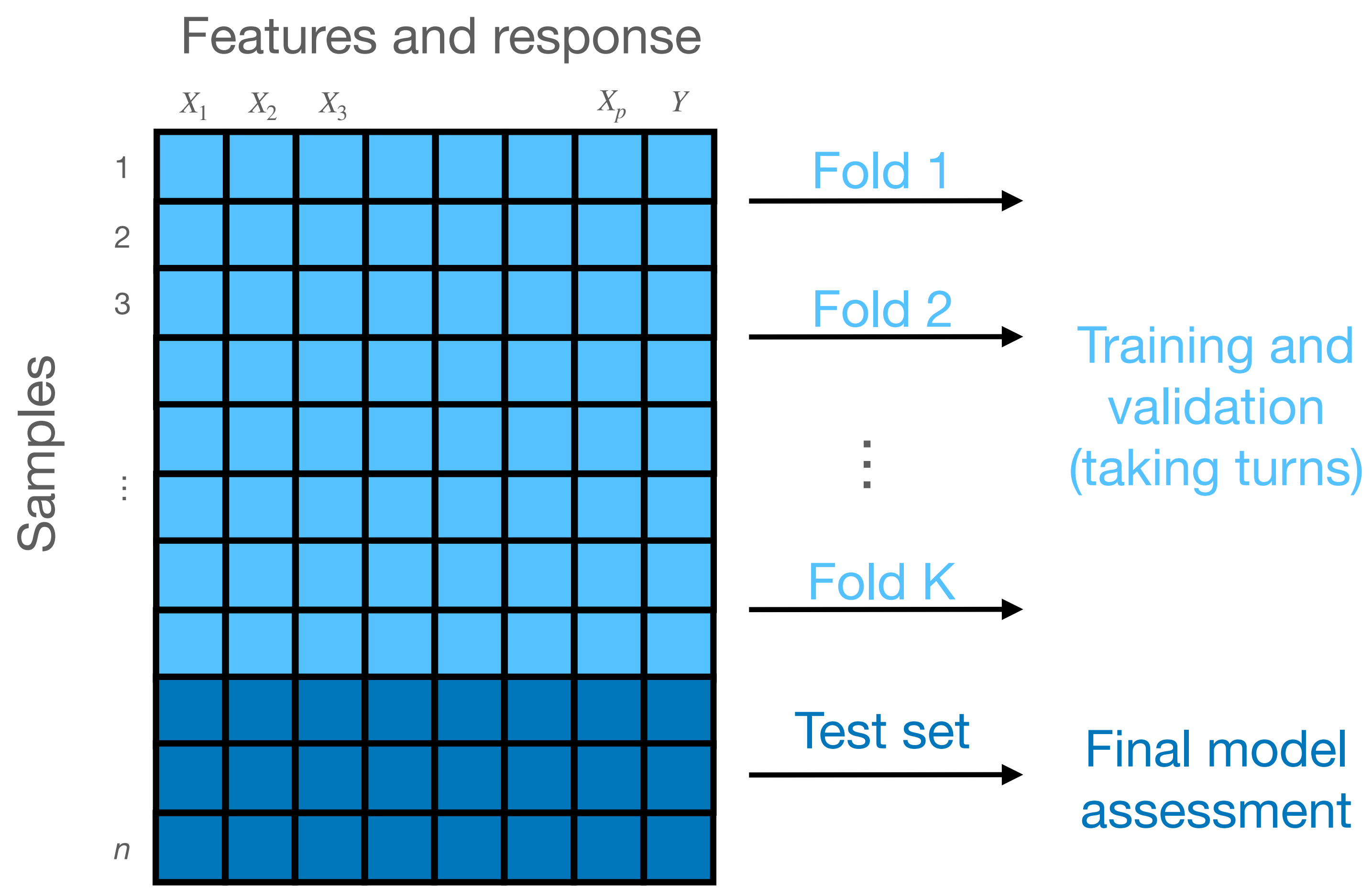
1. Fit models of varying complexity to training set
2. Estimate test error for each model on validation set
3. Choose model complexity to minimize validation error
4. Refit this model on combined training and validation sets
5. Evaluate the final model on the test set

Drawback of validation set approach

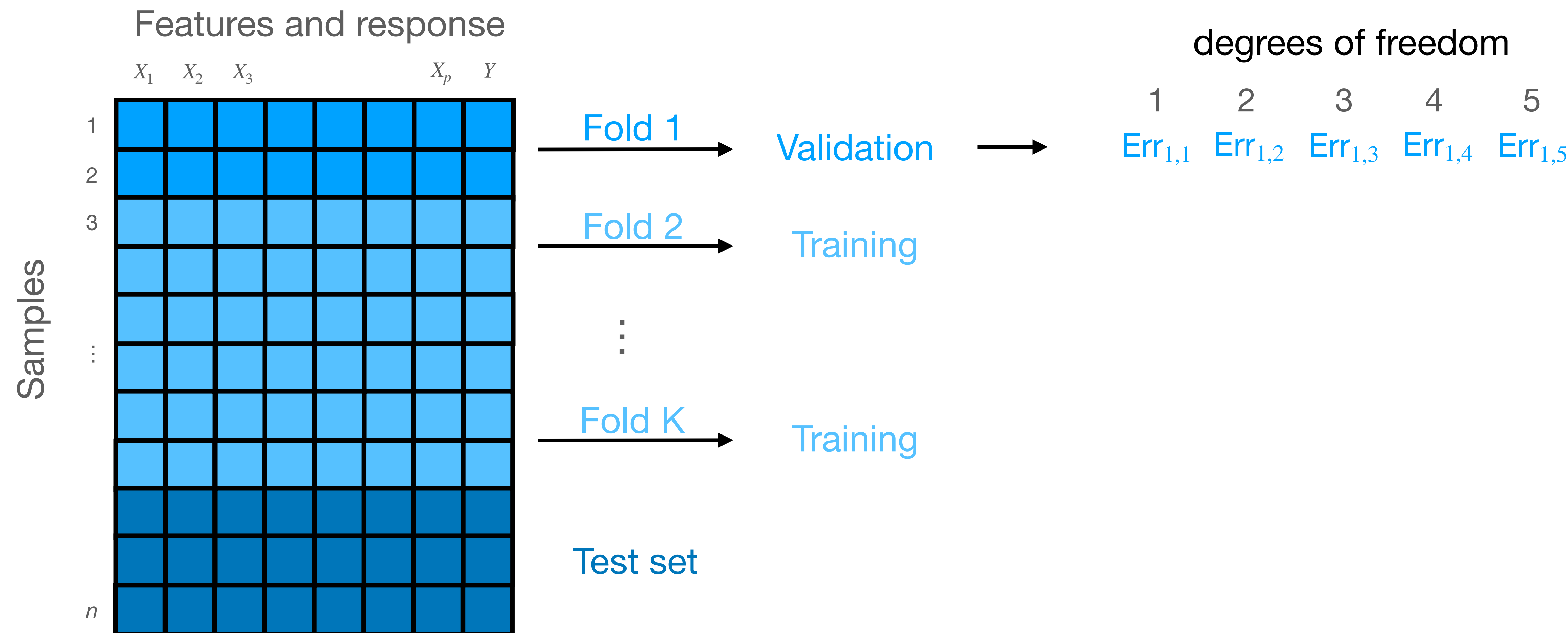
The validation set approach does not make efficient use of samples.

Using a small validation set can lead to a suboptimal model complexity choice.

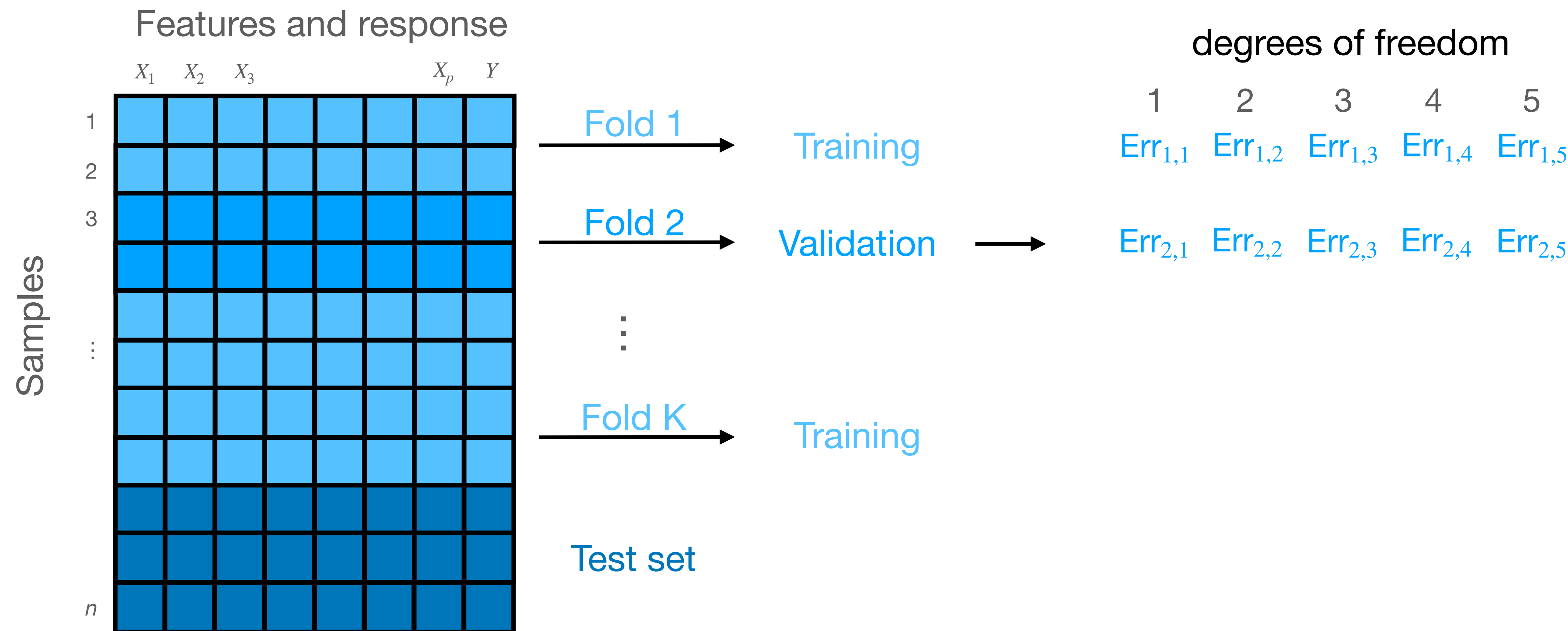
Cross-validation for model selection



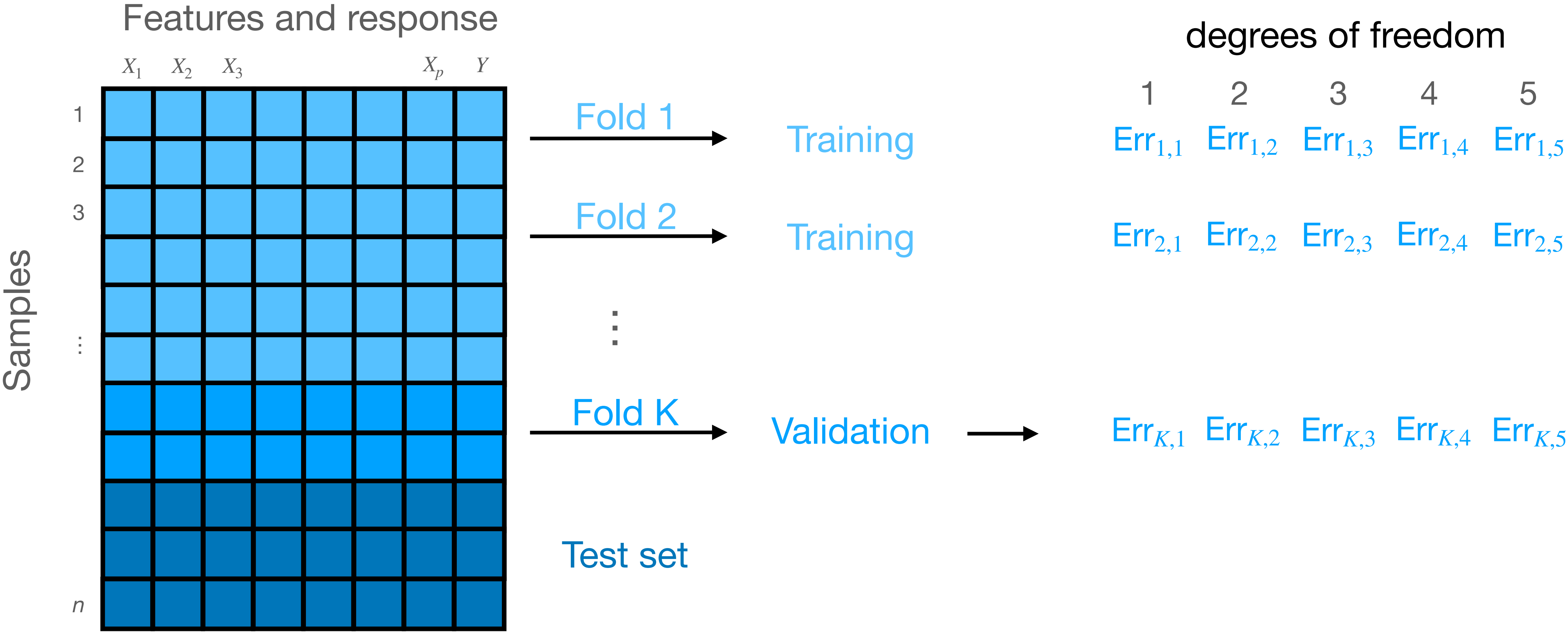
Cross-validation for model selection



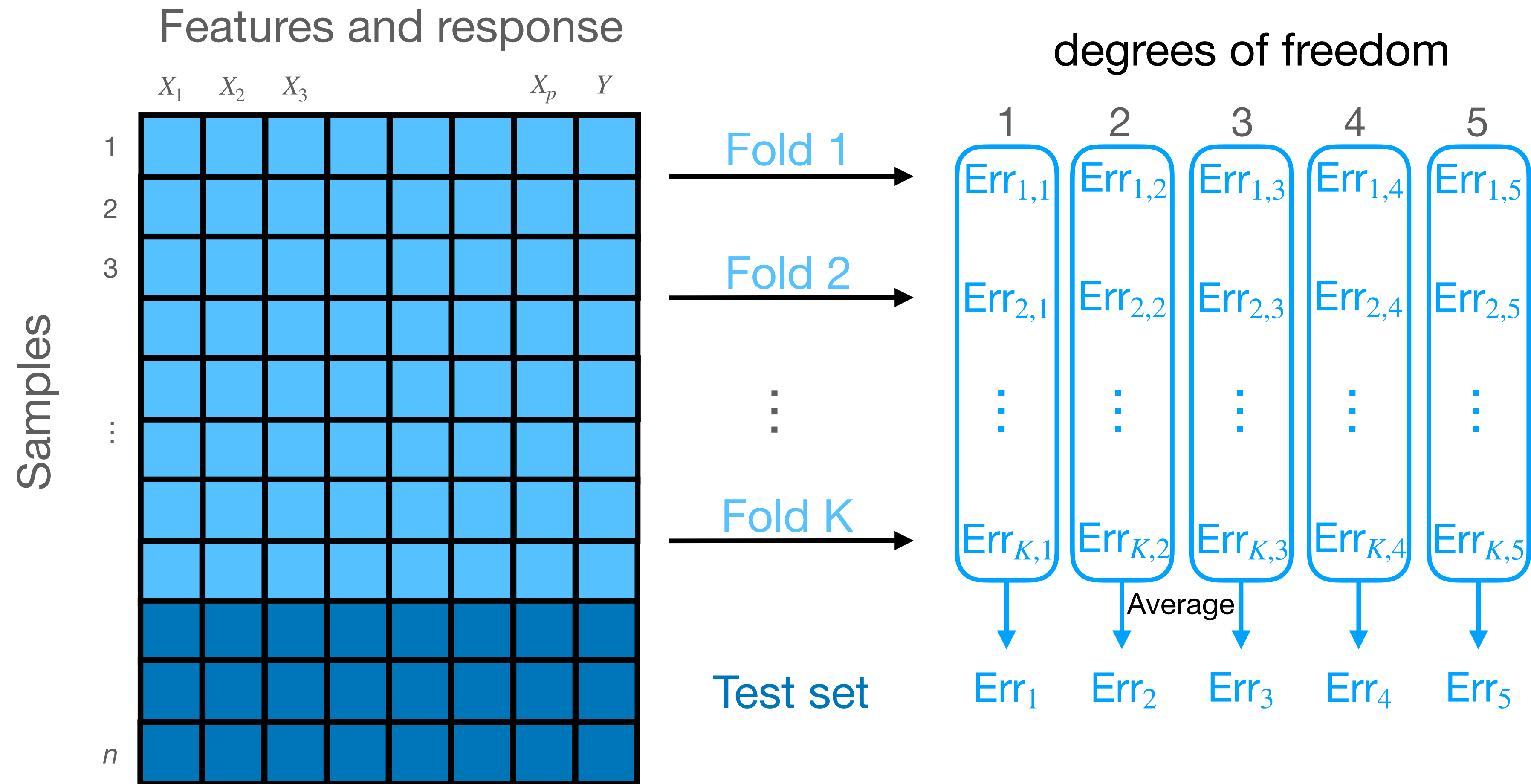
Cross-validation for model selection



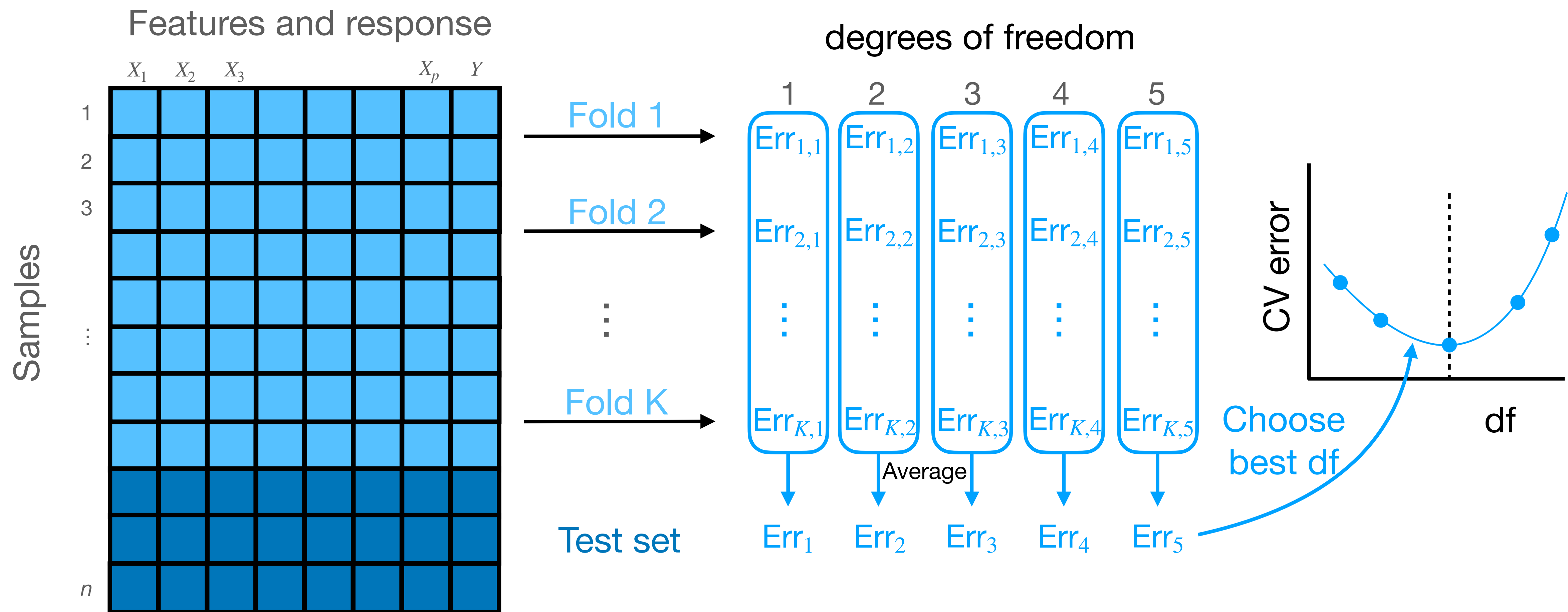
Cross-validation for model selection



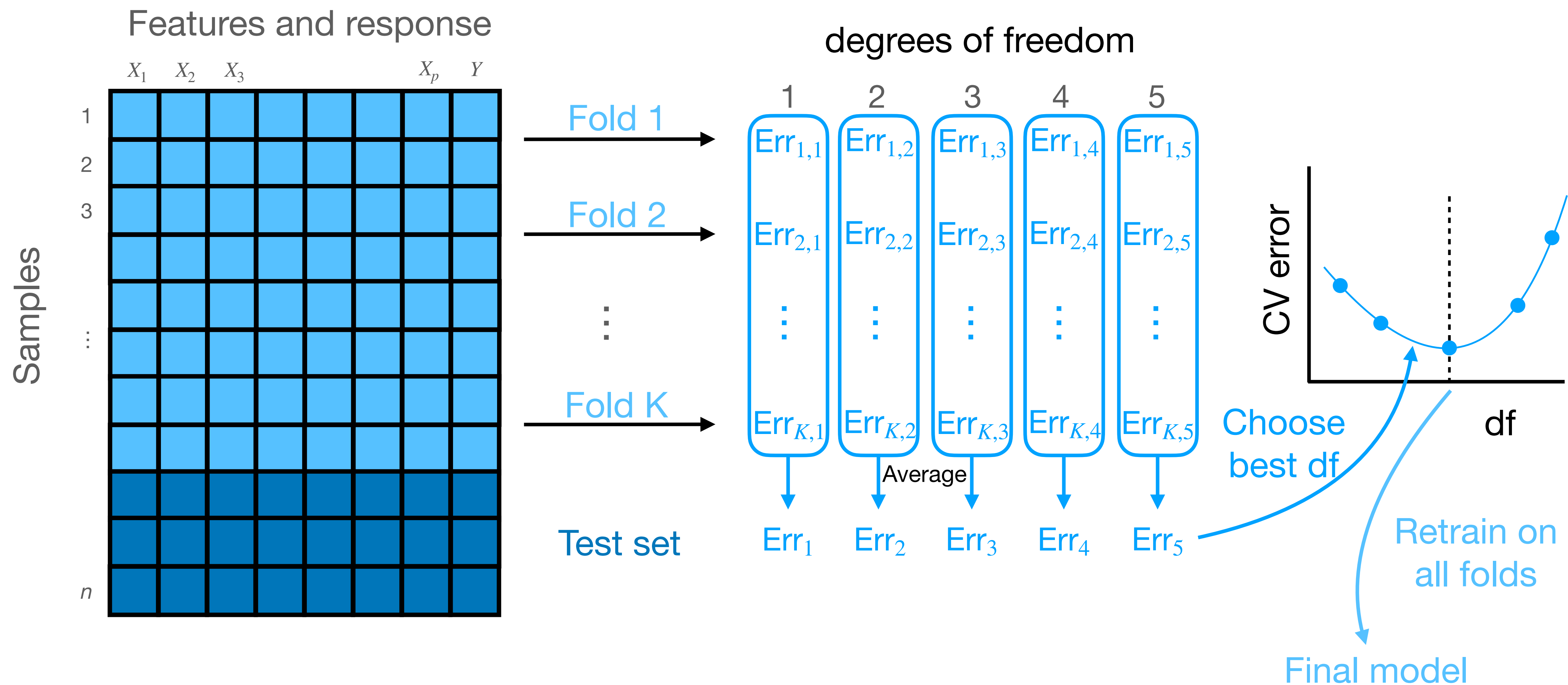
Cross-validation for model selection



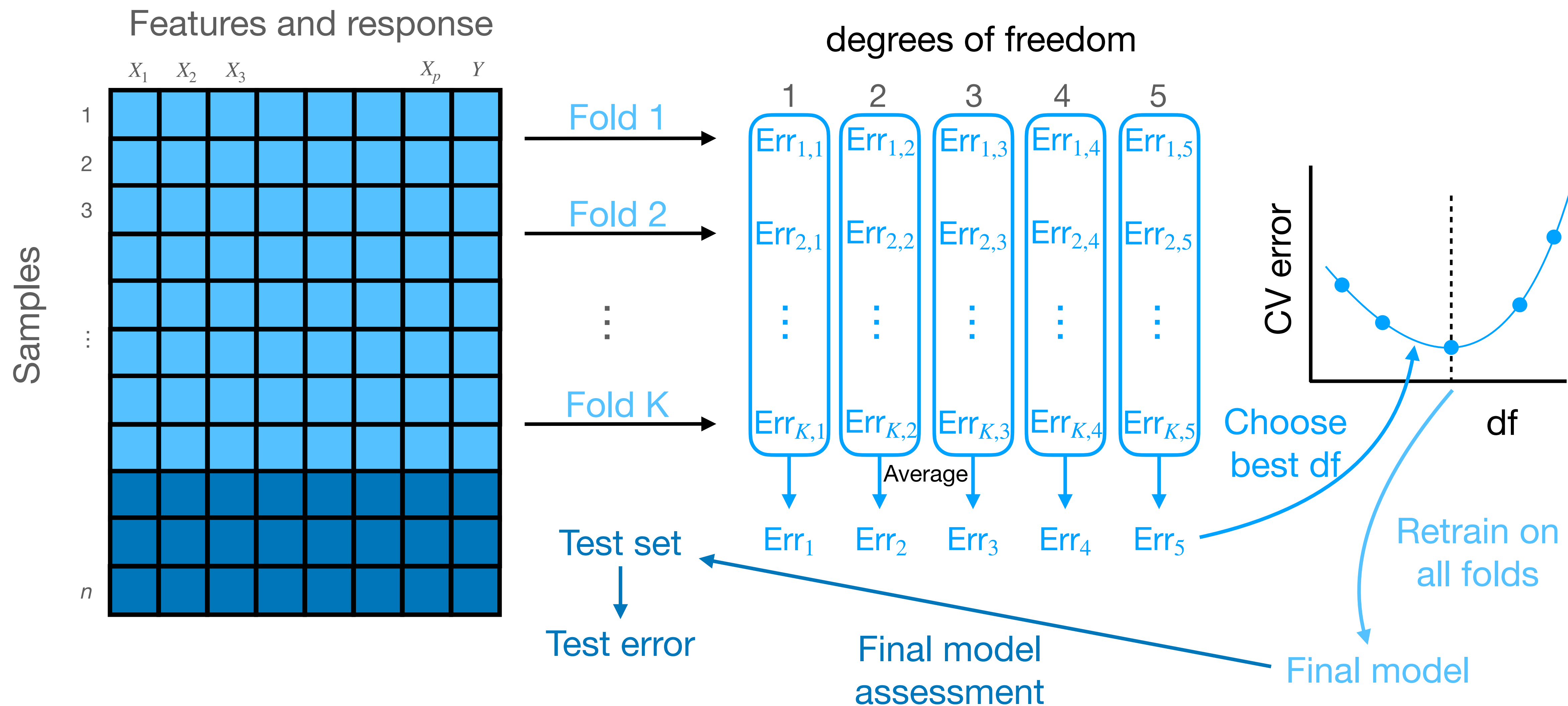
Cross-validation for model selection



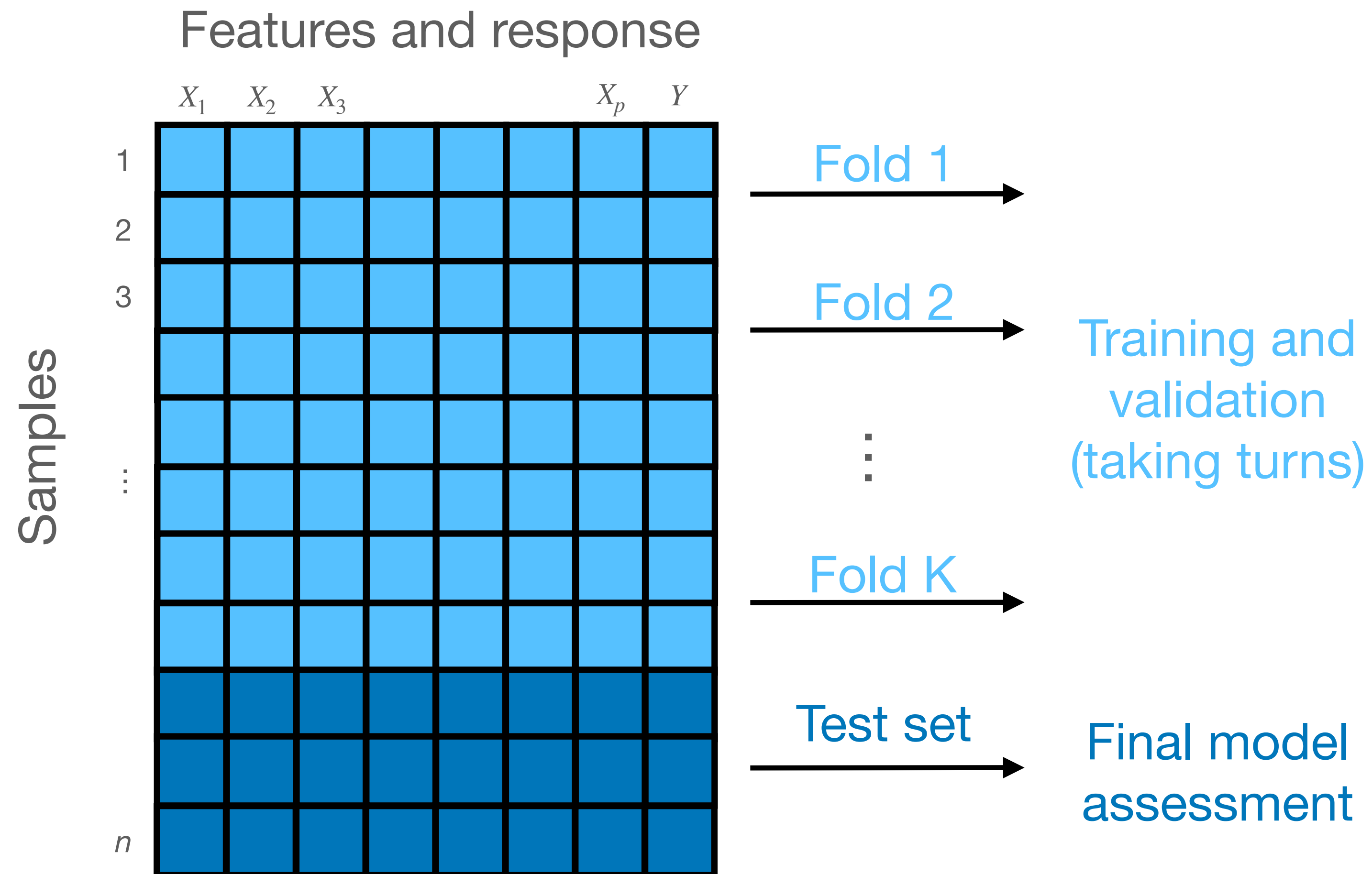
Cross-validation for model selection



Cross-validation for model selection



Cross-validation (summary)



1. Split data into K folds
2. For each fold k ,
 - Fit models of varying complexity to training data, holding out fold k
 - Evaluate validation error for each model on fold k
3. Average across folds to get CV error for each model complexity
4. Choose model complexity to minimize CV error
5. Refit this model on all folds
6. Evaluate final model on the test set

Different kinds of test error

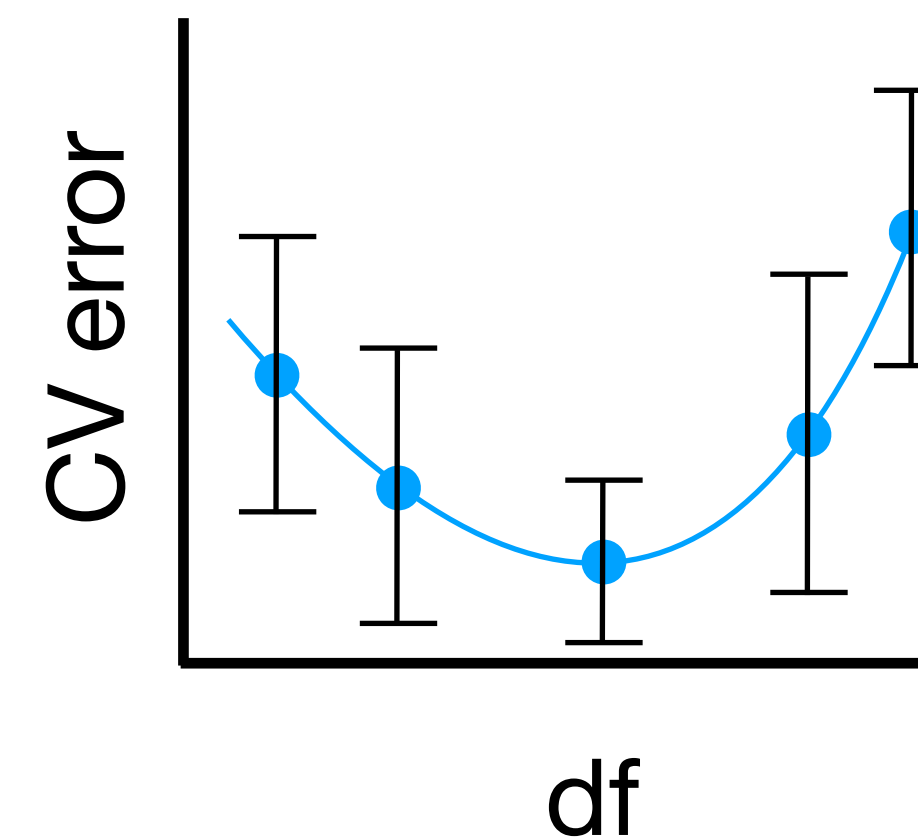
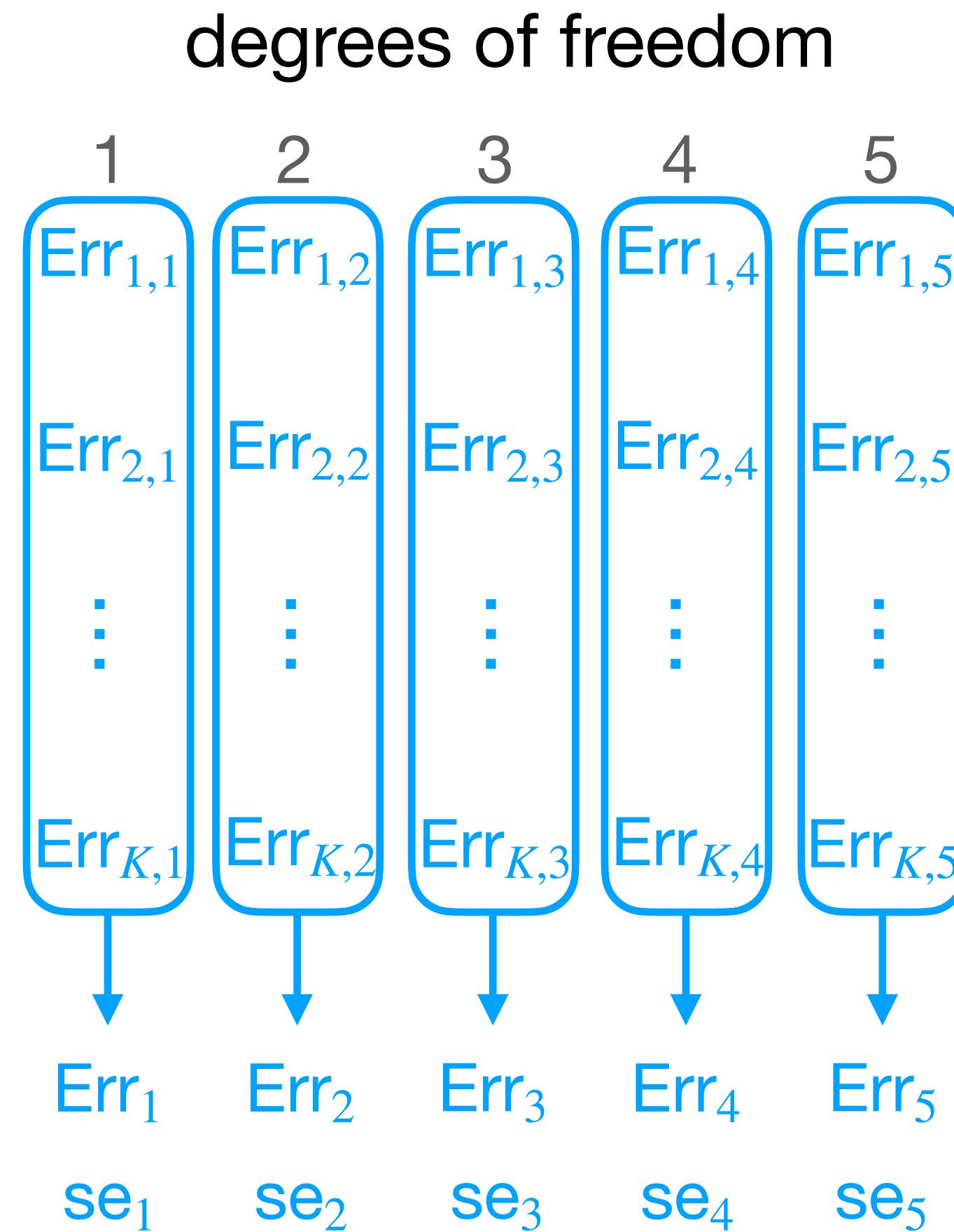
Cross-validation is compatible with any definition of test error (e.g. mean squared error or misclassification error).

Cross-validation should be used with the same error metric as will be used in the final model evaluation.

Choosing the number of folds

- More folds means more computation
- Fewer folds means the training sets used for model selection are much smaller than the actual training set
- In practice, $K = 5$ or $K = 10$ are common choices

Cross-validation standard error

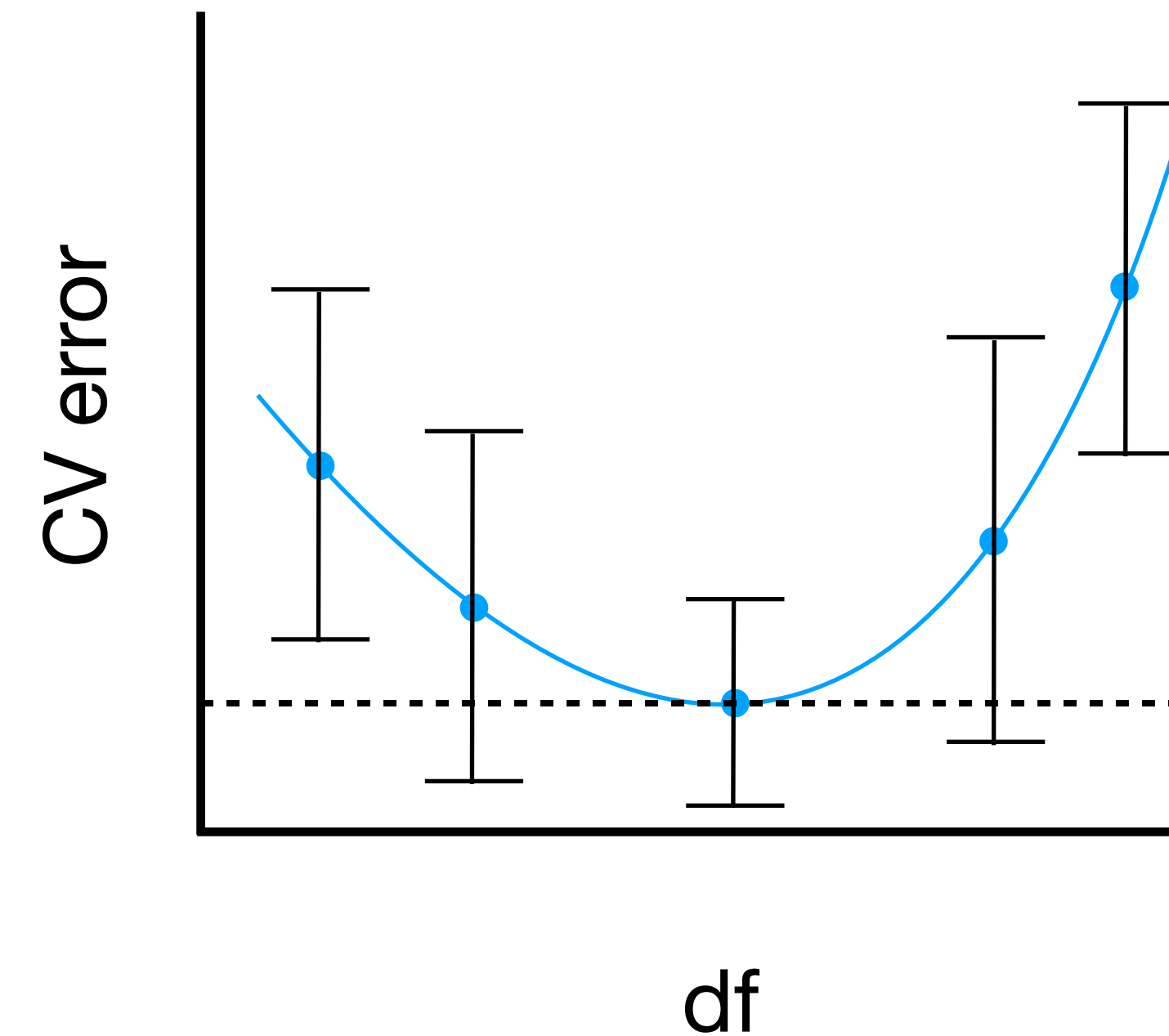


$$se_{df} = \frac{1}{\sqrt{K}} \times \text{s.d.}(Err_{1,df}, \dots, Err_{K,df})$$

One standard error rule

Occam's razor:

Select the smallest model for which the CV error is within one standard error of the lowest point on the curve.



Lecture 4: Classification

A patient comes into the emergency room with stroke symptoms. Based on her CT scan, is the stroke ischemic or hemorrhagic?

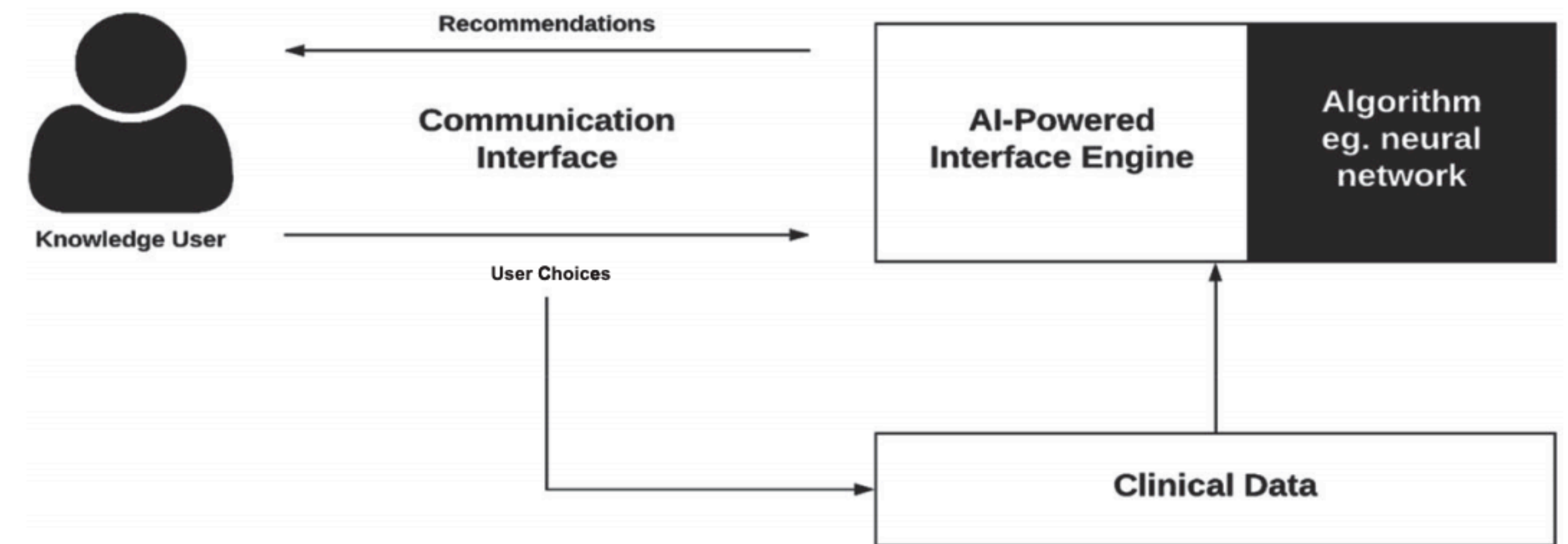


Image source: Sutton et al. 2020 (npj Digit. Med.)

This is a **binary classification problem**: $Y \in \{0,1\}$.

Given features $X = (X_1, \dots, X_p)$, the goal is to guess a response $\hat{Y} = \hat{f}(X)$ that is close to the true response, i.e. $\hat{Y} \approx Y$. Measure of success is usually the

$$\text{test misclassification error} = \frac{1}{N} \sum_{i=1}^N I(Y_i^{\text{test}} \neq \hat{f}(X_i^{\text{test}})).$$

Classification via probability estimation

Suppose that the true relationship between Y and X is

$$\mathbb{P}[Y = 1 | X] = p(X), \quad \text{for some function } p.$$

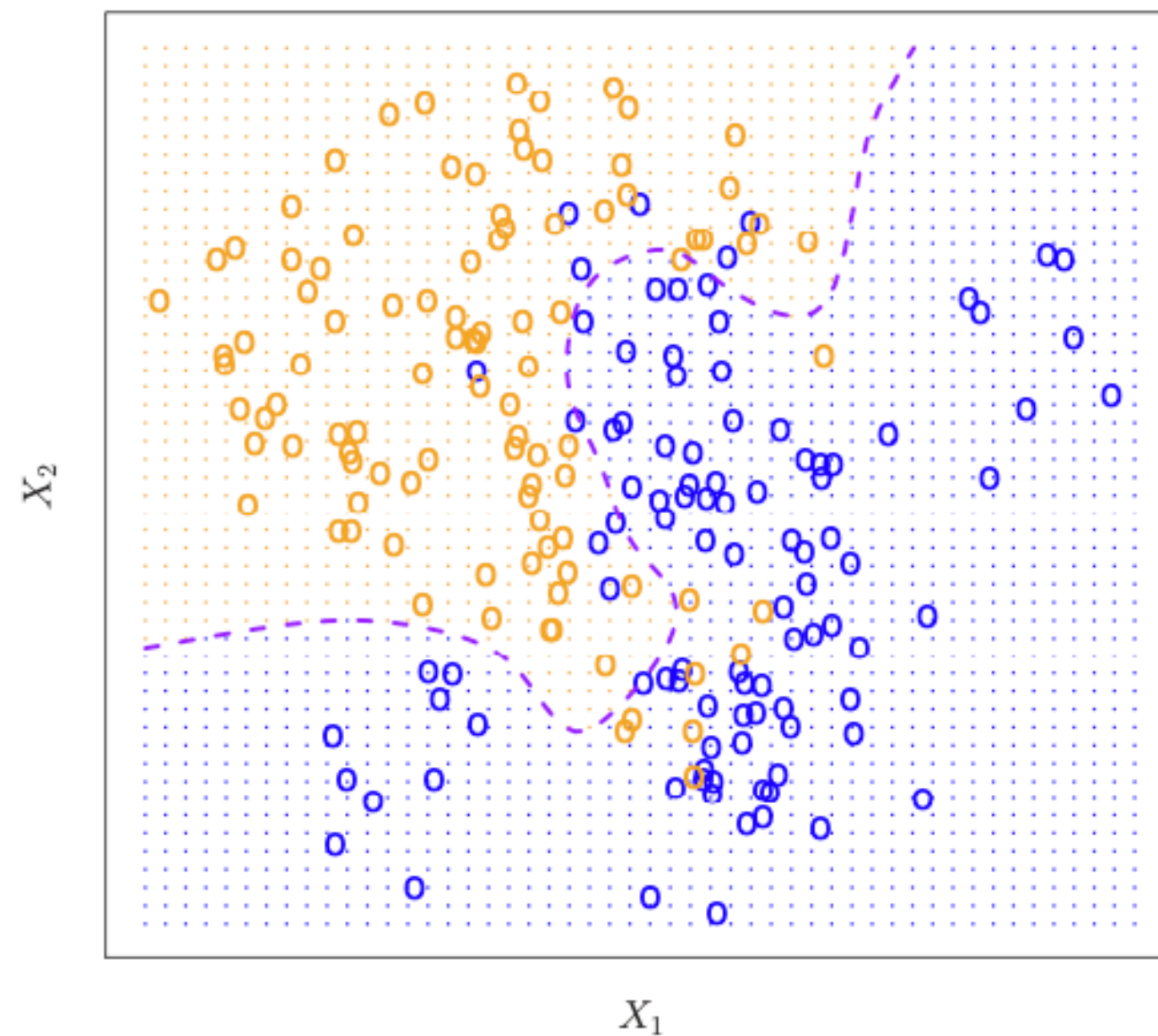
Then, the optimal classifier (called the **Bayes classifier**) is

$$\hat{f}^{\text{Bayes}}(X) = \begin{cases} 1, & \text{if } p(X) \geq 0.5; \\ 0 & \text{if } p(X) < 0.5. \end{cases}$$

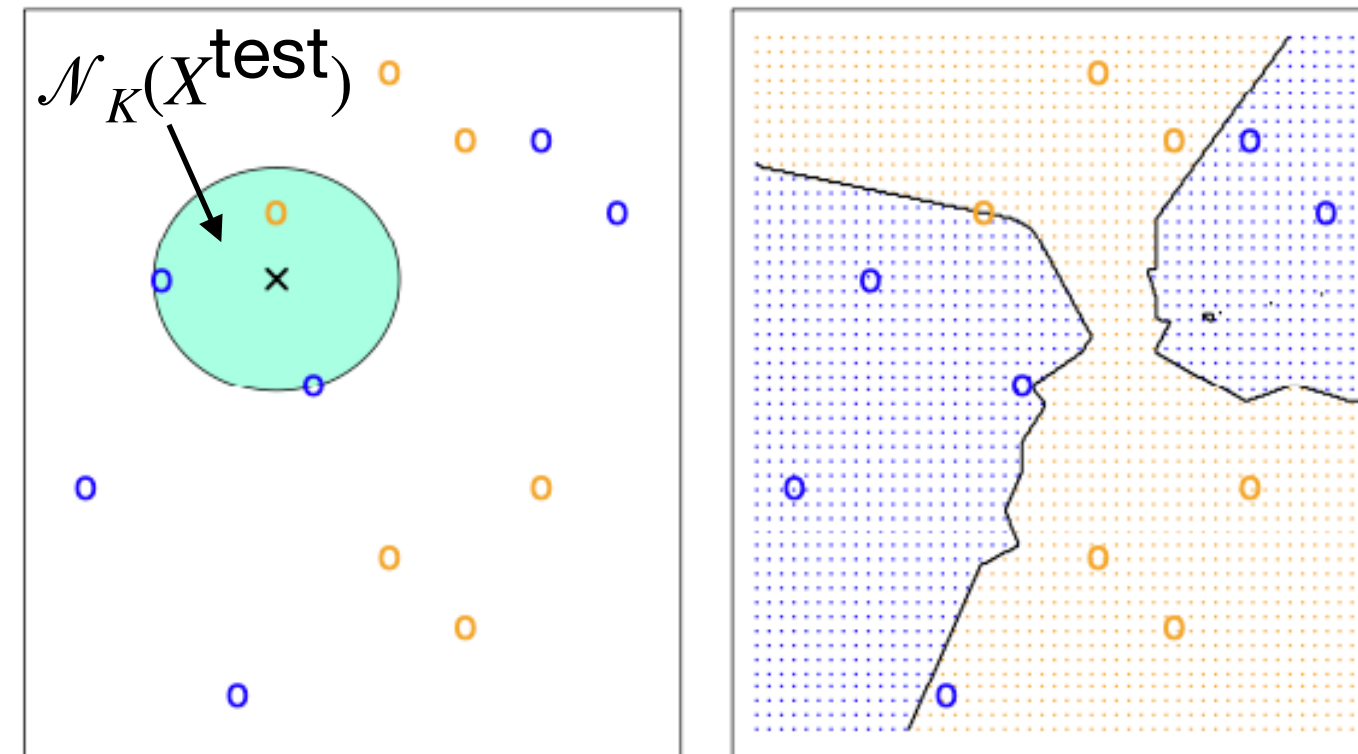
Classifiers usually build an approximation $\hat{p}(X) \approx \mathbb{P}[Y = 1 | X]$, and define

$$\hat{f}(X) = \begin{cases} 1, & \text{if } \hat{p}(X) \geq 0.5; \\ 0 & \text{if } \hat{p}(X) < 0.5. \end{cases}$$

Example: K-nearest neighbors

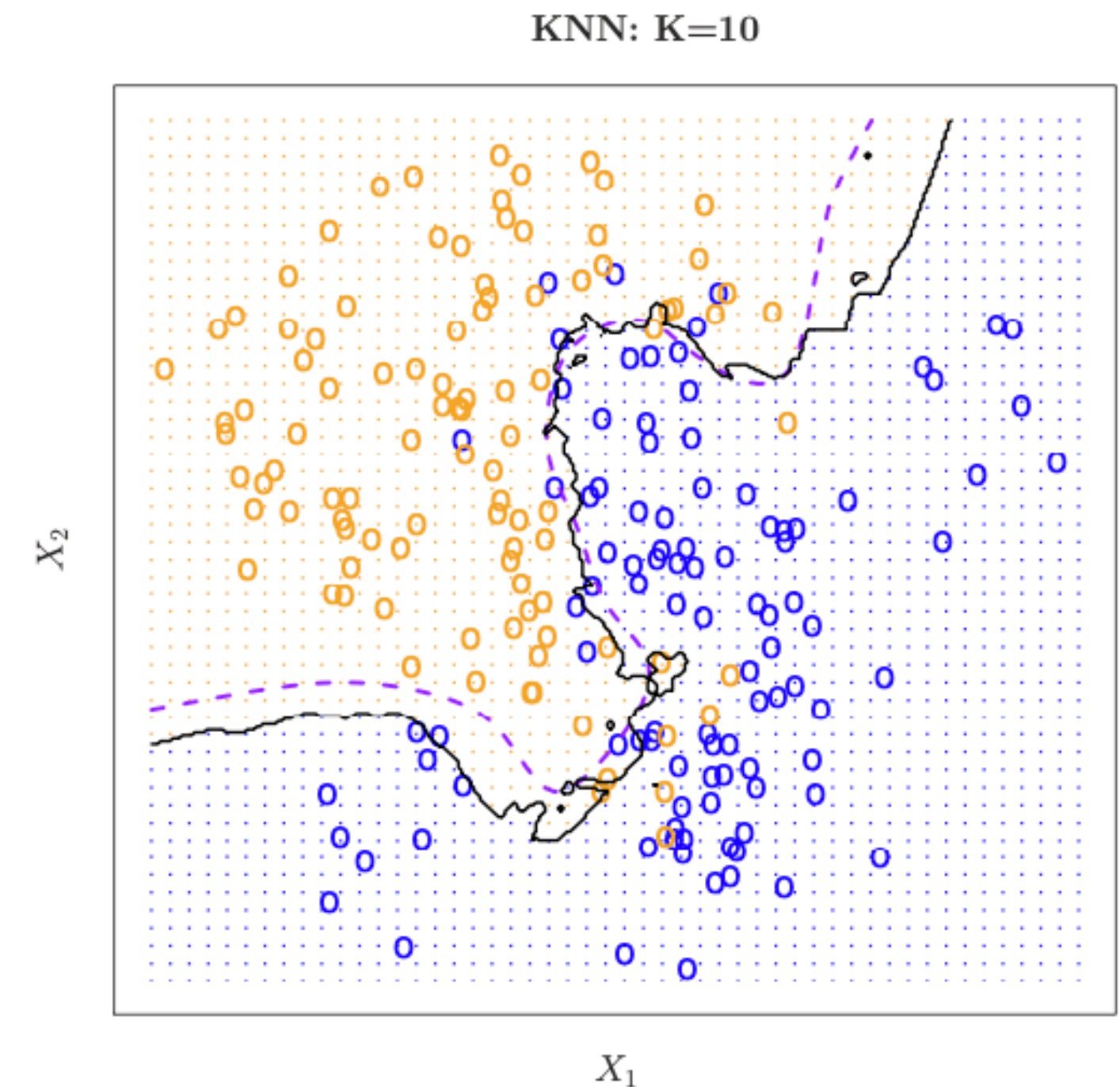


Simulated binary classification data.
Bayes classifier in purple.
E.g., color = stroke type, (X_1, X_2) = CT image.



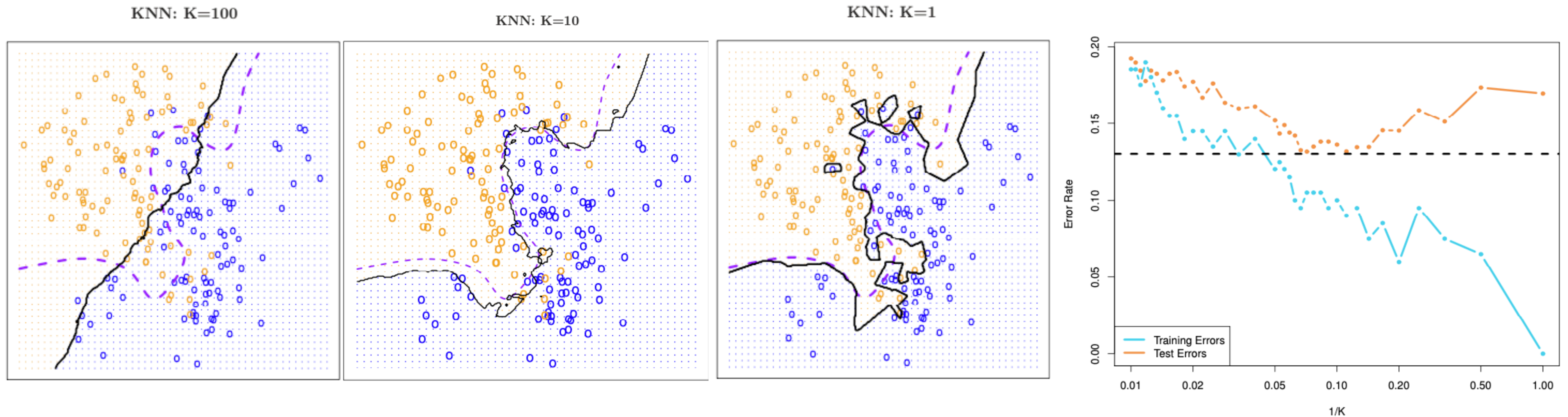
KNN illustration: Classify a test point based on majority vote among 3 nearest neighbors.

$$\hat{p}(X^{\text{test}}) = \frac{1}{K} \sum_{i \in \mathcal{N}_K} I(X_i^{\text{train}} = 1).$$



Applying KNN with $K = 10$ to simulated data.

Model complexity and misclassification error



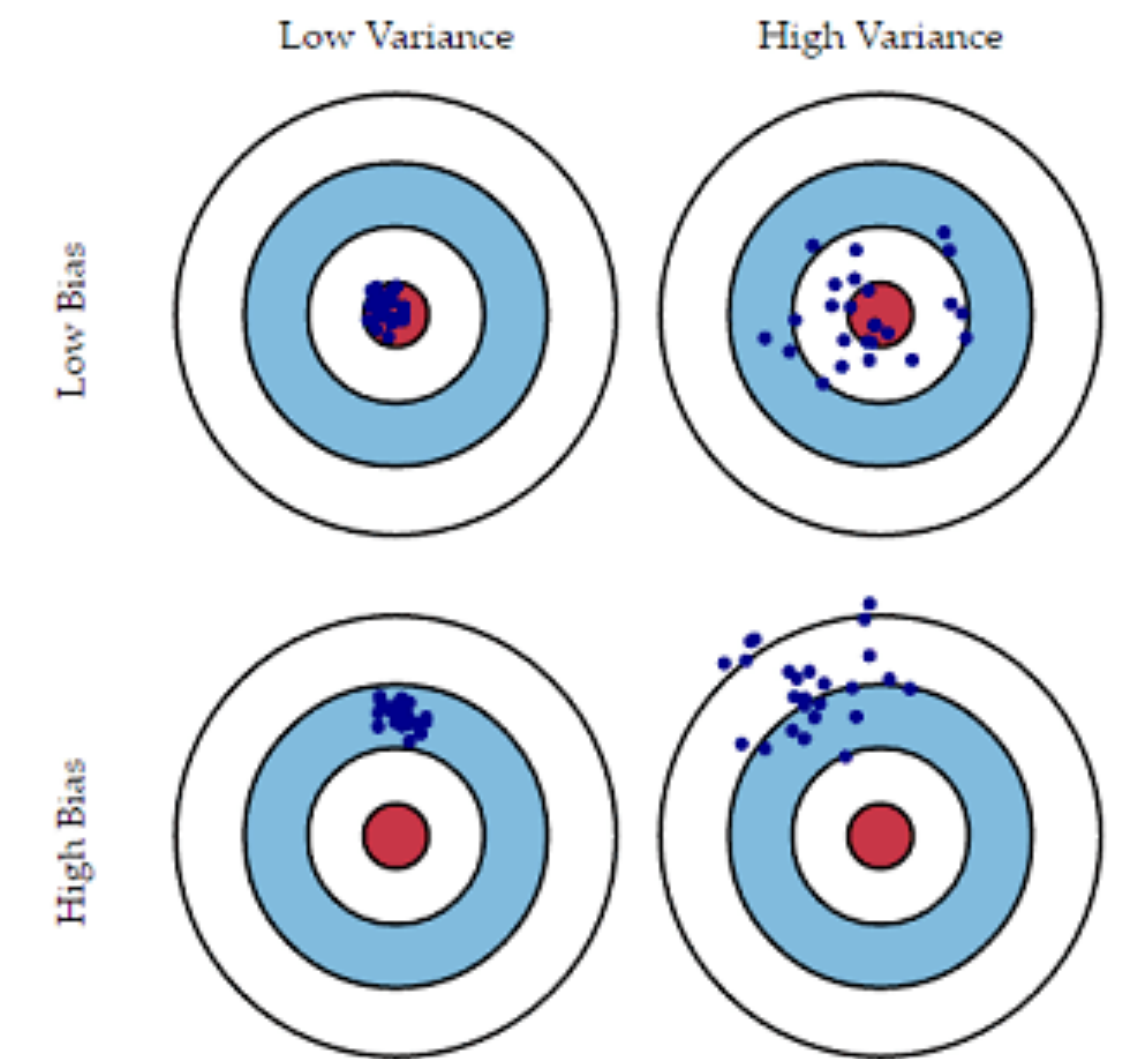
Same Goldilocks principle as in regression case:

- Too little complexity: Can't capture the true trend in the data.
- Too much complexity: Too sensitive to noise in the training data (overfitting).

Bias-variance tradeoff

Mathematically: Applies only to continuous response variables and MSE.

Intuitively: Applies to any prediction problem, including classification.



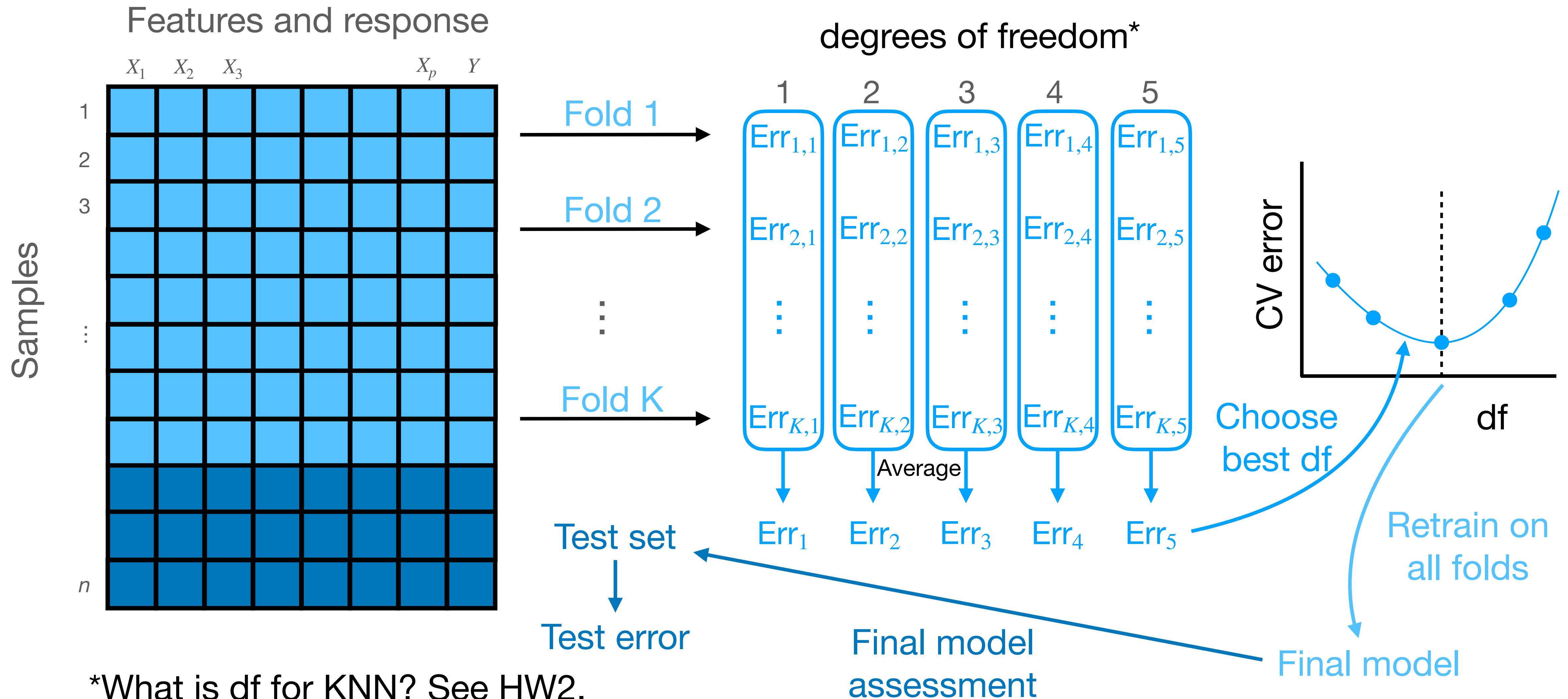
For the estimate $\hat{p}(X)$

- Bias: $\mathbb{E}[\hat{p}(X)] - p(X)$ \longrightarrow
- Variance: $\text{Var}[\hat{p}(X)]$ \longrightarrow

For classifying $\hat{Y} = I(p(X) \geq 0.5)$

- Bias: Predict wrong class on average, to the extent \hat{p} on wrong side of 0.5
- Variance: Prediction varies with training set, to the extent \hat{p} fluctuates above or below 0.5
- Irreducible error (AKA Bayes error): Error incurred by Bayes classifier because $0 < \mathbb{P}[Y = 1 | X] < 1$.

Cross-validation based on misclassification error (otherwise same as before)



Class imbalance

In many real-world classification problems, one class (say $Y = 1$) is significantly less frequent than the other. For example:

- Credit card transaction classification: normal versus fraudulent
- COVID testing: negative versus positive

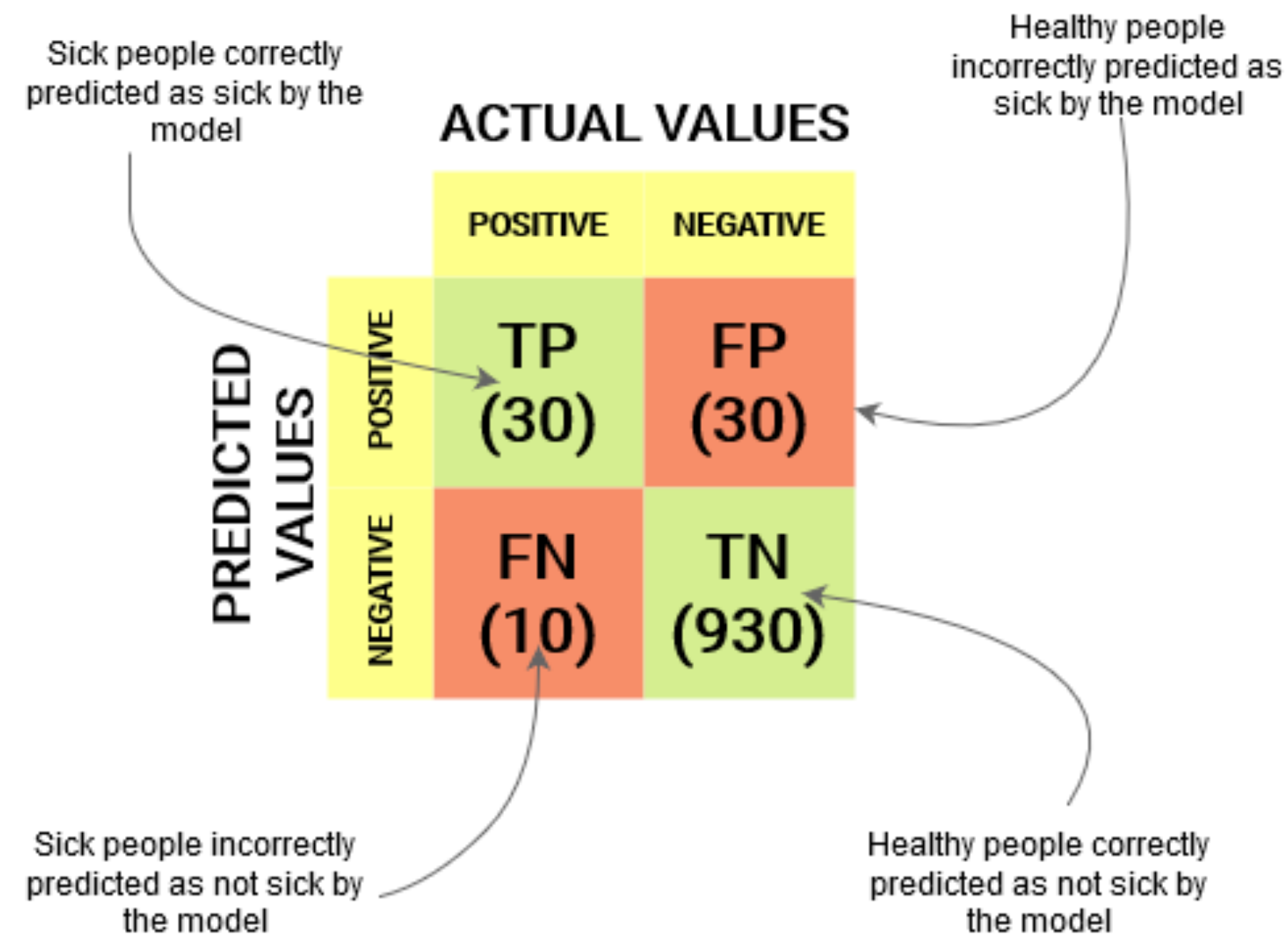
Often in these cases, the costs of misclassification are also asymmetric, i.e. the misclassification error is not the right metric.

Let's say 1% of credit card transactions are fraudulent. Then, **the classifier that always predicts “not fraudulent” will have a misclassification error of only 1%.**

Cross-validation based on misclassification error leads to overly simple models that ignore the minority class.

A more wholistic picture of a classifier

Confusion matrix



A 2x2 confusion matrix for a classifier. The columns are labeled 'ACTUAL VALUES' with 'POSITIVE' and 'NEGATIVE'. The rows are labeled 'PREDICTED VALUES' with 'POSITIVE' and 'NEGATIVE'. The cells contain: TP (30) in green, FP (30) in red, FN (10) in red, and TN (930) in green. Four annotations with arrows point to the cells: 'Sick people correctly predicted as sick by the model' points to TP, 'Healthy people incorrectly predicted as sick by the model' points to FP, 'Sick people incorrectly predicted as not sick by the model' points to FN, and 'Healthy people correctly predicted as not sick by the model' points to TN.

ACTUAL VALUES			
PREDICTED VALUES	POSITIVE	NEGATIVE	
POSITIVE	TP (30)	FP (30)	
NEGATIVE	FN (10)	TN (930)	

Sick people correctly predicted as sick by the model

Healthy people incorrectly predicted as sick by the model

Sick people incorrectly predicted as not sick by the model

Healthy people correctly predicted as not sick by the model

Summaries of confusion matrix

$$\text{False positive rate} = \frac{\text{number false positives}}{\text{total actual negatives}}$$

$$\text{False negative rate} = \frac{\text{number false negatives}}{\text{total actual positives}}$$

Image source: <https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/>

Thinking about misclassification costs

The cost of a false negative might be much greater than a false positive:

- Undetected fraudulent credit card transaction (false negative)
→ drained bank account. Cost: $C_{FN} = \$10,000$.
- False alarm of fraud (false positive)
→ annoying text message and/or replaced credit card. Cost: $C_{FP} = \$10$.

Weighted misclassification error:

$$\frac{1}{N} \sum_{i=1}^N C_{FP} \cdot I(\hat{Y}_i^{\text{test}} = 1, Y_i^{\text{test}} = 0) + C_{FN} \cdot I(\hat{Y}_i^{\text{test}} = 0, Y_i^{\text{test}} = 1).$$

Building misclassification costs into training

There may be two issues with

$$\hat{f}(X) = \begin{cases} 1, & \text{if } \hat{p}(X) \geq 0.5; \\ 0 & \text{if } \hat{p}(X) < 0.5. \end{cases}$$

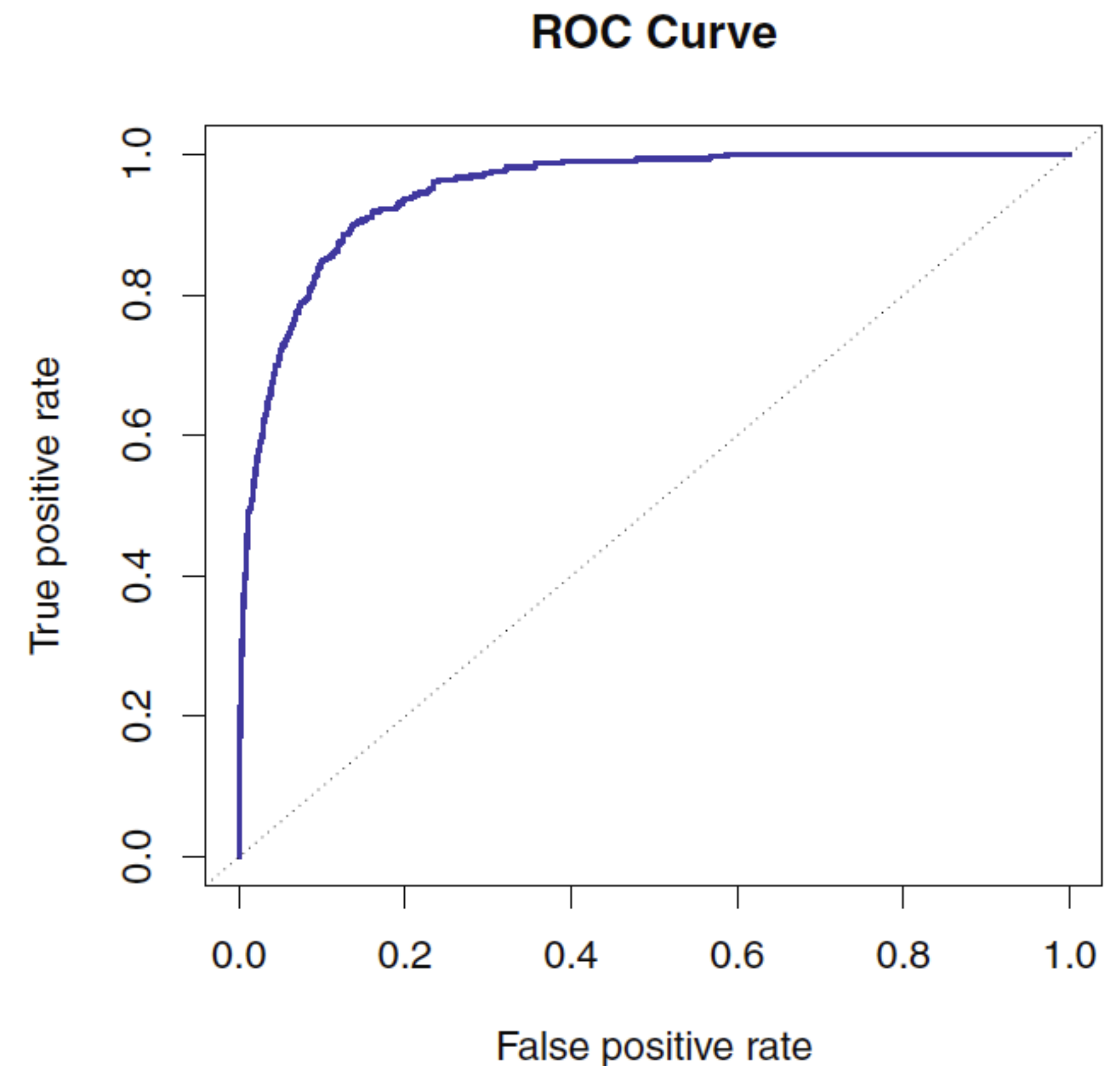
1. The minority class is poorly captured by the probability model $\hat{p}(X)$.
2. The probability threshold of 0.5 is suboptimal.

To fix these, a variety of strategies can be employed:

- Downsample the majority class by a factor C_{FP}/C_{FN} .
- Choose the probability threshold $C_{FP}/(C_{FN} + C_{FP})$ instead of 0.5.
- Build cost directly into the objective function when training.

ROC curve

- The ROC curve plots the true positive rate (one minus the false negative rate) versus the false positive rate, as the threshold is varied from 0 to 1.
- We want the curve to get as close to the upper left-hand corner as possible.
- Area under the curve (AUC) is another measure of the quality of a classifier.



Unit 2 Summary

- Predictive models are evaluated based on their test error, which can be decomposed into bias, variance, and irreducible error (bias-variance tradeoff).
- Model complexity plays a crucial role in test error; models that are not complex enough suffer due to bias, models that are too complex suffer due to variance.
- For splines (and linear regression in general), model complexity captured by the number of parameters being fit.
- Cross-validation is the standard way of tuning model complexity; the trick is to always separate the data used for training from that used for validation/testing.
- Many of these phenomena carry over from regression to classification, but classification brings a few new challenges, like class imbalance.