Week 4: Resampling/Bootstrap & Model Validation and Selection

AI 539: Machine Learning for Non-Majors

Alireza Aghasi

Oregon State University

Super easy to understand !

Bootstrap

$y = \beta_0 + \beta_1 x + \varepsilon$, $\text{var}(\varepsilon) = \sigma^2$, $(x_1, y_1) \sim (x_n, y_n)$

$\hat{\beta}_0 =$

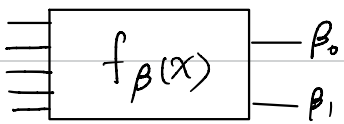
$$\hat{\beta}_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}$$

$E[\hat{\beta}_1] = \beta_1$ # , $\text{Var}(\hat{\beta}_1) = \frac{\sigma^2}{\sum (x_i - \bar{x})^2}$ #

What if $y = \beta_0 + \log|\beta_1 x|$

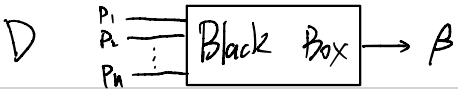
dataset

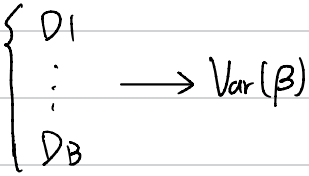
$D \{ (x_1, y_1), \vdots, (x_n, y_n)$

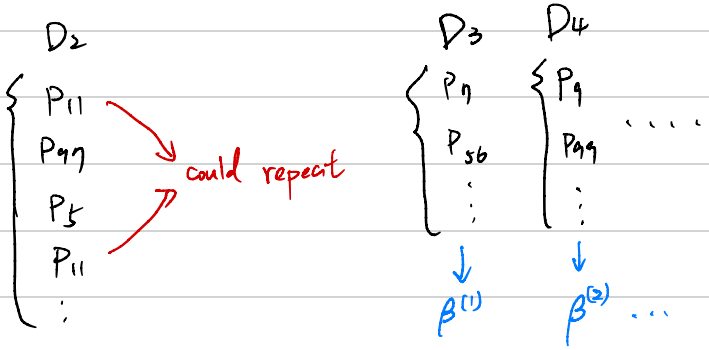


With limited data set, there won't be multiple β_0, β_1

So $\{ D_1 \rightarrow \text{Var}(\beta_1)$, D_2, $\vdots$, D_B







So $\beta^{(1)}, \beta^{(2)} \cdots \beta^{(B)} \rightarrow$

- average
- variance
- histogram
- Confidence Interval

Bootstrap

- The bootstrap is a flexible and very powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator or statistical learning method
- It can provide an estimate of the standard error of a coefficient, or a **confidence interval for that coefficient**, regardless of how complex the derivation of that coefficient is

A Very Simple Example

- Think of a block, which takes 100 randomly generated samples drawn from a distribution and generates their sample mean $\bar{x}$ as the output
- We want to see if we do this many, many times, how the outputs of our block vary
- In principal to do this we would need to run the experiment, for say 1000 times, and then look at the variations of $\bar{x}$, however, this requires constant access to the distribution and a lot of sampling (which in many applications can be expensive to acquire)
- Bootstrap can help us do this without accessing the reference distribution
- Let's see the code

Bootstrap via an Example

- Lets explain bootstrap via an example
- Suppose that we wish to invest a fixed sum of money in two financial assets that yield returns of X and Y (random quantities)
- We will invest α shares in X, and will invest the remaining $1-\alpha$ in Y
- To minimize the risk, we want to minimize $var(\alpha X + (1-\alpha)Y)$
- We can show that (in the class we do it) that the minimizer is

$$\alpha = \frac{var(Y) - cov(X,Y)}{var(X) + var(Y) - 2cov(X,Y)}$$

$X, Y \in RV$, if independent $\Rightarrow Var(X+Y) = Var(X) + Var(Y)$

$X, Y \in RV$, not independent $\Rightarrow Var(X+Y) = Var(X) + Var(Y) + 2\,Cov(X,Y)$

$$V = Var(\alpha X + (1-\alpha) Y)$$

$$= \alpha^2 Var(X) + (1-\alpha)^2 Var(Y) + 2\,Cov(\alpha X, (1-\alpha) Y)$$

$$= \alpha^2 Var(X) + (1-\alpha)^2 Var(Y) + 2\alpha(1-\alpha)\,Cov(X,Y)$$

$$0 = \frac{dV}{d\alpha} = 2\alpha\,Var(X) + 2(1-\alpha)(-1)\,Var(Y) + (2-4\alpha)\,Cov(X,Y)$$

$$\rightarrow \alpha_{opt} = \frac{Var(Y) - Cov(X,Y)}{Var(X) + Var(Y) - 2\,Cov(X,Y)}$$

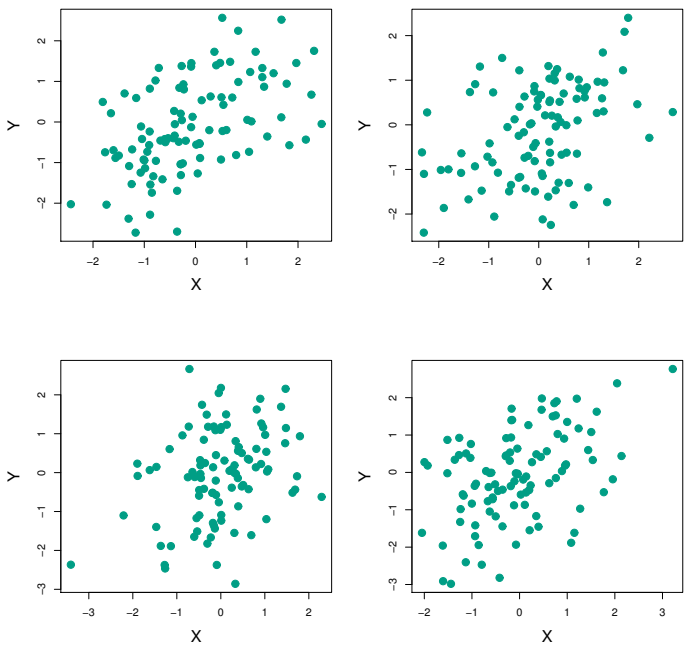
Bootstrap via an Example

- In real-world we do not know $var(X), var(Y), cov(X, Y)$
- Suppose we are given a data set containing pairs of X and Y. We can estimate $var(X), var(Y), cov(X, Y)$ from the sample set and get an estimate $\hat{\alpha}$ for the optimal share
- Ideally we can generate these sample sets many times, and estimate an $\hat{\alpha}$ for each and look into the histogram
- However in a real-world we only have one sample set to use
- Bootstrap yet allows us to generate good estimates of α **using only one sample set!**

Bootstrap via an Example

To see how nicely Bootstrap works, lets compare its outcome with the case that α is generated from many synthetic sample generations

- We generate 1000 sample sets each containing 100 pairs of X, Y
- For the synthetic data generated $var(X) = 1, var(Y) = 1.25$ and $cov(X, Y) = 0.5$ which yield an optimal value of $\alpha = 0.6$

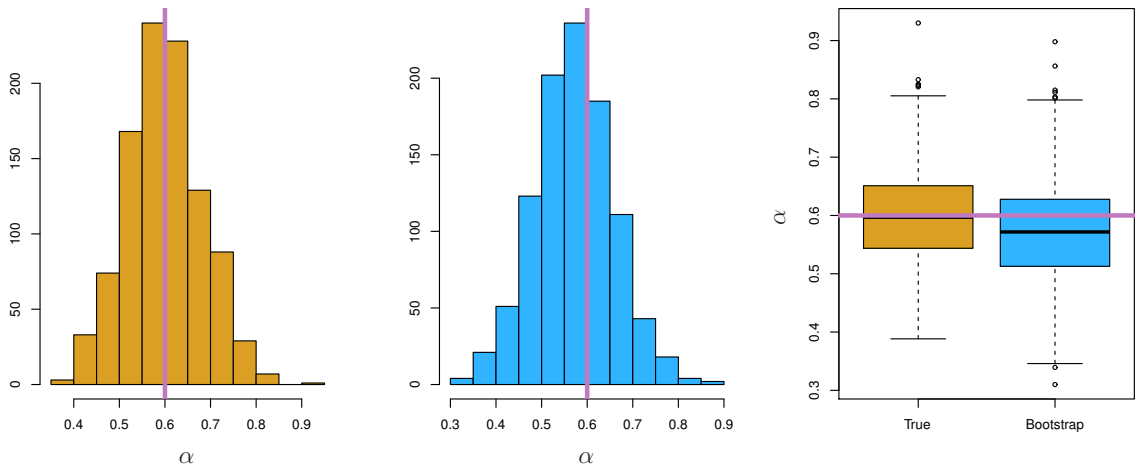


Bootstrap via an Example

- To get the left panel we generate 1000 synthetic sample sets, for each obtain $\hat{\alpha}$ and plot the histogram and calculate

$$\bar{\alpha} = \frac{1}{1000}\sum_{i=1}^{1000}\hat{\alpha}_i, \quad SE(\alpha) = \sqrt{\frac{1}{999}\sum_{i=1}^{1000}(\hat{\alpha}_i - \bar{\alpha})^2}$$

- For the bootstrap we only use one of the sample sets and regenerate new sample set by sampling with replacement
- Surprisingly, the results are very close



Bootstrap General Framework

- Suppose a black-box calculates $\hat{\alpha}$ from a sample set, e.g., a coefficient in linear or logistic regression
- We are interested in estimating the variability of $\hat{\alpha}$ without examining many new sample sets
- Denoting the first bootstrap data set by Z^1, we use Z^1 to produce a new bootstrap estimate for α, which we call $\hat{\alpha}^1$
- This procedure is repeated B (say 100 or 1000) times, in order to produce B different bootstrap data sets, $Z^1, Z^2, \cdots, Z^B$, and the corresponding α estimates, $\hat{\alpha}^1, \cdots, \hat{\alpha}^B$
- We estimate the standard error of these bootstrap estimates using the formula

$$SE_B(\alpha) = \sqrt{\frac{1}{B-1}\sum_{i=1}^{B}(\hat{\alpha}^i - \bar{\hat{\alpha}})^2} \quad \text{where} \quad \bar{\hat{\alpha}} = \frac{1}{B}\sum_{i=1}^{B}\hat{\alpha}^i$$

- This serves as an estimate of the standard error of α estimated from the original data set!

Programming Exercise

Let's go through some programming exercises

- Basic Example
- Linear model example

(x_1, y_1) $\qquad y = \beta_0 + \beta_1 x$

$\vdots$

(x_n, y_n) $\qquad y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$

$$\rightarrow \varepsilon_i = y_i - \beta_0 - \beta_1 x_i \begin{cases} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_{100} \end{cases}$$

$x_1, y_1^{(1)}$	$x_1, y_1^{(2)}$
$\vdots$	$\vdots$
x_{100}, y_{100}	x_{100}, y_{100}

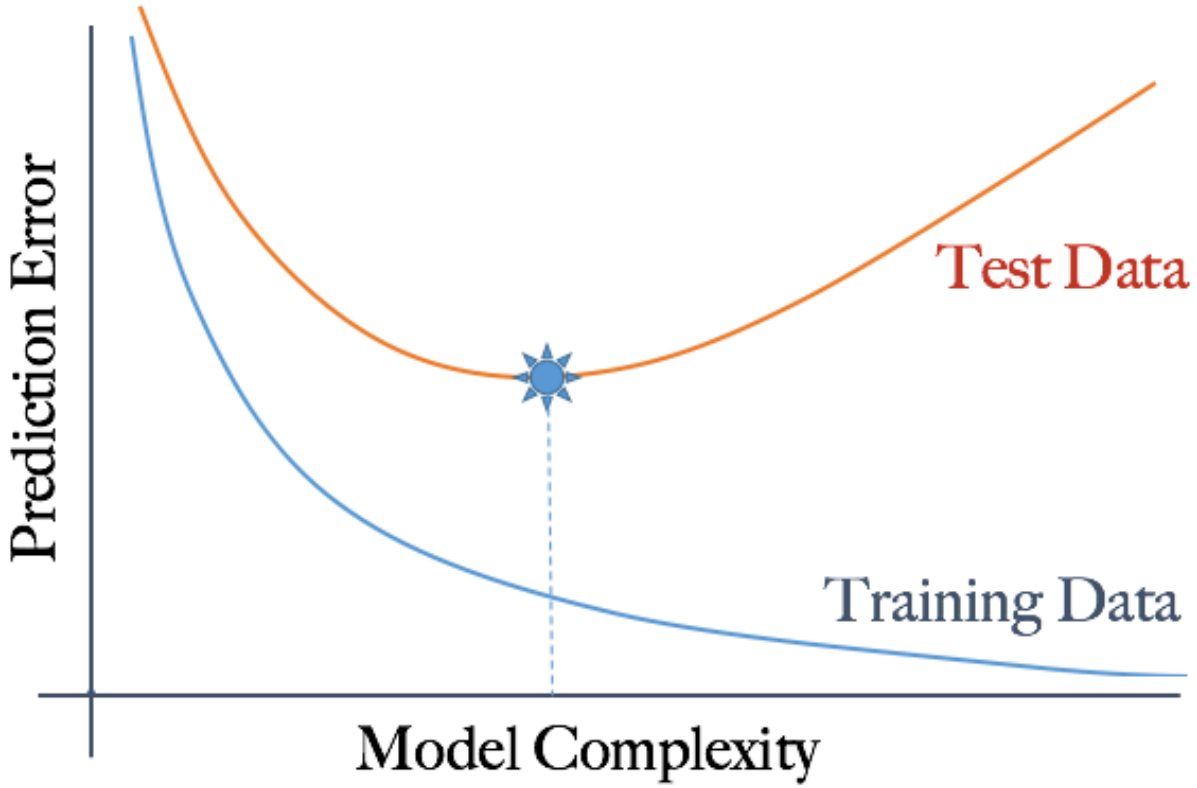
Model Validation and Selection

Introduction

- Recall that we fitted out models using training data and were interested in evaluating the performance with respect to independent test data
- To produce justifiable model reliability arguments, the test data should not be used in the training
- If the model is evaluated against the training data the results can be very distracting
- The training error rate is often quite different from the test error rate, and in particular the former can dramatically underestimate the latter (recall the accuracy vs complexity chart)

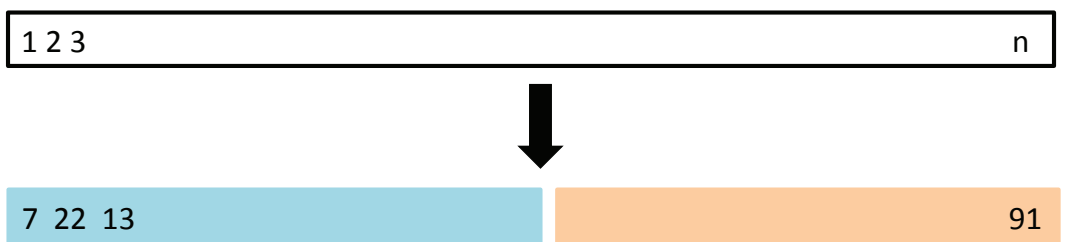
Training vs Test Model Evaluation

Recall this plot from the first session



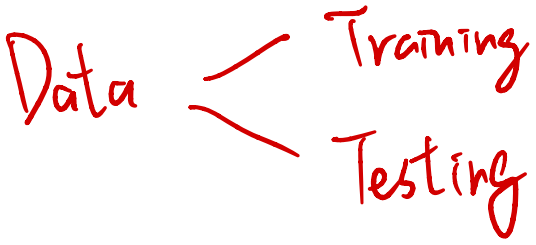
Real-World Data Issues and Test Performance

- A good evaluation is possible when a large test set is available
- Often such set is not available
- We are interested in a class of 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



Validation Set Approach

Approach 1



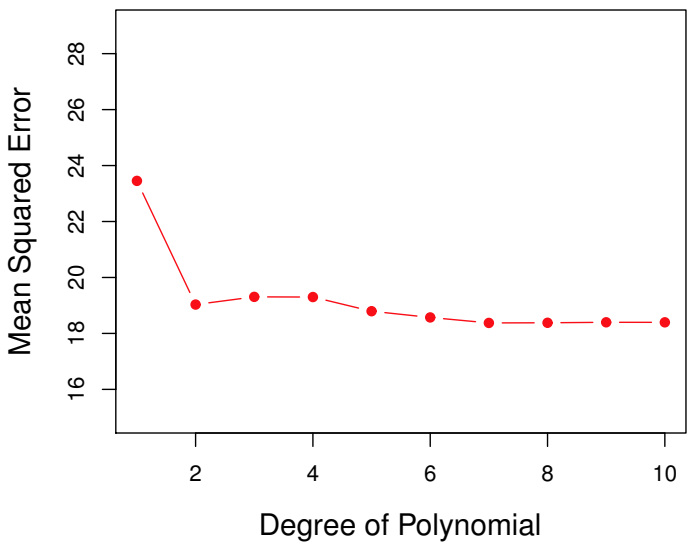
- This is the standard approach we have been using so far
- We randomly divide the available set of samples into two parts: a training set and a validation or hold-out set (sometimes 50%-50% splitting, often 80%-20% splitting)
- The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set
- The error with reference to the hold-out set is an approximation of the test error

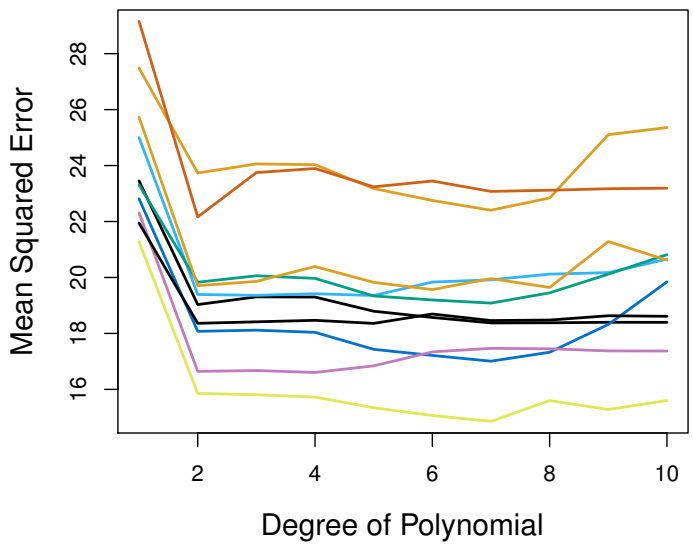
example

- Recall the automobile data: Regressing Mile per Gallon in terms of the Horse Power

$$\texttt{mpg} = \beta_0 + \sum_{i=1}^{p} \beta_i \texttt{(horsepower)}^i$$

- We randomly split the 392 observations into two sets, a training set containing 196 of the data points, and a validation set containing the remaining 196 observations.





Validation Set Cons & Pros

Pro – The procedure is **simple to do** (as we have done so far) and only a subset of the observations — those that are included in the training set rather than in the validation set — are used to fit the model

Con – The validation estimate of the test error can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the validation set

Con – While the estimated test error vary a lot, **finding information such as model selection is still possible**

Con – Since a large portion of the data need to be held aside, the model fits are not accurate enough

Leave-One Out Cross-Validation (LOOCV) Approach 2

- We have n data points $(x_1, y_1), \cdots, (x_n, y_n)$, we use $n-1$ for the training and one instance for the test
- Of course a single test point is no where close to the true test error, but this process is repeated n times, every time $n-1$ points used for training and one point left out for the test
- Considering $MSE_1 = (y_1 - \hat{y}_1)^2, \cdots MSE_n = (y_n - \hat{y}_n)^2$, an approximation of the test error is

$$CV_n = \frac{1}{n} \sum_{i=1}^{n} MSE_i$$

Only one sample for testing each time

Cons & Pros with LOOCV

Pro – It has a very small bias compared to the validation set approach (it almost uses as much data as possible to fit the model)

Pro – The test error overestimation is less than the validation set approach (because of what we mentioned above)

Pro – Its results are reproducible unlike the validation set approach which uses a random subset of the data for test evaluation

Con – It can be computationally very expensive (requires running the algorithm n times)

– For linear models there is a shortcut to calculate CV_n that only requires fitting the model once with the entire data (but **this shortcut only applies to linear models**)

$$CV_n = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2$$

where $\hat{y}_i$ are the fitted values of the original least squares problem and h_i are only data dependent

K-Fold Cross Validation

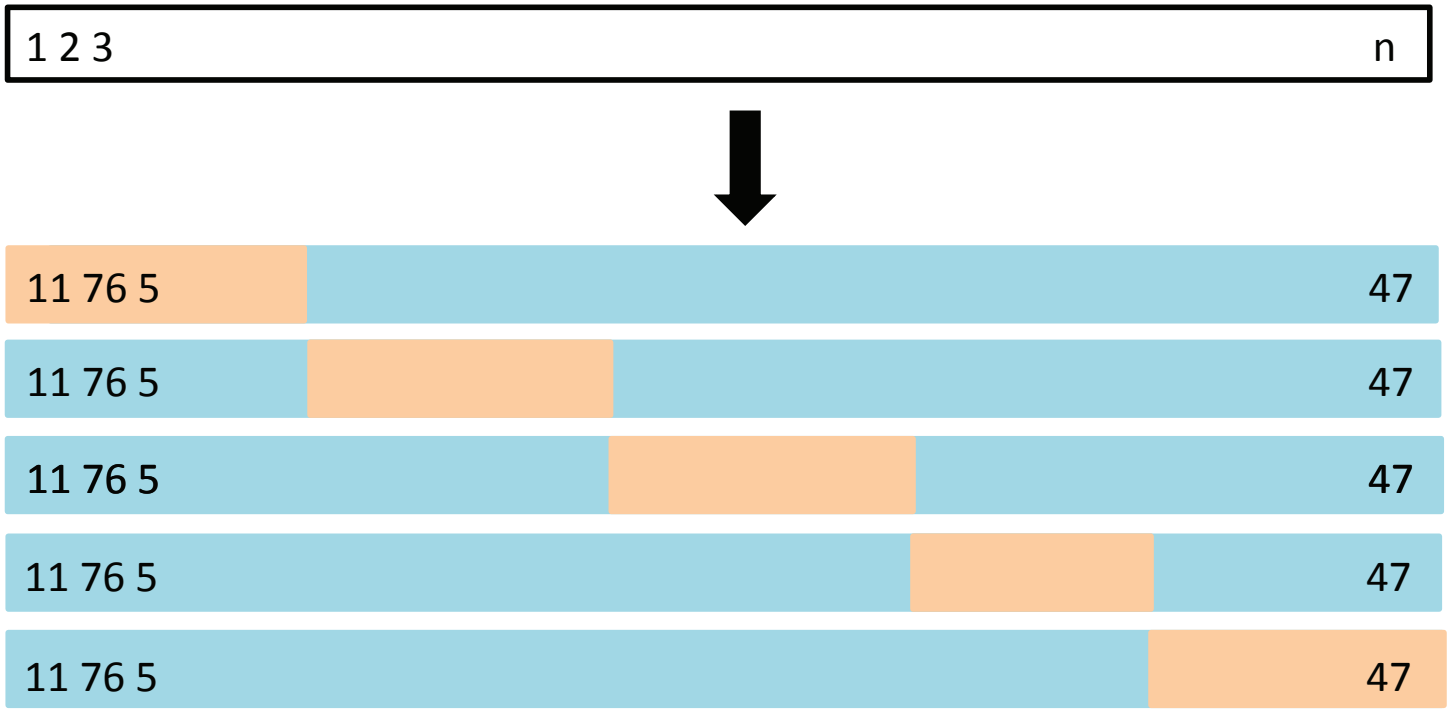
Approach 3

- Widely used approach for estimating test error
- This approach involves **randomly dividing the set of observations into K groups**, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining $K - 1$ folds
- This is done in turn for each part $k = 1, 2, \cdots, K$, and then the results are combined
- Estimates can be used to select best model, and to give an idea of the test error of the final chosen model

$$CV_K = \frac{1}{K} \sum_{k=1}^{K} MSE_k$$

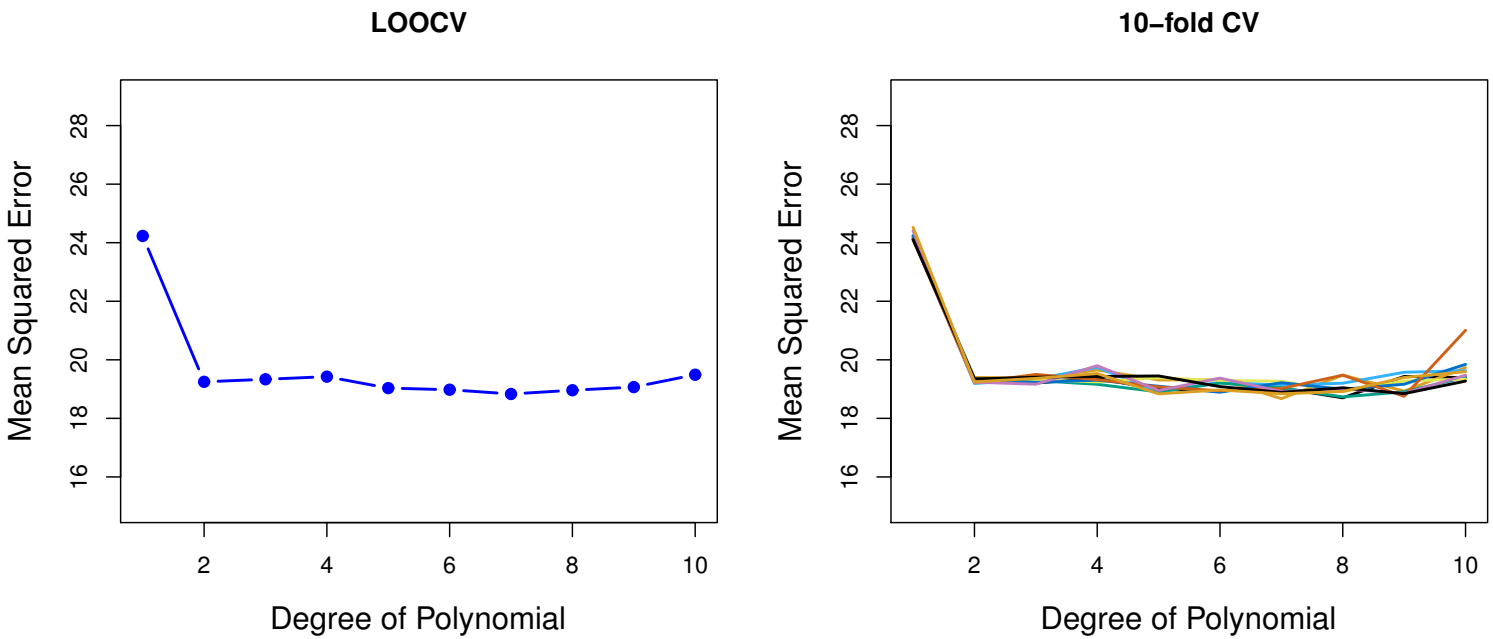
- Often $K = 5$ or $K = 10$ is what is considered in application

K-Fold Cross Validation



K-Fold Cross Validation and LOOCV

- LOOCV is a special case of K-fold CV for $K = n$
- In general K-fold CV is much cheaper than LOOCV because it only requires K model fits vs n model fits
- For model selection, K-fold CV often gives us similar outcomes at a much lower computational cost



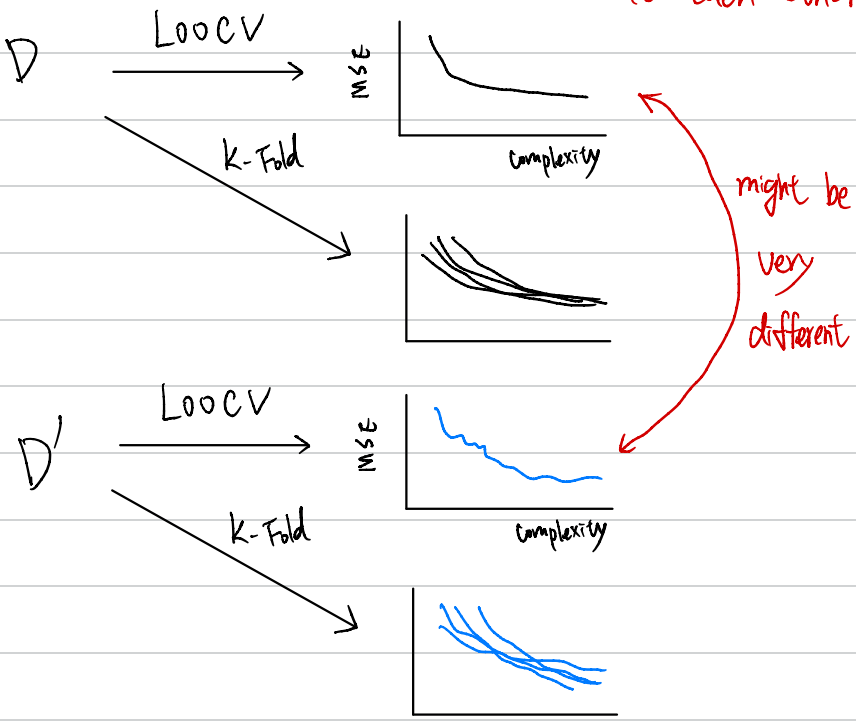
K-Fold Cross Validation and LOOCV

- Aside form the computational issues, even surprisingly K-Fold CV produces better test estimates than the LOOCV
- LOOCV has a lower bias compared to the K-fold CV, since it uses more data to fit the model
- But K-fold CV has a lower variance compared to the LOOCV, since LOOCV is the sum of n highly correlated random variables while the correlation between the MSEs in K-fold CV is lower, recall

$$var(X + Y) = var(X) + var(Y) + 2cov(X, Y)$$

Better than LOOCV in terms of Variance

Since the training sets in LOOCV are highly correlated to each other.



i.g.

$X, Y \in RV$, if independent $\Rightarrow Var(X+Y) = Var(X) + Var(Y)$

$X, Y \in RV$, not independent $\Rightarrow Var(X+Y) = Var(X) + Var(Y) + 2\,Cov(X, Y)$

Suppose $X = Y \Rightarrow Var(X, Y) = Var(2X) = 4\,Var(X)$

CV & Classification

- We divide the data into K roughly equal-sized index sets $C_1, \cdots, C_K$
- Compute

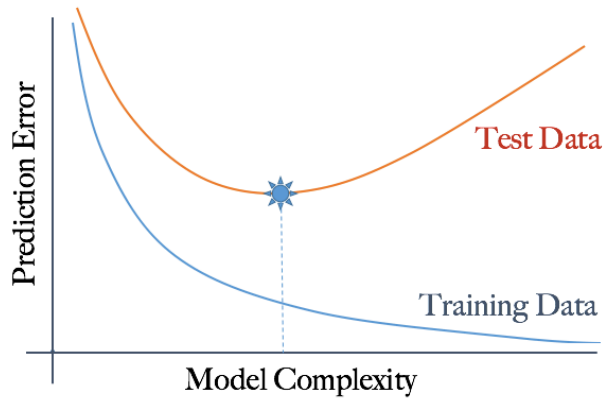
$$CV_K = \frac{1}{K} \sum_{k=1}^{K} Err_k$$

where

$$Err_k = \frac{1}{\#\text{ elements in } C_k} \sum_{i \in C_k} I(y_i \neq \hat{y}_i)$$

Cross Validation Summary

- As mentioned earlier, model selection based on the RSS or R^2 statistics can be misleading, since the training error is not a good representative of the actual test error

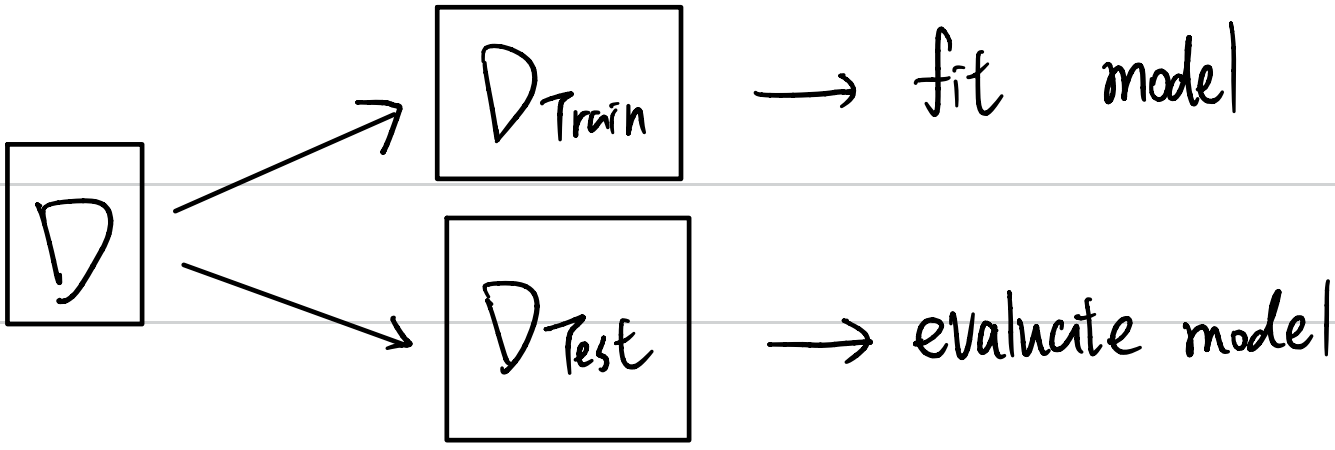


- Instead through a process of splitting the data into training and validations sets, we were able to use LOOCV or K-Fold CV as estimates of the test error
- We discussed why K-Fold CV is a more desirable estimate, computationally and statistically

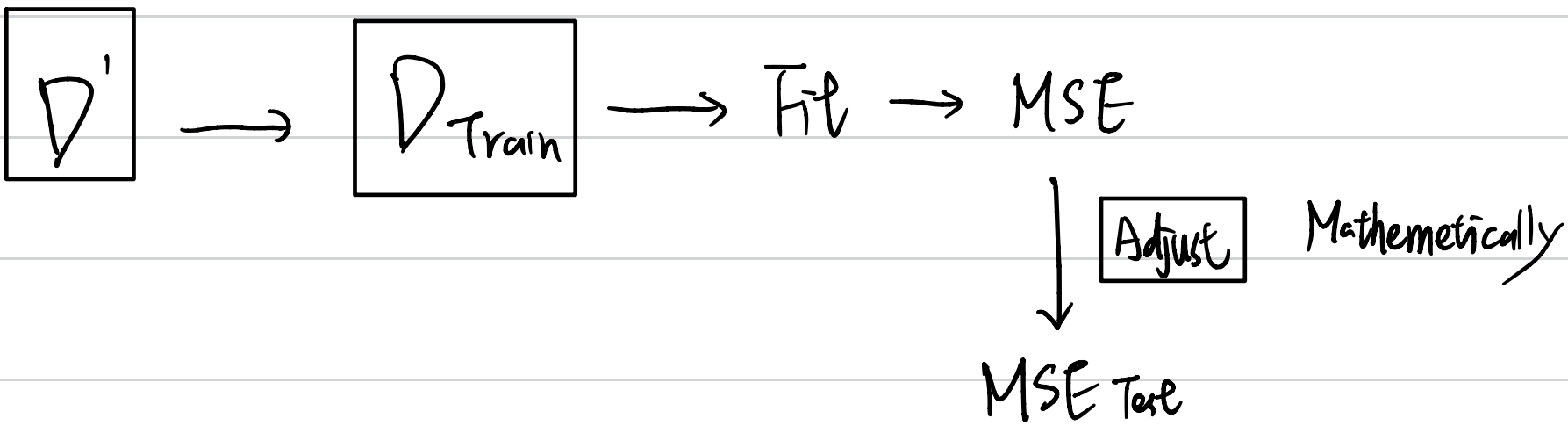
Adjusting the Training Statistics for Test Error Approximation

Adjusting Techniques

- We introduce few other ways of adjusting the training error **to make it a better representative of the test error**
- These adjustments are **not as reliable as Cross validation**, but they are easier to **calculate**
- These quantities were **more widely used before** the widespread use of computers for regression and machine learning
- Now that computers can help performing multiple fits computationally fast enough, often K-Fold CV is considered as the desirable test error approximation
- The test error estimates that we will present in the next couple of slides are **only valid for least-squares models (i.e., linear regression)**, and do not extend to all parametric models



Restricted Classification Model



List of Other Techniques

Only apply to Linear Regression Model

Methods to adjust the training error for the number of variables to estimate the test MSE:

- C_p statistic
- Akaike information criterion (AIC)
- Bayesian information criterion (BIC)
- Adjusted R^2

C_p Statistic

- For a fitted least squares model with d predictors

$$C_p = \frac{1}{n}(RSS + 2d\hat{\sigma}^2)$$

- $\hat{\sigma}^2$ is an estimate of the noise variance
- $\hat{\sigma}^2$ is normally estimated using all the predictors (full model)
- It is an unbiased estimate of the **test MSE**
- The smaller C_p, the better the model (we can pick models with the smallest C_p statistic)
- Becomes a better estimate of the test error as the sample size, n, increases

AIC: Akaike Information Criterion

- Defined for a large class of models based on the maximum likelihood criterion
- When we consider the noise ϵ be of i.i.d Gaussian, the MLE and MSE return identical results and in this case we have

$$AIC = \frac{1}{n\hat{\sigma}^2}(RSS + 2d\hat{\sigma}^2)$$

  which is a multiple of C_p (no preference over using one vs the other)
- $\hat{\sigma}^2$ is an estimate of the noise variance
- $\hat{\sigma}^2$ is normally estimated using all the predictors (full model)
- The smaller AIC, the better the model (we can pick models with the smallest AIC statistic)

BIC: Bayesian Information Criterion

- Takes a Bayesian approach to estimate the test error
- Asymptotically ($n \to \infty$) choosing the model with the highest posterior probability of being the best model
- In the case of least squares the formulation is

$$BIC = \frac{1}{n\hat{\sigma}^2}(RSS + \log(n)d\hat{\sigma}^2)$$

  which takes an *almost* similar form as the previous two statistics
- $\hat{\sigma}^2$ is an estimate of the noise variance
- $\hat{\sigma}^2$ is normally estimated using all the predictors (full model)
- The smaller BIC, the better the model (we can pick models with the smallest AIC statistic)
- When $n < 7$, BIC imposes a smaller penalty on the number of variables, but for $n > 7$ that $\log n > 2$ the penalty is larger
- In other words in standard observation regimes where n is sufficiently large, BIC tends to pick smaller models than AIC or C_p

Adjusted R^2

- Presents a way of making the R^2 statistic dependent on the number of predictors
- Recall the R^2 statistic:

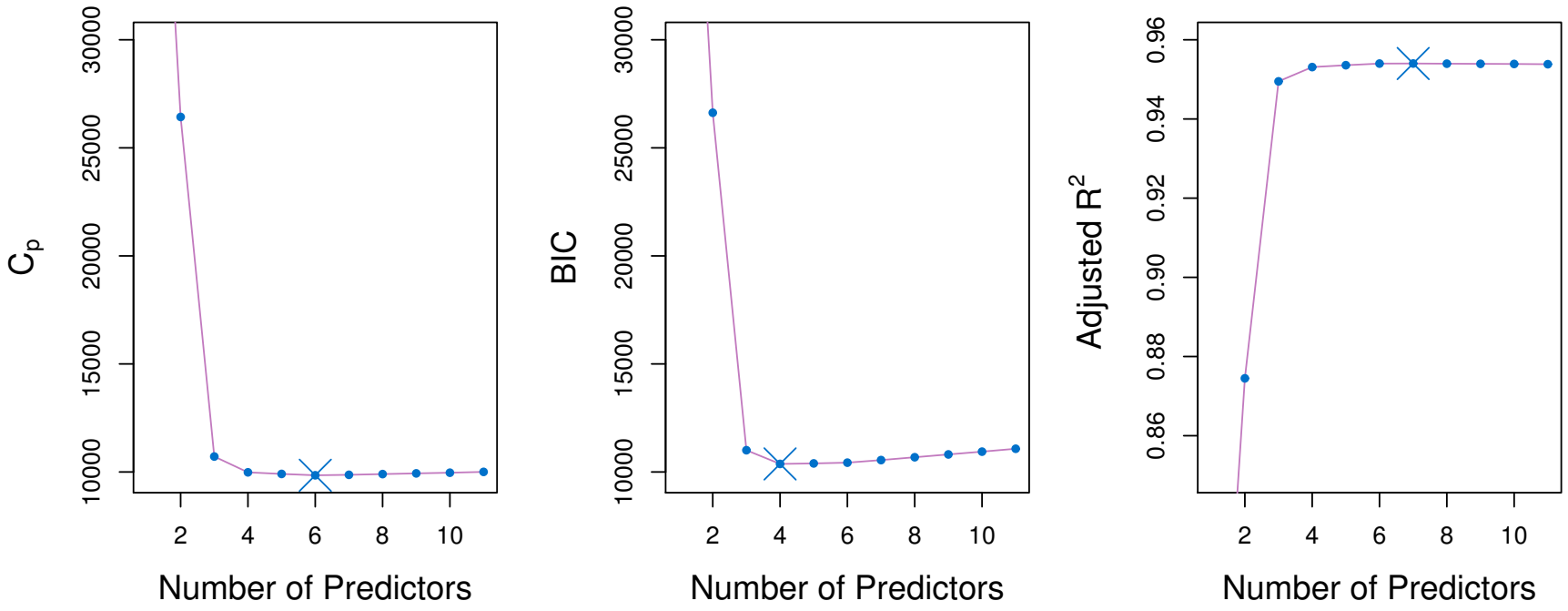
$$R^2 = 1 - \frac{RSS}{TSS}, \qquad \text{where} \quad TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

- The formulation for adjusted R^2 is

$$R^2_{adj} = 1 - \frac{RSS/(n-d-1)}{TSS/(n-1)}$$

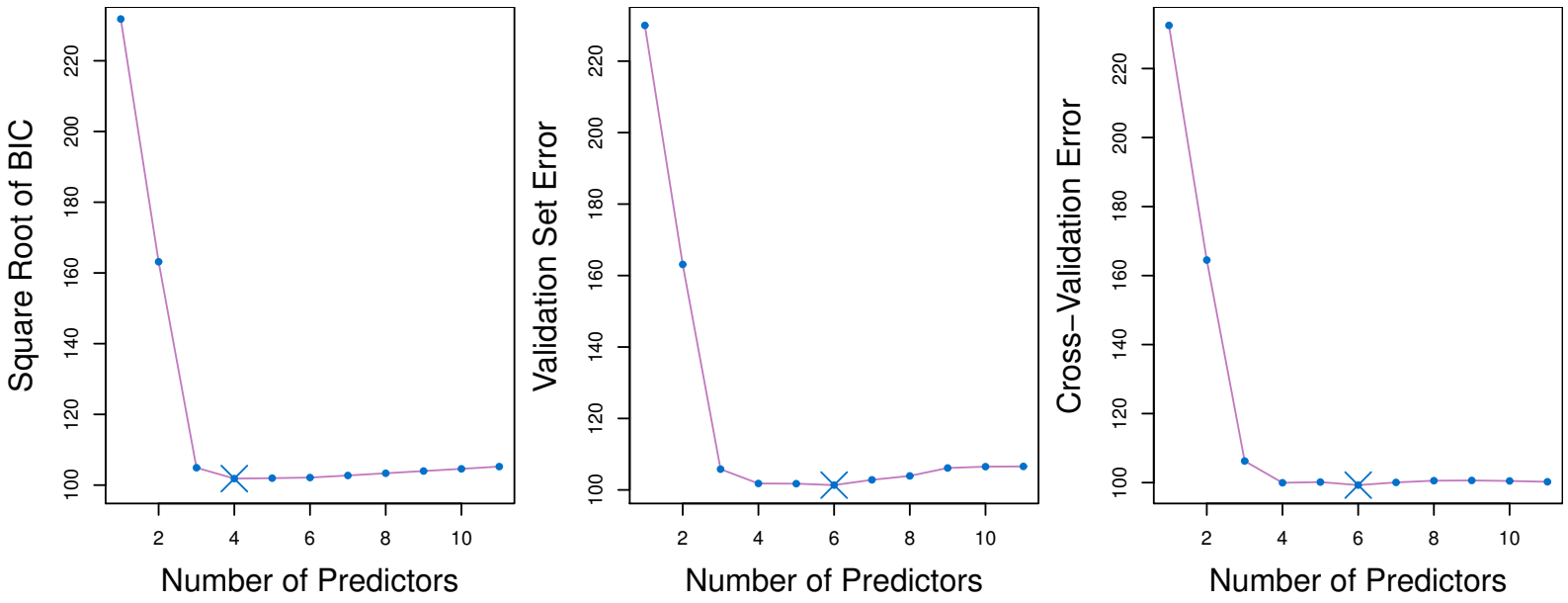
- Unlike the other three statistics that being small indicates a better model, for adjusted R^2 we are interested in models that tend to generate values closer to 1
- The use of C_p, AIC and BIC is more motivated in statistical learning theory than the adjusted R^2

Example Comparing the Performances



C_p, BIC, and adjusted R^2 for the best models of each size for the Credit data set

Comparison Against CV Techniques



- The results are not much different
- Note that nowadays CV methods are computationally fast to implement and regardless of the model can always be used as a reliable selection tool

How to Use These Statistics in Model Selection

– **Best subset selection** formal procedure (NP-hard and computationally not possible for large p)

Algorithm 6.1 *Best subset selection*

---

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.

2. For $k = 1, 2, \ldots p$:

   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.

   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* is defined as having the smallest RSS, or equivalently largest R^2.

3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

$$\{x_1, x_2 \ldots, x_p\}$$

$$\#\text{ of subset} = 2^p$$

$$\sum_{k=0}^{p} \binom{p}{k} = 2^p$$

$M_0 \leftarrow$ Size 0 : no feature : β_0

$M_1 \leftarrow$ size 1 : $\{x_1\}, \{x_2\} \ldots \{x_p\} \rightarrow p$

Best Models

$M_2 \leftarrow$ size 2 : $\{x_1, x_2\}, \{x_1, x_3\}, \ldots \rightarrow \binom{p}{2}$

$\vdots$

M_p

How to Use These Statistics in Model Selection

- **Forward stepwise selection** (computationally tractable)
- At each step the variable that gives the greatest additional improvement to the fit is added to the model

Champions stay

Algorithm 6.2 *Forward stepwise selection*

1. Let $\mathcal{M}_0$ denote the *null* model, which contains no predictors.
2. For $k = 0, \ldots, p-1$:
   (a) Consider all $p-k$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
   (b) Choose the *best* among these $p-k$ models, and call it $\mathcal{M}_{k+1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

- Forward selection can even be used when $n < p$

$\{X_1\}, \{X_2\} \ldots \{X_p\}$

$\{X_{10}\}$

keep

$\{X_{10}, X_1\}, \{X_{10}, X_2\} \cdots \{X_{10}, X_p\}$

$\{X_{10}, X_8\}$

keep

$\{X_{10}, X_8, X_1\}$ $\{X_{10}, X_8, X_2\} \cdots \{X_{10}, X_8, X_p\}$

$\{X_{10}, X_8, X_{20}\}$

$\vdots$

How to Use These Statistics in Model Selection

3

- **Backward stepwise selection** (computationally tractable)
- Begins with the full least squares model containing all p predictors, and then iteratively removes the least useful predictor, one-at-a-time

---

Algorithm 6.3 *Backward stepwise selection*

---

1. Let $\mathcal{M}_p$ denote the *full* model, which contains all p predictors.
2. For $k = p, p-1, \ldots, 1$:
   (a) Consider all k models that contain all but one of the predictors in $\mathcal{M}_k$, for a total of $k-1$ predictors.
   (b) Choose the *best* among these k models, and call it $\mathcal{M}_{k-1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

- Backward selection requires $p < n$ (to allow the full model to be fit)