IST 5535: Machine Learning Algorithms and Applications

Langtao Chen, Spring 2021

5. Resampling Methods

OUTLINE

- (I) Resampling methods
  1. Cross-validation
     - I. K-fold cross validation
     - II. Leave-one-out cross validation
  2. Bootstrap

- (II) Implementing resampling methods in R
  1. Using caret package for quick experimentation
  2. Directly implement the logic for more detailed control

AGENDA

- Introduction to Resampling Methods
- Using Caret Package
- Repeated K-Fold Cross-Validation
- Bootstrap

Resampling Methods

- Resampling methods involve drawing samples from a training set and refitting a model on each sample.

- The objective of resampling is to obtain additional information about the fitted model.

- In this section, we’ll discuss two most commonly used resampling methods:
  - Cross-validation
  - Bootstrap

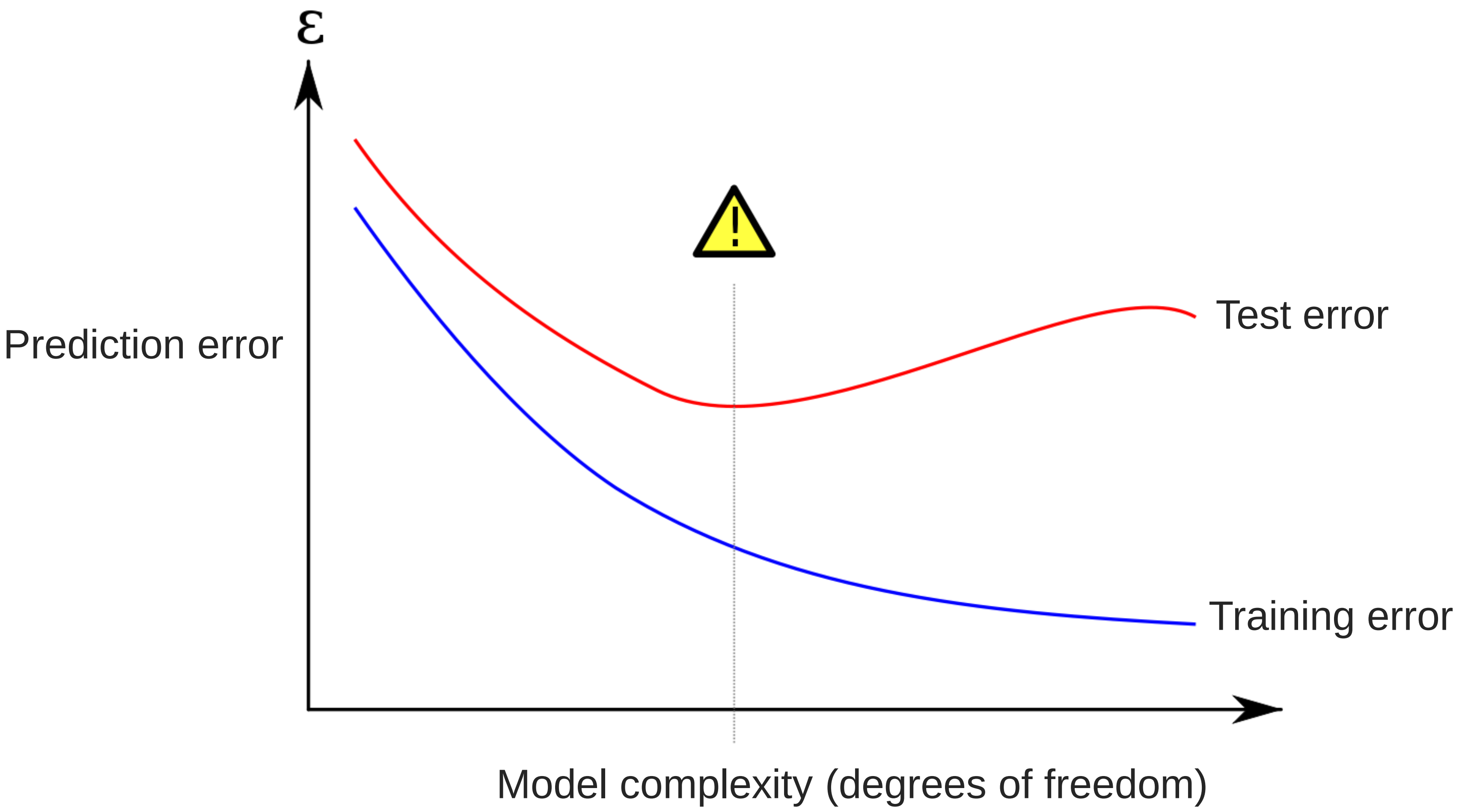
Training Error vs. Test Error

- Test error is the average error that results from using a statistical learning method to predict the response on a new observation—a measurement that was not used in training the method.

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

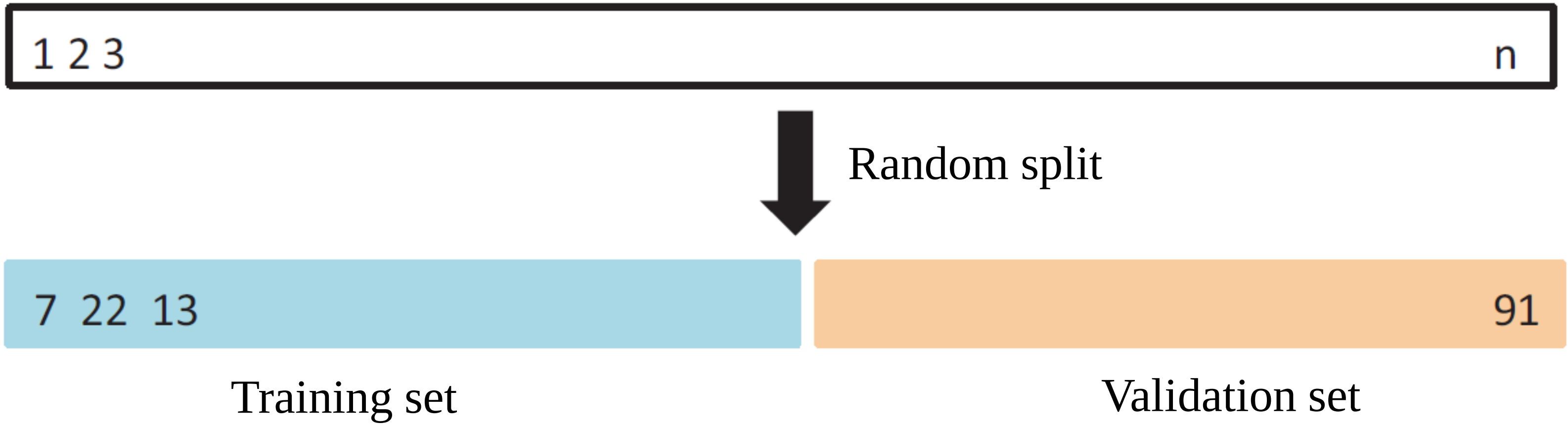
- The training error rate often is quite different from the test error rate, and in particular the former can dramatically underestimate the latter.

Training Error vs. Test Error



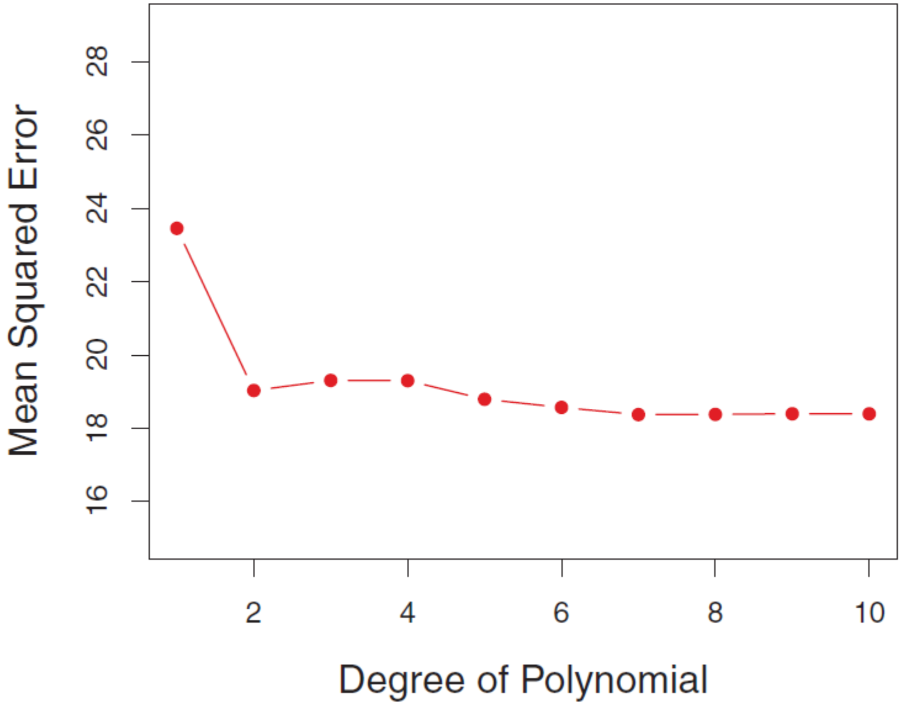
Validation Set Approach

- When we don't have a large designated test set, what can we do?
- Randomly divide the available set of observations into two parts, a *training set* and a *validation set* or *hold-out set*.
- 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 resulting validation set error rate provides an estimate of the test error rate.

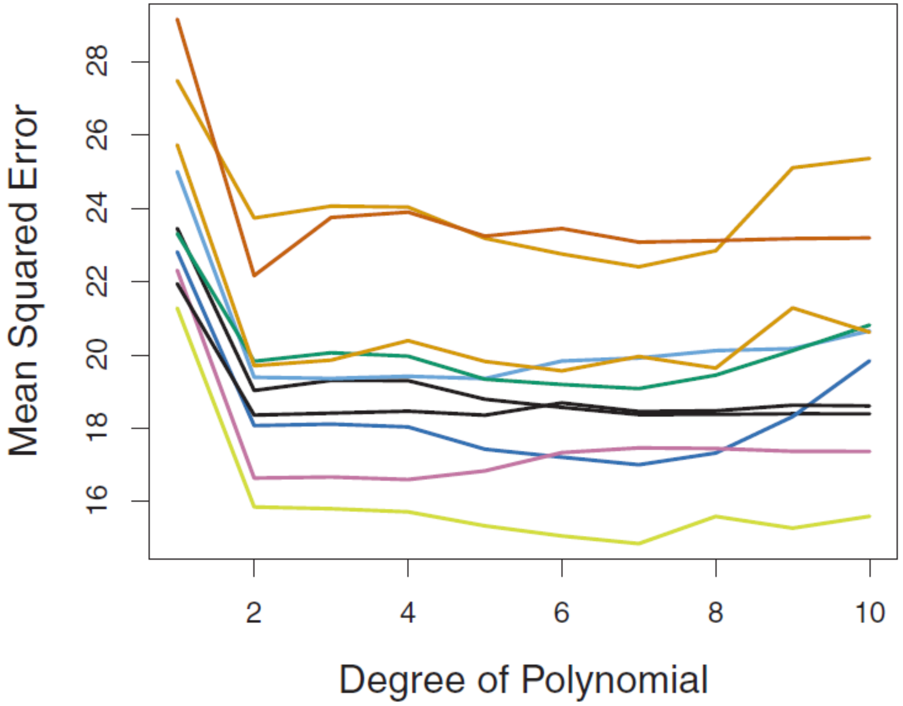


Validation Set Approach on Auto Dataset

- Predict mpg using polynomial functions of horsepower



A single random split



Repeat the process ten times

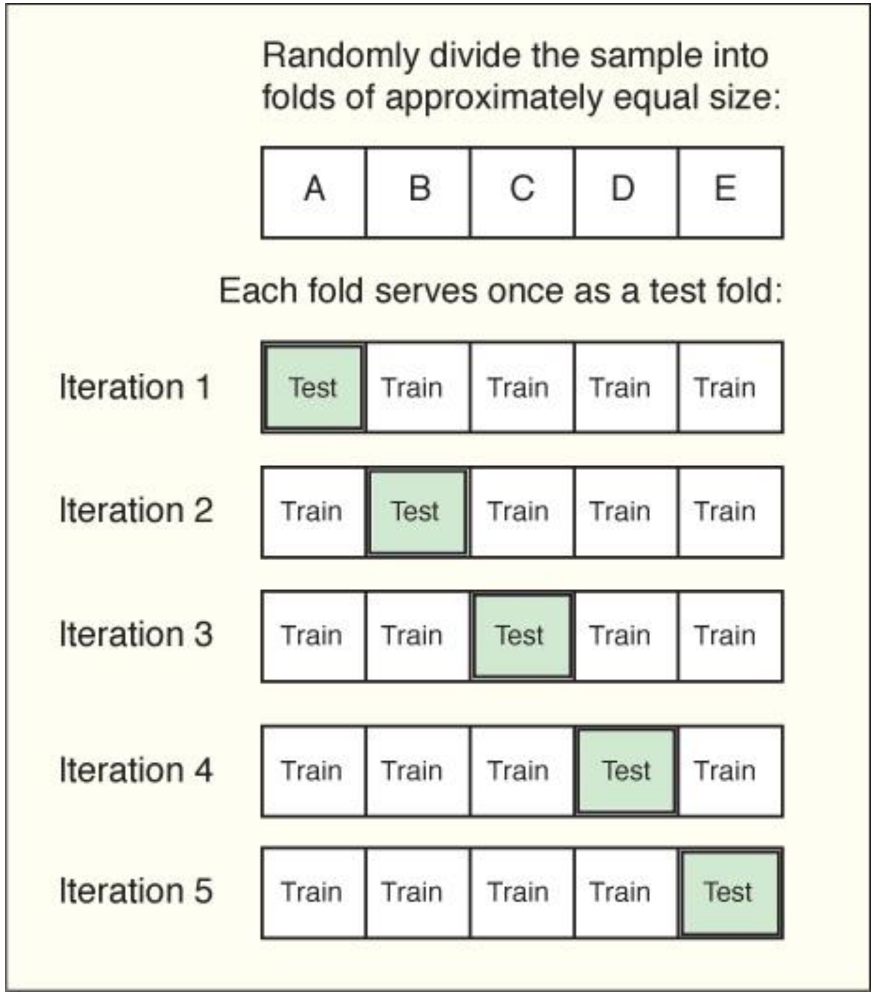
Summary of Validation Set Approach

- Advantages
  - Simple and easy to implement

- Disadvantages
  - The 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.
  - Only a subset of the observations (included in the training set) are used to fit the model.
  - This suggests that the validation set error tends to overestimate the test error for the model fit on the entire data set.

K-Fold Cross-Validation

A 5-fold cross validation



Randomly divide the sample into folds of approximately equal size:

A	B	C	D	E

Each fold serves once as a test fold:

Iteration 1	Test	Train	Train	Train	Train
Iteration 2	Train	Test	Train	Train	Train
Iteration 3	Train	Train	Test	Train	Train
Iteration 4	Train	Train	Train	Test	Train
Iteration 5	Train	Train	Train	Train	Test

In practice, k = 5 or 10.
Magical k?

Cross-Validation Error Rate:

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

Leave-One-Out Cross-Validation (LOOCV)

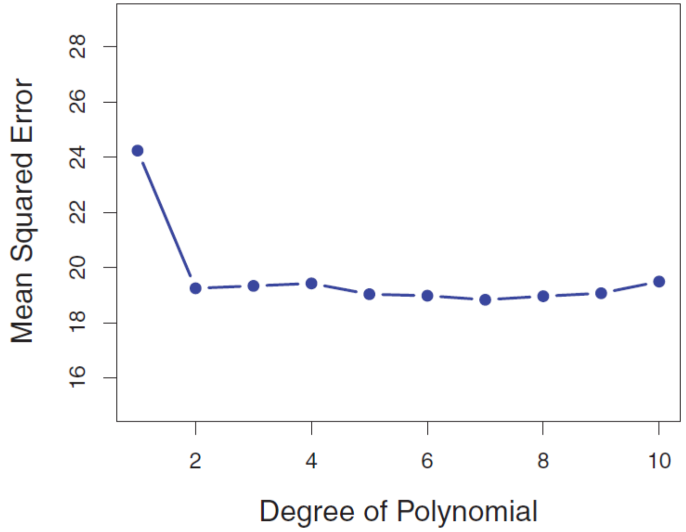
- Set k=n, it is called n-fold or leave-one-out cross-validation (LOOCV)
- Each instance in turn is left out, and the model is trained on all remaining instances.

- Advantages
  - Greatest possible amount of data is used for training.
  - Tends to have lower bias than k-fold cross-validation.
  - The procedure is deterministic: no random sampling is involved, obtain the same result each time.
- Disadvantages
  - Computationally expensive
  - Nonstratified sample (only one instance in the validation/test set) => May lead to poor performance
  - Tends to have higher variance than k-fold cross-validation (bias-variance tradeoff).

LOOCV vs. k-Fold CV and Validation Set

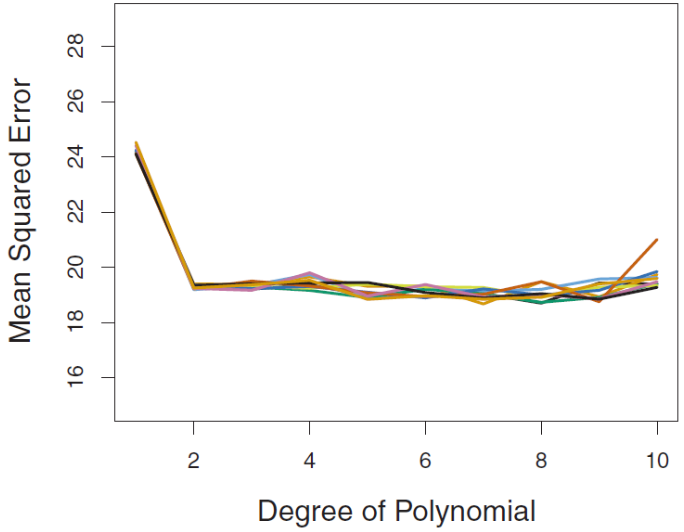
- Predict mpg using polynomial functions of horsepower

LOOCV



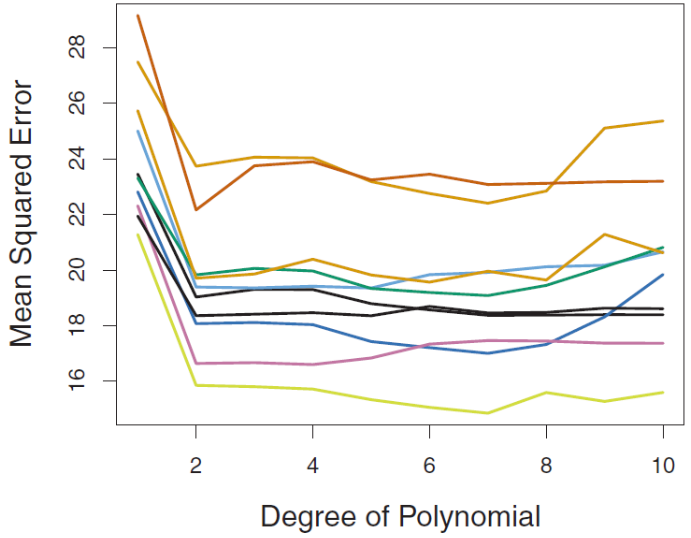
deterministic

K-fold CV (repeated)



low variability

Validation Set (repeated)



high variability

Why k = 5 or 10?

- Computational advantage compared with a large k or $k = n$ (LOOCV).

- Bia-variance trade-off
  - If k is too small, a large portion of the data is not used to train the model. The estimate of prediction error tends to be biased upward.
  - If k is too large, the bias can be reduced. However, the estimated prediction error tends to have a large variance.
  - k = 5 or 10 provides a good trade-off between bias and variance.

Cross-Validation on Classification Problems

- So far, we have discussed cross-validation in regression setting.

- The procedure would be similar in classification setting where the outcome is qualitative, except that we use misclassification rate to quantify test error.
  - K-Fold Cross-Validation

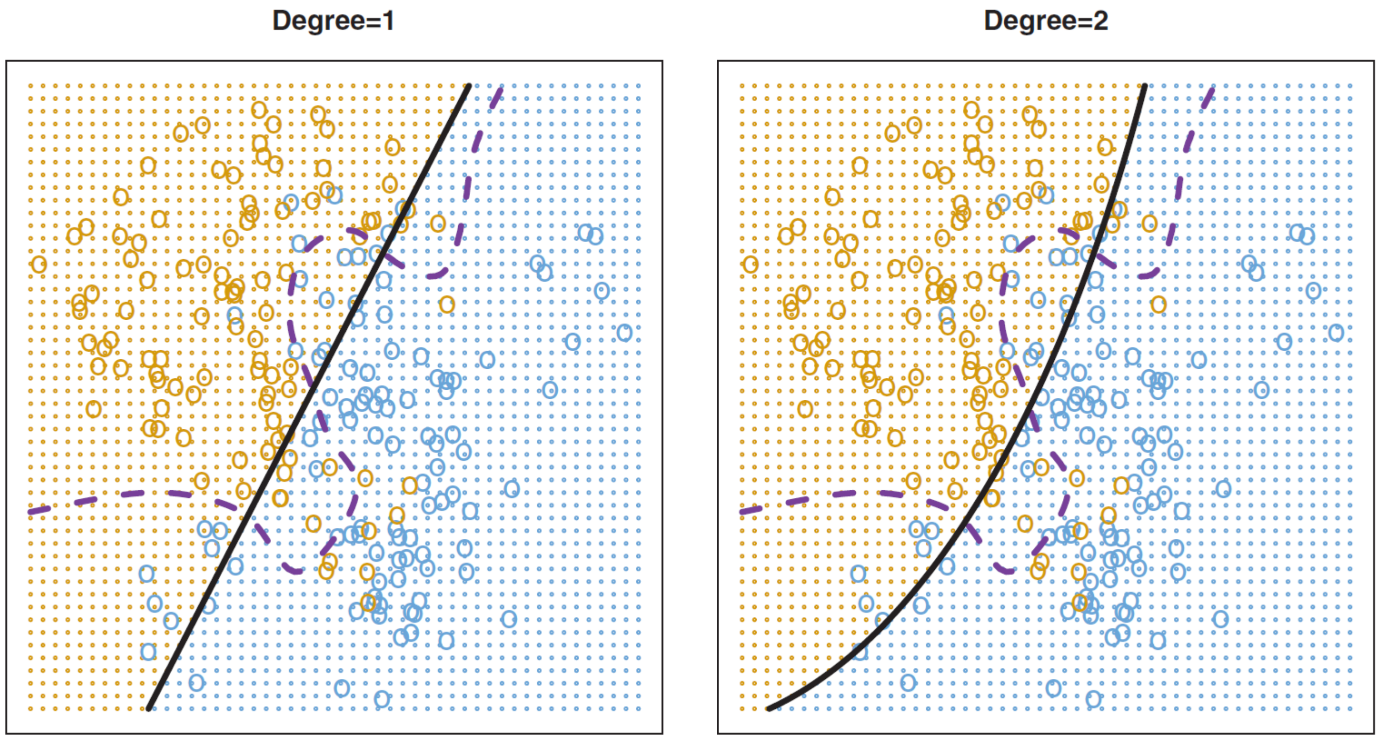
$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} Err_i \qquad \text{where } Err_i = I(y_j \neq \hat{y}_j)$$

  - LOOCV

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

Use CV to Select the Best Model (or Tune Hyperparameters)

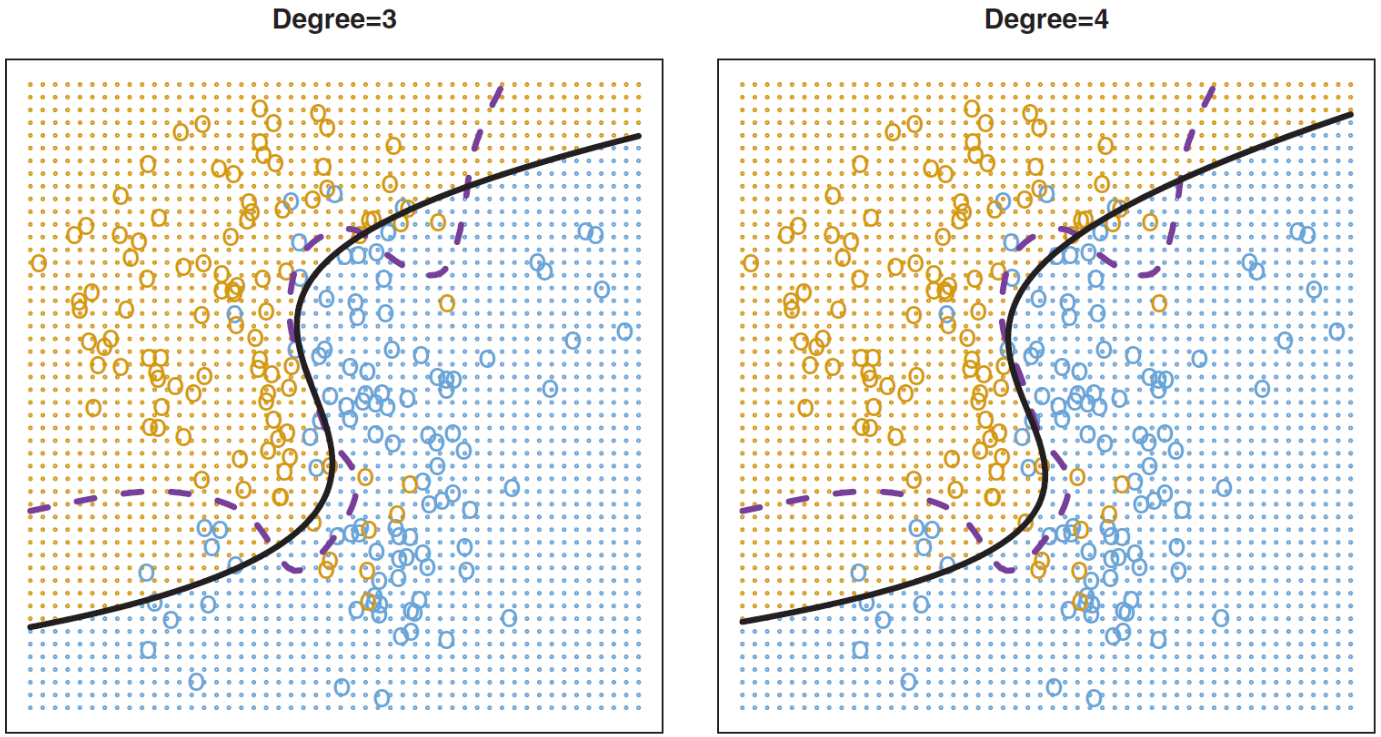
- Select the order of polynomial in logistic regression



Linear Logit
Test Error: 0.201

Quadratic Logit
Test Error: 0.197

Purple dashed: Bayes decision boundary
Black: polynomial logit



Purple dashed: Bayes decision boundary
Black: polynomial logit

Cubic Logit
Test Error: 0.160

Quartic Logit
Test Error: 0.162

Overall Process of Cross Validation for Hyperparameter Tuning

- Some machine learning algorithms have hyperparameters that cannot be directly estimated from the data.
- Cross-validation provides a simple way to tune parameters.

```
Define a grid of parameter values
for each parameter value do
    for each cross-validation iteration do
        Hold-out specification samples
        [Optional] Pre-process the data
        Fit the model on the remainder
        Predict the hold-out samples
    end
    Calculate the average performance across all iterations
end
Determine the optimal parameter value
Fit the final model to all training data using the optimal parameter value
```

Nested Cross-Validation

- Cross-validation can be used to only tune hyperparemeters, or estimate performance of the model.
- It can also be used in a nested structure, to both tune hypermarameters and estimate performance.

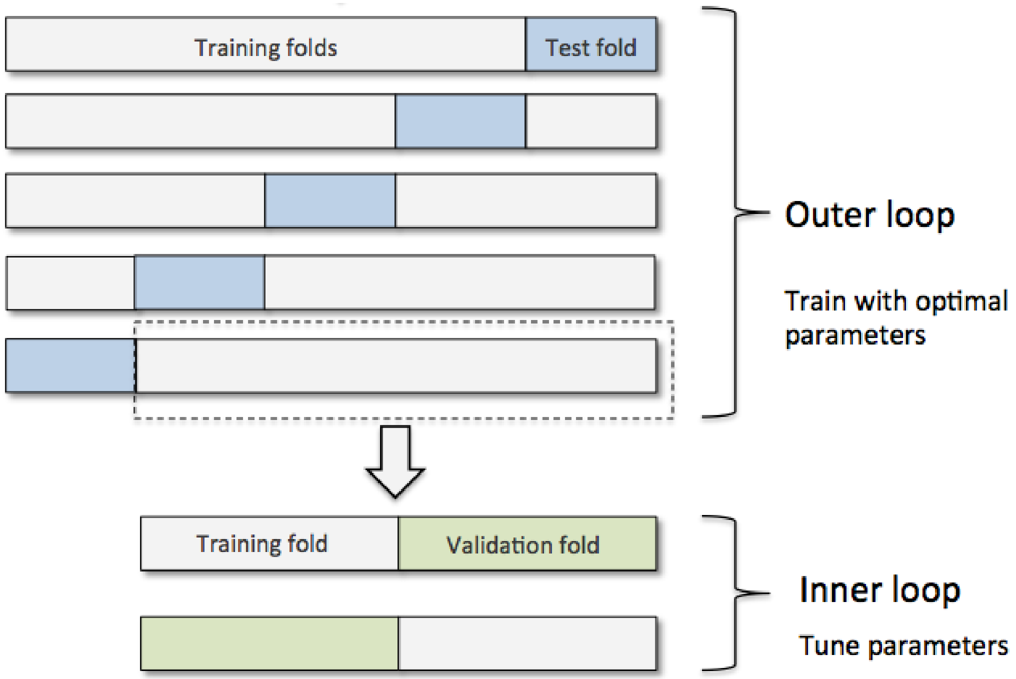


Image: https://sebastianraschka.com/faq/docs/evaluate-a-model.html

- Use inner loop to tune hyperparameters
- Use outer loop to estimate performance

AGENDA

- Introduction to Resampling Methods
- Using Caret Package
- Repeated K-Fold Cross-Validation
- Bootstrap

Use caret R Package

- caret = classification and regression training
- The caret package is a set of functions that attempt to streamline the process for creating predictive models.
- The package contains tools for:
  - data splitting
  - pre-processing
  - feature selection
  - model tuning using resampling
  - variable importance estimation
  - ……

- To learn more, visit http://topepo.github.io/caret/index.html

Validation Set (Simple Split)

- A single 80/20% split of the corolla data

```
# Read data file
df <- read.csv("Telco-Customer-Churn.csv")

# Use caret package
library(caret)

# Data partition
set.seed(1234)
trainIndex <- createDataPartition(df$Churn, p = .8, list = FALSE)
head(trainIndex)

train_data <- df[ trainIndex,]
test_data  <- df[-trainIndex,]
```

Advanced Modeling Training/Tuning

- Use caret::train() to tune model parameters

```
Define a grid of parameter values
for each parameter value do
    for each cross-validation iteration do
        Hold-out specification samples
        [Optional] Pre-process the data
        Fit the model on the remainder
        Predict the hold-out samples
    end
    Calculate the average performance across all iterations
end
Determine the optimal parameter value
Fit the final model to all training data using the optimal parameter value
```

Why the final model is fitted to all training data in the final step?

K-Fold Cross Validation

- Use 5-fold Cross-Validation

```
fitControl <- trainControl(method = "cv",number = 5)

set.seed(123)
svmRadial_fit <- train(Churn ~ ., data = train_data[-1],
                     trControl = fitControl,
                      method = “svmRadial")
print(svmRadial_fit)
```

The train() method in caret only support a few performance measures (overall accuracy and kappa for classification, RMSE, MAE, and R^2 for regression). If you need other measures, you can implement the k-fold cross-validation by your own code.

Example

- Customer Churn Analysis

AGENDA

- Introduction to Resampling Methods
- Using Caret Package
- Repeated K-Fold Cross-Validation
- Bootstrap

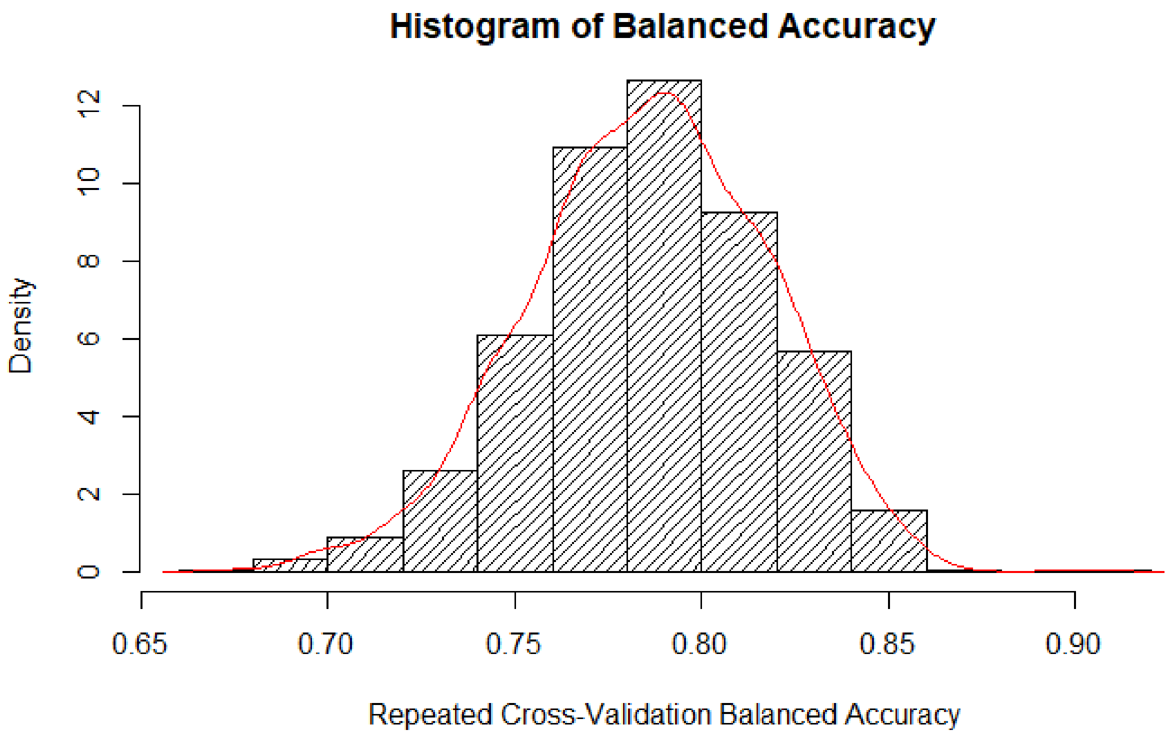
Extension on k-Fold Cross-Validation

- Repeated k-Fold Cross-Validation
  - Alleviate the random effect due to resampling

- Repeated Stratified k-Fold Cross-Validation
  - Alleviate the random effect due to resampling
  - Make sure the folds are balanced, in order to provide a more accurate performance estimate

Repeated K-Fold Cross Validation

- K-fold cross validation does not provide a robust estimate of mean performance.
- We can repeat the k-fold cross validation multiple times to better estimate performance.



How to Implement Repeated K-Fold Cross Validation

- Method 1: Using caret, you can set the train control as:

```
trainControl(method = "repeatedcv",
             number = 5,
             repeats = 200)
```

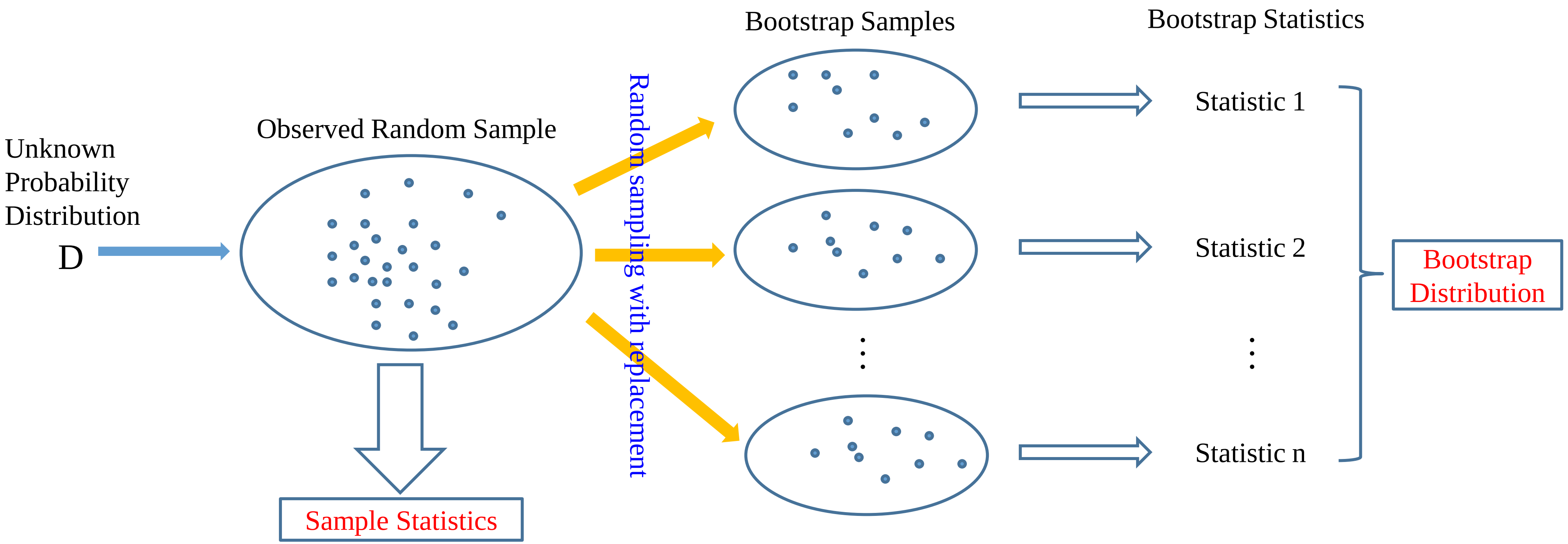
- Method 2: Directly implement the logic by your own code.
  - Refer to Example: Titanic Survival Analysis

AGENDA

- Introduction to Resampling Methods
- Using Caret Package
- Repeated K-Fold Cross-Validation
- Bootstrap

Bootstrap

- Bootstrap provides a general way for quantifying uncertainty of a statistical method based on random sampling with replacement.



Bootstrap distribution is usually closer to true distribution than sample statistics.

Implementing Bootstrap

- Refer to Example: Bootstrap

Summary of Bootstrap

- Bootstrap is based on the law of large numbers: if we sample enough times, we can approximate the true population distribution.

- The number of bootstrap samples should be large (e.g., 1000).

- Bootstrap can easily derive standard errors and confidence intervals for complicated statistics. Hypothesis testing can be very simple.

- Bootstrap works for small sample.

RECAP: OUTLINE

- (I) Resampling methods
  1. Cross-validation
     - I. K-fold cross validation
     - II. Leave-one-out cross validation
  2. Bootstrap

- (II) Implementing resampling methods in R
  1. Using caret package for quick experimentation
  2. Directly implement the logic for more detailed control

Q & A

IST 5535: Machine Learning Algorithms and Applications

Langtao Chen, Spring 2021

Linear Model Selection and Regularization

Reading

- Book Chapter 6 (6.1, 6.2, 6.5, 6.6)

OUTLINE

- (I) Need of Linear Model Selection and Regularization
- (II) Subset Selection
  - Best Subset Selection
  - Stepwise Selection
  - Choosing the Optimal Model
- (III) Shrinkage Methods
  - Ridge Regression
  - The Lasso

AGENDA

- Need of Linear Model Selection and Regularization
- Subset Selection
- Shrinkage/Regularization Methods

Linear Models

- Recap: linear regression model

$$y = \beta_0 + \ \beta_1 x_1 + \beta_2 x_2 + \ \dots \dots + \beta_p x_p + \varepsilon$$

- Despite its simplicity, linear model has distinct advantages:
  - Model interpretation is easy;
  - Predictive performance is often good on real-world problems.

- We usually use OLS to estimate linear model. However, other methods may yield better prediction accuracy and model interpretability.

OLS Failure: Include All Predictors in a Model

- A dataset with complete information is not available or expensive to collect
- May have a serious missing data issue with more predictors
- May not be able to accurately measure some predictors
- Using predictors that are unrelated with the response will increase the variance of the prediction

A parsimonious model helps to unveil the underlying relationships with stable estimates of coefficients (especially for an explanatory model).

When OLS May Not Work?

- 1. Prediction Accuracy
  - Given that the true relationship between the response and predictors is approximately linear, the OLS estimates have low bias.
  - According to the tradeoff between bias and variance, an optimal model should also have low variance.
    - If $n >> p$, OLS estimates tend to have low variability.
    - However, if n is not much larger than p, OLS estimates can have high variability and lead to over fitting and poor prediction.
    - If $n < p$, OLS will fail (variance is infinite).

When OLS May Not Work? (cont.)

- 2. Model Interpretability
  - When we have a large number of predictors, some or many variables will be in fact not associated with the response.
  - Including such *irrelevant* variables leads to unnecessarily complexity in the model, thus making the model hard to interpret.
  - The model could be easier to interpret if we remove those irrelevant variables (or set their coefficients as zeros).

Three Classes of Methods Alternative to OLS

- Subset Selection (a.k.a. Feature Selection or Variable Selection)
  - Identify a subset of the p predictors that we believe to be related to the response.
  - Then fit a model using least squares on the reduced set of variables.

- Shrinkage (a.k.a. Regularization)
  - Fit a model involving all p predictors. However, the estimated coefficients are shrunken towards zero relative to the least squares estimates.
  - This shrinkage has the effect of reducing variance.
  - Some of the coefficients may be estimated to be exactly zero. Hence, shrinkage methods can also perform variable selection.

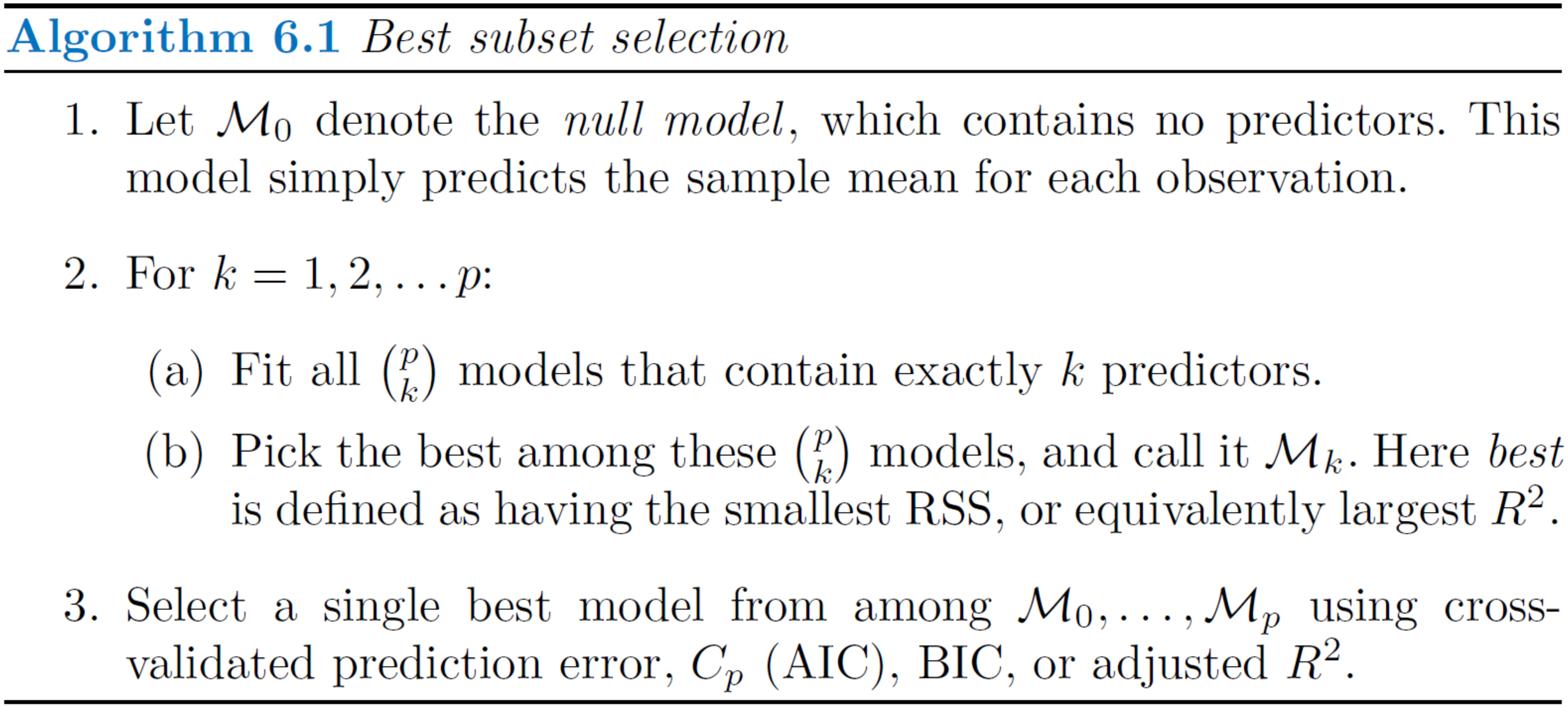
- Dimension Reduction
  - Project the p predictors into a M-dimensional subspace, where $M < p$.
  - Then these M projections are used as predictors to fit a linear regression model by least squares.

AGENDA

- Need of Model Selection and Regularization
- Subset Selection
- Shrinkage/Regularization Methods

Subset Selection

- Best Subset Selection



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 $R^2$.
```

Quiz

- Given a dataset with potentially 3 predictors, how many models will be evaluated during the best subset selection process?
  - (A) 3
  - (B) 4
  - (C) 7
  - (D) 8

Quiz

- Given a dataset with potentially 3 predictors, how many models will be evaluated during the best subset selection process?
  - (A) 3
  - (B) 4
  - (C) 7
  - (D) 8

Answer

Null model $M_{0:}$ 1

Models with one predictor $M_{1:}$ $\binom{1}{3} = 3$

Models with two predictors $M_{2:}$ $\binom{2}{3} = 3$

Models with three predictors $M_{3:}$ $\binom{3}{3} = 1$

In total: 1 + 3 + 3 + 1 = 8 models

Subset Selection

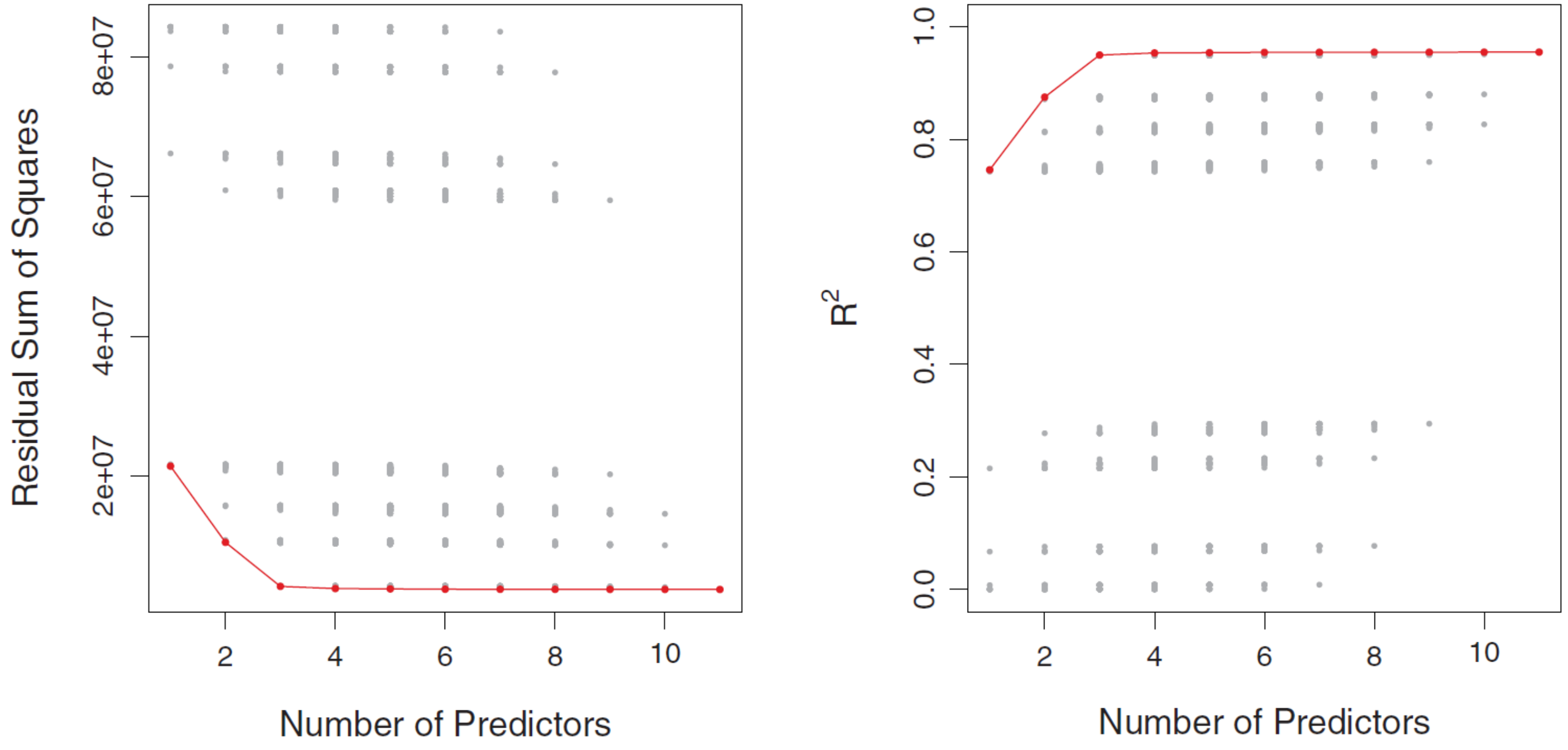
- Best Subset Selection is computationally intensive especially when p is large.

- More attractive alternative: stepwise selection
  - Forward Stepwise Selection
    - Start with no predictors
    - Add them one by one (add the one with largest contribution)
    - Stop when the addition is not statistically significant

  - Backward Stepwise Selection
    - Start with all predictors
    - Successively eliminate least useful predictors one by one
    - Stop when all remaining predictors have statistically significant contribution

Choose the Optimal Model

- As the number of predictors increases, RSS always decreases and R^2 always increases.
- Thus, RSS and R^2 are not suitable for selecting the best model among models with different number of predictors.



Other Measures to Consider for Model Comparison

- The following measures add a heavier penalty on models with many variables:
  - C_p statistic

    $$C_p = \frac{1}{n}(RSS + 2d\hat{\sigma}^2)$$ where d is the number of predictors
  - AIC (Akaike information criterion)

    $$AIC = \frac{1}{n\hat{\sigma}^2}(RSS + 2d\hat{\sigma}^2)$$
  - BIC (Bayesian information criterion)

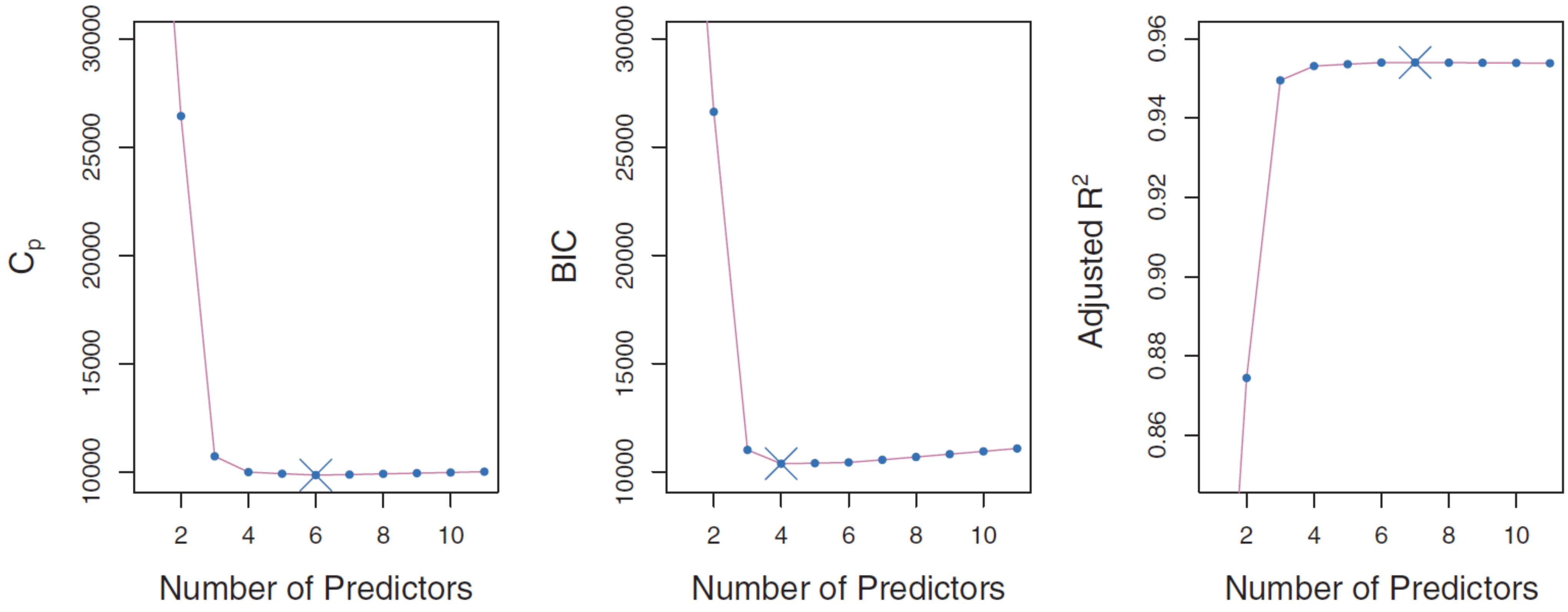
    $$BIC = \frac{1}{n\hat{\sigma}^2}(RSS + \log(n)d\hat{\sigma}^2)$$
  - Adjusted R^2

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

Smaller C_p, AIC, and BIC are better; Larger adjusted R^2 is better.

An Example of Model Selection

- Smaller C_p, AIC, and BIC are better;
- Larger adjusted R^2 is better.



AGENDA

- Need of Model Selection and Regularization
- Subset Selection
- Shrinkage/Regularization Methods

Shrinkage/Regularization Methods

- The above subset selection methods involve using OLS 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 coefficient estimates, or equivalently, that *shrinks* the coefficient estimates towards zero.

- Shrinking the coefficient estimates can significantly reduce their variance.

- Regularization reduces parameters and shrinks the model, thus avoiding over-fit.

- Two best known shrinkage methods: *ridge regression* and the *lasso*

Ridge Regression

- The OLS fitting procedure minimizes the RSS

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

- The ridge regression minimizes

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

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

shrinkage penalty

Shrinkage Penalty

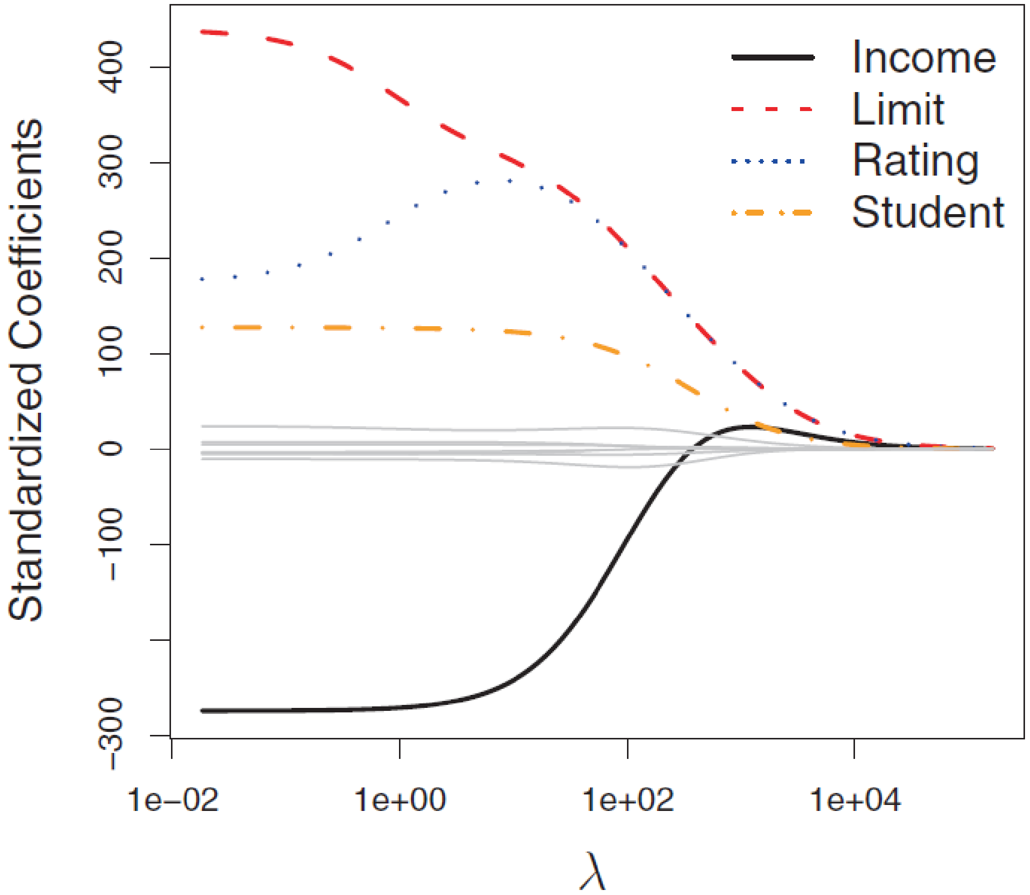
- The ℓ_2 penalty term $||\beta_j||_2 = \sum_{j=1}^{p} \beta_j^2$ has the effect of shrinking coefficient estimates β_j towards zero.

- The tuning parameter $\lambda \geq 0$ controls the relative importance of the penalty term in the overall optimization of the objective function.
  - When $\lambda = 0$, the penalty term does not have effect. Ridge regression results in OLS estimates;
  - When λ is large, the impact of the penalty term grows. $\beta_j (j = 1,2, \dots, p)$ has to be close to zero.

- It's critical to select an appropriate value for λ. In practice, cross-validation is used to tune this parameter.

Example: Ridge Regression on Credit Data

- As λ increases, the ridge coefficient estimates shrink towards zero.



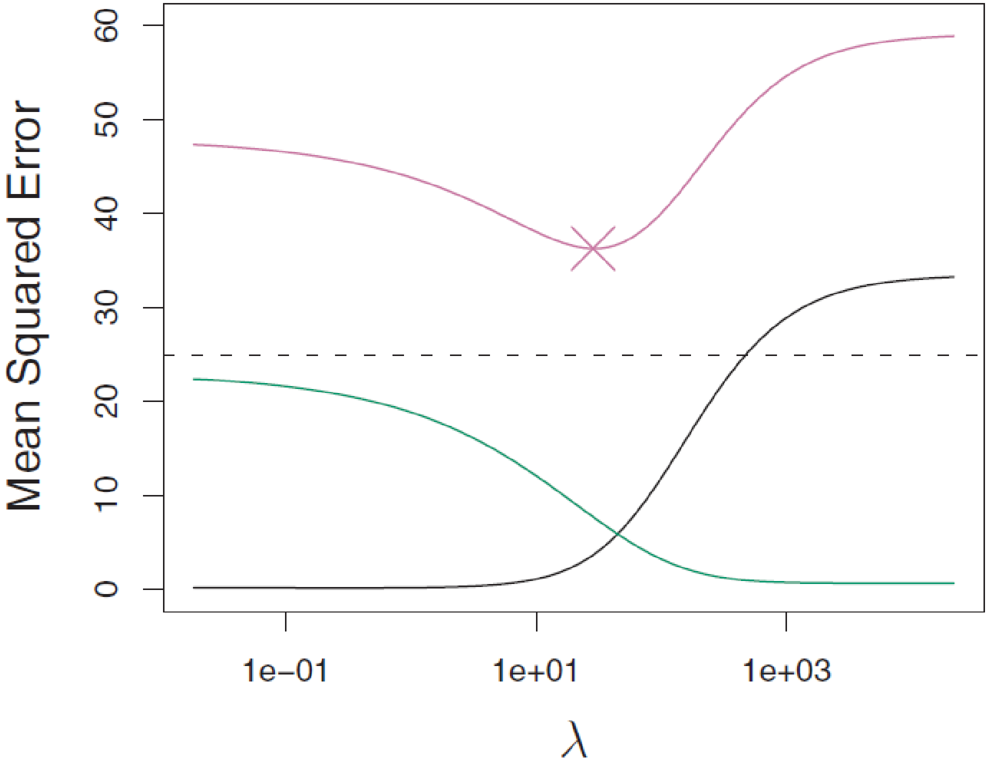
Why Does Ridge Regression Improve Over OLS?

- OLS estimates have low bias. However, if the condition $n >> p$ does not hold, OLS estimates may have large variance.

- By adding the shrinkage penalty, ridge regression leads to more biased but less variable estimates.

- Ridge regression can make a better trade-off between bias and variance, thus improving over OLS.

- Ridge regression works best in situations where OLS estimates have high variance.



Black: Squared Bias
Green: Variance
Purple: Test MSE

The Lasso

- One problem for ridge regression:
  - It shrinks all coefficients towards zero, but it will not set any of them exactly to zero;
  - Thus, ridge regression cannot conduct variable selection.
  - As all *p* variables will be included in the final model, there could be a challenge in model interpretation.

- Lasso is a more recent alternative to ridge regression that overcomes this disadvantage. Lasso coefficients minimize:

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

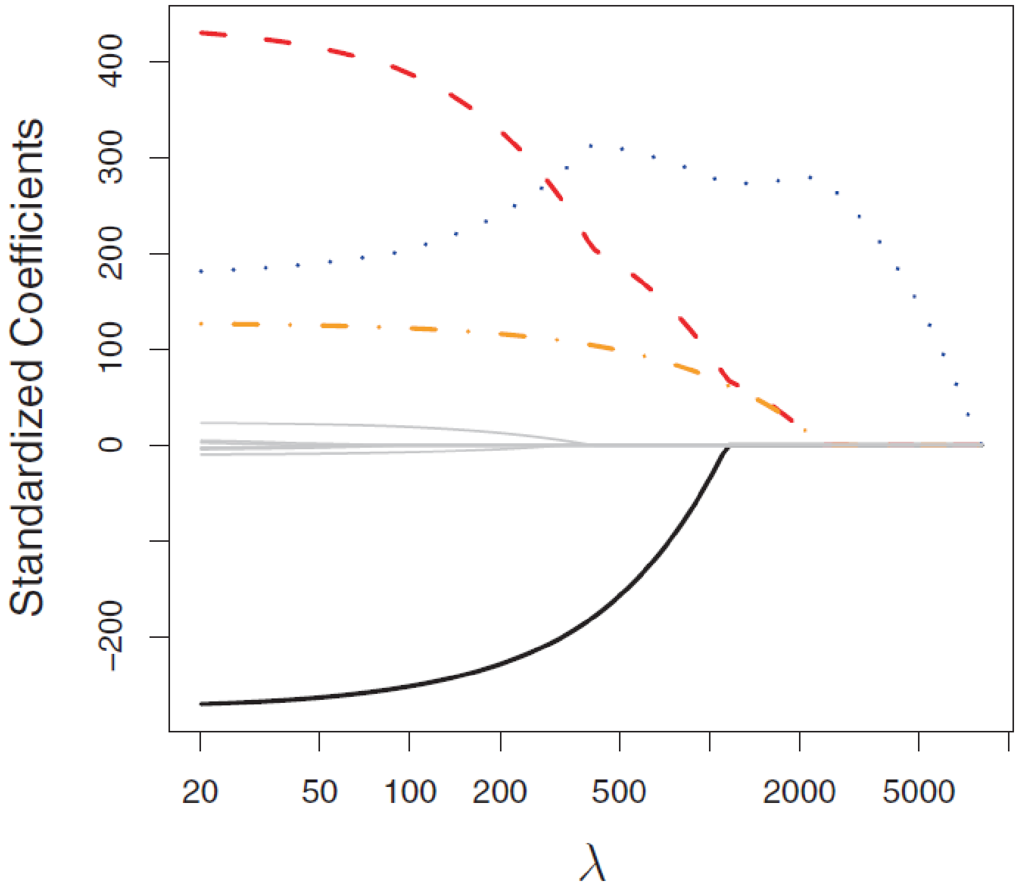
Lasso works in similar way as ridge regression, except using an ℓ_1 penalty.

Lasso Penalty Term

- The lasso ℓ_1 penalty $||\beta_j||_1 = \sum_{j=1}^{p} |\beta_j|$ can force some coefficient estimates to be exactly equal to zero, when the tuning parameter λ is large enough.
- Thus, lasso performs variable selection.
- Models generated from lasso are generally much easier to interpret than those produced by ridge regression.

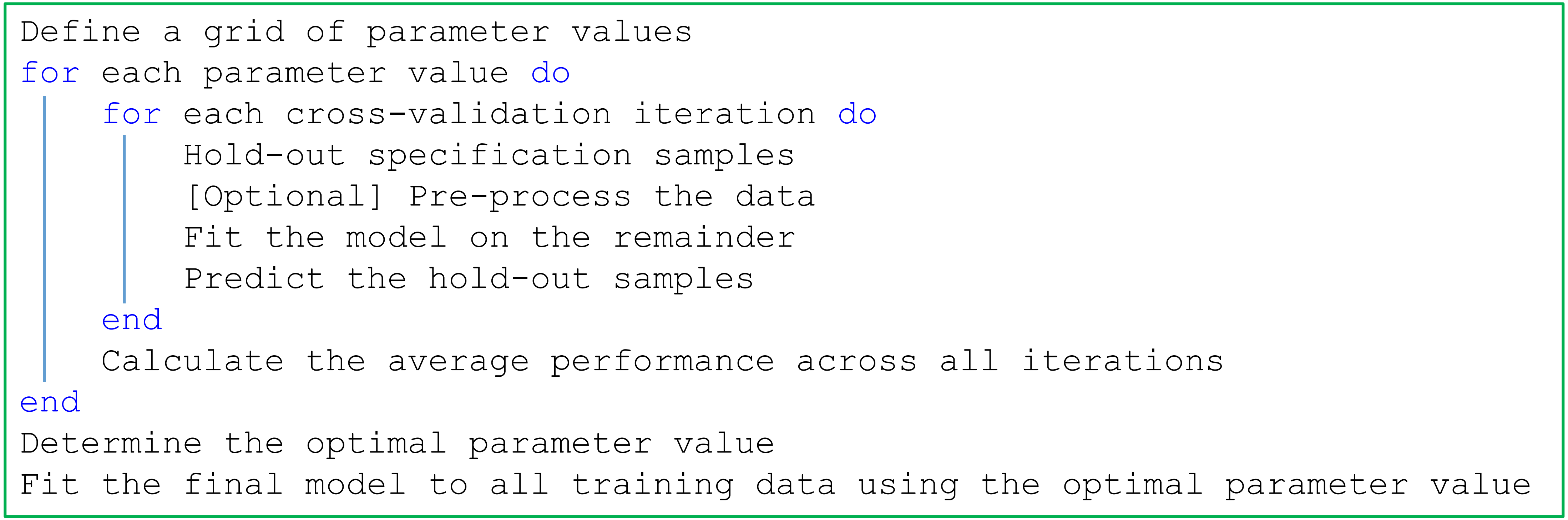
Example: Lasso on Credit Data

- Depending on the value of λ, lasso can produce a model with any number of variables.



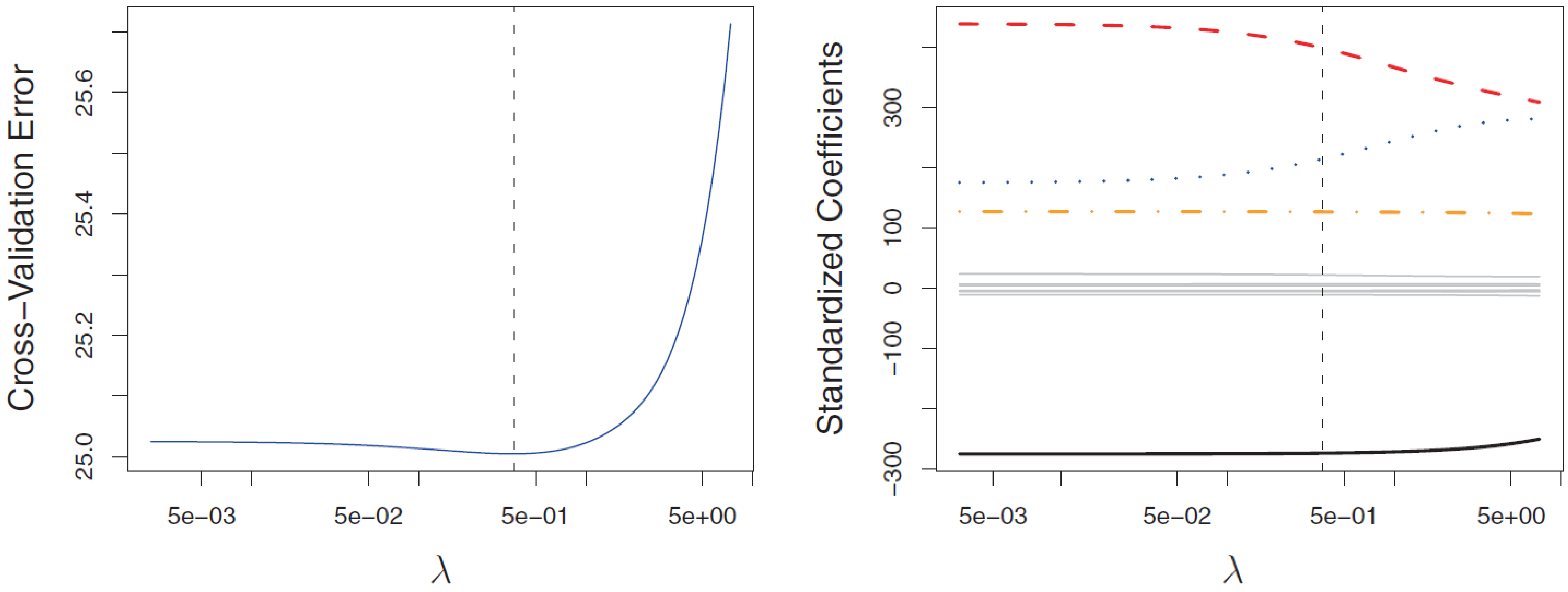
Selecting the Tuning Parameter

- Implementing ridge regression and the lasso requires a method of selecting an optimal value for the tuning parameter λ.
- Cross-validation provides a simple way to tune parameters.



```
Define a grid of parameter values
for each parameter value do
    for each cross-validation iteration do
        Hold-out specification samples
        [Optional] Pre-process the data
        Fit the model on the remainder
        Predict the hold-out samples
    end
    Calculate the average performance across all iterations
end
Determine the optimal parameter value
Fit the final model to all training data using the optimal parameter value
```

Example: Tuning λ for Ridge Regression



Recap: Outline

- (I) Need of Linear Model Selection and Regularization
- (II) Subset Selection
  - Best Subset Selection
  - Stepwise Selection
  - Choosing the Optimal Model
- (III) Shrinkage Methods
  - Ridge Regression
  - The Lasso

Q & A

IST 5535: Machine Learning Algorithms and Applications

Langtao Chen, Spring 2021

Tree-Based Methods

Reading

- Tree-based methods: book chapter 8

Learning Objectives

- Explain the regression tree and classification algorithms.
- Explain the advantages and disadvantages of decision trees compared with other supervised machine learning methods.
- Explain why pruning a decision may improve performance.
- Explain ensemble learning methods and be able to explain three basic types of ensemble.
- Explain bagging trees.
- Explain random forests and compare random forests with bagging trees.
- Implement tree-based methods in R. Use cross-validation to tune hyperparameters.

AGENDA

- Regression and Classification Trees
- Ensemble Learning and Random Forests

Decision Tree Methods

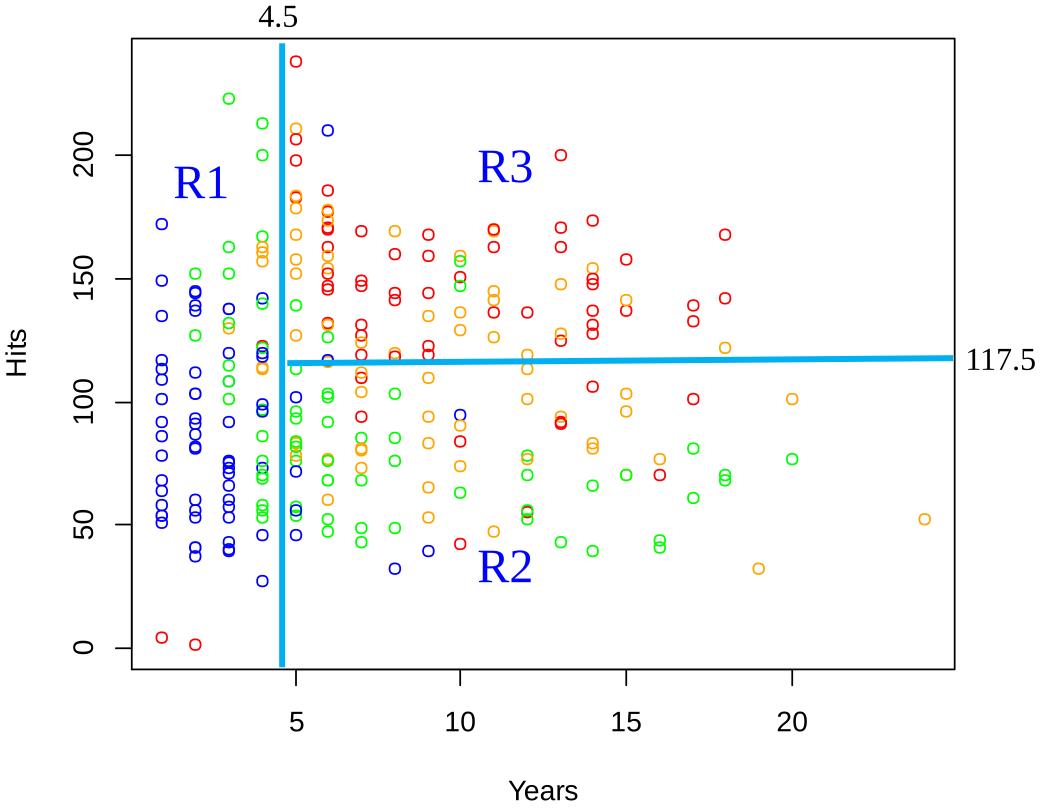
- Decision tree methods can be applied to both regression (regression trees) and prediction (classification trees) problems.

- These methods involve stratifying or segmenting the predictor space into a number of simple regions.

- The splitting rules can be summarized in a tree. That’s why they are called decision tree methods or tree-based methods.

Example: Predicting Baseball Players' Salaries Using Regression Tree

- Years: the number of years that a player has played in major leagues
- Hits: the number of hits that a player made in the previous year



Red: above 75% quantile

Orange: between 50% and 75% quantiles

Green: between 25% and 50% quantiles

Blue: below 25% quantiles

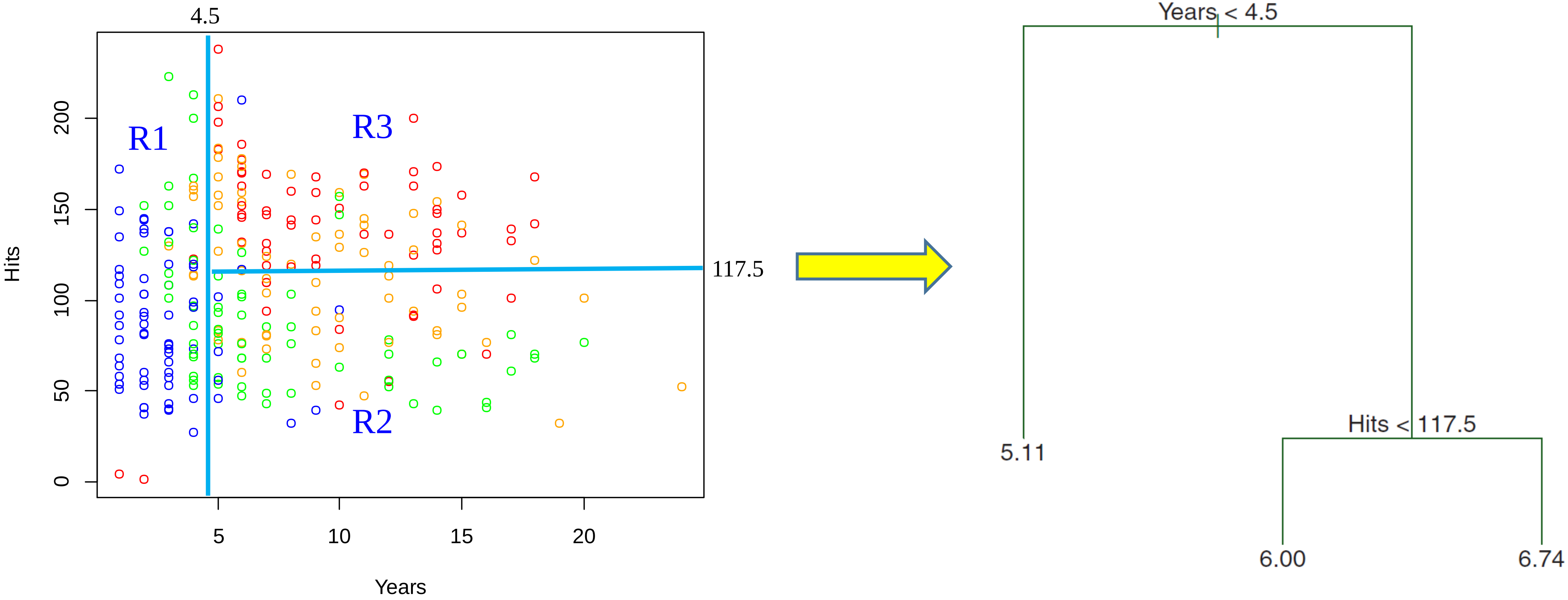
R1 = {X | Years < 4.5}

R2 = {X | Years >= 4.5, Hits < 117.5}

R3 = {X | Years >= 4.5, Hits >= 117.5}

Example: Predicting Baseball Players’ Salaries Using Regression Tree

- Construct a regression tree

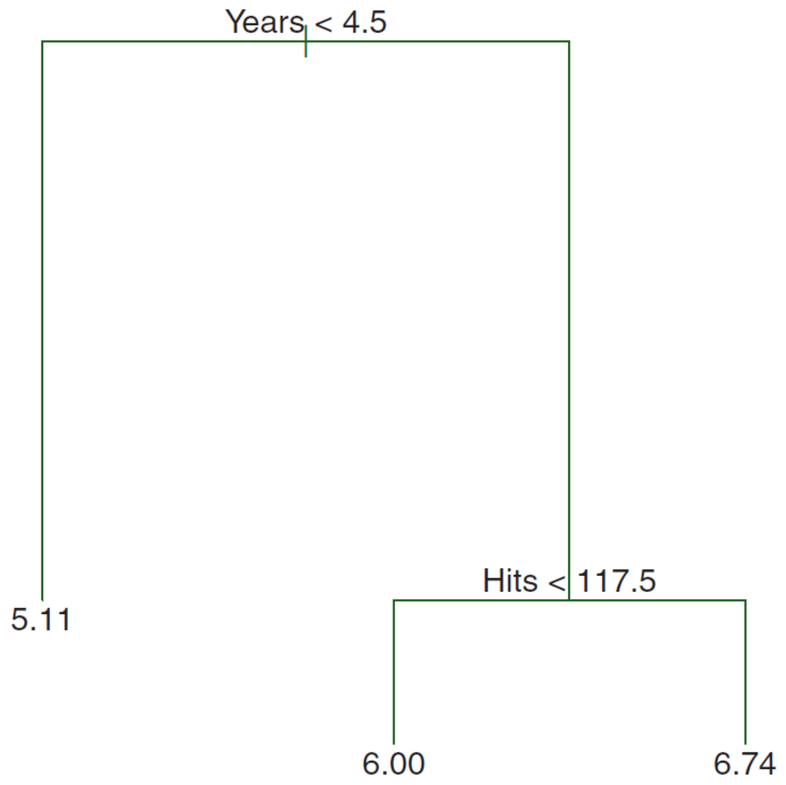


Salary (in $1000) is log transformed; 5.11 is the mean of log(salary) in R1 region.

Predicted salary for a player with more than 5 years and at least 118 hits last year = $1,000*exp(6.74) = $845,561

Interpreting Regression Trees Is Easy

- Years is the most important factor in determining Salary, and players with less experience earn lower salaries than more experienced players.
- Given that a player is less experienced, the number of hits that he made in the previous year seems to play little role in his salary.
- But among players who have been in the major leagues for five or more years, the number of hits made in the previous year does affect salary, and players who made more hits last year tend to have higher salaries.



Building A Regression Tree

- Step 1: We divide the predictor space – that is, the set of possible values for $X_1, X_2, \dots, X_p$ – into J distinct and non-overlapping regions, $R_1, R_2, \dots, R_J$.

- Step 2: For every observation that falls into the region R_j, we make the same prediction, which is simply the mean of the response values for the training observations in R_j.

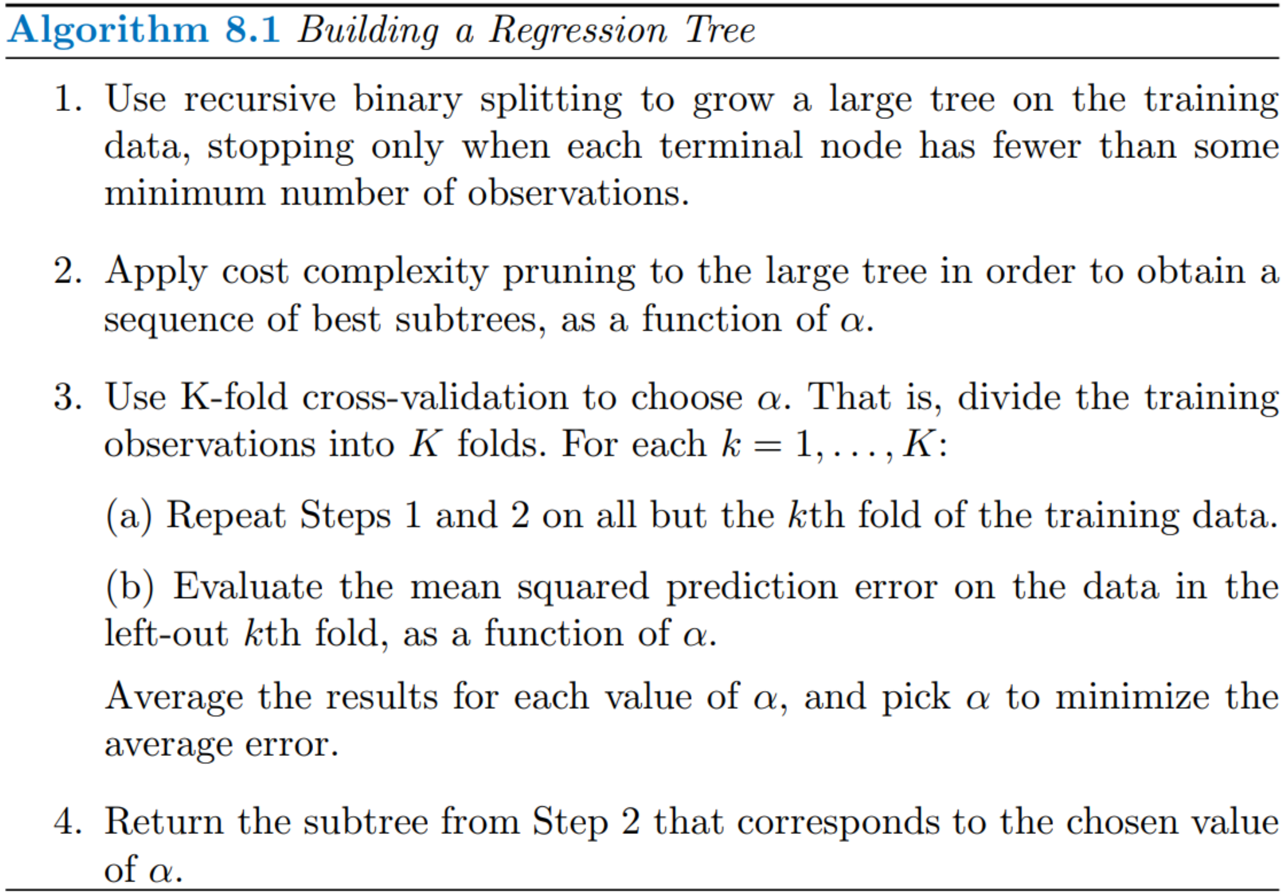
Pruning Tree

- A tree with many terminal nodes tends to over-fit data (low bias, high variance).

- Pruning a tree may improve the prediction performance of the tree by having a better tradeoff between bias and variance.

- Pruned tree has a better interpretation as a smaller set of decisions rules are generated from the pruned tree.

Target: minimize $$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$



Algorithm 8.1 *Building a Regression Tree*

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.

2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of α.

3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$:

   (a) Repeat Steps 1 and 2 on all but the kth fold of the training data.

   (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.

   Average the results for each value of α, and pick α to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of α.

Decision Tree Algorithms

- There are a couple of decision tree algorithms
  - ID3, C4.5, C5, CART, CHAID, M5 etc.

- These decision tree algorithms mainly differ on
  - Splitting criteria: which variable, what value, etc.
  - Stopping criteria: when to stop building the tree
  - Pruning (generalization method): Pre-pruning versus post-pruning

Growing A Classification Tree

- A *classification tree* is very similar to a regression tree, except that we need to predict a qualitative response rather than a continuous one.
- For each region (or node) we predict the most commonly occurring class among the training data in the region (majority vote).
- Growing a decision tree is similar as growing a regression tree, except that minimizing classification error rate rather than RSS is the objective.
- Alternative criteria to classification error include Gini index and cross-entropy. The two alternative measures are more sensitive to node purity.

Trees Vs. Linear Models

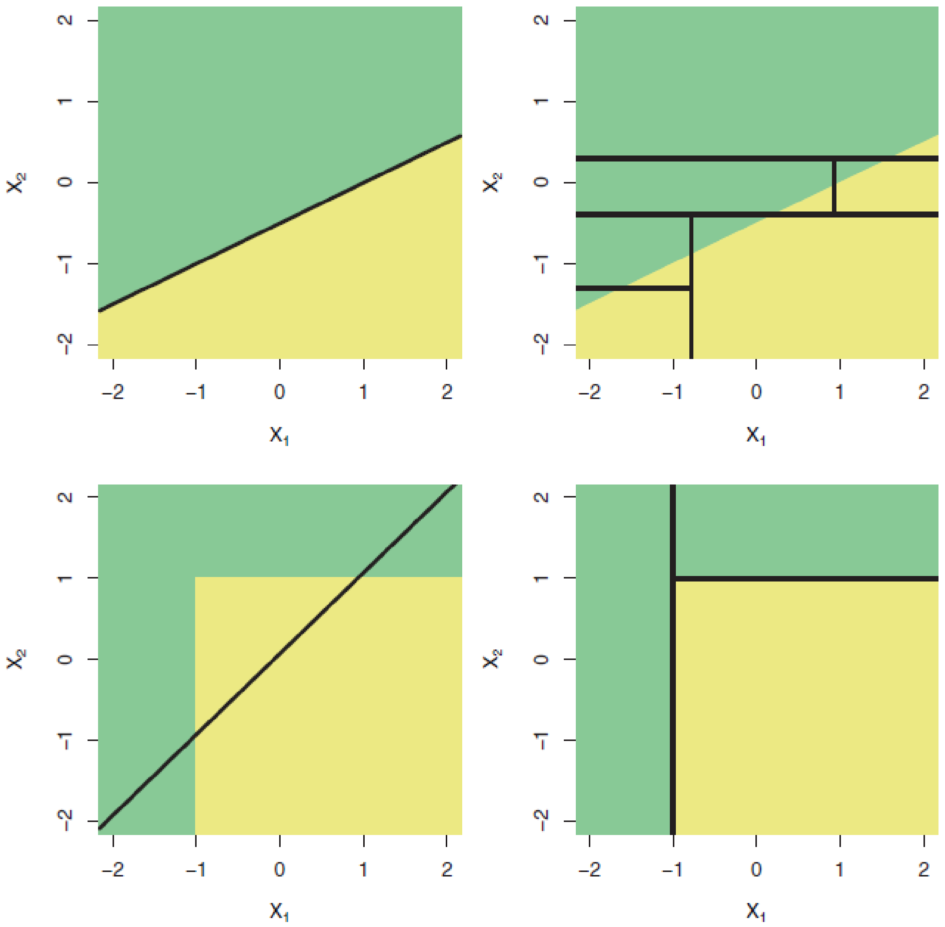
- Linear regression model

$$f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j$$

- Regression tree model

$$f(X) = \sum_{m=1}^{M} c_m \cdot 1_{(X \in R_m)}$$

Top Row: True linear boundary, linear regression works well



Bottom Row: True nonlinear boundary, decision trees are preferred

Summary of Decision Trees

- Advantages
  - Trees are very easy to explain to people. In fact, they are even easier to explain than linear regression!
  - Some people believe that decision trees more closely mirror human decision-making than do the regression and classification approaches.
  - Trees can be displayed graphically, and are easily interpreted even by a non-expert (especially if they are small).
  - Trees can easily handle qualitative predictors without the need to create dummy variables.

- Disadvantages
  - Trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches.
  - Additionally, trees can be very non-robust. In other words, a small change in the data can cause a large change in the final estimated tree.

AGENDA

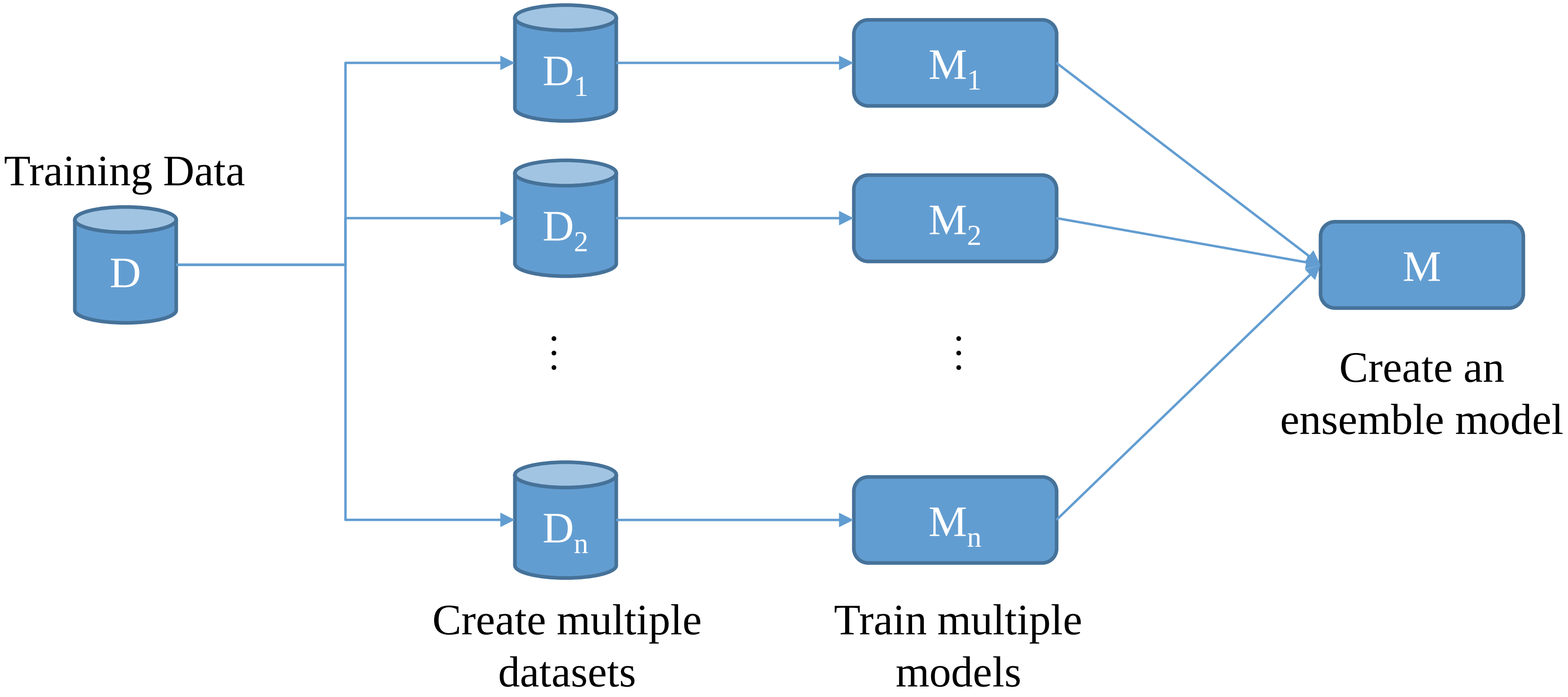
- Regression and Classification Trees
- Ensemble Learning and Random Forests

Improving Decision Trees

- Decision trees usually do not have the same level of predictive accuracy as other supervised machine learning approaches.
- Trees can be very non-robust. A small change in the data can cause a large change in the final estimated tree (large variance).
- To reduce the variance of decision trees, we can aggregate many decision trees to form a single classifier.
- By having a better trade-off between bias and variance, predictive performance can be substantially improved.

Ensemble Learning Methods

- Like the “wisdom of crowd”, ensemble learning methods:
  - Train a collection of simple or weak learning methods;
  - Then combine their results to get a single and better algorithm.



Basic Types of Ensemble

- Bagging (bootstrap aggregating): train learners in parallel on different samples of the data, then combine by voting (classification) or by averaging (regression).

- Stacking: combine model outputs using a second-stage learner like linear regression.

- Boosting: train learners on the filtered output of other learners.

Bagging Trees

- Bagging trees is an ensemble of n_{tree} decision trees by bagging method:
  - Step 1: Draw n_{tree} bootstrap samples from the original data with replacement;
  - Step 2: For each of the bootstrap samples, grow an unpruned decision tree with each node split among all predictors;
  - Step 3: Then combine the n_{tree} decision trees by aggregation:
    - For classification, use majority votes;
    - For regression, use average.

Bagging Trees Vs. Decision Tree

- Bagging generates a large number of decision trees;
- Bagging improves prediction accuracy at the expense of interpretability.

Random Forests

- Random forests (RF) is an extension of bagging trees. RF usually has a better performance than bagging trees.

- Random forests is an ensemble of n_{tree} decision trees by bagging method:
  - Step 1: Draw n_{tree} bootstrap samples from the original data with replacement;
  - Step 2: For each of the bootstrap samples, grow an unpruned decision tree with the following modification:
    - At each node, instead of choosing the best split among all predictors, randomly sample m_{try} predictors;
    - Choose the best split from the m_{try} predictors;
  - Step 3: Then combine the n_{tree} decision trees by aggregation:
    - For classification, use majority votes;
    - For regression, use average.

Random Forests

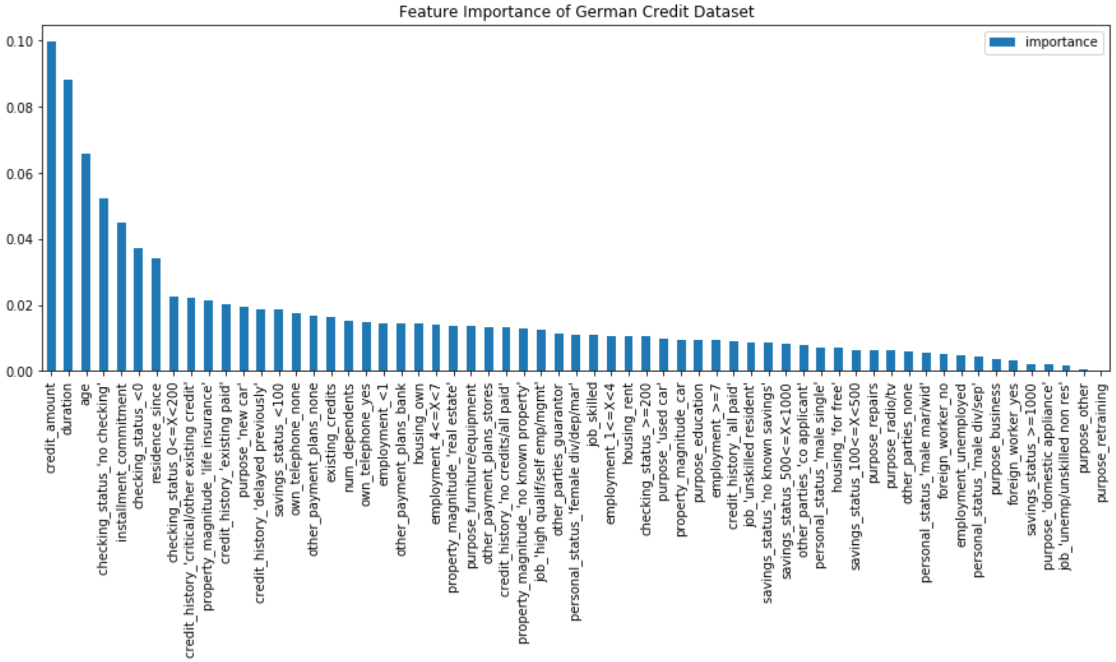
- The principle of diversity: Need to grow different rather than similar decision trees, in order to get the “wisdom of crowd”.

- The diversity principle is implemented by:
  - Draw n_{tree} bootstrap samples from the original data with replacement;
  - When growing a decision tree, randomly choose m_{try} predictors to find the best split for each node.

n_{tree} and m_{try} are two hyperparameters of Random Forests.

Variable/Feature Importance

- Random forests provide the importance score for all predictors.
- The importance of a feature measures how much the feature can help reduce impurity of the data.
- Thus, random forests can be used to select features for other algorithms.
  - In practice, many competitions use random forests for feature selection.

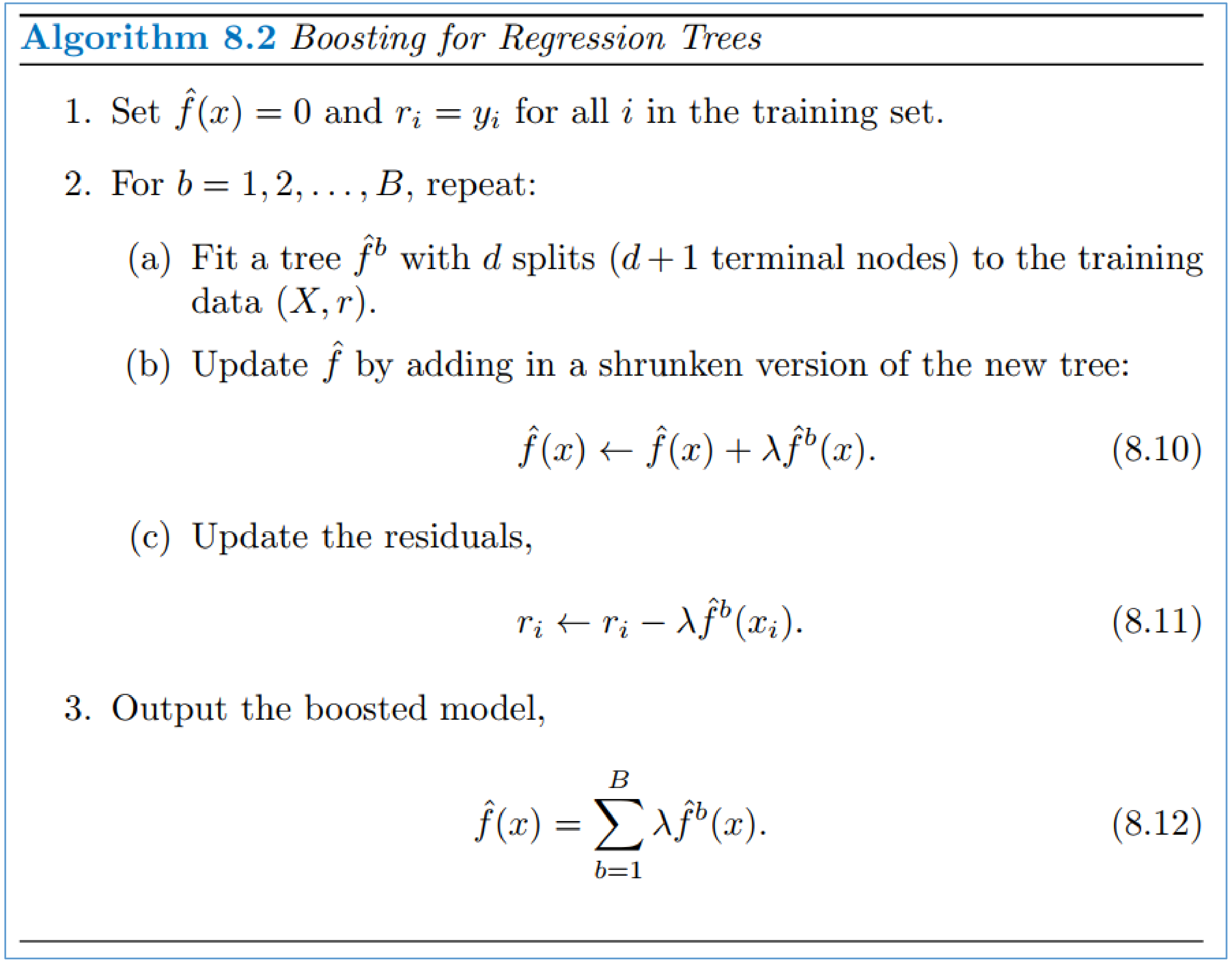


Boosting

- Fit a decision tree using outcome Y;
- Fit a small decision tree to the current residual;
- Repeat the small decision tree building to slowly boost/improve the current model.

No bootstrap sampling involved.

A famous algorithm is called GBM (Gradient Boosting Machine).



Algorithm 8.2 *Boosting for Regression Trees*

```
1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all $i$ in the training set.
2. For $b = 1, 2, \ldots, B$, repeat:
    (a) Fit a tree $\hat{f}^b$ with $d$ splits ($d + 1$ terminal nodes) to the training data $(X, r)$.
    (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:
            $\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x).$    (8.10)
    (c) Update the residuals,
            $r_i \leftarrow r_i - \lambda \hat{f}^b(x_i).$    (8.11)
3. Output the boosted model,
            $\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x).$    (8.12)
```

Q & A