IFI 9000 Analytics Methods
Linear Model Selection and Regularization

by **Houping Xiao**

January 26$^{\text{th}}$, 2021

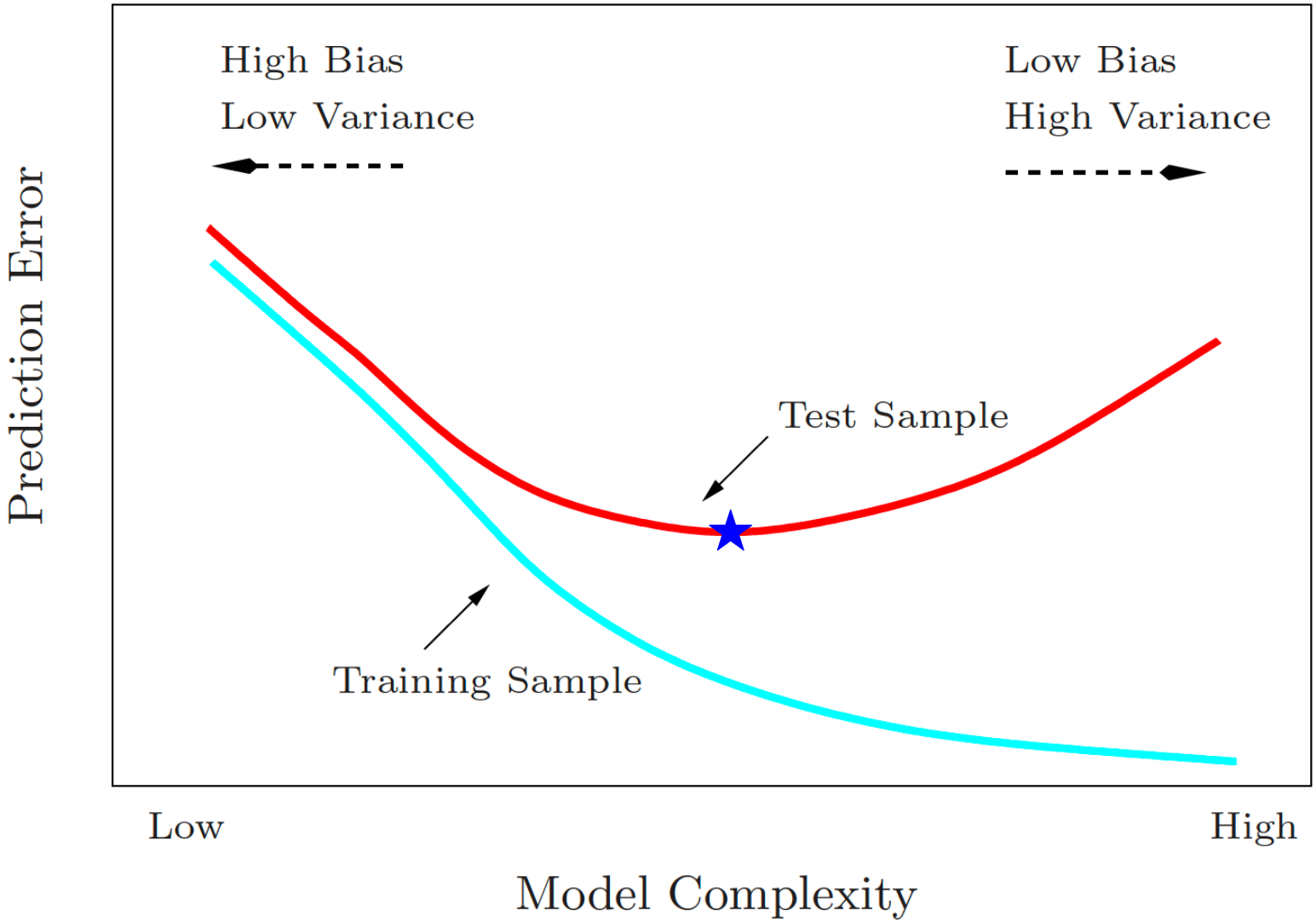
Introduction

Introduction

- Recall that we fitted out models using training data and were interested in evaluating the performance w.r.t. 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 data 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

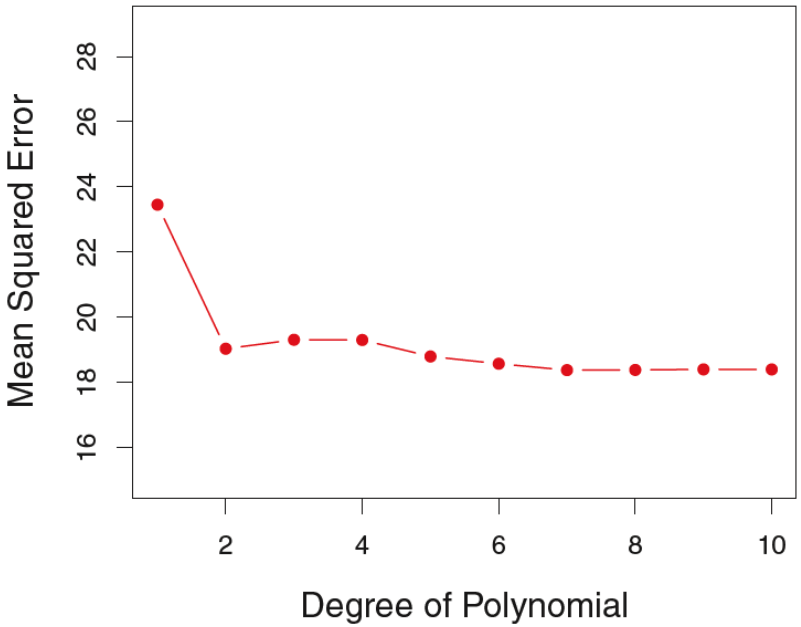
- A standard approach used so far
- Randomly divide the available set of samples into two parts: **a training set** and **a validation/hold-out set** (i.e., 80%-20% splitting)
- The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set
- The error with reference to the hold-out set is an approximation of the test error

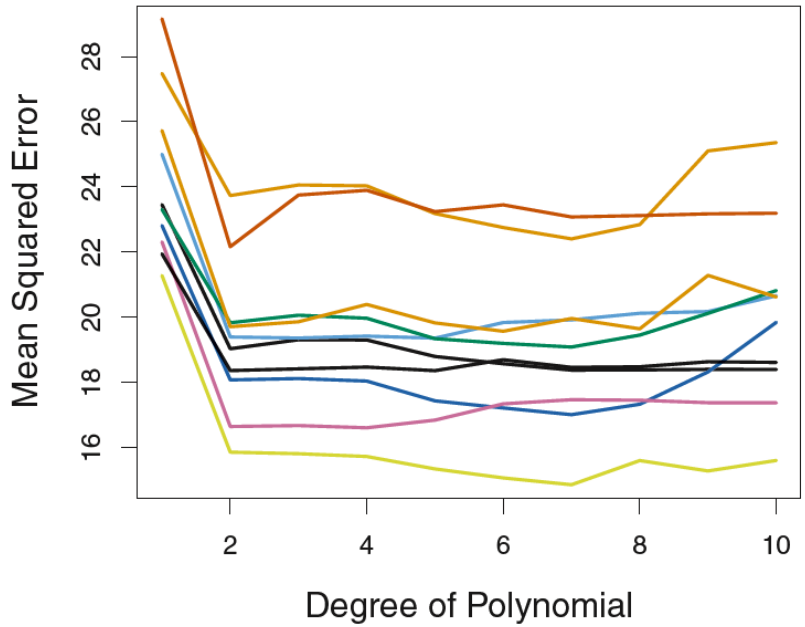
An Example

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

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

- 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





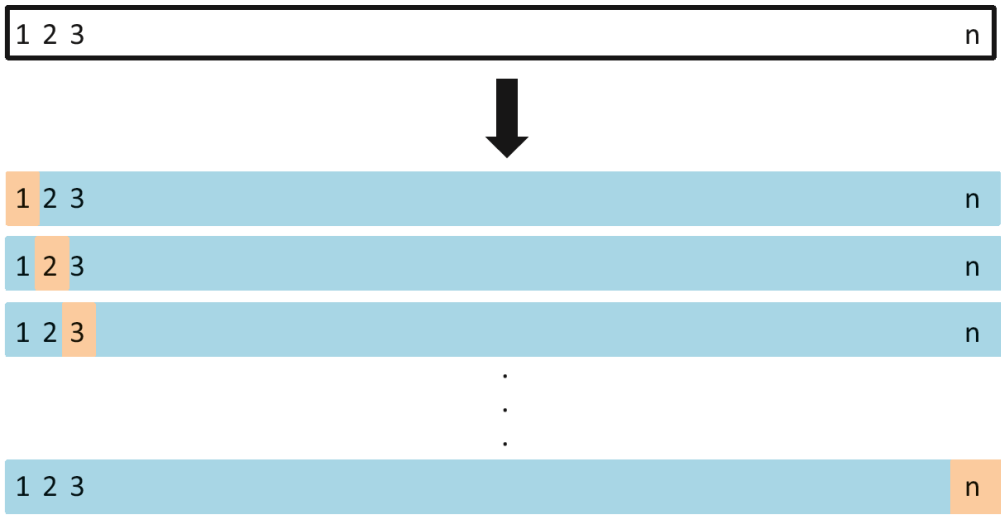
Validation Set Cons & Pros

- The procedure is **simple** (e.g., previous example) 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
- 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
- **While the estimated test error vary a lot, finding information such as model selection is still possible**
- 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)

- 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

$$CV_n = \frac{1}{n} \sum^{n} MSE_i$$



LOOCV Cons & Pros

- It has very small bias compared to the validation set approach (why?)
- The test error overestimation is less than the validation set approach (why?)
- Its results are reproducible unlike the validation set approach which uses a random subset of the data for test evaluation
- It can be computationally very expensive (why?)
- For linear models there is 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

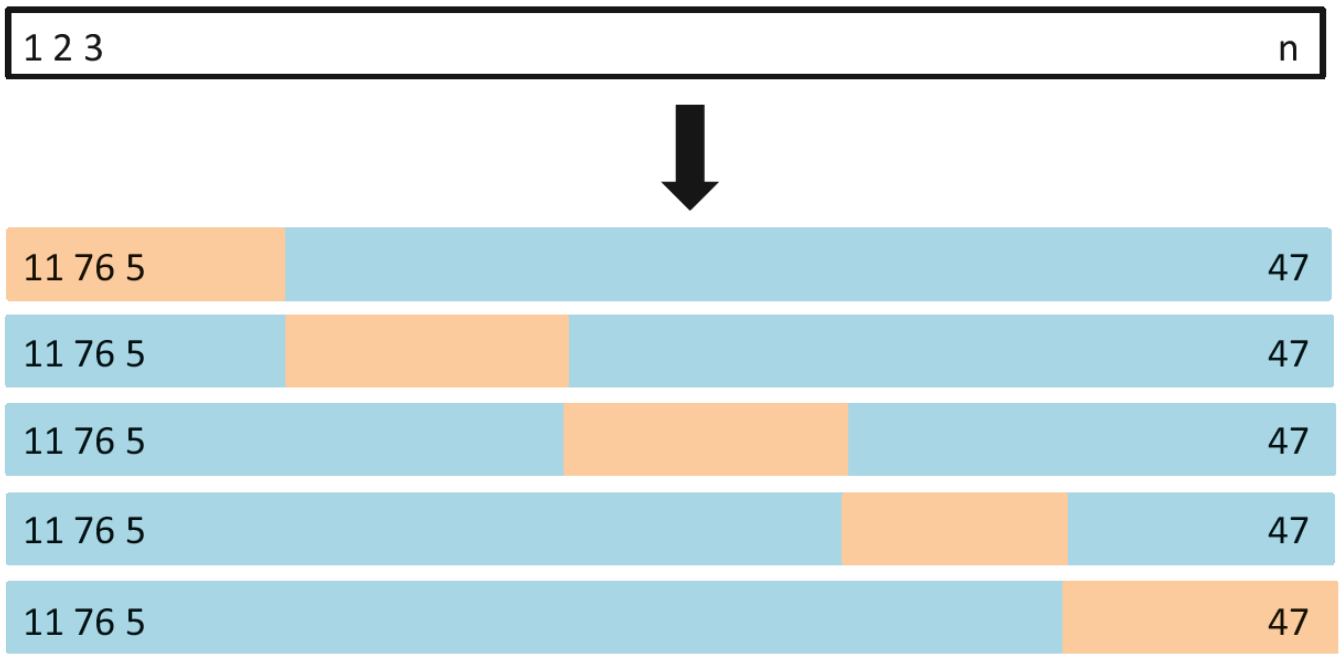
K-Fold Cross Validation

- 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

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

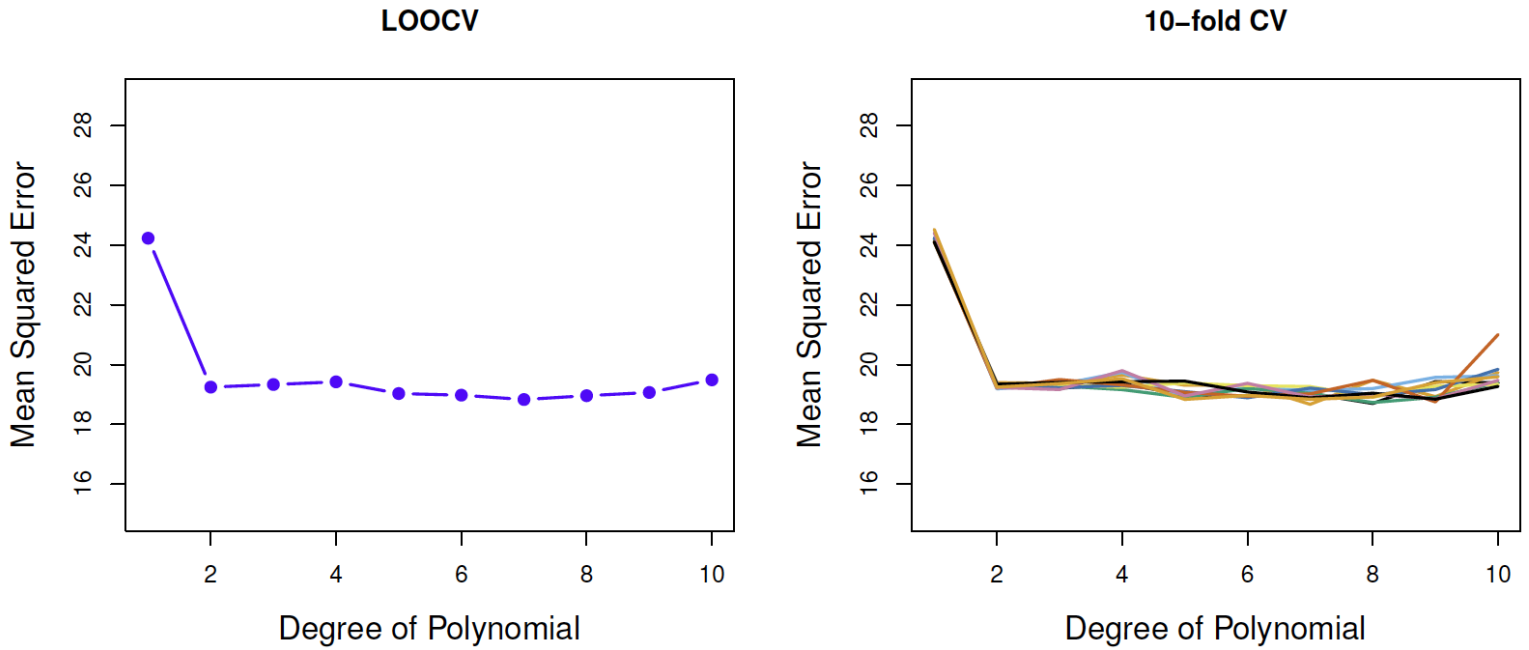
- Often $K = 5$ or 10 is considered in applications

K-Fold Cross Validation



K-Fold CV vs 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 CV vs LOOCV

- Aside from 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 is lower, recall

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

Cross Validation in Classification

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

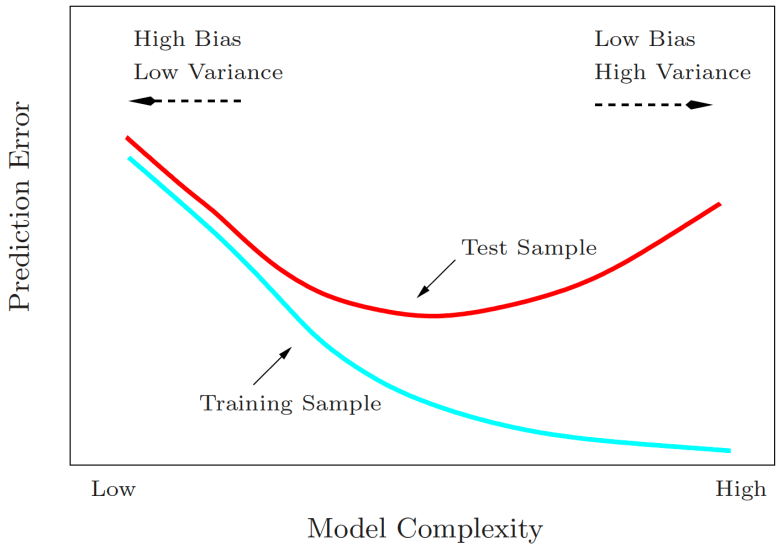
$$CV_k = \frac{1}{K} \sum_{k=1}^{K} Err_k$$

where

$$Err_k = \frac{1}{|C_k|} \sum_{i \in C_k} 1(y_1 \neq \hat{y}_i)$$

Review of Cross Validation

- 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 validation sets, we were able to use LOOCV or K-fold CV as estimates of the test error
- We discussed why K-fold cv is 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 leanring
- Now that computers can help performing multiple fits computationally fast enough, often K-fold CV is considered as the desirable test error approximation

List of Other Techniques

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

- C_p statistic
- AIC: Akaike information criterion
- BIC: Bayesian information criterion
- 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 errors 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 BIC 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 when 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:

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

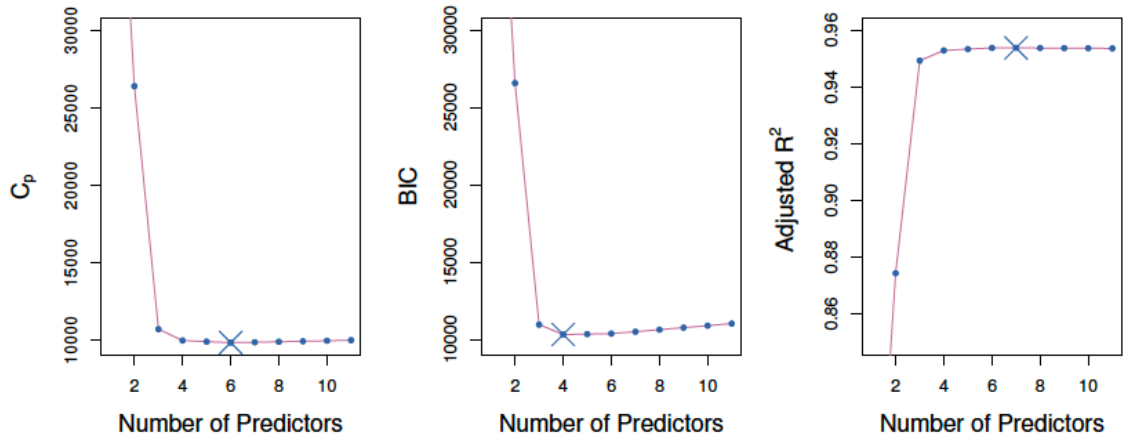
- The formulation for adjusted R^2 is

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

- Unlike the other three statistics that being small indicating a better model, for adjusted R^2 were are interested in mdoels 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

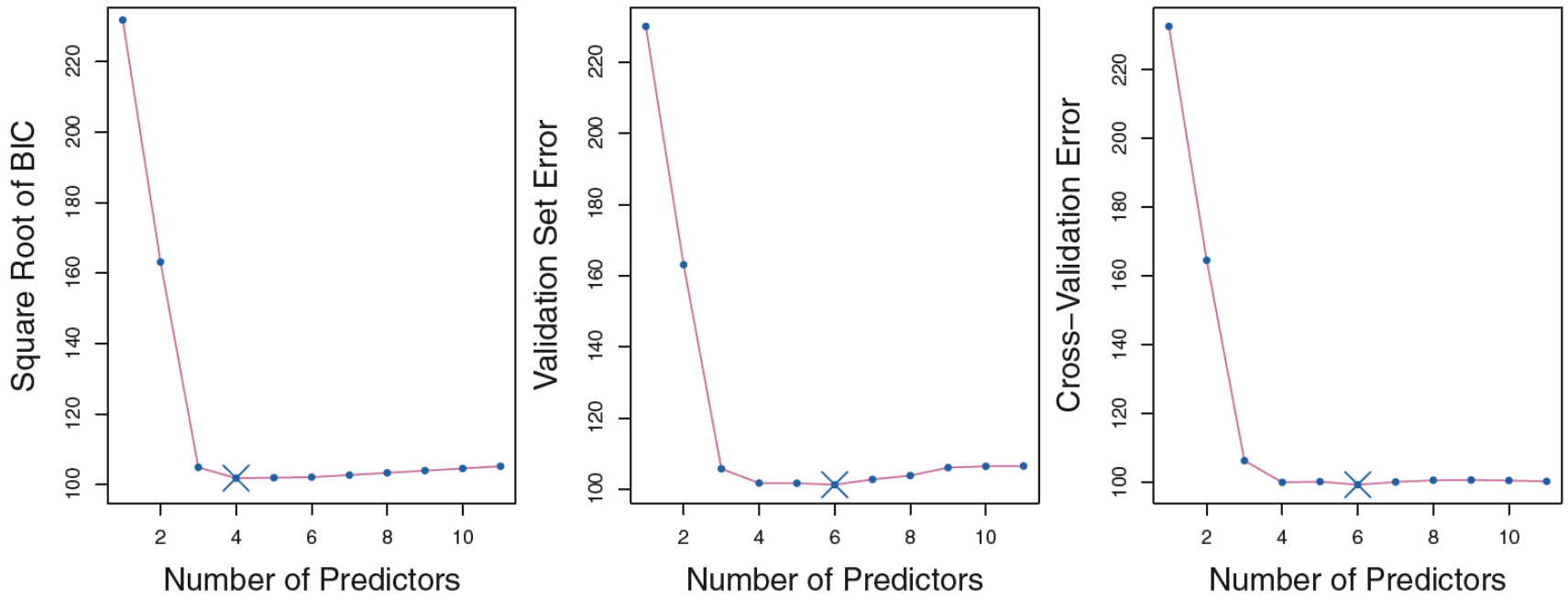
Comparing the Performances: An Example

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



income	income	income
limit	limit	limit
rating	cards	rating
cards	student	cards
age		age
student		student
		Gender

Comparison Against CV Techniques



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

Model Selection

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.

How to Use These Statistics in Model Selection

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

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 $n < p$

How to Use These Statistics in Model Selection

- **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-tiem

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)

What are Shrinkage Methods and Why Useful?

You would probably hear **Ridge Regression** and **LASSO** quite often

- The subset selection methods use least squares to fit a linear model that contains a subset of the predictors
- As an alternative, we can fit a model containing all p predictors using a technique that constrains or regularizes the coefficients estimates, or equivalently, that shrinks the coefficient estimates towards zero
- It may not be immediately obvious why such a constraint should improve the fit, but it turns out that shrinking the coefficient estimates can significantly **reduce the model variance**

Redige Regression

- Recall that the least squares fitting procedure estiamtes $\beta_0, \beta_1, \cdots, \beta_p$ using the values that minimize

$$RSS = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j \cdot x_{ij} \right)$$

- In contrast, the ridge regression coefficient estimates $\hat{\beta}^{ridge}$ are the values that minimize

$$RSS^{ridge} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j \cdot x_{ij} \right) + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

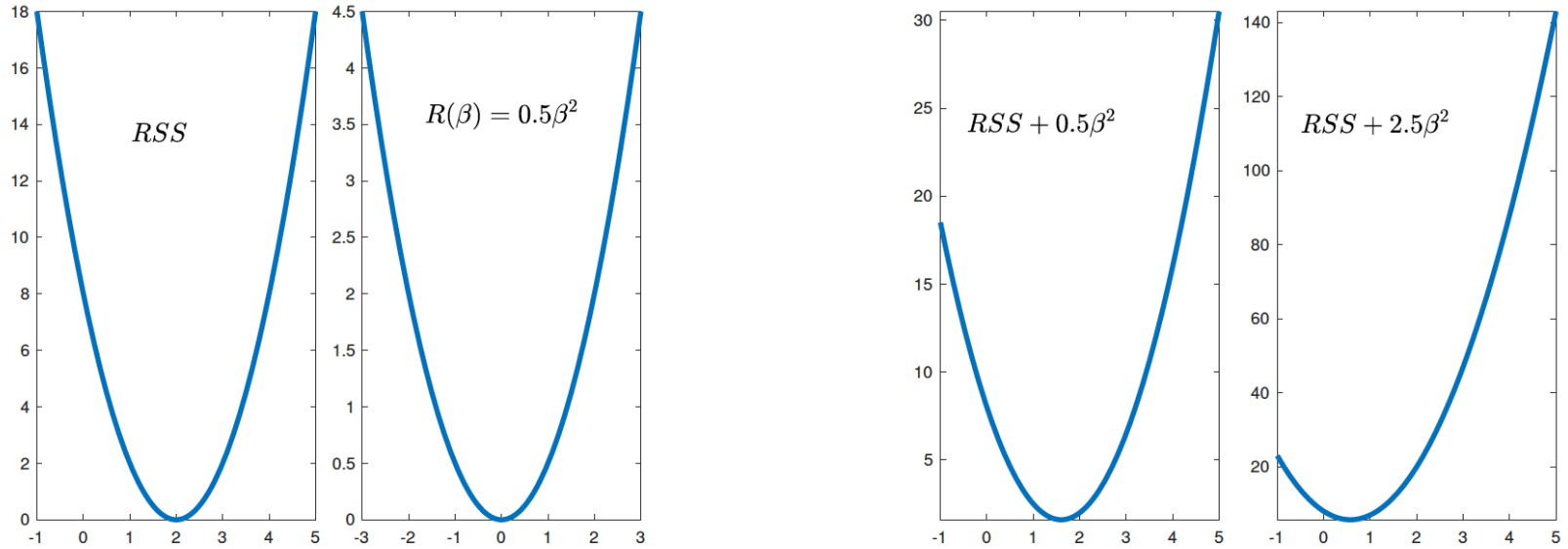
where λ is a hyper/tuning parameter

Ridge Regression

- As with least squares, ridge regression seeks coefficient estimates that fit the data well, by making the RSS small
- However, the second term, $\lambda \sum_{j=1}^{p} \beta_j^2$, called a shrinkage penalty, encourages solutions that are close to zero, and so it has the effect of shrinking the estimates of β_j towards zero
- The tunning parameter λ serves to control the relative impact of these two terms on the regression coefficient estimates (trade off between bias and variance)
- Selecting a good value for λ is critical; often cross-validation is used for this

Effect of Increasing λ on the β

- The figure below shows how increasing the Ridge penalty pushes the minimizers of the mixed RSS objective to zero



Shrinkage Example

- Recall that the least squares solution to fit data point $(x_1, y_1), \cdots, (x_n, y_n)$ was obtained via the minimization

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \beta \cdot x_i)^2; \quad \hat{\beta} = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}$$

- We can show that if we run the ridge regression

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \beta \cdot x_i)^2 + \lambda \beta^2$$

  the new estimaate becomes

$$\hat{\beta}^{ridge} = \frac{\sum_{i=1}^{n} x_i y_i}{\lambda + \sum_{i=1} x_i^2}$$

- Note how increasing λ pushes $\hat{\beta}^{ridge}$ towards zero

In Class Exercise

- For the simple regression problem of fitting $(x_1, y_1), \cdots, (x_n, y_n)$ to the model $y = \beta_0 + \beta_1 x$ show that the least squares estimates for the ridge regularized objective

$$\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2 + \lambda(\beta_0^2 + \beta_1^2)$$

are

$$\hat{\beta}_1^{ridge} = \frac{\sum_{i=1}^n x_i y_i - \frac{n^2}{n+\lambda}\bar{x}\bar{y}}{\lambda + \sum_{i=1}^n x_i^2 + \frac{n^2}{n+\lambda}\bar{x}^2},$$

$$\hat{\beta}_0^{ridge} = \frac{1}{n+\lambda}\left(\sum_{i=1}^{n} y_i - \hat{\beta}_1^{ridge}\sum_{i=1}^{n} x_i\right)$$

What Happens in Multiple Regression?

- In this case we previously had

$$RSS = (\boldsymbol{y} - \boldsymbol{X}\beta)^\top (\boldsymbol{y} - \boldsymbol{X}\beta)$$

which led to

$$\hat{\beta} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

- In the case of regularized problem

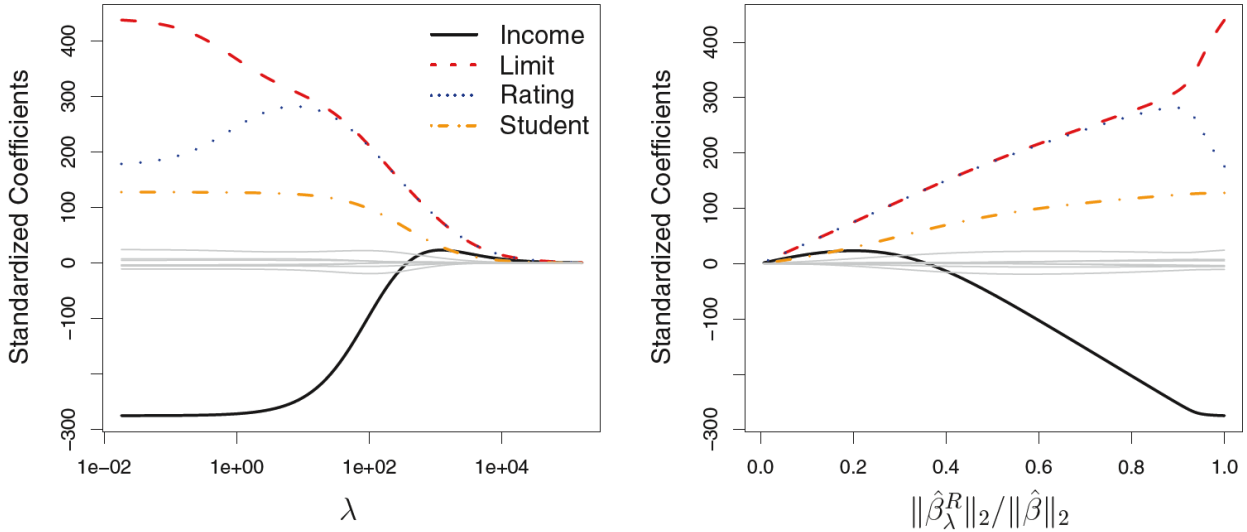
$$(\boldsymbol{y} - \boldsymbol{X}\beta)^\top (\boldsymbol{y} - \boldsymbol{X}\beta) + \lambda ||\beta||^2$$

we will have

$$\hat{\beta}^{ridge} = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

where $||\cdot||^2$ is L_2 norm and $\boldsymbol{I}$ is the identity matrix

Credit Data Example



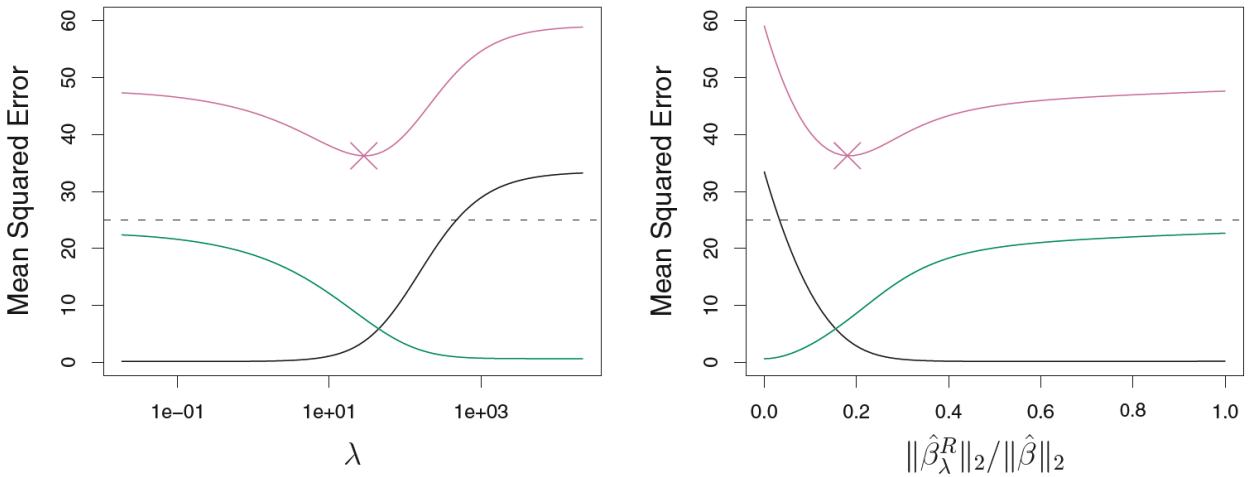
- Left: Each curve corresponds to the ridge regression coefficient estimate for one of the ten variables, plotted as a function of λ
- The right-hand panel displays the same ridge coefficient estimates as the left-hand panel, but instead of displaying λ on the x-axis, we display $||\hat{\beta}^{ridge}||/||\hat{\beta}||$
  - How much **shrinkage** happens by increasing λ

Scaling of the Predictors

- In the standard least-squares if we scale a feature value by c, the corresponding coefficient scales by c^{-1}
- However when we have the ridge regularized objective, this is no more the case
- To see a consistent behavior, for the ridge regularized problem we often work with standardized features:

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2}}$$

Bias-Variance Trade-Off



- squared bias (black), variance (green), and test mean square error (purple) for the ridge regression predictions on a simulated data set, as a function of λ and $||\hat{\beta}^{ridge}||/||\hat{\beta}||$. The horizontal dashed lines indicate the minimum possible *MSE* (**the standard least squares,** $\lambda = 0$ **in nowhere close**). The purple crosses indicate smallest ridge regression model MSE values
- Recall that test error = bias + variance + noise variance

Disadvantage of Ridge Regression

- Ridge regression will include all p predictors in the final model. The penalty $\lambda \sum_{j=1}^{p} \beta_j^2$ will shrink all of the coefficients towards zero, but it will not set any of them exactly to zero (unless $\lambda = \infty$)
- This may not be a problem for prediction accuracy, but it can create a challenge in model interpretation in settings in which the number of variables p is very large
- The lasso is an alternative to ridge regression that overcomes this disadvantage

LASSO

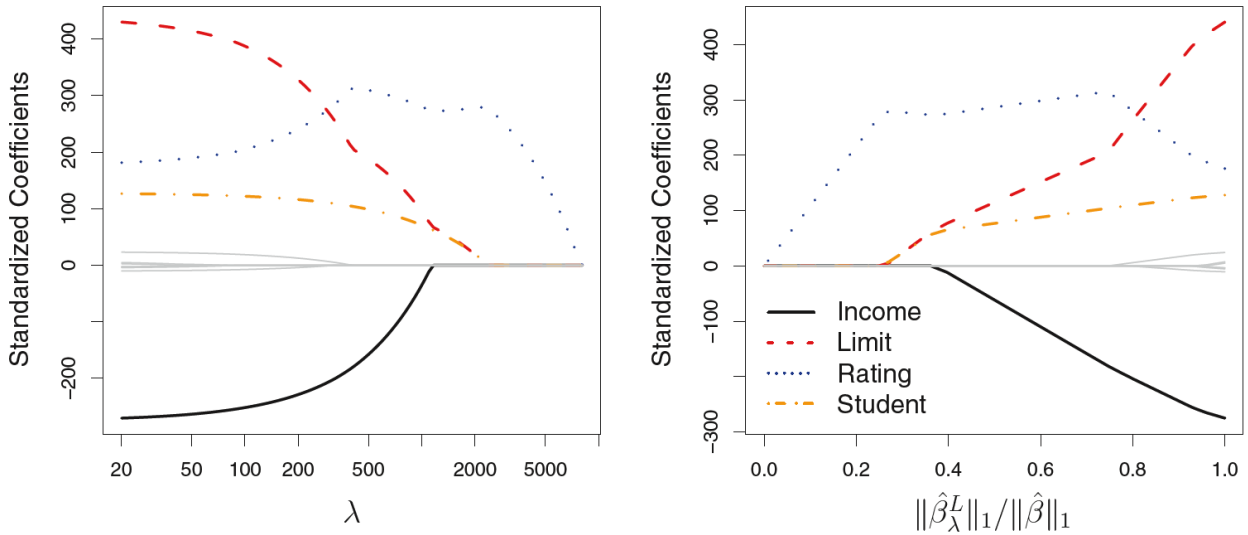
- the lasso coefficient estimates $\hat{\beta}_\lambda^{lasso}$ are the values that minimize

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j|=RSS+\lambda\sum_{j=1}^{p}|\beta_j|$$

  where $\lambda \geq 0$ is a tuning parameter

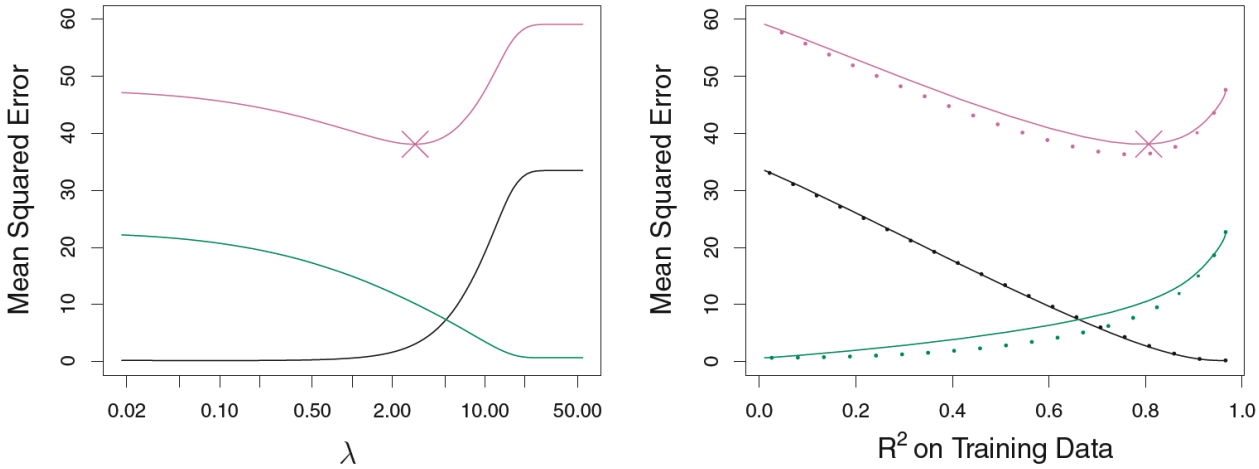
- Like best subset selection, the lasso also performs variable selection

Credit Data Example



- Left: Each curve corresponds to the lasso coefficient estimate for one of the ten variables, plotted as a function of λ
- The right-hand panel displays the same ridge coefficient estimates as the left-hand panel, but instead of displaying λ on the x-axis, we display $||\hat{\beta}^{lasso}||/||\hat{\beta}||$
  - How much **shrinkage** happens by increasing λ

Bias-Variance Trade-Off



- Left: Plots of squared bias (black), variance (green), and test MSE (purple) for the lasso on a simulated data set
- Right: Comparison of squared bias, variance and test MSE between lasso (solid) and ridge (dashed)

Take Away Message for LASSO vs Ridge

- Neither ridge regression nor the lasso will universally dominate the other
- In general, one might expect the lasso to perform better in a setting where a relative **small number of predictors** have substantial coefficients, and the remaining predictors have coefficients that are very small or that equal zero
- Ridge regression will performance better when the response is a function of **many predictors**, all with coefficients of roughly equal size
- However, the number of predictors that is related to the response is never known a priori for real data sets

Take Away Message for LASSO vs Ridge

- A technique such as **cross validation** can be used in order to determine which approach is better on a particular data set
- As with ridge regression, when the least squares estimates have excessively high variance, the lasso solution can yield a reduction in variance at the expense of a small increase in bias, and consequently can generate **more accurate predictions**
- Unlike ridge regression, the lasso performs variable selection, and hence results in models that are **easier to interpret**
- There are very **efficient algorithms** for fitting both ridge and lasso models

The End