Section 8 Resampling Methods

➢ **Bias and Variance Tradeoff.**

Training Data: $\mathcal{D} = \left(\vec{x}^{(i)}, y^{(i)}\right)$ for $i = 1 \ldots n$.

Suppose the data $\mathcal{D}$ follows (linear) model $y = h(\vec{x}) + \epsilon$ with unmodeled error ϵ

$$\epsilon \sim \mathrm{Normal}(0, \sigma^2)$$

An estimate function from $\mathcal{D}$ is denoted by $\hat{h}(\vec{x}; \mathcal{D})$ or $\hat{h}_{\mathcal{D}}(\vec{x})$

Fisher's view: the measurements are a random selection from the set of all possible measurements which form the true distribution!

Evaluate $\hat{h}_{\mathcal{D}}(\vec{x})$ by expected loss on a *new test sample* $(\vec{x}, y)$, we get risk

$$R\left(\hat{h}_{\mathcal{D}}\right) = E_{\vec{x},y}\left[\left(\hat{h}_{\mathcal{D}}(\vec{x}) - y\right)^2\right]$$

The expected predicted error (mean squared error) is calculated by

$$E_{\mathcal{D}}\left[R\left(\hat{h}_{\mathcal{D}}\right)\right] = E_{\vec{x},y,\mathcal{D}}\left[\left(\hat{h}_{\mathcal{D}}(\vec{x}) - y\right)^2\right]$$

$$= E_{\vec{x},y,\mathcal{D}}\left[\left(\hat{h}_{\mathcal{D}}(\vec{x}) - h(\vec{x}) - \epsilon\right)^2\right]$$

$$= E_{\vec{x}}\left[\left(E_D\left[\hat{h}_{\mathcal{D}}(\vec{x}) - h(\vec{x})\right]\right)^2\right] + E_{\vec{x}}\left[E_D\left[\left(\hat{h}_{\mathcal{D}}(\vec{x}) - E_D\left[\hat{h}_{\mathcal{D}}(\vec{x})\right]\right)^2\right]\right] + \sigma^2$$

$$= E_{\vec{x}}\left[\left(E\left[\hat{h}_{\mathcal{D}}(\vec{x})\right] - h(\vec{x})\right)^2\right] + E_{\vec{x}}\left[Var_{\mathcal{D}}\left(\hat{h}_{\mathcal{D}}(\vec{x})\right)\right] + \sigma^2$$

$$= bias^2 + variance + \sigma^2$$

Bias refers to the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much simpler model.

Variance refers to the amount by which $\tilde{h}(\vec{x})$ would change if we estimated it using a different training data set. Since the training data are used to fit the statistical learning method, different training data sets will result in a different $\tilde{h}(\vec{x})$. But ideally the estimate for $h(\vec{x})$ should not vary too much between training sets.

Bias and variance trade-off: The optimal predictive capability is the one that leads to balance between bias and variance.

Example: (Linear Model.) Assume a true model is linear: $h(x_*) = x_*^T \theta$

Find the **bias** and **variance** of $f(x_*|D) = x_*^T \hat{\theta}_{\mathrm{MLE}}$

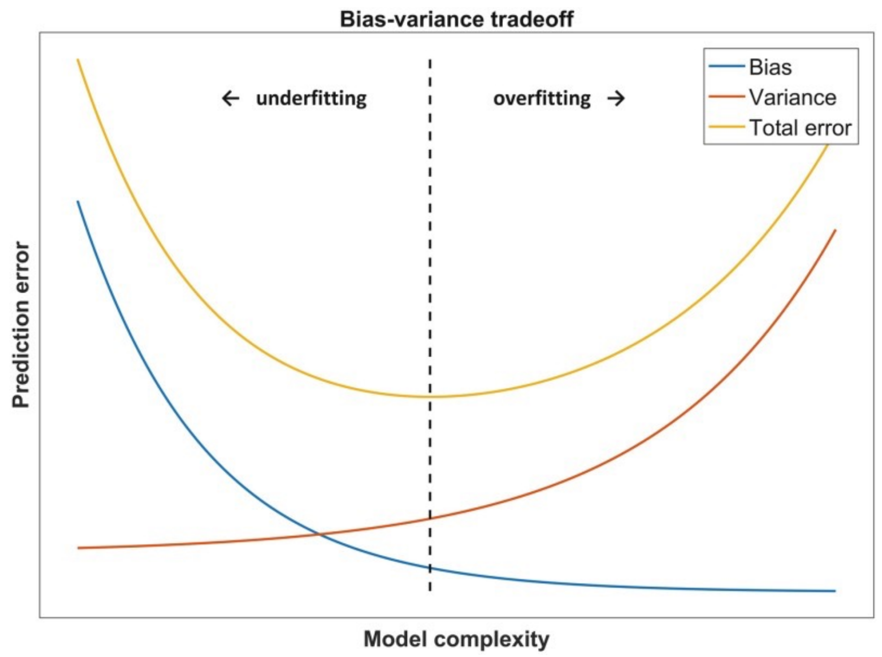
Calculation of bias:

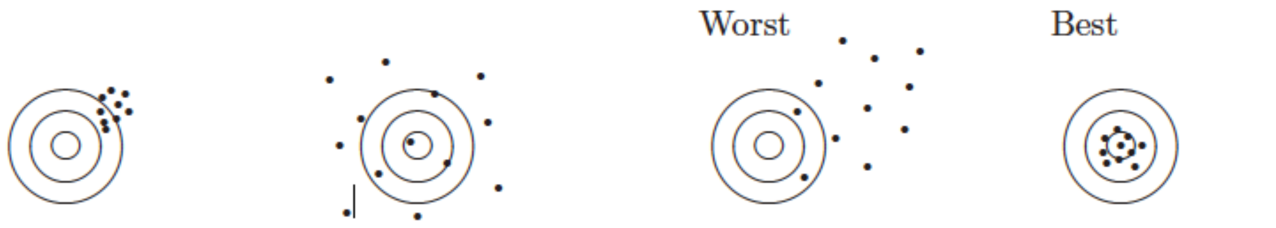
$$\begin{aligned}
\mathrm{bias} &= h(x_*) - \mathbf{E}_D\left[f(x_*|D)\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T\hat{\theta}_{\mathrm{MLE}}\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T(X^TX)^{-1}X^TY\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T(X^TX)^{-1}X^T(X\theta+\epsilon)\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T(X^TX)^{-1}X^TX\theta + x_*^T(X^TX)^{-1}X^T\epsilon\right] \\
&= x_*^T\theta - \mathbf{E}_D\left[x_*^T\theta + x_*^T(X^TX)^{-1}X^T\epsilon\right] \\
&= x_*^T\theta - x_*^T\theta + x_*^T(X^TX)^{-1}X^T\mathbf{E}_D\left[\epsilon\right] \\
&= x_*^T\theta - x_*^T\theta = 0
\end{aligned}$$

Variance of $f(x_*|D) = x_*^T \hat{\theta}_{\text{MLE}}$

$$\begin{aligned}
\text{Var.} &= \mathbf{E}\left[(f(x_*|D) - \mathbf{E}_D\left[f(x_*|D)\right])^2\right] \\
&= \mathbf{E}\left[(x_*^T(X^TX)^{-1}X^T\epsilon)^2\right] \\
&= \mathbf{E}\left[(x_*^T(X^TX)^{-1}X^T\epsilon)(x_*^T(X^TX)^{-1}X^T\epsilon)^T\right] \\
&= \mathbf{E}\left[x_*^T(X^TX)^{-1}X^T\epsilon\epsilon^T(x_*^T(X^TX)^{-1}X^T)^T\right] \\
&= x_*^T(X^TX)^{-1}X^T\mathbf{E}\left[\epsilon\epsilon^T\right](x_*^T(X^TX)^{-1}X^T)^T \\
&= x_*^T(X^TX)^{-1}X^T\sigma_\epsilon^2 I(x_*^T(X^TX)^{-1}X^T)^T \\
&= \sigma_\epsilon^2 x_*^T(X^TX)^{-1}X^TX(x_*^T(X^TX)^{-1})^T \\
&= \sigma_\epsilon^2 x_*^T(x_*^T(X^TX)^{-1})^T \\
&= \sigma_\epsilon^2 x_*^T(X^TX)^{-1}x_*
\end{aligned}$$

- U-shaped bias–variance trade-off curve (Geman et al., 1992).





(a) High bias, Low variance (high precision)

(b) Low bias, High variance (low precision)

(c) High bias, High variance (low precision)

(d) Low bias Low variance (high precision)

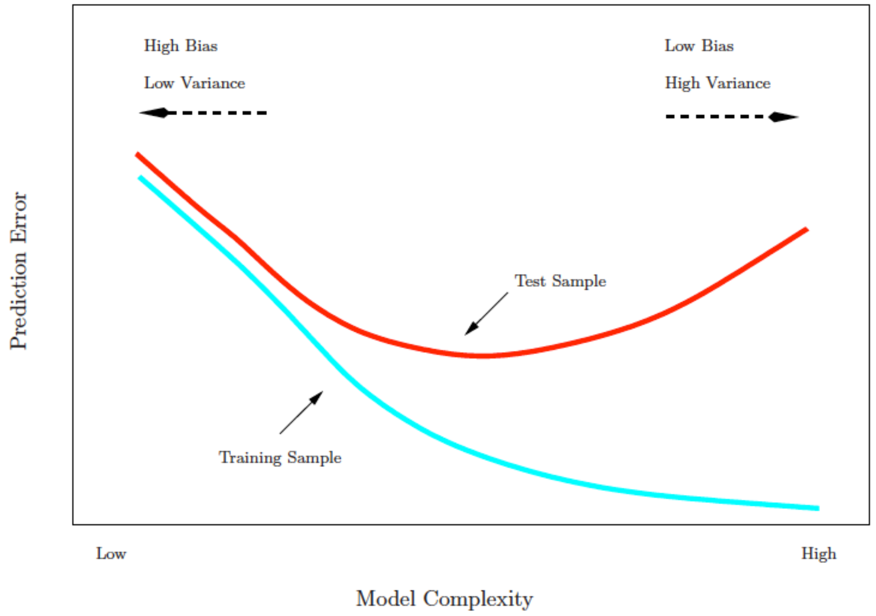
http://scott.fortmann-roe.com/docs/BiasVariance.html

Test error V.S. Training error

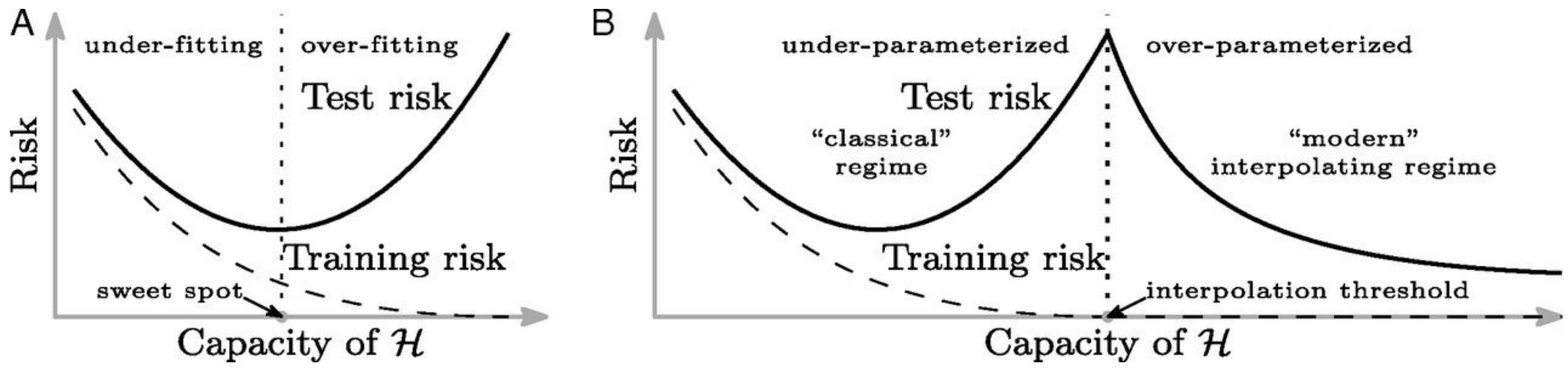
The **training error** can be easily calculated by applying the statistical learning method to the observations used in its training.

The **test error** is the average error that results from using a statistical learning method to predict the response on a new observation, one that was not used in training the method.

But the training error rate often is quite different from the test error rate, and in particular the former can dramatically underestimate the latter.



- Modern point of view of bias-variance trade-off:



1. Reconciling modern machine-learning practice and the classical bias–variance trade-off
Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal
PNAS August 6, 2019 116 (32) 15849-15854; https://doi.org/10.1073/pnas.1903070116

2. Rethinking Bias-Variance Trade-off for Generalization of Neural Networks
Zitong Yang, Yaodong Yu, Chong You, Jacob Steinhardt, Yi Ma
Proceedings of the 37 th International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.
https://arxiv.org/pdf/2002.11328.pdf

3. **A Modern Take on the Bias-Variance Tradeoff in Neural Networks**
Neal, Mittal, Baratin, et.al. https://arxiv.org/pdf/1810.08591.pdf

Prediction-error estimates

Some methods (adjusted R^2, the C_p statistic, AIC and BIC) make a mathematical adjustment to the training error rate in order to estimate the test error rate.

Next, we consider a class of validation methods that estimate the test error, by holding out a subset of the training observations from the fitting process, and then applying the statistical learning method to those held out observations. The resulting validation-set error provides an estimate of the test error.

➢ Overview of Resampling:

Resampling is an important tool to assess the validity/accuracy of statistical methods and models. Pretending the data as population and repeatedly draw sample from the data. Our ultimate goal is to produce the best model with best prediction accuracy.

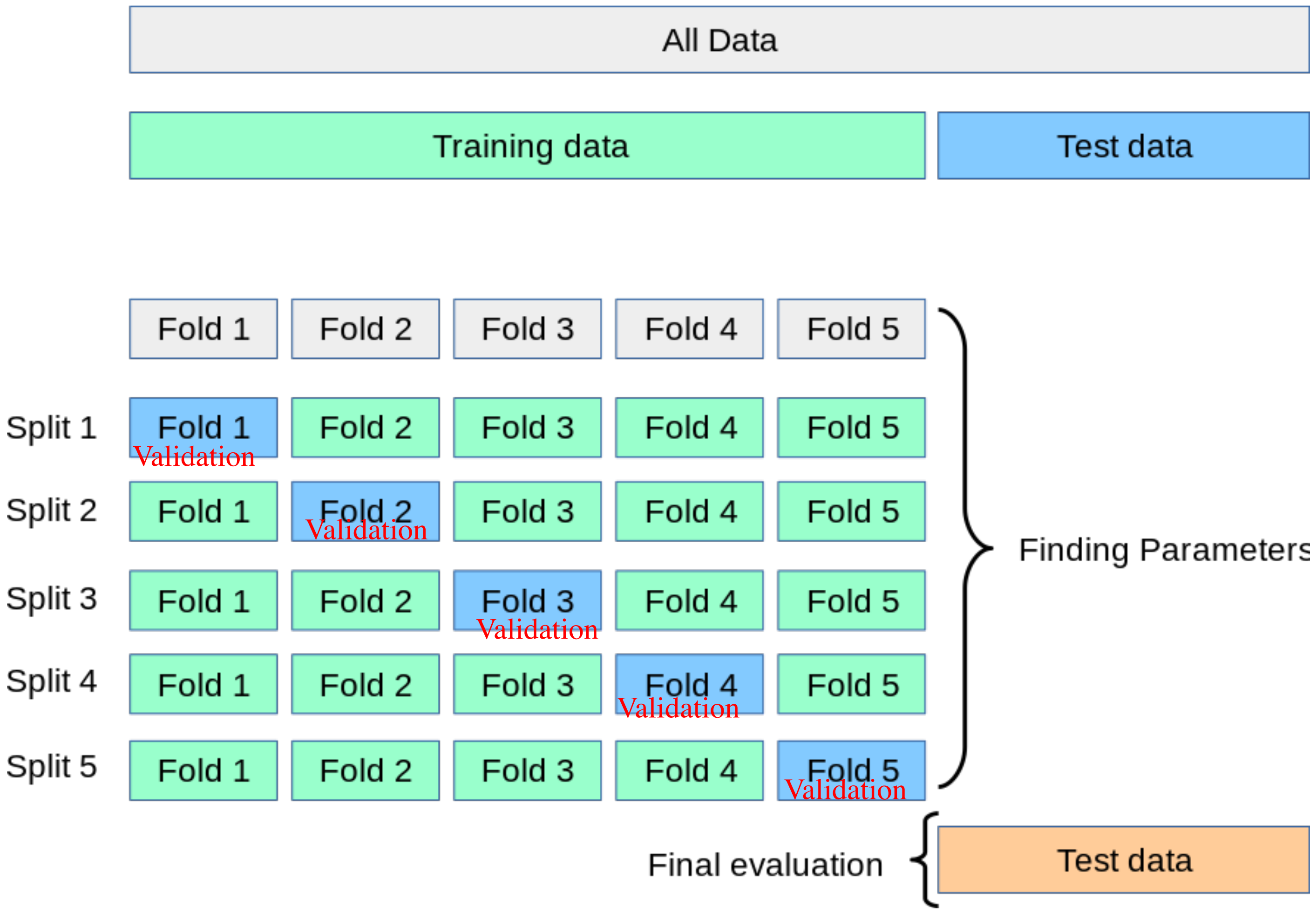
1. **Cross-validation:** estimate the test error of models.

   Artificially separate data into "training data" and "test data" for validation purpose is called *cross-validation.* The "test data" should be more accurately called validation data, which can not be used in training.

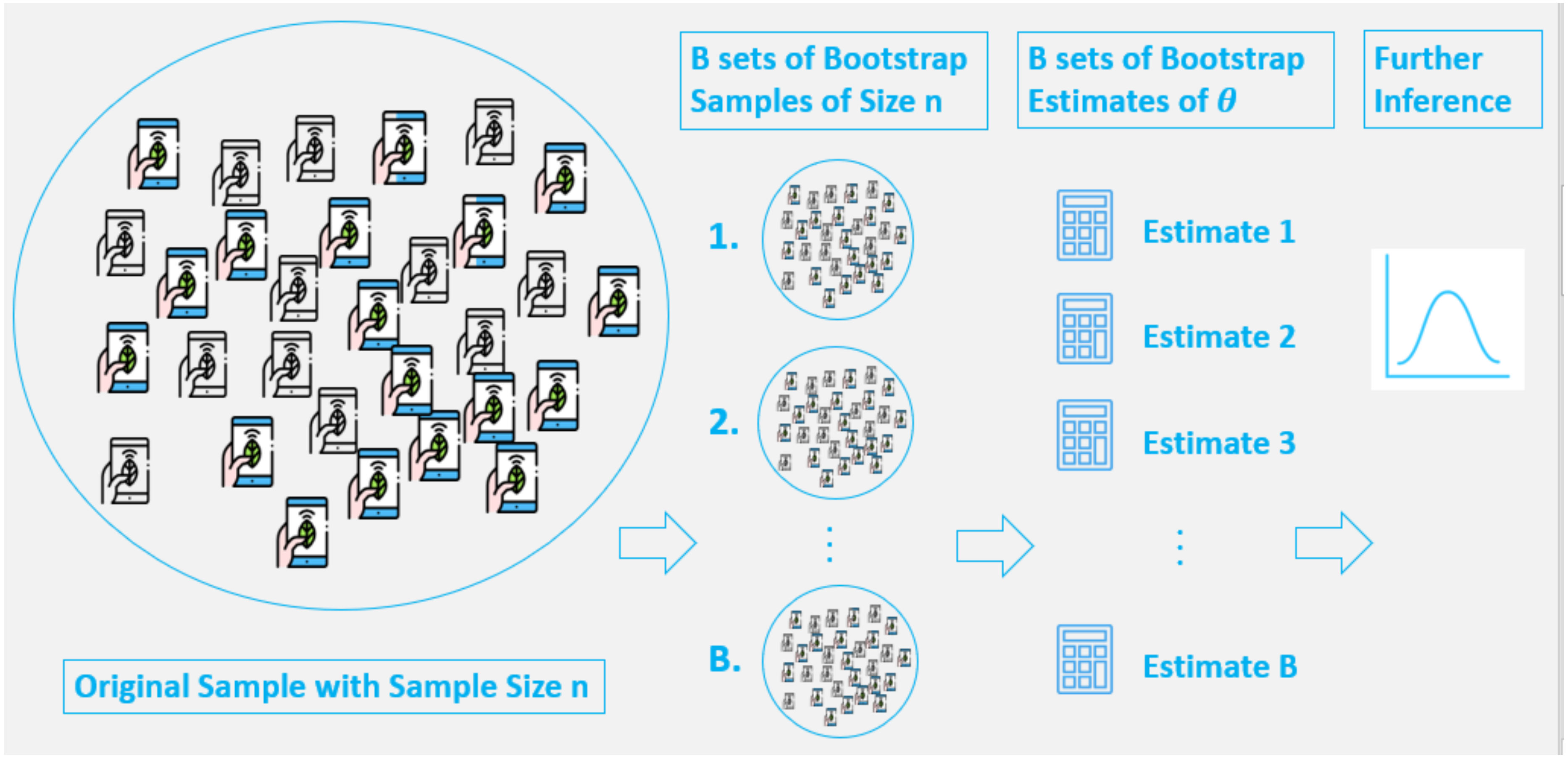
2. **Bootstrap:** quantify the uncertainty of estimators.

   Sampling with replacement n times, n is the sample size of data. Especially useful in statistical inference to quantify the uncertainty of estimates. Used in ensemble methods of machine learning, for example, bagging, random forest.

Cross-validation:



Bootstrap

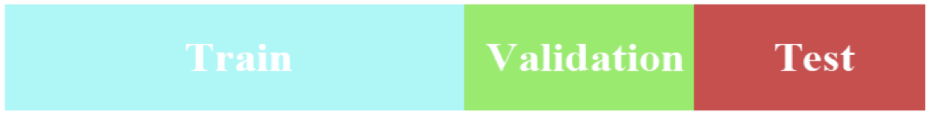

Original Sample with Sample Size n
B sets of Bootstrap Samples of Size n
1.
2.
B.
B sets of Bootstrap Estimates of θ
Estimate 1
Estimate 2
Estimate 3
Estimate B
Further Inference

➢ Cross validation

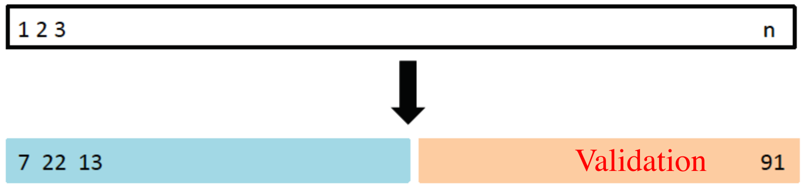
Training error is easily computable with training data. However, the possibility of overfit makes it cannot be used to properly assess test error.

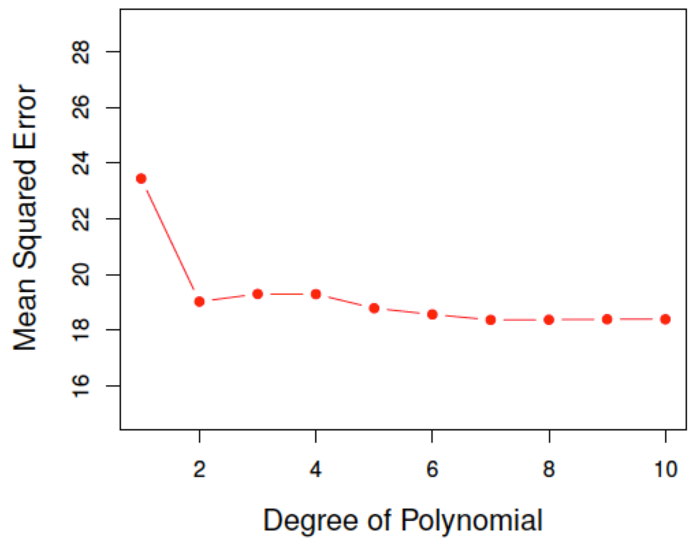
When we have enough data, we can *randomly* split the data into three parts:

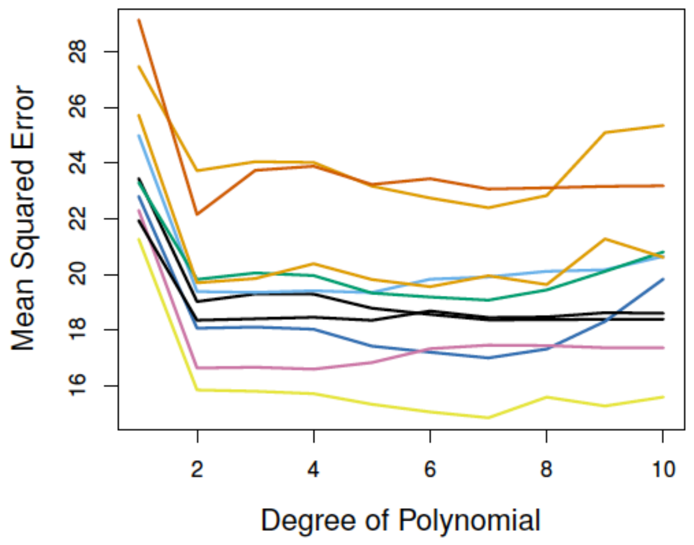
- **Training data:** used to train various models.
- **Validation data:** used to assess the models and identify the best.
- **Test data:** test the results of the best model. (Optional)



Fit various regression models on the training sample. The validation set error rates result from evaluating their performance on the validation sample. MSE as a measure of validation set error.







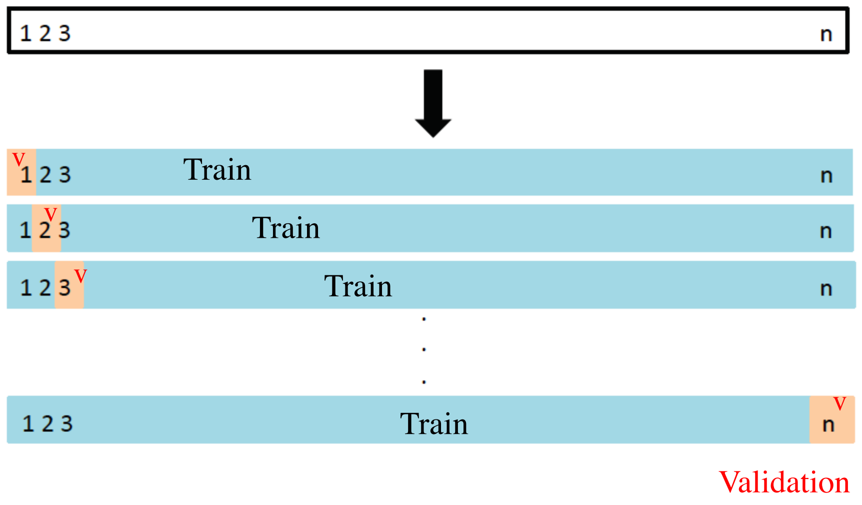
Left: Validation error estimates for a single split into training and validation data sets.

Right: The validation method was repeated ten times, each time using a different random split of the observations into a training set and a validation set. This illustrates the variability in the estimated test MSE that results from this approach

- The validation set MSE for the quadratic fit is considerably smaller than for the linear fit
- Validation set MSE for the cubic fit is slightly larger than for the quadratic fit. This implies that including a cubic term in the regression does NOT lead to better prediction than simply using a quadratic term.
- Repeat the process of randomly splitting the sample set into two parts, we will get a somewhat different estimate for the test MSE.
- Based on the variability among these curves, we can conclude that the linear fit is not adequate for this data.

➢ The leave-one-out cross-validation (LOOCV)

First, pick data point 1 as validation set, the rest as training set. Fit the model on the training set, evaluate the test error, on the validation set, denoted as MSE_1.



Second, pick data point 2 as validation set, the rest as training set. Fit the model on the training set, evaluate the test error on the validation set, denoted as say MSE_2.

…

Repeat the procedure for all data point.

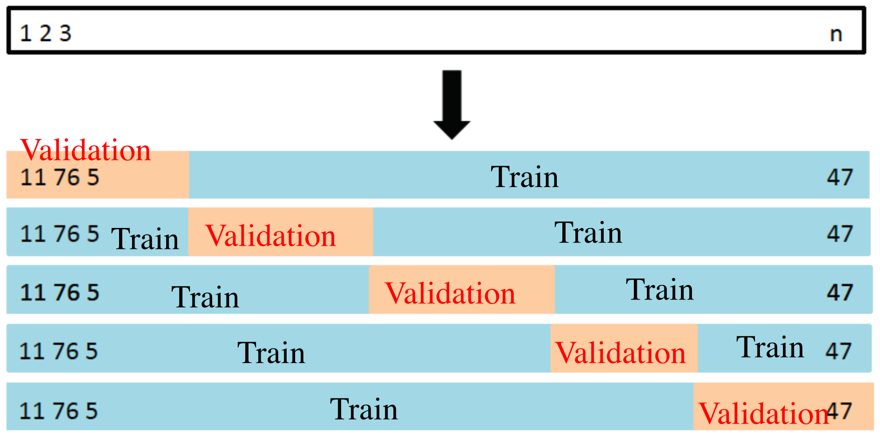
…

Obtain an estimate of the test error by combining the MSE_i for $i = 1,2, \dots n$.

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

➢ **K-fold cross validation (**widely used approach for estimating test error**)**

Divide the data (randomly) into K subsets, usually of equal or similar sizes $\frac{n}{K}$.



Treat one subset as validation set, the rest together as a training set. Run the model fitting on training set. Calculate the test error estimate on the validation set, denoted as MSE_i.

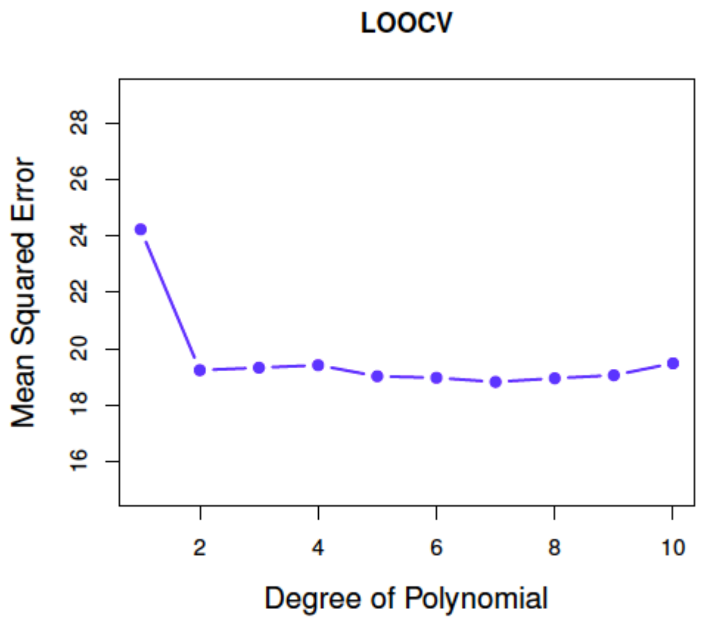
…

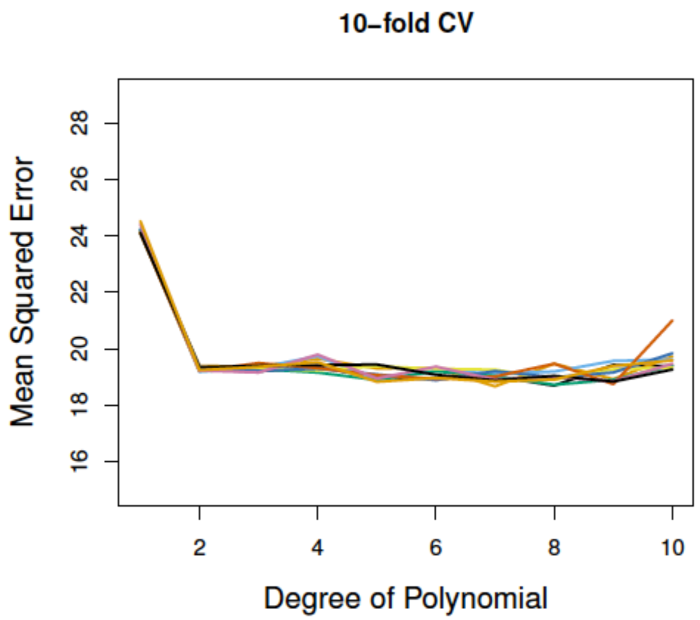
Repeat the procedures over every subset.

…

Average over the above K estimates of the test errors, and obtain

$$CV_{(K)} = \frac{1}{K} \sum_{i=1}^{K} MSE_i$$





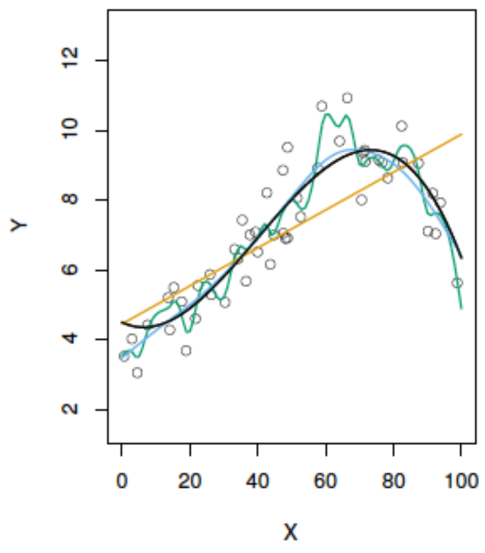
Left: The LOOCV error curve.

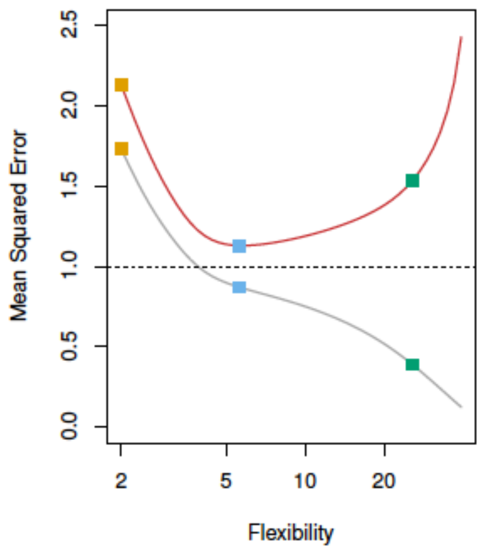
Right: 10-fold CV was run nine separate times, each with a different random split of the data into ten parts.

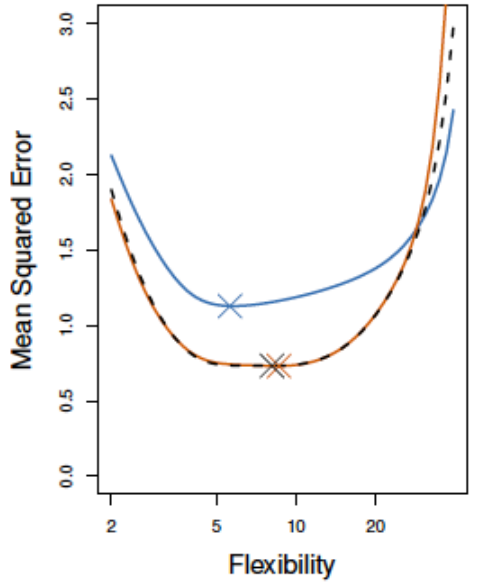
Advantage over LOOCV(K=n):

1. Computationally lighter, especially for complex model with large data.
2. Likely less variance.

Advantage over validation set approach:
Less variability resulting from the data-split, thanks to the averaging.





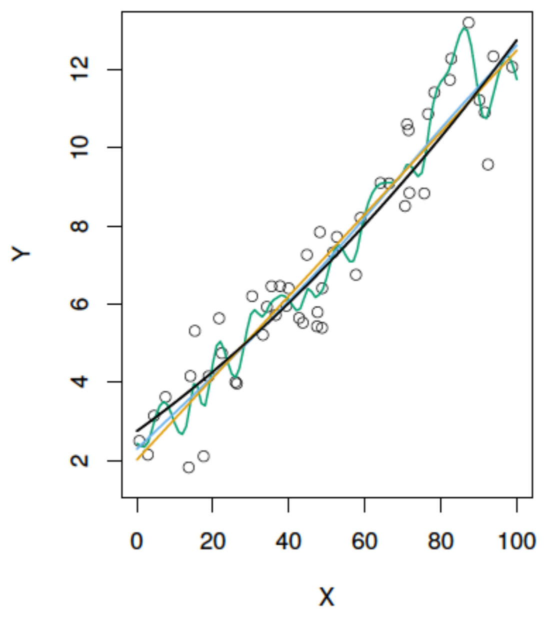


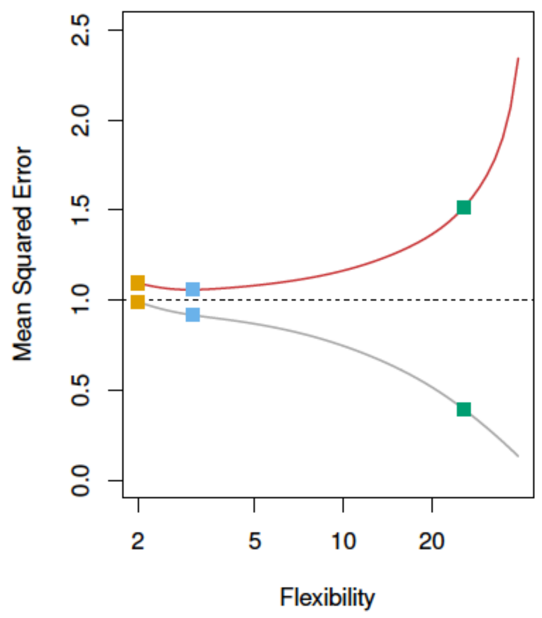
Data simulated from f, shown in black. Three estimates of f are shown: the linear regression line (orange curve), and two polynomial fits (blue and green curves).

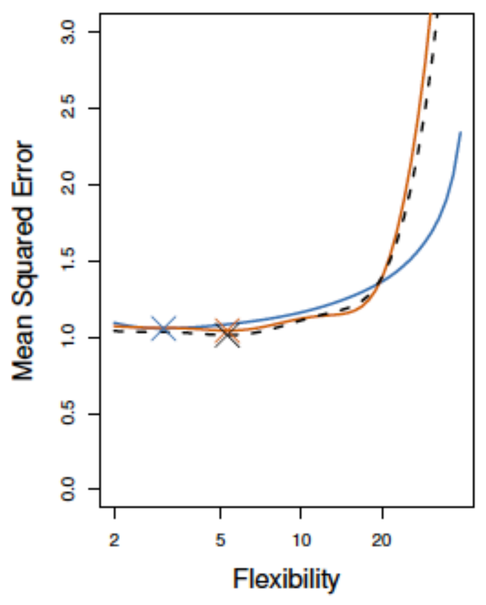
Training MSE (grey curve), test MSE (red curve), and minimum possible test MSE over all methods (dashed line). Squares represent the training and test MSEs for the three fits shown in the left-hand panel.

The true test MSE is shown in blue, the LOOCV estimate is shown as a black dashed line, and the 10-fold CV estimate is shown in orange. The crosses indicate the minimum of each of the MSE curves.

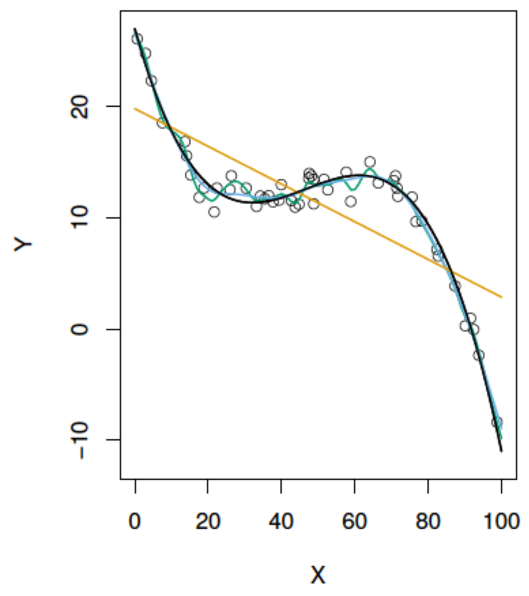
Use a different true f that is much closer to linear. Linear regression provides a very good fit to the data.

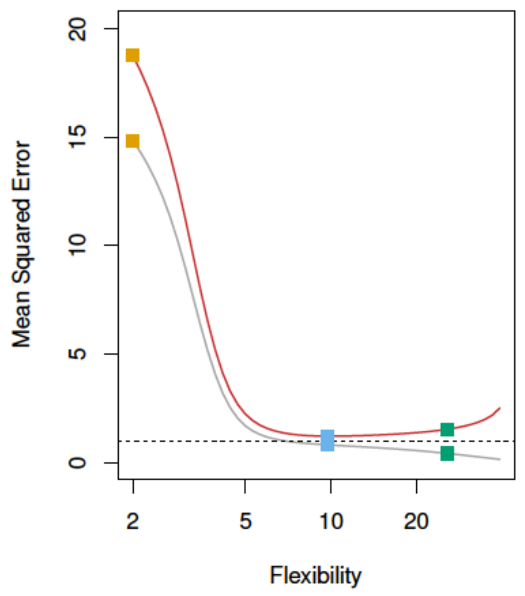


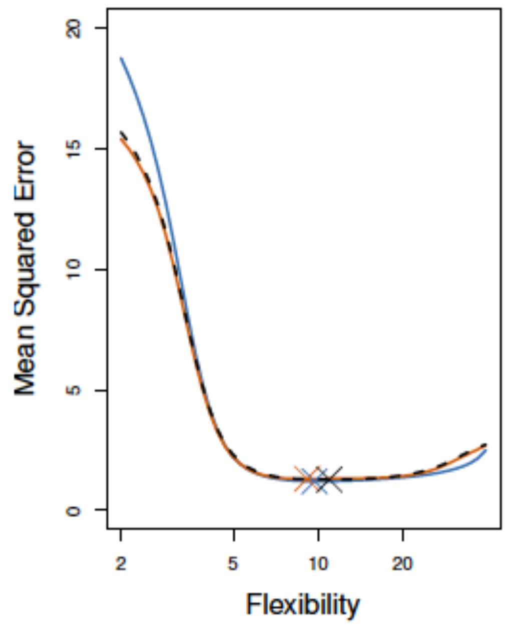




Use a different f that is far from linear. Linear regression provides a very poor fit to the data.







Complexity parameter

A family of models indexed by a parameter, usually representing flexibility or complexity of models, e.g., order of polynomials .
Such parameter is often called tuning parameter; it could even be a number of variables.

Bias variance trade-off

Validation set approach has more bias due to smaller size of training data; LOOCV is nearly unbiased; K-fold (e.g, K=5 or 10) has intermediate bias.

K-fold cross validation has smaller variance than that of LOOCV.

The n training sets LOOCV are too similar to each other. As a result, the trained models are too positively correlated.

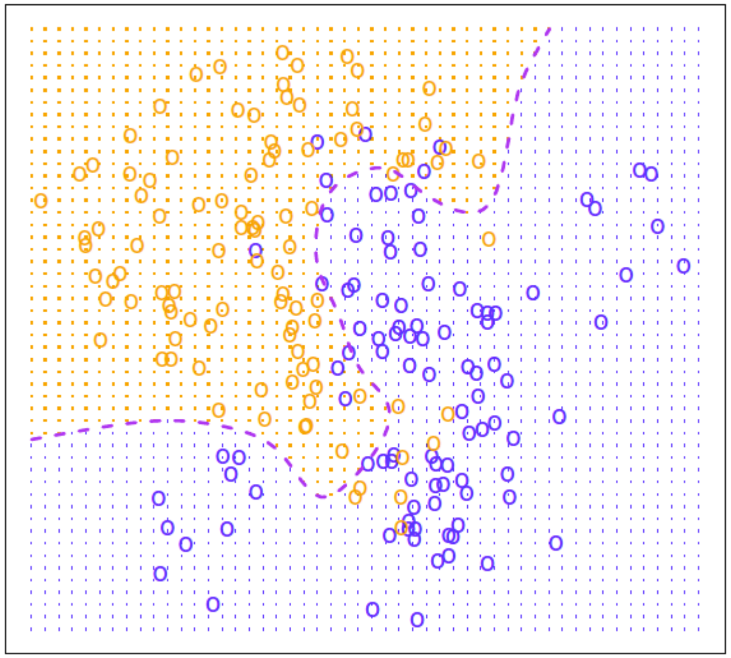
The K training sets of K-fold cross validation are much less similar to each other. As a result, the K-fold cross validation generally has less variance than LOOCV.

➢ Cross validation for classification

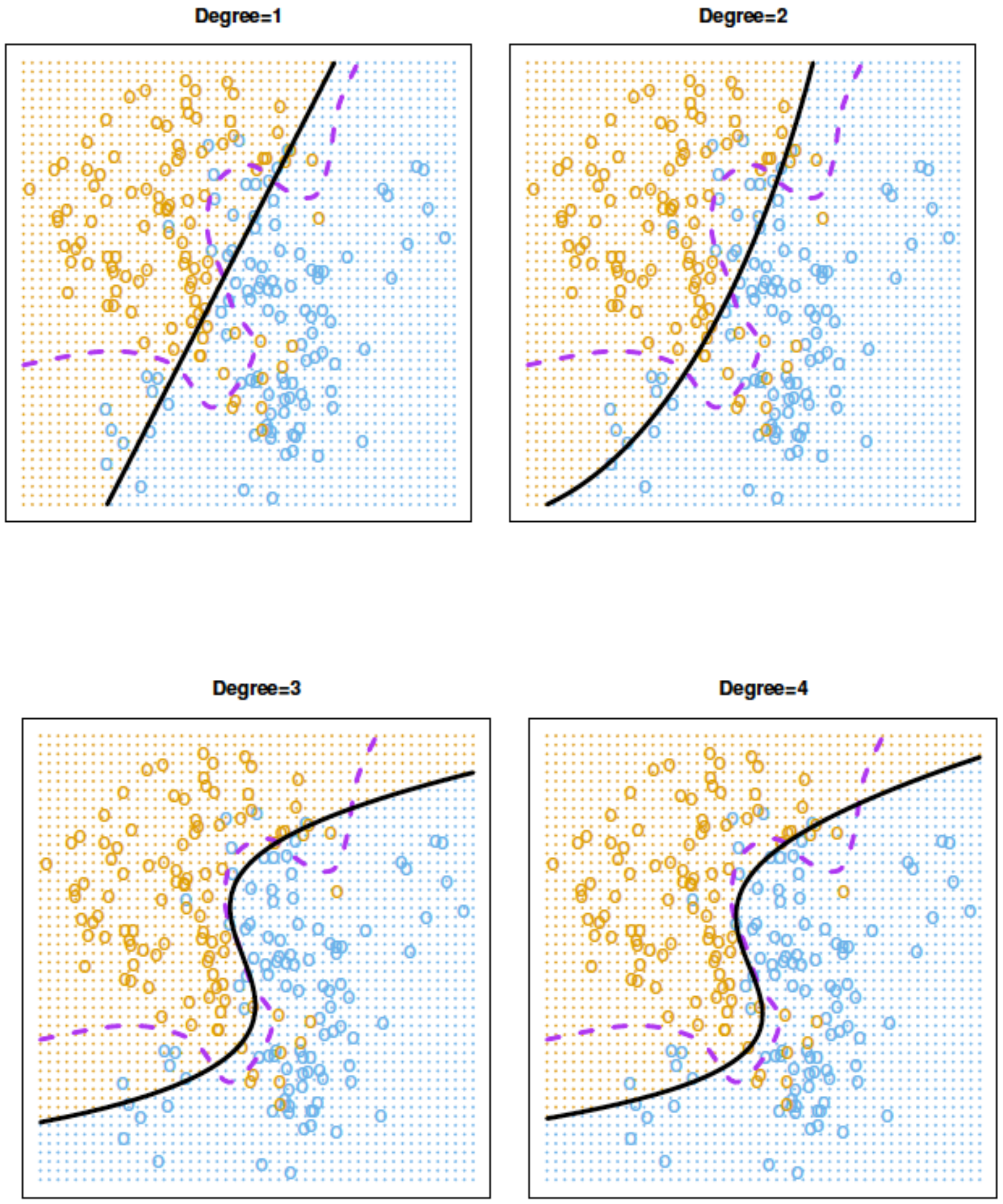
For classification with qualitative response, a natural choice is: 1 for incorrect classification and 0 for correct classification.

We divide the data (randomly) into K equal-sized subsets.

$$CV_{(K)} = \frac{1}{K}\sum_{i=1}^{K} Mean\ Error_{(i)}$$



A simulated data set consisting of 100 observations in each of two groups, indicated in blue and in orange. The purple dashed line represents the Bayes decision boundary. The orange background grid indicates the region in which a test observation will be assigned to the orange class, and the blue background grid indicates the region in which a test observation will be assigned to the blue class.



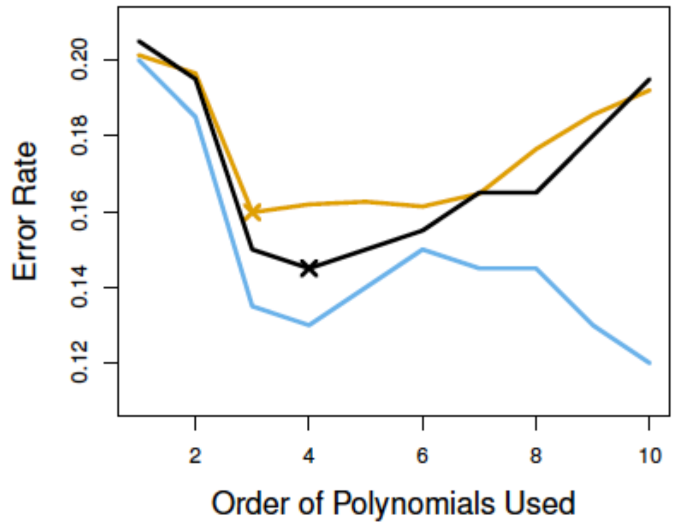
The Bayes decision boundary is represented using a purple dashed line.

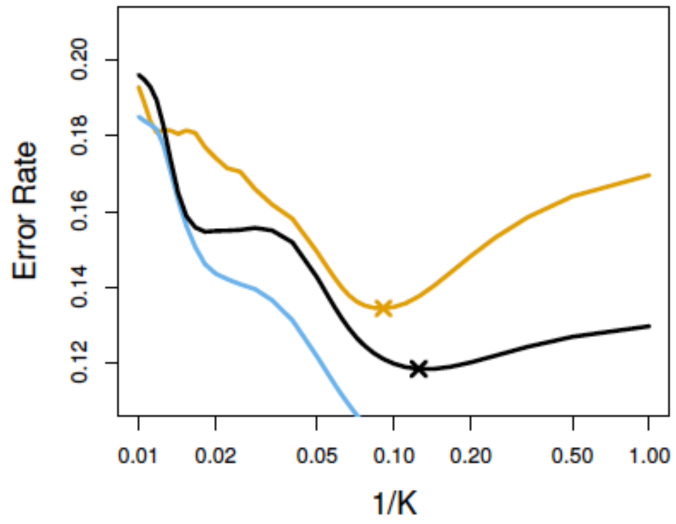
Estimated decision boundaries from linear, quadratic, cubic and quartic (degrees 1 to 4) logistic regressions are displayed in black.

The (true) test error rates for the four logistic regression fits are respectively 0.201, 0.197, 0.160, and 0.162, while the Bayes error rate is 0.133.

In practice the true population distribution is unknown. Thus, the true test error cannot be computed. We use cross validation to solve the problem.

Test error (brown), training error (blue), and 10-fold CV error (black) on the two-dimensional classification.





Left: Logistic regression using polynomial functions of the predictors. The order of the polynomials used is displayed on the x-axis.

Right: The KNN classier with different values of K, the number of neighbors used in the KNN classier.

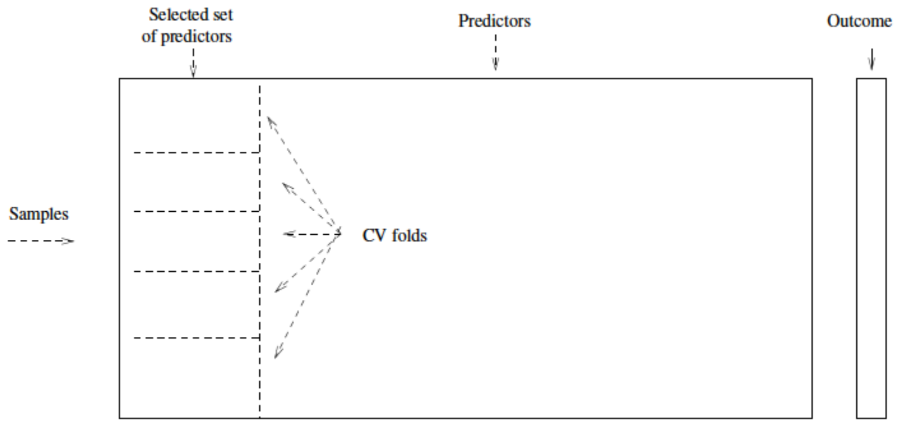
- Training error declines in general when model complexity increases.
- Sometimes even reaches 0.
- Test error general declines first and then increases.
- 10-fold cross validation provides reasonable estimate of the test error, with slight under-estimation.

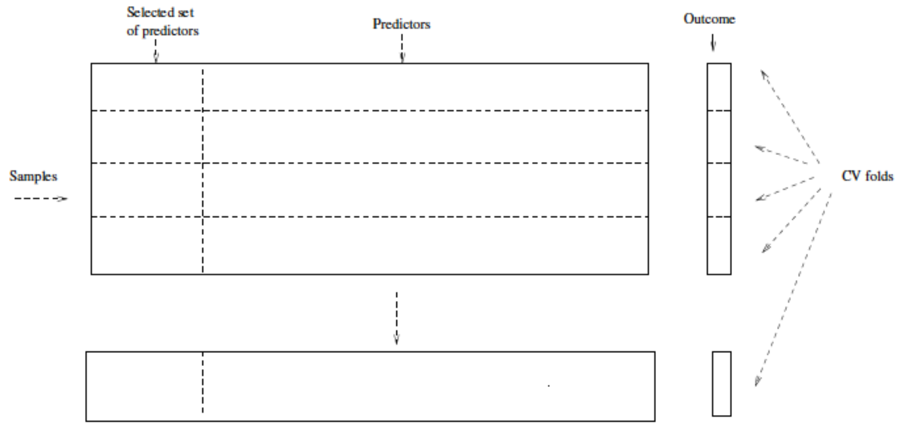
Right and Wrong ways of applying Cross-validation:

Consider a simple classifier applied to some two-class data:
Step 1. Starting with 5000 predictors/features and 50 samples, find the 100 predictors having the largest correlation with the class labels.
Step 2. We then apply a classifier such as logistic regression, using only these 100 predictors.

How do we estimate the test set performance of this classifier? (A or B)

A. Apply cross-validation in step 2.
B. Apply cross-validation to steps 1 and 2.





Split the Data

The test set must simulate a real test scenario, i.e., you want to simulate the setting that you will encounter in real life

- If the data has a temporal component, we will split the data by time.
- If the data is iid, we can split the data uniformly at random.
- **Never** split alphabetically, or by feature values!

- **Bootstrap** (History and background)

The bootstrap (1979 Efron) is a flexible and powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator or statistical learning method. (e.g., estimate of the confidence interval and standard error/deviation of a coefficient.)

The use of the term bootstrap derives from the phrase *to pull oneself up by one's bootstraps*, widely thought to be based on one of the eighteenth century "The Surprising Adventures of Baron Munchausen" by Rudolph Erich Raspe:

> "*The Baron had fallen to the bottom of a deep lake. Just when it looked like all was lost, he thought to pick himself up by his own bootstraps*."

Remark: It is not the same as the term "bootstrap" used in computer science meaning to "boot" a computer from a set of core instructions, though the derivation is similar.

Bootstrap

Suppose we have data$x_{(1)}, \dots, x_{(n)}$with sample mean $\bar{x}$.

Suppose the data are i.i.d. and drawn from a distribution $\boldsymbol{F}$ with unknown mean μ and known variance σ^2. By Central Limit Theorem, we can justify the estimation error $\bar{x} - \mu$ by z-confidence interval or test of hypothesis. The $1 - \alpha$ confidence interval for μ is

$$\left[\overline{x} - z_{\alpha/2}\left(\frac{\sigma}{\sqrt{n}}\right), \overline{x} + z_{\alpha/2}\left(\frac{\sigma}{\sqrt{n}}\right)\right]$$

For example, the 95% confidence interval of μ is

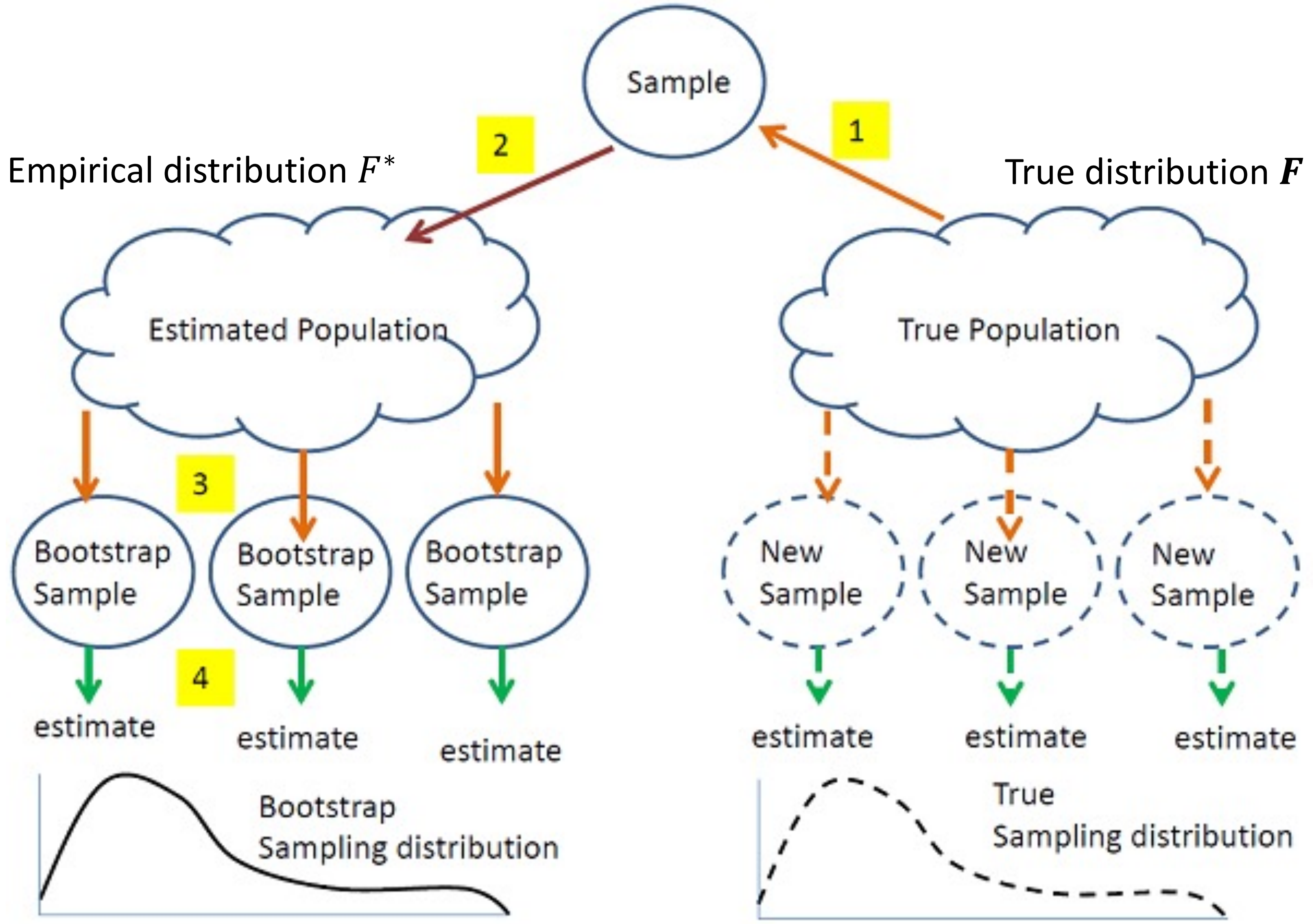
$$\left[\overline{x} - 1.96\frac{\sigma}{\sqrt{n}}, \ \overline{x} + 1.96\frac{\sigma}{\sqrt{n}}\right]$$

If the variance is unknown, we can use the t-confidence interval or test of hypothesis.

$$\left[\bar{x} - \frac{t_{n-1,\alpha/2}\ s}{\sqrt{n}}, \qquad \bar{x} + \frac{t_{n-1,\alpha/2}\ s}{\sqrt{n}}\right]$$

The **bootstrap** is another general tool for assessing statistical accuracy.

Bootstrap principle: $F^* \approx \boldsymbol{F}$.



Procedure of using bootstrap for estimation error $\bar{x} - \mu$:

1) Take $\boldsymbol{n}$ random sample (with replacement) from $x_{(1)}, \ldots, x_{(n)}$. Calculate the sample mean of the "re-sample", denoted as $\bar{x}_1^*$.
2) Repeat the above a large number M times. We have $\bar{x}_1^*, \bar{x}_2^*, \ldots \bar{x}_M^*$.
3) Use the distribution of $\bar{x}_1^* - \bar{x}, \ldots, \bar{x}_M^* - \bar{x}$ to approximate that of $\mu - \bar{x}$.

- The idea is to treat the data distribution (*empirical distribution*) as a proxy of the population distribution.
- Mimic the data generation from the true population.
- Mimic your statistical procedure.
- Evaluate your statistical procedure.

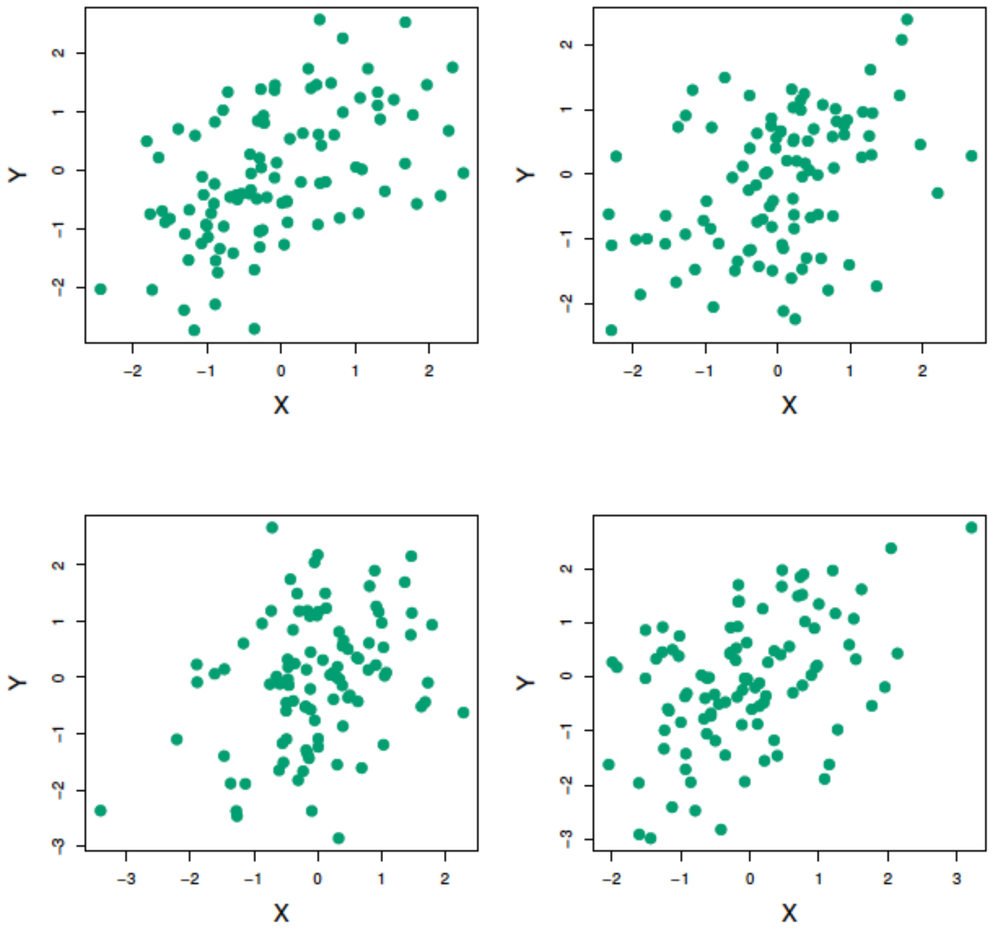
Bootstrap Example: (from page 187 of ISLR)

- Invest a fixed sum of money in two financial assets that yield returns of X and Y , respectively, where X and Y are random quantities.
- Invest a fraction p of our money in X, and will invest the remaining $1 - p$ in Y
- Choose p to minimize the total risk, or variance $\mathrm{Var}(pX + (1 - p)Y)$, of the investment.
- The value p that minimizes the risk $\mathrm{Var}(pX + (1 - p)Y)$ is given by

$$p = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$$

- We can compute **estimates** for these quantities $\sigma_X^2 = Var(X)$, $\sigma_Y^2 = Var(Y)$, and $\sigma_{XY} = Cov(X, Y)$ using a data set that contains measurements for X and Y.

- To estimate the standard deviation of p, we repeated the process of simulating 100 paired observations of X and Y.

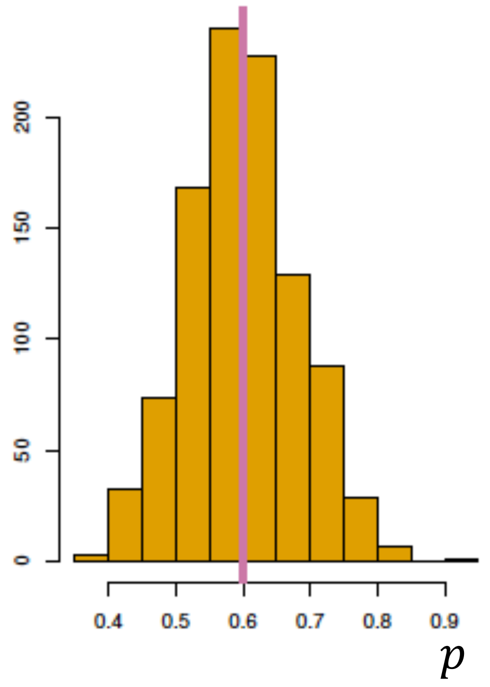


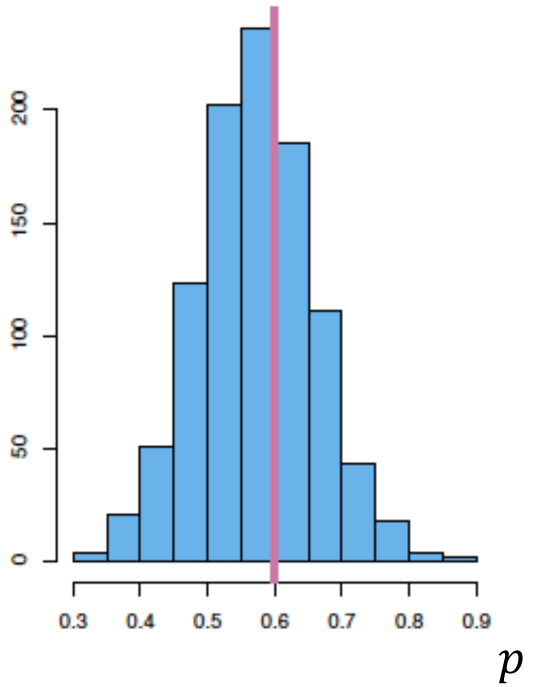
Each panel displays 100 simulated returns for X and Y. From left to right and top to bottom, the resulting estimates for are 0.576, 0.532, 0.657, and 0.651.

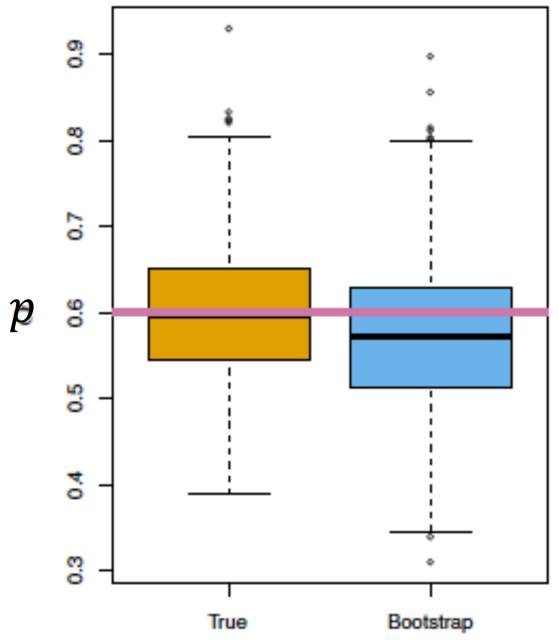
Simulations the parameters were set to $\sigma_X^2 = 1, \sigma_Y^2 = 1.25$, and $\sigma_{XY} = 0.5$, and so the true value of p is 0.6.

1. Repeat the simulation 1000 times and estimates p 1000 times: $p_1, p_2, \ldots, p_{1000}$.

2. From a single data set, bootstrap 1000 new sample sets.



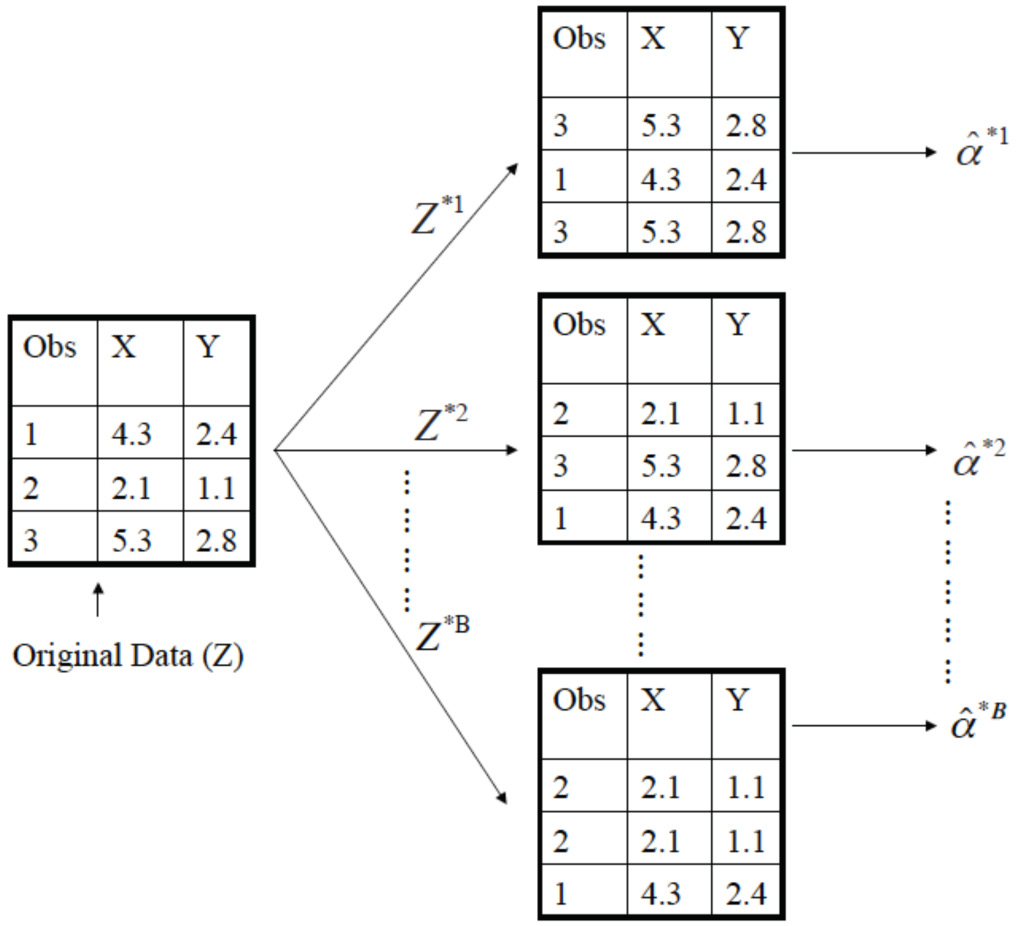




Left: A histogram of the estimates of p obtained by generating 1,000 simulated data sets from the **true population**.

Center: A histogram of the estimates of p obtained from 1,000 **bootstrap** samples from a single data set.

Right: The estimates of p displayed in the left and center panels are shown as boxplots. In each panel, the pink line indicates the **true** value of p.



Obs	X	Y
1	4.3	2.4
2	2.1	1.1
3	5.3	2.8

Obs	X	Y
3	5.3	2.8
1	4.3	2.4
3	5.3	2.8

Obs	X	Y
2	2.1	1.1
3	5.3	2.8
1	4.3	2.4

Obs	X	Y
2	2.1	1.1
2	2.1	1.1
1	4.3	2.4

An illustration of the bootstrap approach on a small sample containing $n = 3$ observations. Each bootstrap data set contains n observations, sampled with **replacement** from the original data set.
Each bootstrap data set is used to obtain an estimate of p.

Parametric bootstrap

Use the parametric bootstrap to estimate a confidence interval for a parameter.

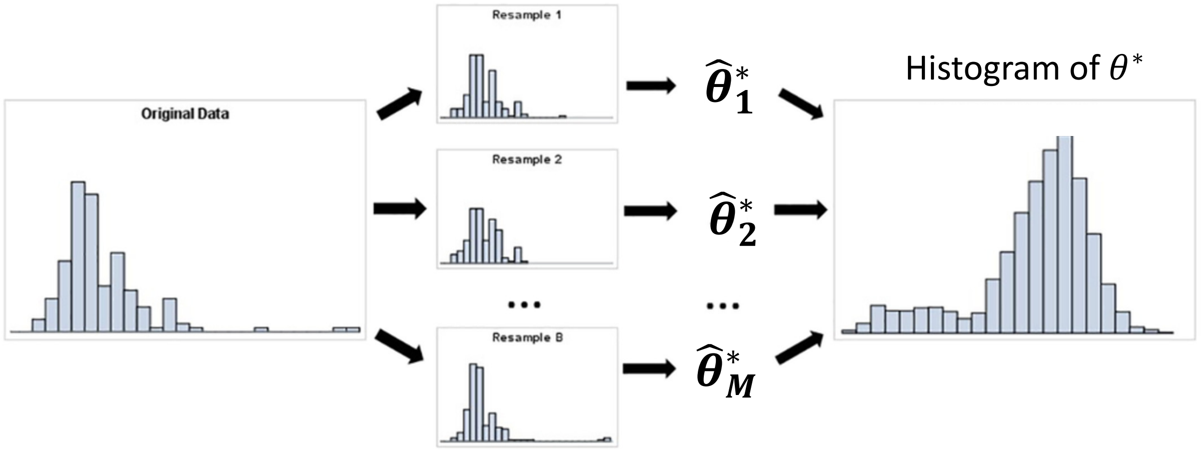
We generate the bootstrap sample from a parametrized distribution.

Data: $x_{(1)}, \dots, x_{(n)}$ drawn from a distribution $h_\theta(\vec{x})$ with unknown parameter θ.
A statistic $\widehat{\boldsymbol{\theta}}$ that estimates θ.

For each bootstrap sample $x^*_{(1)}, \dots, x^*_{(n)}$, compute $\widehat{\boldsymbol{\theta}}^*$.

Repeat the above a large number M times, we get $\widehat{\boldsymbol{\theta}}^*_1, \dots, \widehat{\boldsymbol{\theta}}^*_M$.

We can get the $1-\alpha$ confidence interval for θ from the results $\widehat{\boldsymbol{\theta}}^*_1, \dots, \widehat{\boldsymbol{\theta}}^*_M$


Original Data
Resample 1
Resample 2
...
Resample B
$\widehat{\boldsymbol{\theta}}^*_1$
$\widehat{\boldsymbol{\theta}}^*_2$
...
$\widehat{\boldsymbol{\theta}}^*_M$
Histogram of θ^*

Remarks:

In more complex data situations, figuring out the appropriate way to generate bootstrap samples can require some thought.

For example, if the data is a ***time series***, we can't simply sample the observations with replacement. We can instead create blocks of consecutive observations, and sample those with replacements. Then we paste together sampled blocks to obtain a bootstrap dataset.

Question: Can the bootstrap estimate prediction error, as from cross-validation?

original sample = validation sample
bootstrap dataset = training sample
or the other way around?

➢ **Degrees of freedom** (Optional)

Degrees of freedom represent the number of points of control of a system, model, or calculation. (The Effective Number of Parameters)

degrees of freedom = number of independent values – number of statistics

Suppose data is generated by $y = h(\vec{x}) + \epsilon$, with $\mathrm{E}[\epsilon] = 0$, and $\mathrm{Var}(\epsilon) = \sigma^2$.

Suppose we fit some $\hat{y} = \hat{h}(\vec{x})$ by an iid training set of size n.

The number of degrees of freedom of $\hat{h}$ is

$$\mathrm{df}(\hat{h}) = \frac{1}{\sigma^2}\sum_{i=1}^{n}\mathrm{Cov}\left(\hat{y}^{(i)}, y^{(i)}\right) = \frac{1}{\sigma^2}\mathrm{Trace}(\mathrm{Cov}(\hat{\vec{y}}, \vec{y}))$$

The variance σ^2 is estimated by $\mathrm{s}_{\mathrm{d}}^2 = RSS/(n - d - 1)$

Examples of degree of freedom:

1. $\mathrm{df}(\overline{y}) = 1,$ where $\overline{y} = \frac{1}{n}\left(y^{(1)} + \cdots + y^{(n)}\right)$ is the mean predictor.

2. The identity predictor has degree of freedom n.

3. For a linear model with d inputs, the RSS solutions has $\mathrm{df}\left(\vec{\theta}\right) = d + 1$.

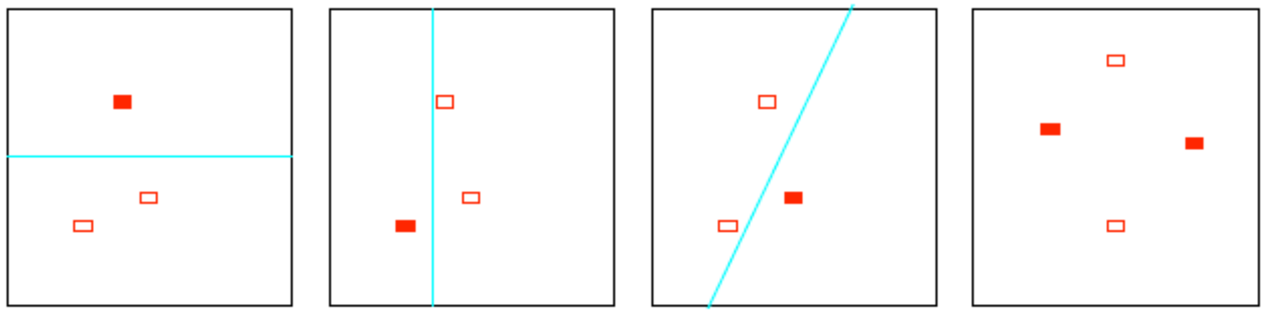
4. The k-nearest neighbors has degree of freedom n/k.

➢ **Vapnik–Chervonenkis (VC) Dimension** (optional)

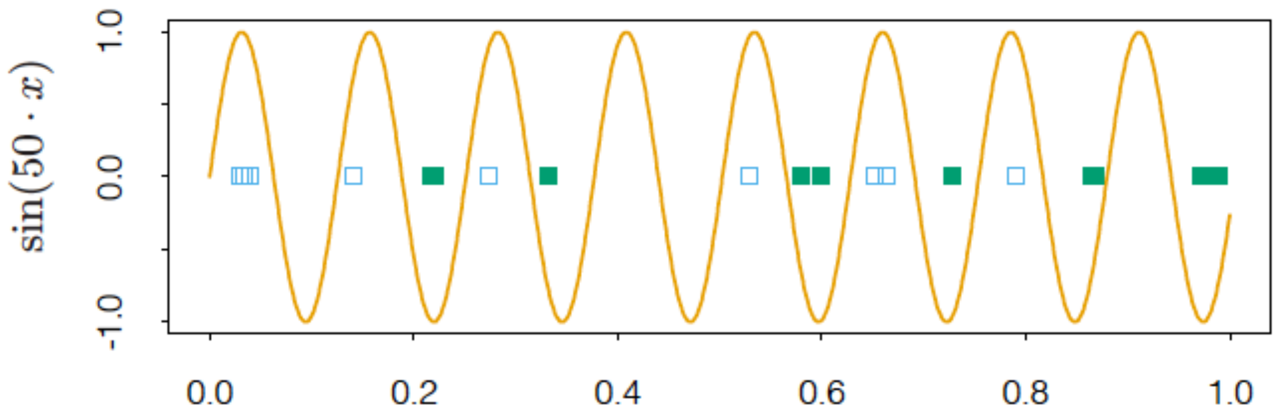
The Vapnik–Chervonenkis dimension is another way of measuring the complexity of a class of functions by assessing how wiggly its members can be.

Suppose we have a class of functions $\{f(\vec{x}, \vec{\alpha})\}$ indexed by a parameter vector $\vec{\alpha}$, with $\vec{x} \in \mathbb{R}^d$.

The VC dimension of the class $\{f(\vec{x}, \vec{\alpha})\}$ is defined to be the largest number of points (in some configuration) that can be shattered by members of $\{f(\vec{x}, \vec{\alpha})\}$.



Assume for now that f is an indicator function taking the values 0 or 1.


$\sin(50 \cdot x)$