# Notes: Introduction to Statistical Learning

A. John Woodill

Revision 1 – October 2018

# Contents


**1 Introduction** **3**

**2 Statistical Learning** **3**

2.1 Overview . . . 3

2.1.1 Predictions . . . 3

2.1.2 Inference . . . 3

2.1.3 Estimating $f()$ . . . 4

2.1.4 Prediction Accuracy versus Model Interpretability . . . 4

2.1.5 Supervised Versus Unsupervised Learning . . . 5

2.1.6 Regression versus Classification Problems . . . 6

2.2 Assessing Model Accuracy . . . 6

2.2.1 Measuring Quality of Fit . . . 6

2.2.2 The Bias-Variance Trade-Off . . . 7

Bias-Variance Trade-Off . . . 8

2.2.3 Classification Strategy . . . 8

The Bayes Classifier . . . 8

K-Nearest Neighbors . . . 8

**3 Linear Regression** **9**

3.1 Simple Linear Regression . . . 9

3.1.1 Estimating the Coefficients . . . 9

3.1.2 Assessing the Accuracy of the Coefficient Estimates . . . 10

3.1.3 Assessing the Accuracy of the Model . . . 10

Residual Standard Error . . . 11

$R^2$ Statistic . . . 11

3.2 Multiple Linear Regression . . . 11

3.2.1 Estimating the Regression Coefficients . . . 12

1. Is there a relationship between the response and predictors? . . . 12

2. Deciding on Important Variables . . . 12

3. Model Fit . . . 12
4. Predictions . . . 13
3.3 Other Considerations in the Regression Model . . . 13
3.3.1 Qualitative Predictors . . . 13
Predictors with two levels . . . 13
Predictors with more than two levels . . . 13
3.3.2 Extensions of the Linear Model . . . 13
Removing the Additive Assumption . . . 14
Non-linear relationship . . . 14
3.3.3 Potential Problems . . . 14
1. Non-linearity of the Data . . . 14
2. Correlation of Error Terms . . . 15
Non-constant Variance of Error Terms . . . 16
Outliers . . . 17
5. High Leverage Points . . . 17
6. Collinearity . . . 18
3.4 Comparison of Linear Regression with K-Nearest Neighbors . . . 18
3.5 Classification . . . 20
3.5.1 Logistic Regression . . . 20
3.5.2 Estimating the Regression Coefficients . . . 21
3.5.3 Making Prediction . . . 21
3.5.4 Multiple Logistic Regression . . . 21
**Logistic Regression for $>$ 2 Response Classes** . . . 21
3.5.5 Linear Discriminant Analysis . . . 21
3.5.6 Using Bayes' Theorem for Classification . . . 22
3.5.7 Linear Discriminant Analysis for p=1 . . . 22
3.5.8 Linear Discriminant Analysis for p $>$ 1 . . . 23
3.5.9 Quadratic Discriminant Analysis (QDA) . . . 24
3.6 Comparison of Classification Methods . . . 25

**4 Resampling Methods** **25**

4.1 Cross-Validation . . . 25

4.1.1 The Validation Set Approach . . . 26

4.1.2 Leave-One-Out Cross-Validation . . . 26

4.1.3 k-Fold Cross-Validation . . . 26

4.1.4 Bias-Variance Trade-Off for k-Fold CV . . . 26

4.1.5 Cross-Validation on Classification Problems . . . 27

4.2 The Bootstrap . . . 27

**5 Linear Model Selection and Regularization** **27**

5.1 Subset Selection . . . 27

5.1.1 Best Subset Selection . . . 27

5.1.2 Stepwise Selection . . . 28

5.1.3 Choosing the Optimal Model . . . 28

Cp, AIC, BIC, and Adjusted R2 . . . 28

Validation and Cross-validation . . . 29

5.2 Shrinkage Methods . . . 29

5.2.1 Ridge Regression . . . 29

Why Does Ridge Regression Improve Over Least Squares? . . . 29

5.2.2 The Lasso . . . 30

Comparing the Lasso and Ridge Regression . . . 30

5.2.3 Selecting the Tuning Parameter . . . 31

5.3 Dimension Reduction Methods . . . 31

5.3.1 Principal Components Regression . . . 31

The Principal Components Regression Approach . . . 31

5.3.2 Partial Least Squares . . . 33

5.4 Considerations in High Dimensions . . . 33

5.4.1 What Goes Wrong in High Dimensions? . . . 34

5.4.2 Regression in High Dimensions . . . 34

5.4.3 Interpreting Results in High Dimensions . . . 34

**6 Moving Beyond Linearity** **35**

6.1 Polynomial Regression . . . 35

6.2 Step Function . . . 35

6.3 Basis Function . . . 37

6.4 Regression Splines . . . 37

6.4.1 Piecewise Polynomials . . . 37

6.4.2 Constraints and Splines . . . 38

6.4.3 The Spline Basis Representation . . . 38

6.4.4 Choosing the Number and Locations of the Knots . . . 39

6.4.5 Comparison to Polynomial Regression . . . 40

6.5 Smoothing Splines . . . 40

6.5.1 An Overview of Smoothing Splines . . . 40

6.5.2 Choosing the Smoothing Parameter $\lambda$ . . . 41

6.6 Local Regression . . . 42

6.7 Generalized Additive Models (GAM) . . . 44

6.7.1 GAMs for Regression Problems . . . 45

6.7.2 GAMs for Classification Problems . . . 46

**7 Tree-Based Methods** **46**

7.1 The Basics of Decision Trees . . . 46

7.1.1 Regression Trees . . . 47

Tree Pruning . . . 47

7.1.2 Classification Trees . . . 49

7.1.3 Trees Versus Linear Models . . . 49

7.1.4 Advantages and Disadvantages of Tress . . . 49

7.2 Bagging, Random Forests, Boosting . . . 50

7.2.1 Bagging . . . 50

Out-of-Bag Error Estimation . . . 50

Variable Importance Measures . . . 51

7.2.2 Random Forests . . . 52

7.2.3 Boosting . . . . . 52

**8 Support Vector Machines** **53**

8.1 Maximal Margin Classifier . . . . . 53

8.1.1 What is a Hyperplace? . . . . . 53

8.1.2 Classification Using a Separating Hyperplane . . . . . 54

8.1.3 The Maximal Margin Classifier . . . . . 55

8.1.4 Construction of the Maximal Margin Classifier . . . . . 56

8.1.5 The Non-separable Case . . . . . 57

8.2 Support Vector Classifiers . . . . . 57

8.2.1 Overview of the Support Vector Classifier . . . . . 57

8.2.2 Details of the Support Vector Classifier . . . . . 57

8.3 Support Vector Machines . . . . . 59

8.3.1 Classification with Non-linear Decision Boundaries . . . . . 59

8.3.2 The Support Vector Machine . . . . . 60

8.4 SVMs with More than Two Classes . . . . . 61

8.4.1 One-Versus-One Classification . . . . . 61

8.4.2 One-Versus-All Classification . . . . . 61

8.5 Relationship to Logistic Regression . . . . . 61

**9 Unsupervised Learning** **62**

9.1 The Challenge of Unsupervised Learning . . . . . 62

9.2 Principal Components Analysis . . . . . 62

9.2.1 What Are Principal Components? . . . . . 63

9.2.2 Another Interpretation of Principal Components . . . . . 64

9.2.3 More on PCA . . . . . 65

Scaling the Variables . . . . . 65

Uniqueness of the Principal Components . . . . . 66

The Proportion of Variance Explained . . . . . 66

Deciding How Many Principal Components to Use . . . . . 66

9.2.4 Other Uses for Principal Components . . . . . 66

9.3 Clustering Methods . . . . . 67
9.3.1 K-Means Clustering . . . . . 67
9.3.2 Hierarchical Clustering . . . . . 69

# 1 Introduction

The following notes are from Introduction to Statistical Learning by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. For updated document and Python code see https://github.com/johnwoo Intro-to-Stat-Learning. For online book content see www.statlearning.com

# 2 Statistical Learning

## 2.1 Overview

Suppose we observe a quantitative response $Y$ with different predictors, $X_1, X_2, ..., X_p$. We assume there is some relationship between $Y = X_p$. A general form is,

$$Y = f(x) + \epsilon$$

where $f$ is some systematic information that $X_p$ provides about $Y$. $\epsilon$ is the random error term, which is **independent** of $X_p$ and mean zero.

Statistical learning refers to approaches that estimate $f$ where we can estimate to provide predictions or inference.

### 2.1.1 Predictions

Predictions assume a set of inputs $X_p$ are available but outputs $Y$ may not be. Predictions follow the form,

$$\hat{Y} = f(\hat{X})$$

The accuracy of $\hat{Y}$ by predicting $f(\hat{X})$ is not perfect, which introductions an error of two quantities:

**Reducible Error**: Error can be improved with appropriate modeling strategies.

**Irreducible Error**: variability of $\epsilon$ affects the accuracy of the prediction; thus, cannot reduce the error introduced by $\epsilon$. The error may contain (1) unmeasured variables, (2) unmeasured variation.

Derive reducible and irreducible errors by simply differencing $Y - \hat{Y}$, then find mean squared.

$$E(Y - \hat{Y})^2 = E[f(X) + \epsilon - \hat{f}(\hat{X})]^2$$

$$E(Y - \hat{Y})^2 = \underbrace{[f(X) - \hat{f}(X)]^2}_{\text{Reducible}} + \underbrace{Var(\epsilon)}_{\text{Irreducible}}$$

Thus, the irreducible error depends on the variation in the error. Statistical Learning focuses on improving (minimizing) the reducible error. Note that the irreducible error will always provide an upper bound on the accuracy of the prediction, which is almost always unknown in practice

### 2.1.2 Inference

Inference relates to understanding the relationship between $X$ and $Y$, or how $Y$ changes in response to $X$.

- Which predictors are associated with the response?

- What is the relationship between the response and each predictor?
- Can the relationship between Y and each predictor be adequately summarized using a linear equation, or is the relationship more complicated?

### 2.1.3 Estimating $f()$

Our goal is to apply statistical learning method to train data to estimate and unknown function $f$. Methods include parametric and nonparametric methods.

**Parametric**: Methods that use distributional assumptions are called parametric methods, because we estimate the parameters of the distribution assumed for the data. In OLS, assumptions about function form are linear with fixed parameters. No matter how much data you have, there will always be fixed parameters.

Examples:

- Logistic Regression
- Linear Discriminant Analysis
- Perceptron
- Naive Bayes
- Simple Neural Networks

Advantage: simplifies estimating $f()$ because it is easier to estimate a set of parameters, $\beta_0, \beta_1$.

Disadvantage: model does not usually match $f()$.

**Non-Parametric**: Methods do not make explicit assumptions about the functional form of $f()$. Goal is to get as close to the data points as possible without being to rough or wiggly.

Examples:

- k-Nearest Neighbors
- Decision Trees like CART and C4.5
- Support Vector Machines

Advantage: potential to accurately fit a wider range of possible shapes for $f()$.

Disadvantage: do not reduce the problem of estimating $f$ to a small number of parameters, thus a large number of observations is required to accurately estimate $f$.

### 2.1.4 Prediction Accuracy versus Model Interpretability

*Why would we ever choose to use a more restrictive method instead of a very flexible approach?*

Restrictive models, such as linear models, are more interpretable. In contrast, flexible approaches, such as splines, can provide complicated estimates of $f()$ that may improve prediction accuracy.

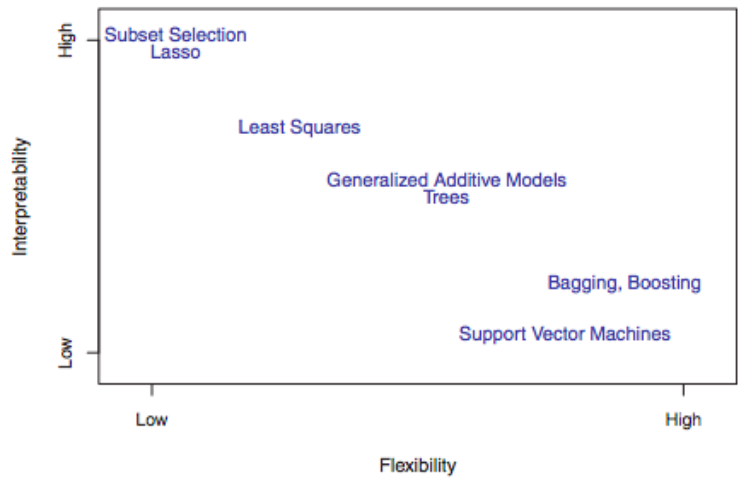


**FIGURE 2.7.** *A representation of the tradeoff between flexibility and interpretability, using different statistical learning methods. In general, as the flexibility of a method increases, its interpretability decreases.*

The choice of modeling strategy comes down to the end goal: prediction or inference. Less flexible models are easy to interpret, thus preferred when the goal is inference. If prediction is preferred, then more flexible models may be preferred – although, more flexible models are not always preferred for prediction accuracy due to overfitting.

### 2.1.5 Supervised Versus Unsupervised Learning

**Supervised**: Each observation of the predictor measurements $x_i$ there is an associated response measurement $y_i$. Goal is to fit a model that relates to the response predictors with an aim to accurately predict the response variable in the future.

Examples:

- Linear Regression
- Logistic Regression
- GAM
- Boosting
- Support Vector Machines

**Unsupervised**: Observations of the predictor $x_i$ does not contain a response variable, $y$.

Examples:

- Cluster Analysis
- PCA

### 2.1.6 Regression versus Classification Problems

Variables are characterized by quantitative or qualitative (categorical). Quantitative values are numerical whereas qualitative variables take values in classes or categories.

- Regression Analysis: Uses quantitative variables
- Classification Analysis: Uses qualitative variables

We select statistical learning methods based on the response variable being quantitative or qualitative.
**Note**: distribution of predictors being qualitative or quantitative is less important.

## 2.2 Assessing Model Accuracy

### 2.2.1 Measuring Quality of Fit

To assess performance of statistical learning methods, we need to quantify the extent to which the predicted response value is close to the true value. The most commonly-used measure is mean-squared-error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

Calculating training MSE (MSE on training data) doesn't tell us much about out-of-sample performance, which is preferred. Suppose $(x_0, y_0)$ are previously unseen test observations. The test MSE is

$$Ave(y_0 - \hat{f}(x_0))^2)$$

It is important to note that minimizing the training data provides no guarantee that the method will also minimize the test data.

There is a trade-off between inflexibility versus flexible models. Degrees of freedom define the flexibility of a curve. A more restricted (smoother) curve has fewer degrees of freedom than a wiggly curve. As flexibility increases training MSE declines monotonically.

In the figure below, as the flexibility of the statistical learning method increases, we observe a monotone decrease in the training MSE and a U-shape in the test MSE. As model flexibility increases, training MSE will decrease, but the test MSE may not.

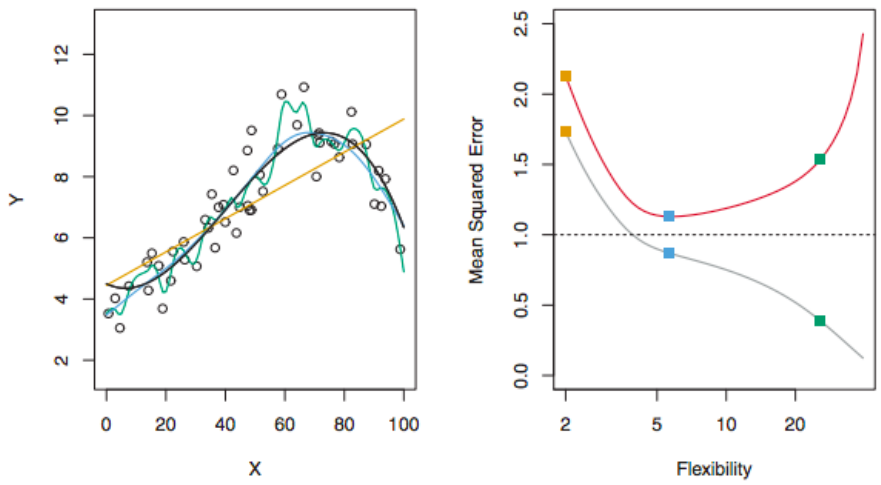


**FIGURE 2.9.** Left: *Data simulated from $f$, shown in black. Three estimates of $f$ are shown: the linear regression line (orange curve), and two smoothing spline fits (blue and green curves).* Right: *Training MSE (grey curve), test MSE (red curve), and minimum possible test MSE over all methods (dashed line). Squares represent the training and test MSEs for the three fits shown in the left-hand panel.*

**Note**: Overfitting occurs when a small training MSE but a large test MSE exists. This happens because the training model is finding patterns in the data and not establishing the signal.

### 2.2.2 The Bias-Variance Trade-Off

The U-shaped observed in the test MSE curve is a result of two competing properties in statistical learning:

The test MSE, for a given value $x_0$, can be decomposed into the sum of three fundamental quantities: variance of $\hat{f}(x_0)$, the squared bias of $\hat{f}(x_0)$, and the variance of the error terms $\epsilon$,

$$\underbrace{E(y_0 - \hat{f}(x_0))^2}_{\text{Expected Test MSE}} = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon)$$

To minimize the expected test error, we need to select a stat. method that achieves a low variance and a low bias.

- Variance: amount by which $\hat{f}$ changes if estimated using different raining data.
- Bias:error that is introduced by approximating a real-life problem

Variance between training data sets shouldn't change $\hat{f}$ too much; however, methods that are more flexible have higher variance that will shift the MSE larger whereas restricted methods have low variance and will only cause small shifts.

In terms of bias, the inverse is true. Restricted methods do not identify the true response variable, which results in large bias; however, flexible methods are usually better at predicting the true response variable which provides less bias.

### Bias-Variance Trade-Off

- Flexible Methods: Variance increases and bias will decrease MSE
- Restricted Methods: Variance decreases and bias will increase MSE

**Note**: The challenge lies in finding a method for which both the variance and the squared bias are low.

### 2.2.3 Classification Strategy

Model accuracy transfers over to classification problems. The most common approach is to quantify the accuracy of $\hat{f}$ using a training error rate, or the proportion of mistakes that are made to the training observations,

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

where $I(y_i \neq \hat{y}_i)$ is an indicator variable that equals 1 if $y_i \neq \hat{y}_i$, and zero if $y_i = \hat{y}_i$. If $I(y_i \neq \hat{y}_i) = 0$, then the observation was classified correctly. The test error is calculated as,

$$Ave(I(Y - 0 \neq \hat{y}_0))$$

**The Bayes Classifier** The error rate can be classified by assigning each observation to the most likely class, given its predictor values. The Bayes Classifier is,

$$Pr(Y = j|X = x_0)$$

or the probability that $Y = j$ given the observed predictor vector $x_0$. The Bayes Classifier establishes a Bayes decision boundary that falls on one side or the other of the classification.

Bayes error rate maximizes the probability of selecting,

$$1 - E(\max_j PR(Y = j|X))$$

and is analogous to the irreducible error.

**K-Nearest Neighbors** In theory, Bayes Classifier is the gold standard, but we don't always know the conditional distribution of $Y$ given $X$;thus, we need to estimate the probability – K-Nearest Neighbor (KNN) is one method.

Given a positive integer $K$, and a test observation $x_0$, the KNN classifier identifies the $K$ points in the training data that are closest to $x_0$, represented by $N_0$. Condition probabilities are estimated for class $j$ as a fraction of $N_0$ whose response values equal $j$:

$$Pr(Y = j|X = x_0) = \frac{1}{K}\sum_{i \in N_0} I(y_i = J)$$

The choice of $K$ has a drastic effect on the classifier obtained; small $K = 1$ provides a boundary that is overfly flexible and has a low bias but high variance. As $K$ increases, method becomes less flexible and is closer to linear (high bias low variance). No strong relationship between test and train error rates. Flexible $K = 1$ have a low training rate (0), but test error will be high.

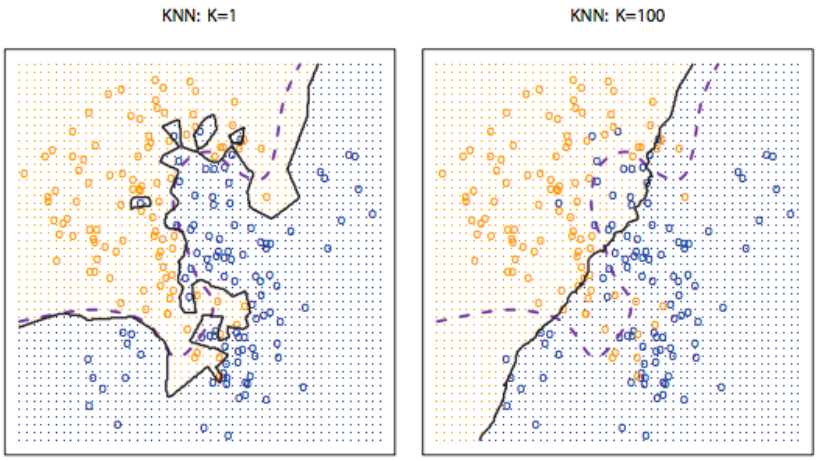


**FIGURE 2.16.** *A comparison of the KNN decision boundaries (solid black curves) obtained using $K = 1$ and $K = 100$ on the data from Figure 2.13. With $K = 1$, the decision boundary is overly flexible, while with $K = 100$ it is not sufficiently flexible. The Bayes decision boundary is shown as a purple dashed line.*

**Note:** In both the regression and classification settings, choosing the correct level of flexibility is critical to the success of any statistical learning method.

# 3 Linear Regression

## 3.1 Simple Linear Regression

Approach to predicting a quantitative response $Y$ on the basis of a single predictor variable $X$, assuming an approximate linear relationship.

$$Y \approx \beta_0 + \beta_1 X$$

where $\beta_0, \beta_1$ are unknown constants that represent an intercept and slope, known as coefficients or parameters.

### 3.1.1 Estimating the Coefficients

Goal is to minimize the relationship between a linear line and the actual value, also known as residuals. Most approaches involve minimizing the least squares criterion.

Residual from linear regression is,

$$e_i = y_i - \hat{y}_i$$

where residual sum of squares (RSS) is,

$$RSS = e_1^2 + e_2^2 + ... + e_n^2$$

and the minimization problem reduces to,

$$\beta_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{2}(x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \hat{\beta}\bar{x}$$

### 3.1.2 Assessing the Accuracy of the Coefficient Estimates

A sample mean is unbiased in the sense that on average the estimated sample equals population mean. By selecting multiply samples, calculating mean, and estimate mean of sample means, the mean should be close to the population mean, which produces unbiased mean. The regression mean provides a reasonable estimate of this sampling procedure.

To calculate how over-or-under the average estimate of the population mean is, we use the standard error,

$$Var(\hat{\mu}) = SE(\hat{\mu})^2 = \frac{\sigma}{n}$$

A regression line provides a reasonable estimate of the sample mean assuming the sample mean is randomly drawn multiple times and averaged.

**Standard Error:** tells us the average amount that is estimate $\hat{\mu}$ differs from the actual value of $\mu$

$$SE(\hat{\beta}_1)^2 = \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

where $\sigma^2 = Var(\epsilon)$ and we assume the errors are uncorrelated with common variance $\sigma^2$. Generally, we don't know $\sigma$, so we estimate from the residual standard error,

$$RSE = \sqrt{RSS/(n-2)}$$

Standard errors can then be used to calculate confidence intervals at a 95% confidence interval as,

$$[\hat{\beta}_1 - 2 \cdot SE(\hat{\beta}_1), \hat{\beta}_2 + 2 \cdot SE(\hat{\beta}_1)]$$

SE can also be used to perform hypothesis tests on coefficients (stat. sign),

$$H_0 : \beta_1 (\text{There is no relationship between X and Y.})$$

$$H_a : \beta_1 \neq (\text{There is some relationship between X and Y.})$$

A t-stat measures the number of standard deviations that $\hat{\beta}_1$ is away from 0. Generally, a t-stat above 2 implied statistical significance.

$$t = \frac{\hat{\beta}_1 - 0}{SE(\hat{\beta}_1)}$$

**p-value**: a small p-value indicates that it is unlikely to observe such a substantial association between the predictor and the response due to chance, in the absence of any real associations.

In other words, a small p-value infers that there is an association between the predictor and the response, in which case we reject the null hypothesis.

### 3.1.3 Assessing the Accuracy of the Model

Model accuracy is typically assessed with residual standard error (RSE) and the $R^2$

**Residual Standard Error** RSE is an estimate of the standard deviation of $\epsilon$, or the average amount that the response will deviate from the true regression line. RSE is thought of as a measure of lack of fit – low RSE indicates model fits data well, high RSE indicates poor fit.

$$RSE = \sqrt{\frac{1}{n-2}RSS}$$

**$R^2$ Statistic** $R^2$ takes the form of a proportion – the portion of the variance explained – and will be between 0 and 1

$$R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}$$

where $TSS = \sum(y_i - \bar{y})^2$ is the total sum of squares. TSS measures the total variance in the response and is thought of as the amount of variability inherent in the response before the regression is performed. RSS measures the amount of variability that is left unexplained after performing the regression.

$R^2 = 0$ regression did not explain much of the variability in the response

$R^2 = 1$ indicates a large proportion of the variability in the response has been explained in the regression.

TSS uses mean of $y_i$ whereas RSS uses residual differences.

An R-squared of 0.65 might mean that the model explains about 65% of the variation in our dependent variable.

**Problems with R-squared** (https://data.library.virginia.edu/is-r-squared-useless/)

- R-squared does not measure goodness of fit. It can be arbitrarily low when the model is completely correct. By making $\sigma^2$ large, we drive R-squared towards 0, even when every assumption of the simple linear regression model is correct in every particular.
- R-squared can be arbitrarily close to 1 when the model is totally wrong.
- R-squared says nothing about prediction error, even with $\sigma^2$ exactly the same, and no change in the coefficients. R-squared can be anywhere between 0 and 1 just by changing the range of X. We're better off using Mean Square Error (MSE) as a measure of prediction error.
- R-squared cannot be compared between a model with untransformed Y and one with transformed Y, or between different transformations of Y. R-squared can easily go down when the model assumptions are better fulfilled.

## 3.2 Multiple Linear Regression

Estimating separate simple linear regression models for each predictor is not entirely satisfactory: (1) unclear how single predictions affect or variables (2) individual regressions ignore the other regressors. A better solution is to provide individual slopes for each of the regressors,

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \epsilon$$

$\beta_j$ quantifies the association between that variable and the effect on a one unit increase in $X_j$, holding all other predictors fixed.

### 3.2.1 Estimating the Regression Coefficients

The parametric estimates are obtained using the same least squares minimization problem as before, although slightly more complicated. The differences are that covariates are adjusted based on correlations and will obtain different results than individual least squares estimates[1].

**1. Is there a relationship between the response and predictors?** In multiple variable linear regressions, we need to consider whether there is a relationship between all variables. Thus, the null hypothesis is,

$$H_0 = \beta_1 = \beta_2 = ... = \beta_p = 0$$

An F-statistic establishes the hypothesis test,

$$F = \frac{(TSS - RSS)/p}{RSS/(n-p-1)}$$

where a value close to 1 establishes no relationship between the response and predictors where as an F-stat greater than 1 establishes the covariates represent a relationship to the response variable $Y$.

We can also test a subset of covariates to determine whether a relationship exists.

**If we use the individual t-statistics and associated p-values in order to decide whether or not there is any association between the variables and the response, there is a very high chance that we will incorrectly conclude that there is a relationship. However, the F-statistic does not suffer from this problem because it adjusts for the number of predictors.**

**2. Deciding on Important Variables** It is possible that all of the predictors are associated with the response, but it is more often the case that the response is only related to a subset of the predictors. This association is referred to as variable selection.

To determine the best variable selections, we can utilize BIC, AIC, $R^2$, or even RMSE. However, the size of model selection grows exponentially, so costs increase substantially.

Three approaches exist to validate model selection:

- Forward selection: start with intercept with no predictors and add variables that minimize the RSS.
- Backward selection: start with all variables and remove variables that are the least stat sign.
- Mixed selection: combination of forward and backward selection. Start with no variables, add variables that provides the best fit, but only add variables below a certain threshold.

**3. Model Fit** Most common numerical measures of model fit are RSE and $R^2$. It is important to note that $R^2$ will always increase with additional variables, so care needs to be taken when utilize $R^2$ as a model fit discussion. Additional RSE can increase when variables are added.

[1] A regression of shark attacks and ice cream sales reveals a significant result that shark attacks increase with ice cream sale due to increases in temperatures; however, when temperatures are included in the analysis to adjust for correlations between increases in temperature and ice cream sales, the estimate becomes insignificant.

**4. Predictions** Once the model has been fit, predictions are relatively straightforward. However, uncertainty exists,

- Are the coefficient estimates of the true population? Inaccuracies related to the reducible error.
- Does the linear model provide accurate approximations? Model bias may bias results.
- Even if we know the true values, we cannot perfectly predict the response because of the random error. Therefore, irreducible errors always exist in linear approximations.

**Note:** Confidence intervals are used to quantify uncertainty around model estimates.

## 3.3 Other Considerations in the Regression Model

### 3.3.1 Qualitative Predictors

Predictors can be qualitative.

**Predictors with two levels** Create a dummy variable, $D$, for two possible numerical values, such as 0 or 1. The level that is associated with 1 can be interpreted as,

$$D(1) = \beta_0 + \beta_1 + \epsilon$$

$$D(0) = \beta_0 + \epsilon$$

It is also possible to code with 1 and -1. In this case, the interpretation of the coefficient's changes.

$$D(1) = \beta_0 + \beta_1 + \epsilon$$

$$D(-1) = \beta_0 - \beta_1 + \epsilon$$

**Predictors with more than two levels** With more than two levels, dummy variables need to be spread out for each factor (level). When including more than two factors, there will always be one fewer dummy variables. The level with no dummy variable is known as the baseline and includes the constant and errors.
**Note:** The baseline establishes the number for which coefficients are differenced or added. For example, a baseline (intercept) reports 500. $\beta_1$ reports a coefficient of -5. Therefore, the dummy variable representing $\beta_1$ has a value of 495.

### 3.3.2 Extensions of the Linear Model

Two of the most important assumptions in linear regressions is,

- **Additive**: the effect of changes in predictor $X$ on the response $Y$ is independent of the values of other predictors.
- **Linear**: change in response $Y$ due to a one-unit change in $X$ is constant, regardless of the value of $X$

**Removing the Additive Assumption** Additive assumption assumes no relationship between predictors, which may not always hold (temp and precipitation). In statistics, this is known as an interaction effect. We can relax the additive assumption by including an **interaction term**.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + \epsilon$$

Example,

$$\begin{aligned} \texttt{sales} &= \beta_0 + \beta_1 \times \texttt{TV} + \beta_2 \times \texttt{radio} + \beta_3 \times (\texttt{radio} \times \texttt{TV}) + \epsilon \\ &= \beta_0 + (\beta_1 + \beta_3 \times \texttt{radio}) \times \texttt{TV} + \beta_2 \times \texttt{radio} + \epsilon. \end{aligned} \tag{3.33}$$

We can interpret $\beta_3$ as the increase in the effectiveness of TV advertising for a one unit increase in radio advertising (or vice-versa). The coefficients that result from fitting the model (3.33) are given in Table 3.9.

textbfNote: The **hierarchical principle** states that if we include an interaction in a model, we should also include the main effects, even if the p-values associate with their coefficients are not sign.

Quantitative and qualitative variables can be interacted to remove the additive assumption.

**Non-linear relationship** The relationship between the response and predictor may be non-linear. We can accommodate this relationship by using a polynomial regression.

A simple way to fit a polynomial is to use quadratic functional form of variables.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \epsilon$$

Note that this is still a linear model!!!

### 3.3.3 Potential Problems

Most common problems when fitting a linear regression,

- Non-linearity of the response-predictor relationships.
- Correlation of error terms.
- Non-constant variance of error terms.
- Outliers.
- High-leverage points.
- Collinearity.

**1. Non-linearity of the Data** If a true linear relationship exists between response and predictors, then we can utilize the linear interpretation discussed. However, nonlinearities can throw off modeling aspects and interpretations.

Residual plots are useful for identify non-linearities.

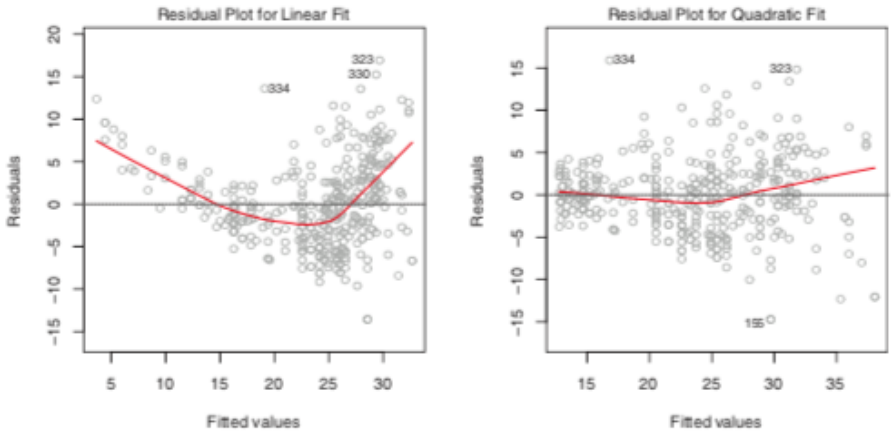


**FIGURE 3.9.** *Plots of residuals versus predicted (or fitted) values for the* `Auto` *data set. In each plot, the red line is a smooth fit to the residuals, intended to make it easier to identify a trend.* Left: *A linear regression of* `mpg` *on* `horsepower`*. A strong pattern in the residuals indicates non-linearity in the data.* Right: *A linear regression of* `mpg` *on* `horsepower` *and* $\texttt{horsepower}^2$*. There is little pattern in the residuals.*

Simple approaches to transform variables include log $X$, $sqrtX$, and $X^2$.

**2. Correlation of Error Terms** An important assumption is that the error terms are uncorrelated. Moreover, standard errors are calculated assuming uncorrelated error terms, thus may underestimate the true standard errors.

Correlation of errors terms may exist in time series data (serial correlation). One way to check for correlation in error terms is to plot residuals versus time series.

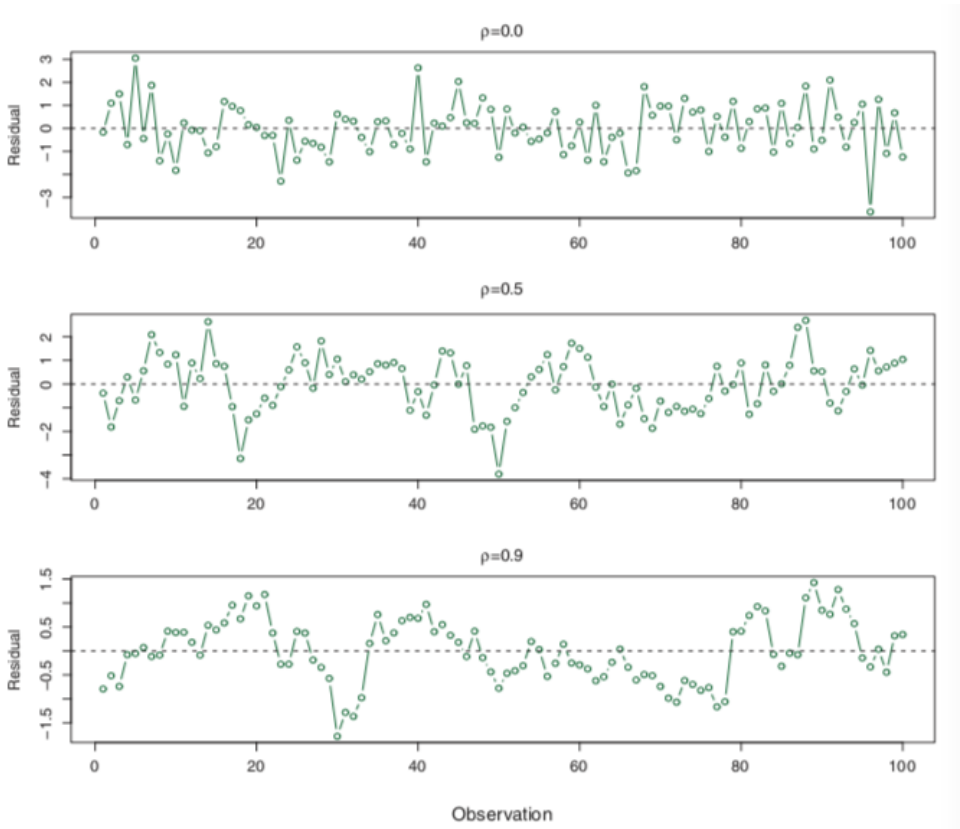


**FIGURE 3.10.** *Plots of residuals from simulated time series data sets generated with differing levels of correlation $\rho$ between error terms for adjacent time points.*

Correlation of error terms can exist outside of time series if groups (states or family members) are included in the variables.

**Non-constant Variance of Error Terms** Another important assumption is that the error terms have a constant variance, $Var(\epsilon) = \sigma^2$. Non-constant error terms exist with heteroscedastic data.

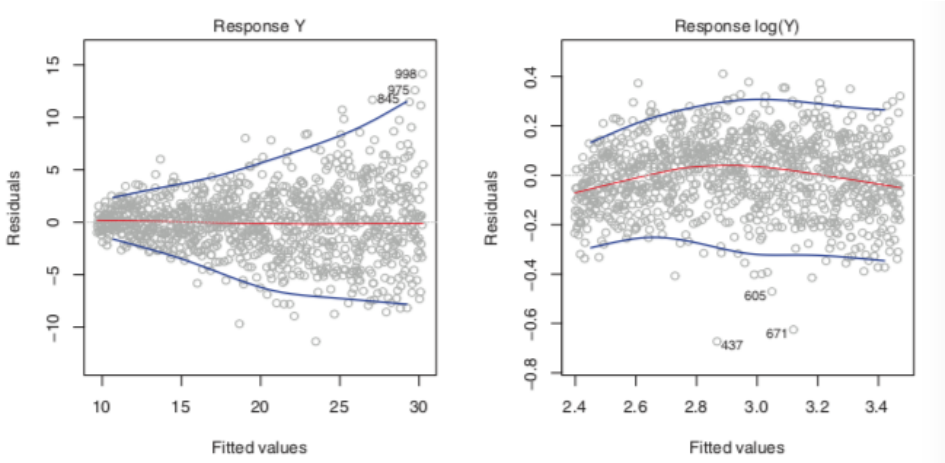


**FIGURE 3.11.** *Residual plots. In each plot, the red line is a smooth fit to the residuals, intended to make it easier to identify a trend. The blue lines track the outer quantiles of the residuals, and emphasize patterns.* Left: *The funnel shape indicates heteroscedasticity.* Right: *The response has been log transformed, and there is now no evidence of heteroscedasticity.*

Ways to deal with heteroskedasticity is to log the response variable. Another option is to fit a weighted least squares.

**Outliers** Outlier is a point far beyond the value predicted by the model. An outlier may or may not affect a predictors slope and may also affect the RSE, which can affect confidence intervals and p-values, and can also affect the $R^2$.

Residual plots can be used to identify outliers or standardized residual (divide residuals by standard error.) plots,

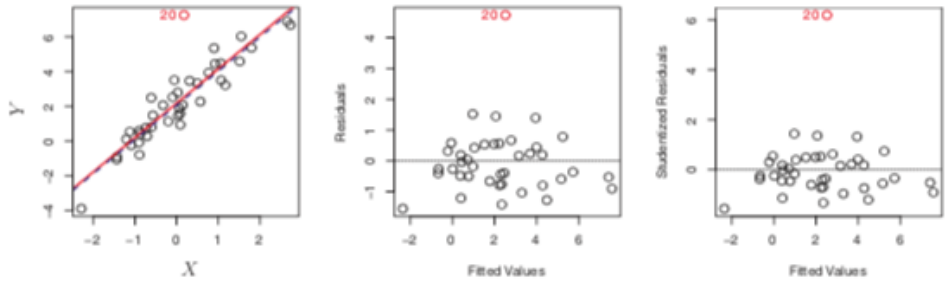


**FIGURE 3.12.** Left: *The least squares regression line is shown in red, and the regression line after removing the outlier is shown in blue.* Center: *The residual plot clearly identifies the outlier.* Right: *The outlier has a studentized residual of 6; typically we expect values between −3 and 3.*

**5. High Leverage Points** High leverage points have an unusual value for $x_i$. These observations can heavily affect the least squares line.

These can be identified similar to outliers or through a leverage statistic, calculated as,

$$h_i = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{i'=1}^{n}(x_{i'} - \bar{x})^2}$$

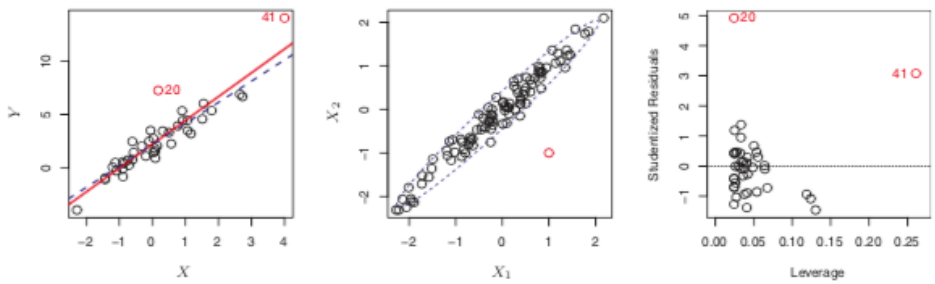


**FIGURE 3.13.** Left: *Observation 41 is a high leverage point, while 20 is not. The red line is the fit to all the data, and the blue line is the fit with observation 41 removed.* Center: *The red observation is not unusual in terms of its $X_1$ value or its $X_2$ value, but still falls outside the bulk of the data, and hence has high leverage.* Right: *Observation* 41 *has a high leverage and a high residual.*

**6. Collinearity** Collinearity occurs when two or more predictor variables are closely related to one another. High correlation relates to variables being collinear.

Collinearity introduces problems because the effects cannot be parsed out which can produce uncertainty around the coefficient estimates. Other problems exist, such as reduction in accuracy of coefficients causes standard errors to grow (due to calculation of t-stat and coefficient).

Ways to deal to collinearity include looking at correlation matrix of the predictors. A better way to assess multicollinearity (correlation of more than two variables) is to use variance inflation factor (VIF). The smallest possible value for VIF is 1, which indicates complete absence of collinearity. A VIF exceeds 5 or 10 indicates a problem.

VIF is the ratio of the variance of $\hat{\beta}_j$ when fitting the full model divided by the variance of $\hat{\beta}_j$ on its own.

$$VIF(\hat{\beta}_j) = \frac{1}{1 - R^2_{X_j|X_{-j}}}$$

To deal with collinearity, two solutions exist: (1) drop the problematic variables; (2) combine the collinear variables into a single predictor.

## 3.4 Comparison of Linear Regression with K-Nearest Neighbors

K-nearest neighbors regression (KNN regression) is one of the most well-known non-parametric regressions. KNN regressions first identify the K training observations that are closest to $x_0$. Then estimates $f(x_0)$ using the average of all the training responses in $N_0$,

$$\hat{f}(x_0) = \frac{1}{K} \sum_{x_i \in N_0} y_i$$

Small $K$ results in step-function that is most flexible of data while larger values smooth the plane and less flexible. The optimal value of $K$ depends on the bias-variance tradeoff.

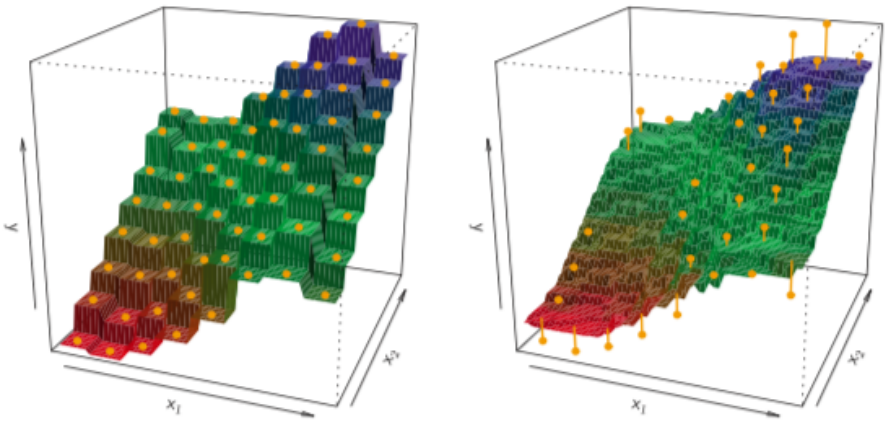

**FIGURE 3.16.** *Plots of $\hat{f}(X)$ using KNN regression on a two-dimensional data set with 64 observations (orange dots).* Left: $K = 1$ *results in a rough step function fit.* Right: $K = 9$ *produces a much smoother fit.*

**Note:** the parametric approach will outperform the non-parametric approach if the parametric form that has been selected is close to the true form of $f$.

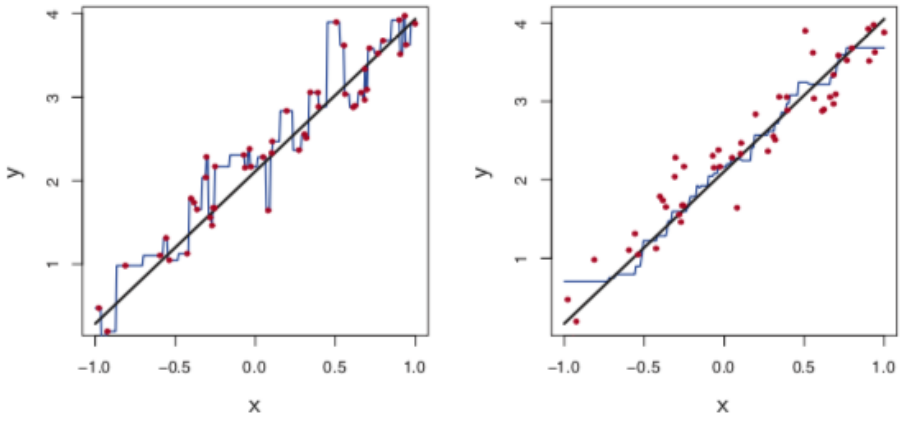


**FIGURE 3.17.** *Plots of $\hat{f}(X)$ using KNN regression on a one-dimensional data set with 100 observations. The true relationship is given by the black solid line.* Left: *The blue curve corresponds to $K = 1$ and interpolates (i.e. passes directly through) the training data.* Right: *The blue curve corresponds to $K = 9$, and represents a smoother fit.*

**Note:** Generally, KNN regressions will outperform linear regressions with low number of variables. As the number of variables increase, KNN predictive power degrades (problem of dimensionality). As a general rule, parametric methods will tend to outperform non-parametric approaches when there is a small number of observations per predictor.

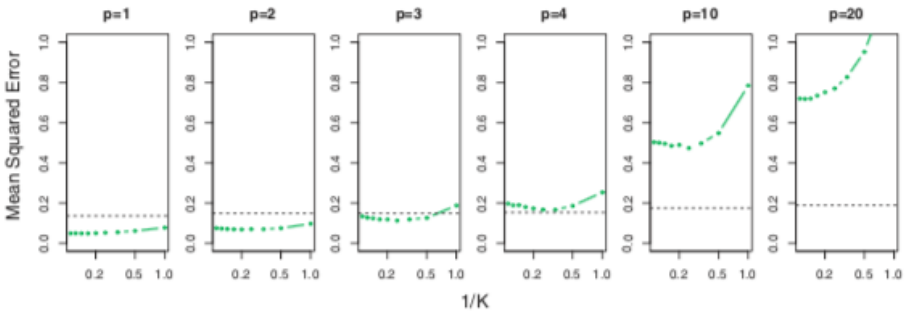


**FIGURE 3.20.** *Test MSE for linear regression (black dashed lines) and KNN (green curves) as the number of variables p increases. The true function is non–linear in the first variable, as in the lower panel in Figure 3.19, and does not depend on the additional variables. The performance of linear regression deteriorates slowly in the presence of these additional noise variables, whereas KNN's performance degrades much more quickly as p increases.*

## 3.5 Classification

Classification problems involve dealing with categorical (qualitative) variables. These problems, generally, predict the probability of each category, so they behave like a regression problem.

Three most common classification problems: **logistic regression, linear discriminant analysis, and KNN**.

**Why Not Linear Regression?**: No natural way to convert a qualitative response variable with more than two levels into a quantitative response that is ready for linear regression, e.g. can't convert 1, 2, 3 to 1 to 3.

### 3.5.1 Logistic Regression

Logistic regression models the probability that $Y$ belongs to a particular category, $p(X) = Pr(Y = 1|X)$. The general form is from a linear regression is,

$$p(X) = \beta_0 + \beta_1 X$$

However, the linear regression form includes a balance between negative values and positive values, which does not apply to probabilities.

The logistic function form is,

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1}}{1 + e^{\beta_0+\beta_1 X_1}}$$

To fit the model between zero and one, we use a maximum likelihood method (see next section). Solving for right-hand side, $\beta_0 + \beta_1 X$ equals,

$$\underbrace{log(\frac{p(X)}{1-p(X)})}_{\text{Log-odds or logit}} = \beta_0 + \beta_1 X$$

Thus, the logistic regression model has a logit that is linear in $X$. However, in a logistic regression, a one-unit increase in $X$ changes the log odds by $\beta_1$.

### 3.5.2 Estimating the Regression Coefficients

General intuition behind maximum likelihood is to estimate $\beta_0$ and $\beta_1$ such that the predicted probability $\tilde{p}(x_i)$ of default for each individual from the logistic regression, corresponds as closely as possible to the individuals observed default status. In other words, we find coefficients that yields a number close to one for all individuals who defaulted and number close to zero for all individuals who did not. Formally, the likelihood function is,

$$l(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'}))$$

$\beta_0, \beta_1$ are chosen to maximize the likelihood function.

Many aspects of logistic regression are similar to linear regression: measure accuracy of coefficients with standard errors, t-stats, null hypothesis testing. The intercept is generally not of interest and is used to fit probabilities to the proportion of ones in the data.

### 3.5.3 Making Prediction

Predictions are made from the simple logistic model,

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 X}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 X}}$$

### 3.5.4 Multiple Logistic Regression

Using multiple variables follows a similar approach to simple logistic regression,

$$p(X) = \frac{e^{\beta_0 + \beta_1 X_1 + ... + \beta_p X_p}}{1 + e^{\beta_0 + \beta_1 X_1 + ... + \beta_p X_p}}$$

As in the linear regression setting, the results obtained using one predictor may be quite different from those obtained using multiple predictors, especially when there is correlation among the predictors. In general, the phenomenon is known as confounding.

**Logistic Regression for $>$ 2 Response Classes** Multiple-class logistic regressions are available, but discriminant analysis is popular for multiple-class classification.

### 3.5.5 Linear Discriminant Analysis

Linear Discriminant Analysis involves modeling the distribution of the predictors $X$ separately in each of the response classes, and then use Bayes theorem to flip those around into estimate for $Pr(Y = k|X = x)$.

Why choose LDA?

- When classes are well separated in logistic regressions, the results are unstable. LDA does not suffer from this.
- If $X$ predictors are approx normal and $n$ is small, LDA are more stable
- Popular with two response class

### 3.5.6 Using Bayes' Theorem for Classification

The Bayes' Theorem states,

$$Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

where $\pi_k$ represents the overall, or prior, probability that a given observation is associated with the kth category of the response variable $Y$. $f_k(x)$ denotes the density function of $X$ for an observation that comes from the kth class.

Generally, estimating $\pi_k$ is easy if we have a random sample of $Y$s from the population (compute fraction of the training observations that belong to the kth class). However, estimating $f_k(x)$ is more challenging unless a form of density is assumed. The Bayes' classifier provides the lowest error rate, so if we can estimate $f_k(x)$ then we can get a way to classify Bayes.

### 3.5.7 Linear Discriminant Analysis for p=1

With only one predictor, assuming a normal or Gaussian, it is simply to estimate the normal density.

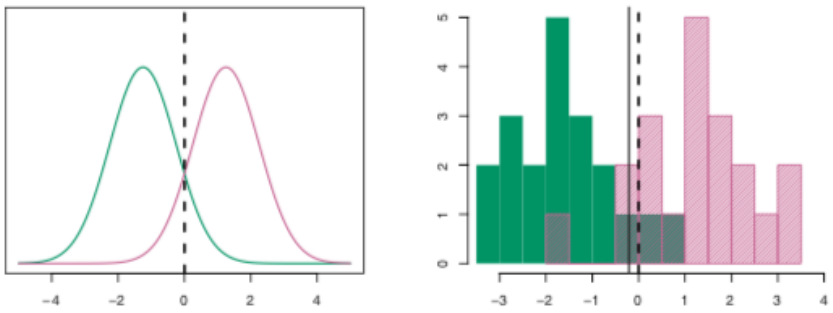


**FIGURE 4.4.** Left: *Two one-dimensional normal density functions are shown. The dashed vertical line represents the Bayes decision boundary.* Right: *20 observations were drawn from each of the two classes, and are shown as histograms. The Bayes decision boundary is again shown as a dashed vertical line. The solid vertical line represents the LDA decision boundary estimated from the training data.*

In practice, even if we are quite certain of our assumption that X is drawn from a Gaussian distribution within each class, we still have to estimate the parameters $\mu_1, ..., \mu_K, \pi_1, ..., \pi_K, and \sigma^2$. The linear discriminant analysis (LDA) method approximates the Bayes classifier by plugging estimates for $\pi_k, \mu_k, and \sigma^2$ as follows,

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2$$

where $n$ is total number of training observations, $n_k$ is the number of training obs in the kth class. The estimate for $\mu_k$ is simply the average of all the training observations from the kth class, while

$\sigma\hat{}2$ can be seen as a weighted average of the sample variances for each of the $K$ classes. We can estimate $\hat{\pi}_k$ as,

$$\hat{\pi}_k = n_k/n$$

The LDA classifier is,

$$\hat{\delta}_k(x) = x \cdot \frac{\hat{\mu}_k}{\hat{\sigma}^2} - \frac{\hat{\mu}_k^2}{2\hat{\sigma}^2} + log(\hat{\pi}_k)$$

**Note:** the word linear in LDA comes from the fact the discriminant function $\hat{\delta}_k(x)$ are linear functions of $x$.

To reiterate, the LDA classifier results from assuming that the observations within each class come from a normal distribution with a class-specific mean vector and a common variance $\sigma^2$, and plugging estimates for these parameters into the Bayes classifier.

### 3.5.8 Linear Discriminant Analysis for p > 1

In the case of $p > 1$ predictors, the LDA classifier assumes that the observations in the kth class are drawn from a multivariate Gaussian distribution $N(\mu_k, \sum)$, where $\mu_k$ is a class-specific mean vector, and $\sum$ is a covariance matrix that is common to all $K$ classes. The multivariate density function is plugged into LDA.

**Problem:** binary classifiers, such as LDA, can make two types of errors: (1) can incorrectly assign an individual who defaults to the no default category, or (2) it can incorrectly assign an individual who does not default to the default category. The solution to this is a confusion matrix,

| | | *True default status* | | |
|---|---|---|---|---|
| | | No | Yes | Total |
| *Predicted default status* | No | 9,644 | 252 | 9,896 |
| | Yes | 23 | 81 | 104 |
| | Total | 9,667 | 333 | 10,000 |

**TABLE 4.4.** *A confusion matrix compares the LDA predictions to the true default statuses for the 10,000 training observations in the* `Default` *data set. Elements on the diagonal of the matrix represent individuals whose default statuses were correctly predicted, while off-diagonal elements represent individuals that were misclassified. LDA made incorrect predictions for 23 individuals who did not default and for 252 individuals who did default.*

which describes the number predicted correctly versus not to compare strength of the model.

While the Bayes' Classifier will provide lowest error rate, it doesn't always do a good job predicting because of the threshold for the posterior probability, default 50%. If concerned about incorrect predictions, it's best to lower the threshold, but lowering too much will cause increases in prediction error. Deciding on the threshold is dependent on domain knowledge.

The ROC curve (receiver operating characteristics) is used to simultaneously display the two types of errors for all thresholds. The performance is based on the area under the curve (AUC) of the ROC. ROC curves are useful for comparing different classifiers.

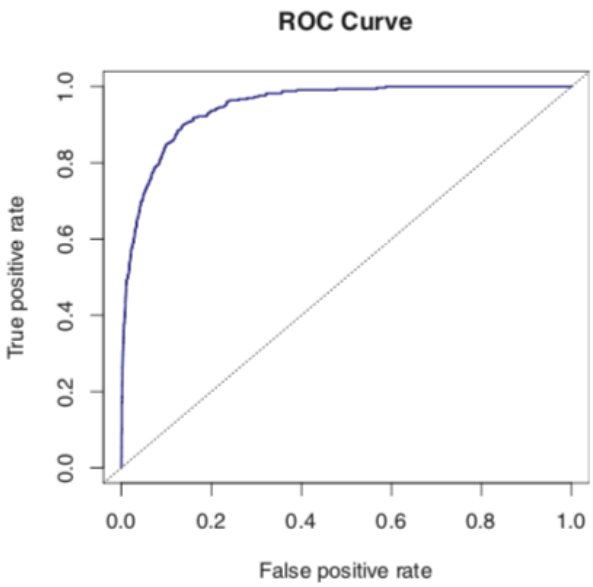


**FIGURE 4.8.** *A ROC curve for the LDA classifier on the* `Default` *data. It traces out two types of error as we vary the threshold value for the posterior probability of default. The actual thresholds are not shown. The true positive rate is the sensitivity: the fraction of defaulters that are correctly identified, using a given threshold value. The false positive rate is 1-specificity: the fraction of non-defaulters that we classify incorrectly as defaulters, using that same threshold value. The ideal ROC curve hugs the top left corner, indicating a high true positive rate and a low false positive rate. The dotted line represents the "no information" classifier; this is what we would expect if student status and credit card balance are not associated with probability of default.*

### 3.5.9 Quadratic Discriminant Analysis (QDA)

Like LDA, the QDA classifier results from assuming that the observations from each class are drawn from a Gaussian distribution, and plugging estimates for the parameters into Bayes' theorem in order to per- form prediction. However, unlike LDA, QDA assumes that each class has its own covariance matrix. Further, QDA assumes $x$ is quadratic as opposed to linear.

Generally, LDA is more flexible with a lower variance, so model performance is improved over QDA. However, with larger training sets, QDA may perform better, so the variance is not of concern.

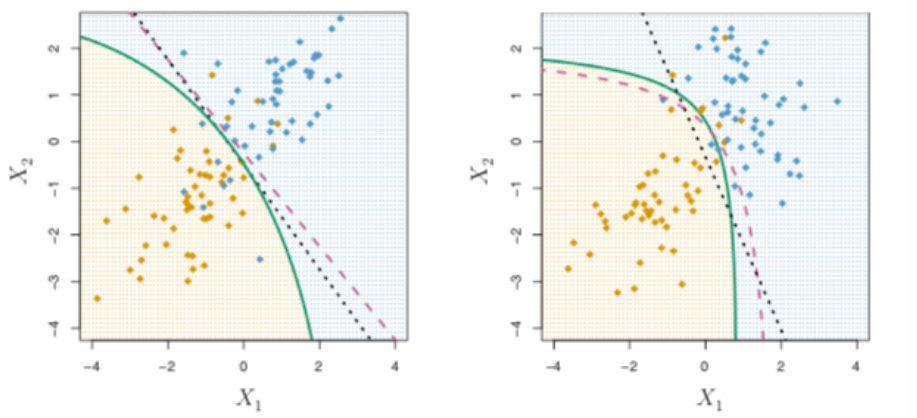


**FIGURE 4.9.** Left: *The Bayes (purple dashed), LDA (black dotted), and QDA (green solid) decision boundaries for a two-class problem with* $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2$. *The shading indicates the QDA decision rule. Since the Bayes decision boundary is linear, it is more accurately approximated by LDA than by QDA.* Right: *Details are as given in the left-hand panel, except that* $\boldsymbol{\Sigma}_1 \neq \boldsymbol{\Sigma}_2$. *Since the Bayes decision boundary is non-linear, it is more accurately approximated by QDA than by LDA.*

### 3.6 Comparison of Classification Methods

LDA and logistic regressions are closely connected and differ only in their fitting procedure: logistic regression is estimating using maximum likelihood whereas LDA is estimated using mean and variance from a normal distribution.

LDA assumes normal distributions and common variances, so is an improvement over logistic; however, without those assumptions, logistic regressions will do better.

When the decision barrier is highly non-linear, KNN will perform better because of its non-parametric approach; however, KNN doesn't provide a table of coefficients to compare significance of variables.

QDA serves as a compromise between KNN, LDA, and logistic regressions.

In summary, when the true decision boundaries are linear, then the LDA and logistic regression approaches will tend to perform well. When the boundaries are moderately non-linear, QDA may give better results. Finally, for much more complicated decision boundaries, a non-parametric approach such as KNN can be superior.

## 4 Resampling Methods

Resampling methods involve drawing samples from training set and refitting a model to obtain additional information.

**Model assessment:** The process of evaluating a models performance.

**Model selection:** selecting the proper level of flexibility.

### 4.1 Cross-Validation

Given a data set, the use of a particular statistical learning method is warranted if it results in a low test error. The test error can be easily calculated if a designated test set is available. In contrast, the training error can be easily calculated by applying the statistical learning method to the observations used in its training.

### 4.1.1 The Validation Set Approach

**Validation set approach:** involves randomly dividing the available set of observations into two part, training set and validation set or hold-out set. The MSE from test is used to validate the performance of the model.

Two drawbacks:

- validation estimate of the test error rate can be highly variable depending on which observations are in the training set and validation set.
- Reduced observation by splitting data suggests the validation set error rate may overestimate the test error rate for the model fit on the entire data set.

Cross-validation can address these two issues.

### 4.1.2 Leave-One-Out Cross-Validation

Leave-One-Out Cross-Validation (LOOCV) addresses the drawbacks by leaving out one observation as the validation set and the remaining are used to train the model. The procedure is repeated $n$ times and average of the test error is calculated as,

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

Advantages:

- Less bias than the validation set approach
- no randomness is selection so LOOCV always yield the same results.

Disadvantage: can be expensive to implement.

### 4.1.3 k-Fold Cross-Validation

K-fold CV randomly divides the set of observations into $k$ groups, or folds, of equal size. Each fold is treated as a validation set and the observations not in the fold is the training set. The procedure is repeated $k$ times and average MSE is computed,

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

k-fold CV is equivalent to LOOCV when $k = n$.

Advantage of k-fold CV: computationally less expensive than LOOCV.

### 4.1.4 Bias-Variance Trade-Off for k-Fold CV

An important advantage of k-fold CV is that it often gives more accurate estimates of the test error rate than does LOOCV. This has to do with a bias-variance trade-off. From the perspective of bias reduction, it is clear that LOOCV is to be preferred to k-fold CV. However, LOOCV has a higher variance then k-fold CV because LOOCV results are highly correlated with one another because the procedure uses most of the same data set. k-Fold will be less correlated since the overlap is smaller.

Typically, $k = 5$ or $k = 10$ is preferred because it has been shown empirically to yield test error rate estimates that suffer neither from high bias or high variance.

#### 4.1.5 Cross-Validation on Classification Problems

Works similar to regressions methods but instead of calculating MSE you calculate the error, such as accuracy or kappa.

### 4.2 The Bootstrap

Statistical tool for quantifying uncertainty associated with a given estimator or statistical learning method. (e.g. calculating standard errors). The procedure involves repeatedly sampling from the original data set (with replacement) and estimate to get the parameter of interest. From the number of repetitions, the uncertainty can be computed by taking the standard deviations of the mean (standard error). This can then be used to calculate confidence intervals.

## 5 Linear Model Selection and Regularization

Before moving on to nonlinear models, we deal with replacing plain least squares fitting with alternative fitting problems.

Reasons to use another fitting procedure:

- **Prediction Accuracy:** Relationship between response and prediction is based on assumptions in model. By constraining or shrinking the estimated coefficients, we can reduce the variance at the cost of a negligible increase in bias, which can improve prediction accuracy.
- **Model Interpretability:** Variables that are irrelevant lead to unnecessary complexity in the model, so it is best to remove them to improve interpretability.

Three methods for feature/variable selection:

- **Subset Selection:** identify a subset of predictors that are related to the response.
- **Shrinkage:** coefficients are shrunken towards zero relative to the least squares estimates, thus reducing the variance. Depending on shrinkage, some coefficients may be zero, so can be used for feature selection.
- **Dimension Reduction:** projecting predictors through linear combinations which are used as predictors to fit a linear model.

### 5.1 Subset Selection

#### 5.1.1 Best Subset Selection

Fit a linear model for each combination of predictors and identify the one that is best through AIC, BIC, R2, RSE. However, as predictors are added R2 increases and RSS decreases. For this reason, cross-validation is used to validate model selection.

Subset selection can be computationally costly, so there are efficient alternatives.

### 5.1.2 Stepwise Selection

- Forward selection: start with intercept with no predictors and add variables that minimize the RSS.
- Backward selection: start with all variables and remove variables that are the least stat sign.
- Mixed selection: combination of forward and backward selection. Start with no variables, add variables that provides the best fit, but only add variables below a certain threshold.

### 5.1.3 Choosing the Optimal Model

RSS and R2 are not suitable for model selection because more variables will be the preferred model. Best to calculate the test error,

- indirectly estimate test error by making adjustments to the training error to account for bias due to overfitting
- directly estimate the test error using validation or cross-validation

**Cp, AIC, BIC, and Adjusted R2** Four approaches exist for adjusting the training error, Cp, AIC, BIC, Adjusted R2.

The $C_p$ estimate of the test MSE is computed as,

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

where $d$ is number of predictors.

$C_p$ statistic adds a penalty to $2d\hat{\sigma}^2$ to the training RSS in order to adjust for the fact that the training error tends to underestimate the error. The penalty increases as number of predictors which adjusts for the corresponding decrease in RSS as predictors are added.

AIC criterion is defined as,

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

BIC is derived from Bayesian point of view,

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

Adjusted $R^2$ incorporates RSS and RSS but with $n$ and $d$.

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

The intuition behind the adjusted R2 is that once all of the correct variables have been included in the model, adding additional noise variables will lead to only a very small decrease in RSS. In theory, the model with the largest adjusted R2 will have only correct variables and no noise variables. (R2 is generally not used over AIC, BIC or $C_p$). *Note: A small value for $C_p, AIC, and BIC$ indicates a model with a low test error. A large value for adjusted R2 indicates a model with a small test error.*

**Validation and Cross-validation** We can compute the validation set error or the cross-validation error for each model under consideration, and then select the model for which the resulting estimated test error is smallest. This procedure has an advantage relative to AIC, BIC, Cp, and adjusted R2, in that it provides a direct estimate of the test error, and makes fewer assumptions about the true underlying model. It can also be used in a wider range of model selection tasks, even in cases where it is hard to pinpoint the model degrees of freedom (e.g. the number of predictors in the model) or hard to estimate the error variance $\sigma^2$.

However, cross-validation is computationally expensive, but is generally preferred.

**One-standard error rule:** Calculate the standard error of the estimated test MSE for each model size, then select the smallest model where the test error is within one standard error of the lowest point on the curve. If models are more or less equally good, then we want the simplest model.

## 5.2 Shrinkage Methods

Instead of cross-validating models with combinations of predictors, we can estimate a model with all predictors and constrain or regularize the coefficients toward zero.

### 5.2.1 Ridge Regression

Ridge regression is similar to least squares expect the minimization problem includes a tuning parameter, $\lambda$.

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

where $\lambda \geq 0$ is the tuning parameter. Ridge regression reduces RSS but incorporates a shrinkage term $\lambda \sum_j \beta_j^2$. The tuning parametric series to control the relative impact of these two terms on the regression coefficient estimates.

When $\lambda = 0$ the penalty term has no effect, which reduces to OLS. As $\lambda$ increases to infinity, the impact of the shrinkage penalty grows and the coefficients will approach zero.

**Note: Shrinkage does not apply to the intercept.**

**Why Does Ridge Regression Improve Over Least Squares?** Ridge regression's advantage over least squares is rooted in the bias-variance trade-off. As $\lambda$ increases, the flexibility of the ridge regression fit decreases, leading to decreased variance but increased bias – the shrinkage of the ridge coefficient estimates leads to a substantial reduction in the variance of the predictions, at the expense of a slight increase in bias.

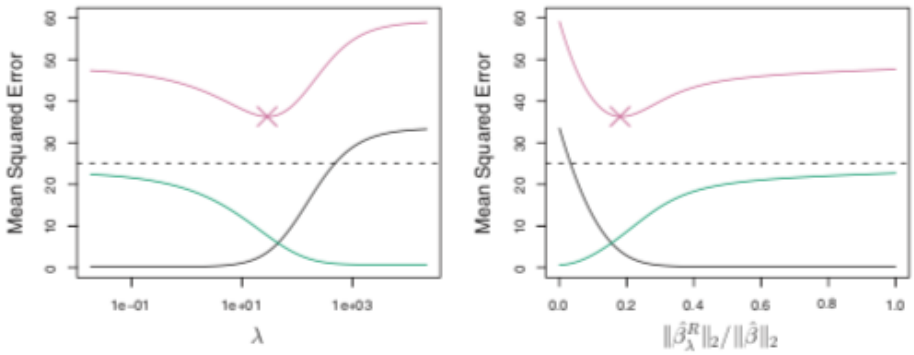


**FIGURE 6.5.** *Squared bias (black), variance (green), and test mean squared error (purple) for the ridge regression predictions on a simulated data set, as a function of* $\lambda$ *and* $\|\hat{\beta}_\lambda^R\|_2/\|\hat{\beta}\|_2$*. The horizontal dashed lines indicate the minimum possible MSE. The purple crosses indicate the ridge regression models for which the MSE is smallest.*

In situations where the relationship between the response and the predictors is close to linear, the least squares estimates will have low bias but may have high variance. With a large number of predictors versus number of observations, linear regression will be extremely variable. Ridge regressions performs well by reducing the variance with only a slight increase in bias; thus, ridge regressions work best in situations where the least squares estimates have high variance.

One disadvantage of ridge regression is the model will always include all predictors as opposed to subset of variables through recursive selection.

### 5.2.2 The Lasso

The lasso is an alternative to the ridge regression that allows predictors to equal zero.

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

The main difference between ridge and lasso is that $\beta_j^2$ is replaced with $|\beta_j|$. Similarly, the lasso regressions shrink coefficients towards zero and can include zero. Lasso produces sparse models or models that involve only a subset of the variables. Selecting $\lambda$ is done in the same way too.

**Comparing the Lasso and Ridge Regression** The lasso leads to qualitatively similar behavior to ridge regression, in that as $\lambda$ increases, the variance decreases and the bias increases.
**Note:** In general, one might expect the lasso to perform better in a setting where a relatively small number of predictors have substantial coefficients, and the remaining predictors have coefficients that are very small or that equal zero. Ridge regression will perform 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. A technique such as cross-validation can be used in order to determine which approach is better on a particular data set.

### 5.2.3 Selecting the Tuning Parameter

Cross-validation provides a simple way to tackle this problem. We choose a grid of $\lambda$ values, and compute the cross-validation error for each value of $\lambda$. We then select the tuning parameter value for which the cross-validation error is smallest. Finally, the model is re-fit using all of the available observations and the selected value of the tuning parameter.

## 5.3 Dimension Reduction Methods

Dimension reductions methods transform the predictors and then fit a least squares model using the transformed variable.

$$Z_m = \sum_{j=1}^{p} \phi_{jm} X_j$$

where $Z_m$ represent linear combinations of original predictors $p$ and constants, $\phi_{jm}$. $Z_m$ are used to fit the linear regression model and can lead to better results than fitting using least squares.

Dimension reduction serves to constrain the estimated $\beta_j$ coefficients, but can bias the coefficients. However, when predictors, $p$, is large relative to $n$, selecting $M << p$ can reduce the variance of the fitted coefficients.

If $M = p$ no dimension reduction occurs and same as least squares with original predictors.

### 5.3.1 Principal Components Regression

PCA is a technique for reducing the dimension of $X$. The first component PC1 vary the most, then PC2, ... PCN. Or, in other words *the first principal component vector defines the line that is as close as possible to the data (original observations) and captures most of the information contained in the data.*

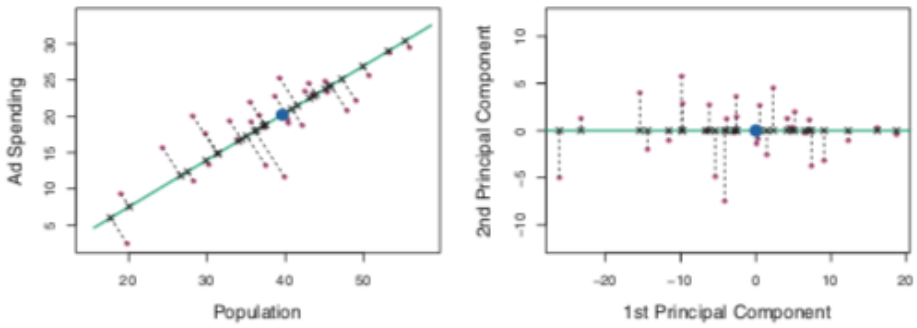


**FIGURE 6.15.** *A subset of the advertising data. The mean* pop *and* ad *budgets are indicated with a blue circle.* Left: *The first principal component direction is shown in green. It is the dimension along which the data vary the most, and it also defines the line that is closest to all* $n$ *of the observations. The distances from each observation to the principal component are represented using the black dashed line segments. The blue dot represents* $(\overline{\text{pop}}, \overline{\text{ad}})$. Right: *The left-hand panel has been rotated so that the first principal component direction coincides with the* $x$*-axis.*

**The Principal Components Regression Approach** PCR involves constructing the first $M$ principals $Z_1, ...Z_m$ and using components as the predictors in a linear regression model. A small number of principals are selected that explain most of the variability in the data.

If assumptions underlying PCR holds, then using principal components instead of original predictors will lead to better results. Further, reducing dimensions can mitigate overfitting.

The following figure shows that as more principal components are used the bias decreases, but the variance increases. This can result in substantial improvements over least squares.

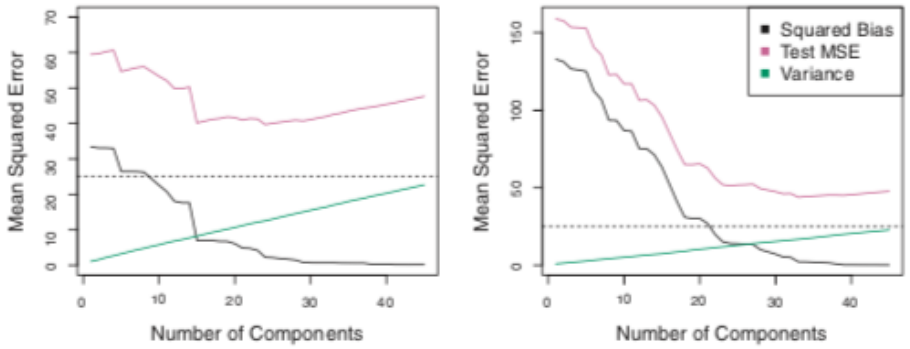


**FIGURE 6.18.** *PCR was applied to two simulated data sets.* Left: *Simulated data from Figure 6.8.* Right: *Simulated data from Figure 6.9.*

However, contrast to ridge/lasso results show PCR does not perform as well. The worst performance of PCR is a consequence of using many principal components. PCR will do better when the first few principals components explain most of the variation. PCR number of components are selected using cross-validation.

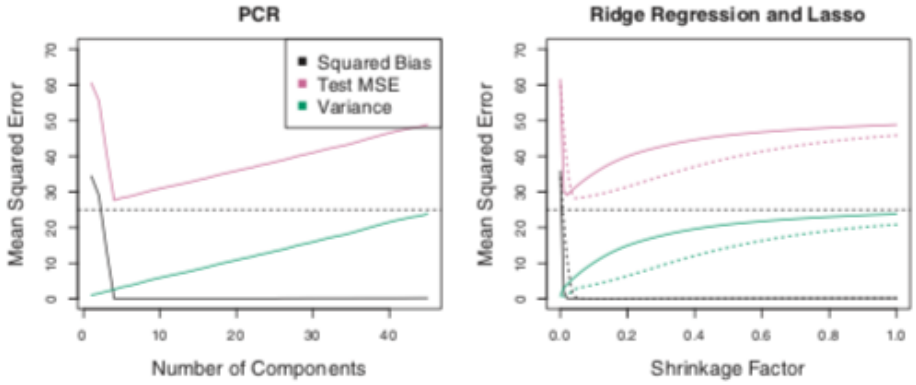


**FIGURE 6.19.** *PCR, ridge regression, and the lasso were applied to a simulated data set in which the first five principal components of $X$ contain all the information about the response $Y$. In each panel, the irreducible error $Var(\epsilon)$ is shown as a horizontal dashed line.* Left: *Results for PCR.* Right: *Results for lasso (solid) and ridge regression (dotted). The $x$-axis displays the shrinkage factor of the coefficient estimates, defined as the $\ell_2$ norm of the shrunken coefficient estimates divided by the $\ell_2$ norm of the least squares estimate.*

**Note:** PCR is not a feature selection method, which does not result in the development of a model that relies on a small set of the original features.

Principal components are standardized to ensure all variables are on the same scale. Without standardization, high-variance variables will tend to play a larger role in the principal components obtained.

A major drawback of PCA is there is no guarantee that the directions that best explain the predictors will also be the best directions for use for predicting the response (unsupervised learning methods).

### 5.3.2 Partial Least Squares

PLS overcomes the limitations of PCA by making use of the response $Y$. PLS attempts to find directions that help explain both the response and the predictors through dimension reduction similar to PCA (supervised learning method).

Steps to compute PLS:

- Standardize predictors and response.
- Compute first direction $Z_1$ by setting each constant, $\phi_{j1}$, equal to the coefficient from OLS.
- PLS places the highest weight on the variables that are most strongly related to the response.

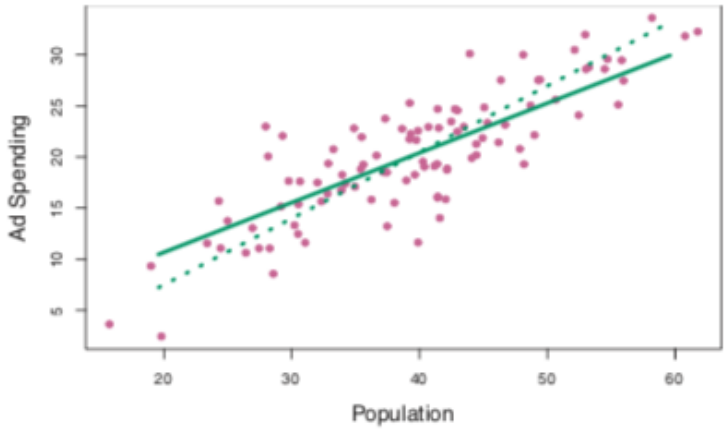


**FIGURE 6.21.** *For the advertising data, the first PLS direction (solid line) and first PCR direction (dotted line) are shown.*

**Note:** PLS does not fit the predictors as closely as PCA, but it does a better job explaining the response.

The second PLS direction is calculated by first adjusting each of the variables for $Z_1$ by regressing each variable on $Z_1$ and obtaining the residuals. The residuals are interpreted as the remaining information that has not been explained by the first PLS direction. $Z_2$ is computed using the *orthoganalized data*. This process is continued through each direction.

**The number of directions used are chosen through cross-validation.**

While supervised dimension reduction of PLS can reduce bias, it also increases the variance so the benefits of PLS relative to PCR are a wash.

## 5.4 Considerations in High Dimensions

High dimension data is defined as more predictors, $p$, than number of observations, $n$. As a result, classical approaches, such as OLS, are not appropriate.

### 5.4.1 What Goes Wrong in High Dimensions?

In high dimension data, least squares will yield a set of coefficient estimates that result in a perfect fit to the data (residuals = 0). This will lead to overfitting and low performance on test predictions.

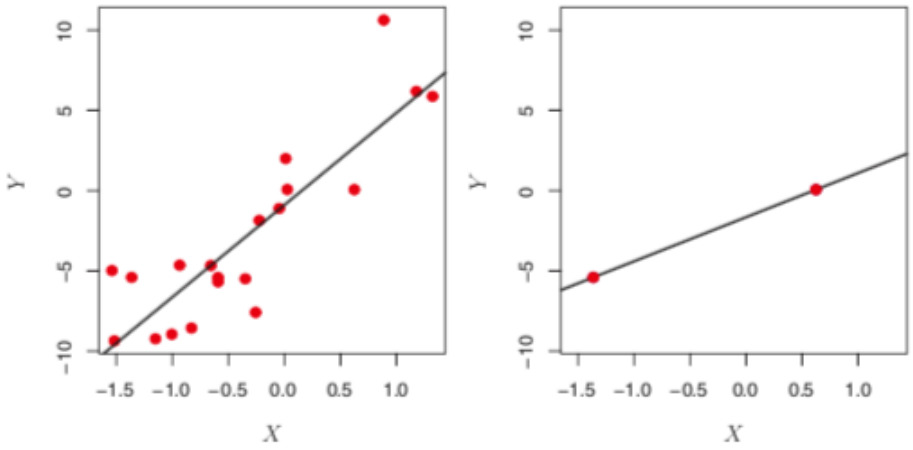


**FIGURE 6.22.** Left: *Least squares regression in the low-dimensional setting.* Right: *Least squares regression with $n = 2$ observations and two parameters to be estimated (an intercept and a coefficient).*

When performing cross-validation on models, $C_p$, AIC, and BIC approaches are not appropriate because estimate $\hat{\sigma}^2$ is problematic. $R^2$ is also problematic because fit will equal 1.

### 5.4.2 Regression in High Dimensions

Models for fitting less flexible least squares are useful for performing regression in high-dimensions, such as forward stepwise selection ,ridge, lasso, and PCR. These methods avoid overfitting by using less flexible fitting approaches than least squares.

The following figure highlights three important points:

(1) regularization or shrinkage plays a key role in high-dimensional problems,

(2) appropriate tuning parameter selection is crucial for good predictive performance, and

(3) the test error tends to increase as the dimensionality of the problem (i.e. the number of features or predictors) increases, unless the additional features are truly associated with the response.

**Curse of Dimensionality**: noisy features increase the dimensionality of the problem, exacerbating the risk of overfitting (noise features may be assigning nonzero coefficients due to change).

**Note:** adding additional signal features associated with the response will improve the fitted model, but noisy features will lead to deterioration of the fitted model and increase the test error.

### 5.4.3 Interpreting Results in High Dimensions

In high-dimensional settings, multicollinearity problem is extreme: any variable in the model can be written as a linear combination of all of the other variables in the model.

Essentially, this means that we can never know exactly which variables (if any) truly are predictive of the outcome, and we can never identify the best coefficients for use in the regression.

It is important to take care when reporting errors and measure of model fit in high-dim settings. When number of predictors is larger than number of observations, it is easy to obtain a useless model that has zero residuals.
**Note:** Never use sum of squares errors, p-values, $R^2$, or other traditional measure of model fit. It is important to report results on an independent test set, or cross-validation errors.

# 6 Moving Beyond Linearity

In this chapter we relax the linearity assumption while still attempting to maintain as much interpretability as possible. We do this by examining very simple extensions of linear models like polynomial regression and step functions, as well as more sophisticated approaches such as splines, local regression, and generalized additive models.

- Polynomial regression: extends linear model by adding extra predictors, such as $X, X^2, X^3$
- Step Function: cut the range of variables into K distinct regions (fitting a piecewise constant function)
- Regression Splines: more flexible than polynomials and step functions. Involved dividing the range of $X$ and $K$ distinct regions, fit a polynomial that join smooths the boundaries, or *knots*.
- Smoothing Splines: Similar to regression splines, but minimizing a residual sum of squares subject to smoothness penalty.
- Local Regression: similar to splines, but regions are allowed to overlap
- Generalized Additive Models (GAM): extend methods above to deal with multiple predictors.

## 6.1 Polynomial Regression

A polynomial function follows,

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_i^2 + ... + \beta_d x_i^d + \epsilon_i$$

with a large enough $d$, this regression can produce a highly non-linear curve, but usually don't want to use larger than 4 because will produce overly flexible fits.

## 6.2 Step Function

We can use step function in order to avoid imposing a global structure such as in a linear model. Here we break the range of $X$ into bins or cut points $c_1, c_2, ..., c_K$, and fit a different constant in each bin from a continuous variable into an *ordered categorical variable*.

$$
\begin{aligned}
C_0(X) &= I(X < c_1), \\
C_1(X) &= I(c_1 \leq X < c_2), \\
C_2(X) &= I(c_2 \leq X < c_3), \\
&\vdots \\
C_{K-1}(X) &= I(c_{K-1} \leq X < c_K), \\
C_K(X) &= I(c_K \leq X),
\end{aligned}
$$

where $I(\cdot)$ is an indicator function that returns a 1 if the condition is true and 0 otherwise (dummy variables). We then fit a linear model as,

$$y_i = \beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + ... + \beta_K C_K(x_i) + \epsilon_i$$

A limitation of step-function regressions is that unless there is a natural cut (breakpoint) in the predictors, piece-wise-constant functions can miss the action. See left panel below where the first bin clearly misses the increasing trend.

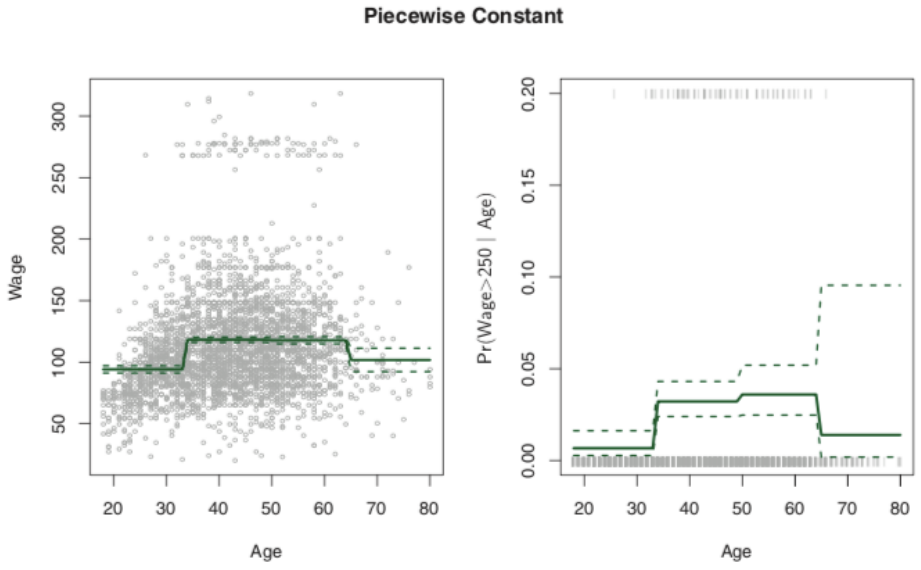


**FIGURE 7.2.** *The* `Wage` *data.* Left: *The solid curve displays the fitted value from a least squares regression of* `wage` *(in thousands of dollars) using step functions of* `age`*. The dotted curves indicate an estimated 95 % confidence interval.* Right: *We model the binary event* `wage>250` *using logistic regression, again using step functions of* `age`*. The fitted posterior probability of* `wage` *exceeding* $250,000 *is shown, along with an estimated 95 % confidence interval.*

## 6.3 Basis Function

Polynomial and piecewise-constant regression models are in fact special cases of a basis function approach where the variables are transformed,

$$y_i = \beta_0 + \beta_1 b_1(x_1) + ... + \beta_K b_K(x_i) + \epsilon_i$$

where $b_K(\cdot)$ are fixed and known. Utilizing a linear regression in this way provides an easy estimate for polynomial and step-function regressions, so inference tools such as s.e., f-stats, etc. are available.

## 6.4 Regression Splines

Regression splines provide flexible classes of the basis functions.

### 6.4.1 Piecewise Polynomials

Piecewise polynomial regressions involve fitting separate low-degree polynomials over different regions of $X$. A piecewise cubic polynomial with a single known at a point $c$ takes the form,

$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + \epsilon_i & \text{if } x_i < c; \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + \epsilon_i & \text{if } x_i \geq c. \end{cases}$$

Here, we fit two different polynomial functions based on the subset of observations at cut $c$. More knots lead to a more flexible piecewise polynomial. For $K$ knots, we fit $K + 1$ different cubic polynomials.
Note: when $K = 0$, this produces a piecewise linear function.

Note that in the figure below, the top left panel provides a piece wise cubic function that is not smooth across age. The discontinuity can be a problem, so it is best to use a continuous piecewise cubic that constrains the polynomial to be continuous; although, notice it is not perfectly continuous.

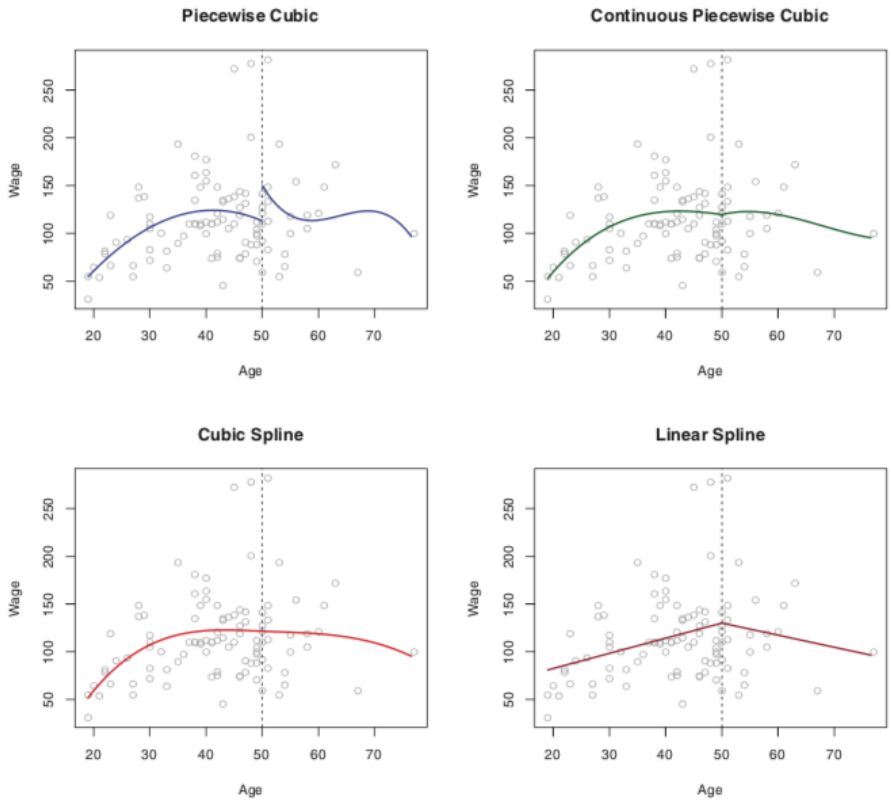


**FIGURE 7.3.** *Various piecewise polynomials are fit to a subset of the* `Wage` *data, with a knot at* `age=50`. Top Left: *The cubic polynomials are unconstrained.* Top Right: *The cubic polynomials are constrained to be continuous at* `age=50`. Bottom Left: *The cubic polynomials are constrained to be continuous, and to have continuous first and second derivatives.* Bottom Right: *A linear spline is shown, which is constrained to be continuous.*

### 6.4.2 Constraints and Splines

Another way to constrain the function form is to allow for a first and second derivative. A benefit to this is that it frees up degrees of freedom. A cubic spline (lower left in figure above) reduces the complexity of the piecewise polynomial which uses $K$ knots and ensures continuity across the polynomial. Similarly, a linear spline provides continuity in the derivative up to the degree at each knot (lower right panel in figure above).

### 6.4.3 The Spline Basis Representation

How do you fit a polynomial with a first and second derivative? A basis model can represent a regression spline under these conditions which can then be fit with a linear regression.

$$y_i = \beta_0 + \beta_1 b_1(x_1) + ... + \beta_{K+3} b_{K+3}(x_i) + \epsilon_i$$

The way to fit this type of basis function is to start off with a basis for a cubic polynomial $(x, x^2, x^3)$ then add a *truncated power basis* function per knot,

$$h(x,\xi) = (x-\xi)_+^3 = \begin{cases} (x-\xi)^3 & \text{if } x > \xi \\ 0 & \text{otherwise,} \end{cases}$$

where $\xi$ is the knot. This results in a least squares regression with an intercept and $3+K$ predictors: (1) $X, X^2, X^3$; (2) $h(X,\xi_1), h(X,\xi_2), h(X,\xi_3)$ where $xi_1, ..., \xi_K$ are knots.

A limitation to splines is they can have high variance at the outer range of the predictors. A *natural spline* is a regression spline where the function is required to be linear on the boundaries; thus, producing more stable estimates on the boundaries.

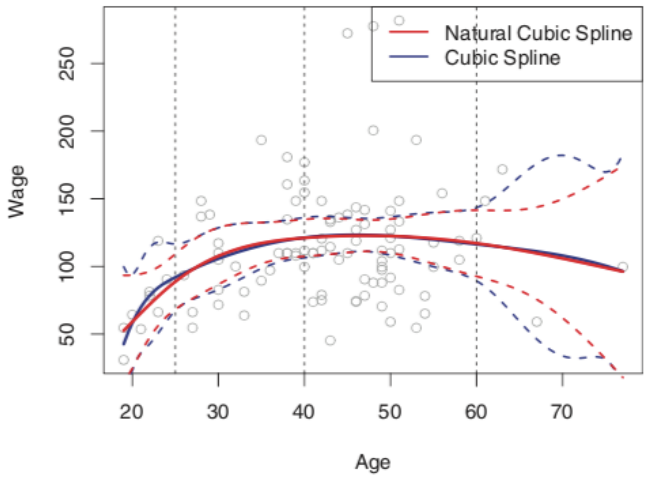


**FIGURE 7.4.** *A cubic spline and a natural cubic spline, with three knots, fit to a subset of the* `Wage` *data.*

### 6.4.4 Choosing the Number and Locations of the Knots

It is common to place knots in uniform quantiles across the variable. The number of knots depends on how best the spline fits the data. This is generally done through cross-validating RSS.

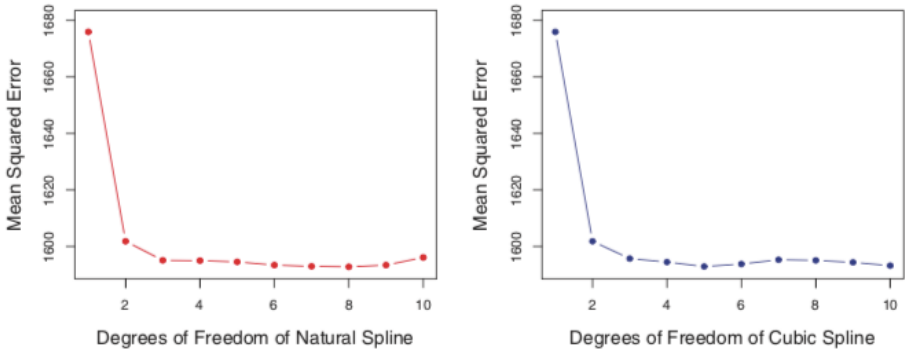


**FIGURE 7.6.** *Ten-fold cross-validated mean squared errors for selecting the degrees of freedom when fitting splines to the* `Wage` *data. The response is* `wage` *and the predictor* `age`. Left: *A natural cubic spline.* Right: *A cubic spline.*

### 6.4.5 Comparison to Polynomial Regression

Regression splines often give superior results to polynomial regressions because spline introduce flexibility by increasing the number of knots but keeping the degree fixed, thus producing more stable estimates. Splines also allow us to place more knots, and hence flexibility, over regions where the function $f$ seems to be changing rapidly, and fewer knots where $f$ appears more stable.

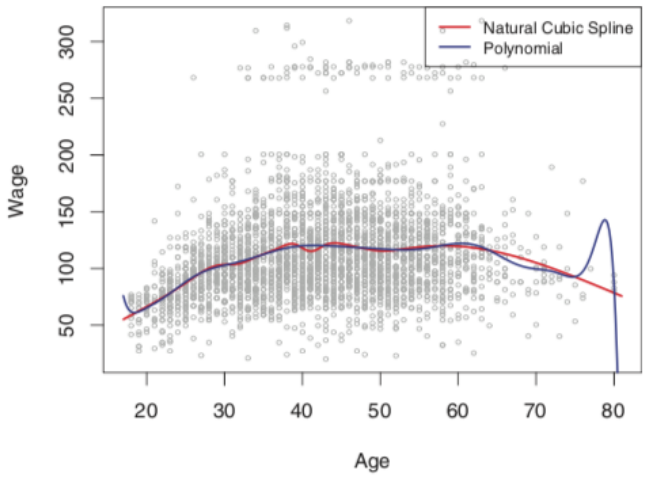


**FIGURE 7.7.** *On the* `Wage` *data set, a natural cubic spline with 15 degrees of freedom is compared to a degree-15 polynomial. Polynomials can show wild behavior, especially near the tails.*

## 6.5 Smoothing Splines

### 6.5.1 An Overview of Smoothing Splines

In fitting a smooth curve, we want a function, $g(x)$, that fits the data well where we seek to solve,

$$RSS = \sum_{i=1}^{n}(y_i - g(x_i))^2$$

However, if we put constraints on $g(x)$ then the functional form will fit the data perfectly (overfit), thus RSS will equal zero. What we want is a function that makes RSS small, but that is also smooth.

A natural approach is to minimize,

$$\underbrace{\sum_{i=1}^{n}(y_i - g(x_i))^2}_{\text{Loss Function}} + \underbrace{\lambda \int g''(t)^2 dt}_{\text{Penalty Term}}$$

where $\lambda$ is a nonnegative tuning parameter. The function $g(\cdot)$ that minimizes the previous equation is the smoothing spline.

Note: The first derivative $g'(t)$ measures the slope of a function at t, and the second derivative corresponds to the amount by which the slope is changing. Hence, broadly speaking, the second derivative of a function is a measure of its roughness: it is large in absolute value if $g(t)$ is very wiggly near $t$, and it is close to zero otherwise.

In other words, $\int g''(t)^2 dt$ is simply a measure of the total change in the function $g'(t)$ over its entire range. If $g$ is very smooth, then $g'(t)$ will be close to constant and $g''(t)^2$ will take on a small value. Conversely, if $g$ is jumpy and variable then $g'(t)$ will vary significantly and $g''(t)^2 dt$ will take on a large value. The larger the value of $\lambda$, the smoother $g$ will be.

The function $g(x)$ that solves the minimization problem has specific properties,

- is a piecewise cubic polynomial with knots at the unique values of $x_1, ..., x_n$
- continuous first and second derivatives at each knot
- linear in the region outside of the extreme knows

**Note: In other words, the function $g(x)$ that minimizes the problem is a natural cubic spline with knots at $x_1, ..., x_n$. This is not the same natural cubic spline applied using the basis function in a linear regression.**

### 6.5.2 Choosing the Smoothing Parameter $\lambda$

The tuning parameter $\lambda$ controls the roughness of the smoothing spline, thus the effective degrees of freedom, $df_\lambda$. $df_\lambda$ is a measure of the flexibility of the smoothing spline – the higher it is, the more flexible (and the lower-bias but higher-variance) the smoothing spline is. The definition of $df_\lambda$ can be written as,

$$\hat{\mathbf{g}}_\lambda = \mathbf{S}_\lambda \mathbf{y}$$

where the vector of fitted values when applying a smoothing spline to the data can be written as an $nxn$ matrix $\mathbf{S}_\lambda$ times the response vector $\mathbf{y}$. Therefore, the effective degrees of freedom are defined to be the sum of the diagonal elements of the matrix $\mathbf{S}_\lambda$

$$df_\lambda = \sum_{i=1}^{n} \{\mathbf{S}_\lambda\}_{ii}$$

When fitting a smoothing spline, it is not necessary to choose the number of knots, but instead the value of $\lambda$, which is chosen through cross-validation by making RSS as small as possible. This is done using LOOCV, which is the same cost as computing a single fit, using the following formula,

$$RSS_{cv}(\lambda) = \sum_{i=1}^{n} (y_i - \hat{g}_\lambda^{(-i)}) = \sum_{i=1}^{n} [\frac{y_i - \hat{g}_\lambda(x_i)}{1 - \{\mathbf{S}_\lambda\}_{ii}}]^2$$

where $\hat{g}_\lambda^{(-i)}$ represents the fitted value for the smoothing spline evaluated at $x_i$ for all training data except for the $i$th observation. In contrast, $\hat{g}_\lambda(x_i)$ represents the smoothing spline function fit to all of the training observations and evaluated at $x_i$.

**Note:** The formula suggests that we can compute each of these LOOCV fits using only $\hat{g}_\lambda$, the original fit to all of the data. Notice in the figure below that LOOCV provides a reasonable estimate of the data while reducing effective degrees of freedom.

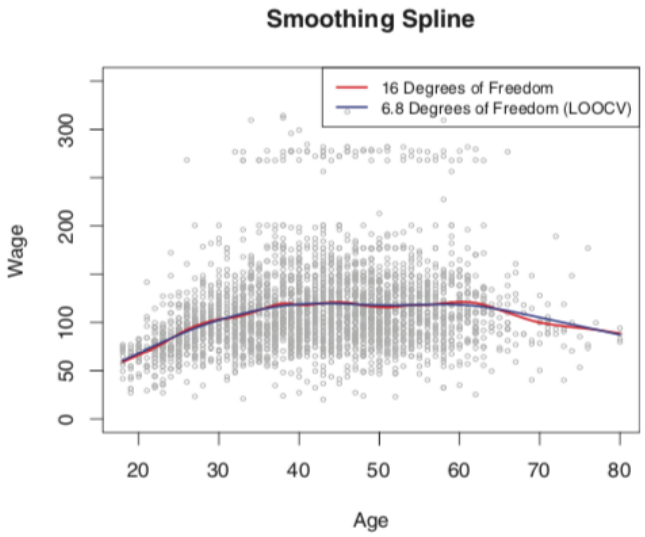


**FIGURE 7.8.** *Smoothing spline fits to the* `Wage` *data. The red curve results from specifying* 16 *effective degrees of freedom. For the blue curve,* $\lambda$ *was found automatically by leave-one-out cross-validation, which resulted in* 6.8 *effective degrees of freedom.*

## 6.6 Local Regression

Local regression is a different approach for fitting flexible non-linear functions, which involves computing the fit at a target point x0 using only the nearby training observations. To obtain the local regression fit at a new point, we need to fit a new weighted least squares regression model by minimizing for a new set of weights. Local regression is sometimes referred to as a memory-based procedure, because like nearest-neighbors, we need all the training data each time we wish to compute a prediction.

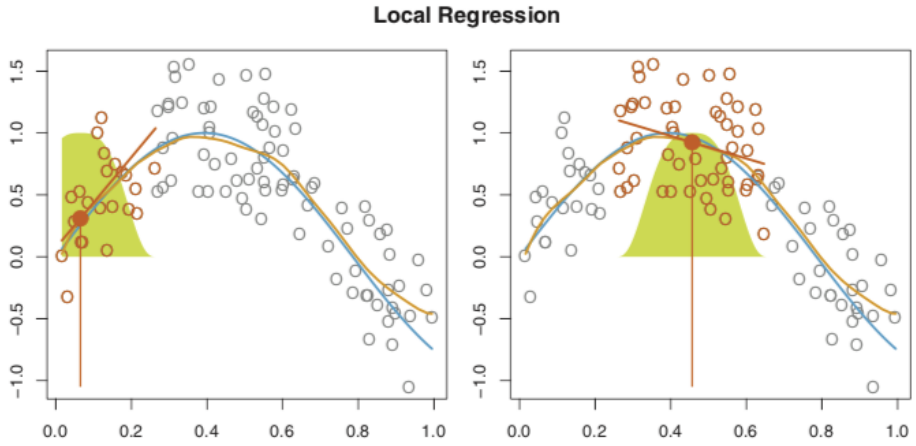


**FIGURE 7.9.** *Local regression illustrated on some simulated data, where the blue curve represents $f(x)$ from which the data were generated, and the light orange curve corresponds to the local regression estimate $\hat{f}(x)$. The orange colored points are local to the target point $x_0$, represented by the orange vertical line. The yellow bell-shape superimposed on the plot indicates weights assigned to each point, decreasing to zero with distance from the target point. The fit $\hat{f}(x_0)$ at $x_0$ is obtained by fitting a weighted linear regression (orange line segment), and using the fitted value at $x_0$ (orange solid dot) as the estimate $\hat{f}(x_0)$.*

Things to consider:

- how to define the weighting function $K$
- whether to fit a linear, constant, or quadratic regression
- (Most important) defining span, $s$, that controls the flexibility of the non-linear fit.

The smaller the value of $s$, the more local and wigglier the fit will be. Cross-validation is used to determine the optimal value of $s$.

**Algorithm 7.1** *Local Regression At $X = x_0$*

1. Gather the fraction $s = k/n$ of training points whose $x_i$ are closest to $x_0$.

2. Assign a weight $K_{i0} = K(x_i, x_0)$ to each point in this neighborhood, so that the point furthest from $x_0$ has weight zero, and the closest has the highest weight. All but these $k$ nearest neighbors get weight zero.

3. Fit a *weighted least squares regression* of the $y_i$ on the $x_i$ using the aforementioned weights, by finding $\hat{\beta}_0$ and $\hat{\beta}_1$ that minimize

$$\sum_{i=1}^{n} K_{i0}(y_i - \beta_0 - \beta_1 x_i)^2. \tag{7.14}$$

4. The fitted value at $x_0$ is given by $\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0$.

*Varying coefficient model* is a way to adapt a model to the most recently gathered data. **Local regressions can perform poorly when p-dimensional neighborhoods are small, say 3 or 4, because there will generally be very few training observations close to $x_0$ or the observation being interpolated.**

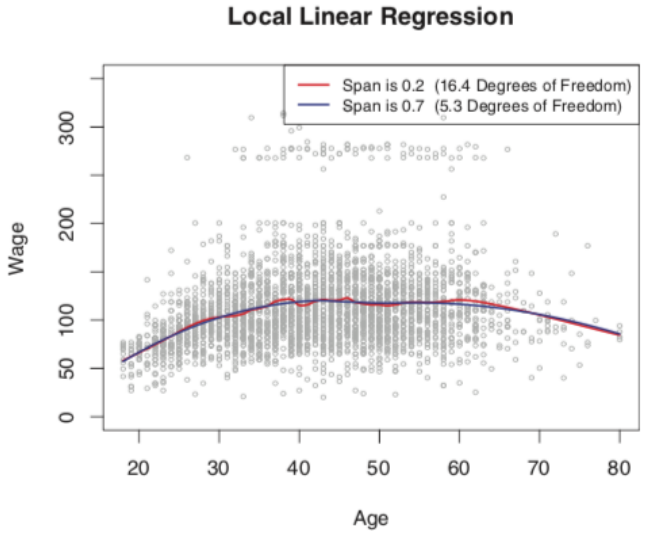


**FIGURE 7.10.** *Local linear fits to the* `Wage` *data. The span specifies the fraction of the data used to compute the fit at each target point.*

## 6.7 Generalized Additive Models (GAM)

Generalized additive models (GAMs) provide a general framework for extending a standard linear model by allowing non-linear functions of each of the variables, while maintaining additivity. GAMs can be applied to both quantitative and qualitative response.

### 6.7.1 GAMs for Regression Problems

In order to allow for non-linear relationships between each feature and the response it is best to replace each linear component $\beta_j x_{ij}$ with a smooth non-linear function $f_j(x_{ij}$. An *additive* model, GAM, can be written as,

$$y_i = \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + ... + f_p(x_{ip}) + \epsilon_i$$

It is called an additive model because we **calculate a separate $f_j$ for each $X_j$ and then add together all of their contributions**.

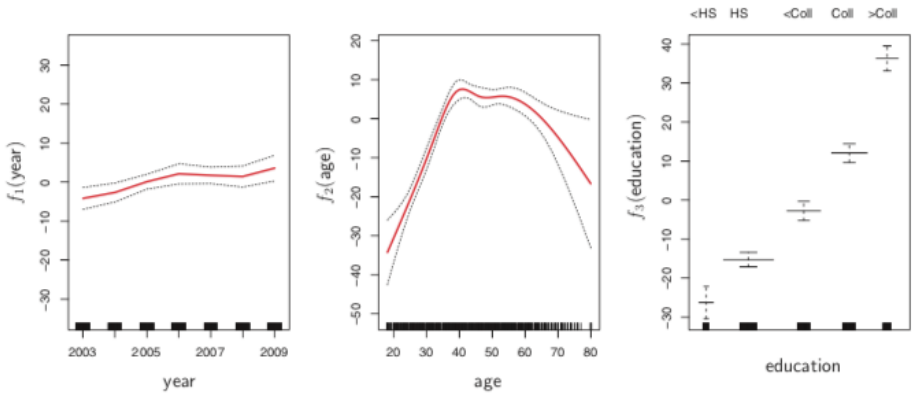


**FIGURE 7.11.** *For the* `Wage` *data, plots of the relationship between each feature and the response,* `wage`*, in the fitted model (7.16). Each plot displays the fitted function and pointwise standard errors. The first two functions are natural splines in* `year` *and* `age`*, with four and five degrees of freedom, respectively. The third function is a step function, fit to the qualitative variable* `education`*.*

A GAM regression model using a natural spline is simply a linear regression where covariates are fit using a function form of the data $f_p(x_{ip})$. However, fitting a GAM model with a smoothing spline is not as simple and is usually fit using backfitting.

**backfitting: involves multiple predictors by repeatedly updating the fit for each predictor in turn, holding the others fixed.**

Generally, the differences in GAMs using smoothing splines versus natural splines are small.

Pros and Cons of GAMs:

- (Pros) Do not need to manually try out many different transformations of each variable because the fit is automatically modeled in a non-linear framework.
- (Pros) Can be more accurate
- (Pros) Because the model is additive, we can examine the effects of each covariate on the response by holding all other variables fixed.
- (Pros) smoothness of function can be summarized via degrees of freedoms
- (Cons) Model is restricted to be additive, thus with many variables important interactions can be missed; although, we can simply add interactions to the model to account for the missing interactions.

### 6.7.2 GAMs for Classification Problems

GAMs can also be used in situations where Y is qualitative, such as,

$$log(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X_1 + ...\beta_p X_p$$

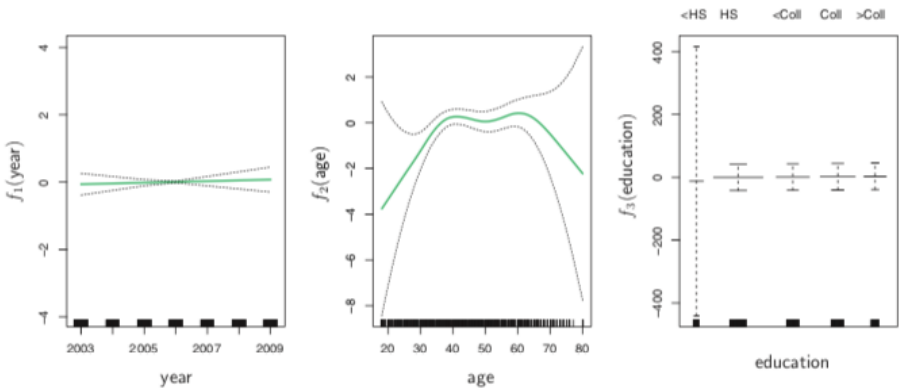


**FIGURE 7.13.** *For the* `Wage` *data, the logistic regression GAM given in (7.19) is fit to the binary response* `I(wage>250)`*. Each plot displays the fitted function and pointwise standard errors. The first function is linear in* `year`*, the second function a smoothing spline with five degrees of freedom in* `age`*, and the third a step function for* `education`*. There are very wide standard errors for the first level* `<HS` *of* `education`*.*

**Differences between poly, spline, and GAM:**

- Poly: Fits a specific quadratic functional form; solved through OLS
- Spline Regression: Fit using specific functional forms (natural spline, basis spline); solved through OLS
- GAM: Fits each covariate through a nonlinear functional form using high number of degrees; solved through penalty using $\lambda$ tuning parameter and integral of second derivative.

# 7 Tree-Based Methods

Tree-based methods stratify or segment the predictor space into a number of simple regions. In order to make a prediction for a given observation, we typically use the mean or the mode of the training observations in the region to which it belongs. Decision trees are not as accurate as other methods, such as bagging, random forests and boosting which rely on estimating multiple trees and combining to produce predictions.

## 7.1 The Basics of Decision Trees

Can be applied to both regressions and classifications.

### 7.1.1 Regression Trees

Process of building a regression tree:

- Divide the predictor space $(X_1, X_2, ...X_p)$ into $J$ distinct and non-overlapping regions, $R_1, ..., R_j$
- 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 observation, $R_j$

The primary goal of dividing the predictor space is to minimize the RSS,

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - y_{\hat{R}_j})^2$$

where $y_{\hat{R}_j}$ is the mean response for the training observations. This is computationally infeasible to consider every possible partition; therefore, we take a top-down approach know as recursive binary splitting.

**Recursive Binary Splitting:** Starting at the top of the tree, successive splits of the predictor space split via two new branches down the tree. The best split is made at the particular step.

To perform recursive binary splitting, the general outline is to consider all predictors, $X_1, ..., X_p$ and all possible values of the cut point $s$ for each of the predictors, and then choose the predictor and cut point such that the resulting tree has the lowest RSS. This is designated as,

$$R_1(j, s) = \{X | X_j <\} and R_2(j, s) = \{X | X_j \geq s\}$$

and seeks to minimize,

$$\sum_{i: x_i \in R_1(j,s)} (y_i - y_{\hat{R}_1})^2 + \sum_{i: x_i \in R_2(j,s)} (y_i - y_{\hat{R}_2})^2$$

Next, repeat the process by looking for the best predictor and best cut point in order to split the data further so as to minimize the RSS within each of the resulting regions. However, we split one of the two previously identified regions. The process continues until a stopping criterion is reach. Then predict responses.

**Tree Pruning** Splitting in such a way described above may produce good predictions but not on the test set. The tree may be too complex to produce out-of-sample predictive power. A smaller tree with few splits might lower variance and better interpretation at the cost of a little bias.

A better strategy is to grow a very large tree and then prune back to obtain a subtree. However, this is computationally expensive, so *cost complexity pruning or weakest link pruning* is one way to do this.

**Cost complexity pruning or weakest link pruning**: Consider a sequence of trees indexed by a nonnegative tuning parameter, $\alpha$.

**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 $\alpha$.

3. Use K-fold cross-validation to choose $\alpha$. 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 $k$th fold of the training data.

   (b) Evaluate the mean squared prediction error on the data in the left-out $k$th fold, as a function of $\alpha$.

   Average the results for each value of $\alpha$, and pick $\alpha$ to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of $\alpha$.

The goal is to include number of terminal nodes, $T$ when minimizing RSS,

$$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - y_{\hat{R}_m})^2 + \alpha |T|$$

The tuning parameter, $\alpha$, controls a trade-off between the subtree's complexity and its fit to the training data. As $\alpha$ increases, there is a price to pay for having a tree with many terminal modes. The tuning parameter is calculated using cross-validation.

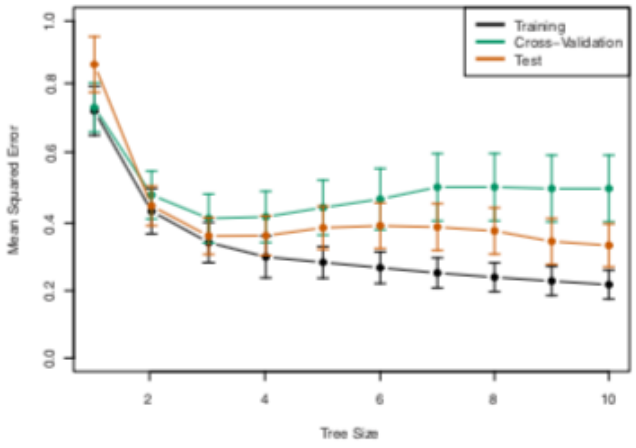


**FIGURE 8.5.** *Regression tree analysis for the* `Hitters` *data. The training, cross-validation, and test MSE are shown as a function of the number of terminal nodes in the pruned tree. Standard error bands are displayed. The minimum cross-validation error occurs at a tree size of three.*

### 7.1.2 Classification Trees

Classification trees are very similar to regression trees except the response is different. For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region.

Instead of RSS, classification trees use the error rate to optimize splitting. However, the classification error rate may not be appropriate, so a Gini index can be used instead,

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

which measures a total variance across the $K$ classes. Gini index is a measure of node purity where a small value indicates that a node contains predominantly observations from a single class.

Entropy can also be used as an alternative to the Gini index,

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} log \hat{p}_{mk}$$

A smaller value indicates the $m$th node is pure.

Gini or entropy are used to evaluate the quality of a particular split. The classification error rate is preferable if prediction accuracy of the final pruned tree is the goal.

Note: A split occurs when it increases node purity.

### 7.1.3 Trees Versus Linear Models

Which model is better: Trees or Linear Models?

**Linear Model:** If the relationship between the features and the response is well approximated by a linear model will outperform trees.

**Trees:** If highly non-linear relationships between variables then decision trees may outperform classical approaches.

Note: estimating the test error through cross-validation methods can determine which approach is best.

### 7.1.4 Advantages and Disadvantages of Tress

- (Good) Trees are very easy to explain to people (easier than linear regression)
- (Good) Mirror human decision making than regressions
- (Good) Displayed graphically, so interpretation is easier
- (Good) Handle qualitative predictors without needing to create dummy variables
- (Bad) Do not have same level of predictive accuracy as some of the other regression approaches
- (Bad) Trees can be very non-robust; a small change in the data can cause a large change in the final estimated tree.

Note: Bagging, RF, and Boosting methods can improve performance of decision trees.

## 7.2 Bagging, Random Forests, Boosting

### 7.2.1 Bagging

*Bootstrap aggregation, or bagging* is a general purpose procedure for reducing the variance of a statistical learning method, which is useful and frequently used in the context of decision trees.

**Note:** Averaging a set of observations reduces variance; thus, to increase the prediction accuracy of a model is to take many training sets, build a separate prediction model using each training set, and average the resulting predictions,

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(x)$$

Bagging involves taking repeated samples from a single training data set, $B$, then train method on $b$th bootstrapped training set, and average all predictions,

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x)$$

To apply bagging to regression trees, simply construct B regression tress using B bootstrapped training sets and average the resulting predictions. Trees are grown deep and not pruned, so individual tree has high variance, but low bias.

For classification problems, a majority vote is taken to predict group.

Note: The number of trees $B$ is not a critical parameter with bagging; using a very large value of B will not lead to overfitting. In practice we use a value of B sufficiently large that the error has settled down. Using $B = 100$ is sufficient to achieve good performance in this example.

**Out-of-Bag Error Estimation** Two thirds of the observations are used in each bagged tree, so the remaining one-third is referred to as the out-of-bag (OOB) observations – this is used as a valid estimate of the test error for the bagged model. A benefit to using OOB is that it is convenient when performing bagging on large data sets for which cross-validation would be computationally onerous.

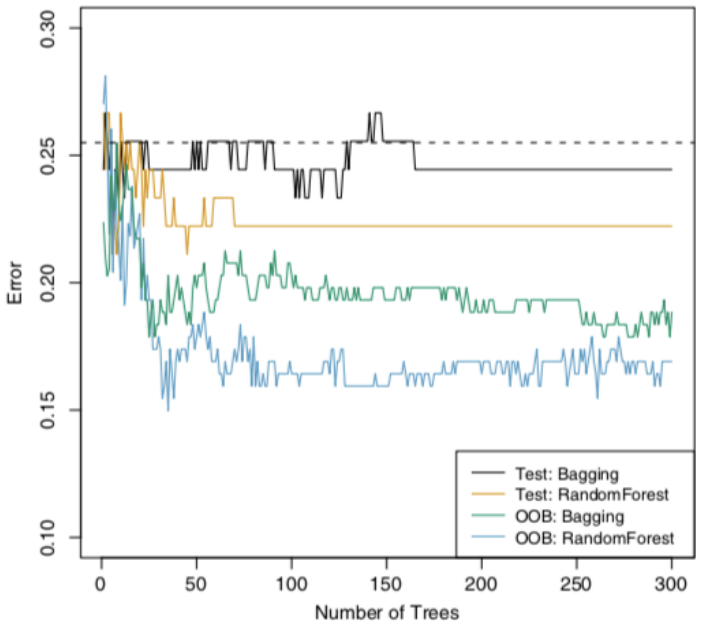


**FIGURE 8.8.** *Bagging and random forest results for the* `Heart` *data. The test error (black and orange) is shown as a function of* $B$*, the number of bootstrapped training sets used. Random forests were applied with* $m = \sqrt{p}$*. The dashed line indicates the test error resulting from a single classification tree. The green and blue traces show the OOB error, which in this case is considerably lower.*

**Variable Importance Measures** when we bag a large number of trees, it is no longer possible to represent the resulting statistical learning procedure using a single tree, and it is no longer clear which variables are most important to the procedure. Thus, bagging improves prediction accuracy at the expense of interpretability.

Variable important is calculated as the mean decrease in Gini index for each variable, relative to the largest.

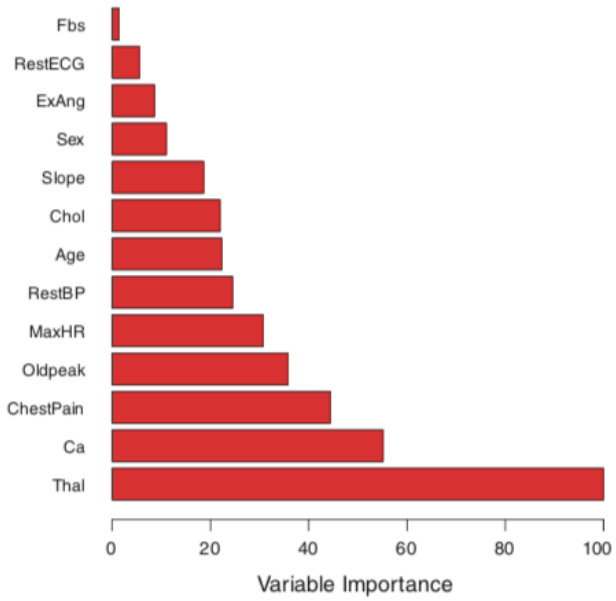


**FIGURE 8.9.** *A variable importance plot for the* `Heart` *data. Variable importance is computed using the mean decrease in Gini index, and expressed relative to the maximum.*

### 7.2.2 Random Forests

Random forests provide an improvement over bagged trees by way of a small tweak that decorrelates the trees. When building these decision trees, each time a split in a tree is considered, a random sample of $m$ predictors is chosen as split candidates from the full set of $p$ predictors. The split is allowed to use only one of those m predictors.

Note: A fresh sample of $m$ predictors is taken at each split, and typically we choose $m \approx \sqrt{p}$ - that is, the number of predictors considered at each split is approximately equal to the square root of the total number of predictors.

When splitting, the algorithm is not even allowed to consider a majority of the available predictors. This feature overcomes limitations in bagging where bagged trees are highly correlated, thus leading to large correlated predictions. RF only consider a subset of predictors, which decorrelates the trees thereby making the average of the result trees less variable and hence more reliable.

The main difference between bagging and RF is the choice of the predictor subset size $m$. If $m = p$, or subset of predictors equals total predictors then this is simply bagging. However, if $m = \sqrt{p}$ then a random forest is initiated which leads to a reduction in both test error and OOB error over bagging.

A small number of $m$ is typically used for building a RF when a large number of correlated predictors exist.

### 7.2.3 Boosting

Boosting works similarly to bagging, but the trees are grown sequentially: each tree is grown using information from previously grown trees. Boosting does not involve bootstrap sampling; instead each tree is fit on a modified version of the original data set.

The main idea behind boosting is that the approach learns slowly. We fit a tree using the current residuals rather than the outcome variable. The new decision tree is then used in the fitted function

to update the residuals. The trees can be rather small with a few nodes, which is determined by the parameter $d$ in the algorithm.

Sitting small trees to the residuals slowly improves $\hat{f}$ where the shrinkage parameter $\lambda$ slows the process down even further.

Note: learning approaches that learn slowly tend to perform well.

Tuning parameters for boosting:

- $B$: number of trees. Cross-validation selects B because boosting can overfit if $B$ is too large
- $\lambda$: learning rate, typically between 0.01 or 0.001. A small $\lambda$ can require a very large value of $B$
- $d$: number of splits in each tree, which controls the complexity.

Note: Difference between RF and boosting is that boosting generally requires small trees, or even a single predictor, whereas RF use a subset of predictors.

**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:
   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x). \tag{8.10}$$
   (c) Update the residuals,
   $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i). \tag{8.11}$$
3. Output the boosted model,
   $$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x). \tag{8.12}$$

# 8 Support Vector Machines

The support vector machine is a generalization of a simple and intuitive classifier called the maximal margin classifier; however, the classifier cannot be applied to most data sets because it requires that the classes be separable by a linear boundary.

## 8.1 Maximal Margin Classifier

### 8.1.1 What is a Hyperplane?

In a p-dimensional space, a hyperplane is a flat affine subspace of dimension p-1, defined as,

$$\beta_0 + \beta_1 X_1 + ... + \beta_p X_p = 0$$

However, if the hyperplane is not satisfied ($> 0 or < 0$) then $X$ lies to one side of the hyperplane. Therefore, a hyperplane divides p-dimensional space into two halves. To calculate the side of the hyperplane, simply solve for the equation above.

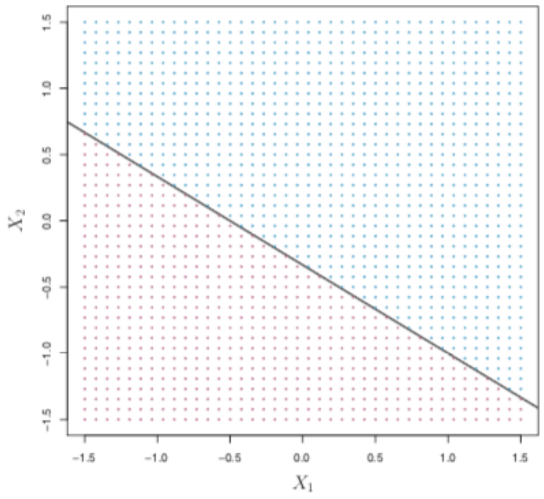


**FIGURE 9.1.** *The hyperplane* $1 + 2X_1 + 3X_2 = 0$ *is shown. The blue region is the set of points for which* $1 + 2X_1 + 3X_2 > 0$*, and the purple region is the set of points for which* $1 + 2X_1 + 3X_2 < 0$*.*

### 8.1.2 Classification Using a Separating Hyperplane

Suppose that it is possible to construct a hyperplane that separates the training observations perfectly according to their class labels.

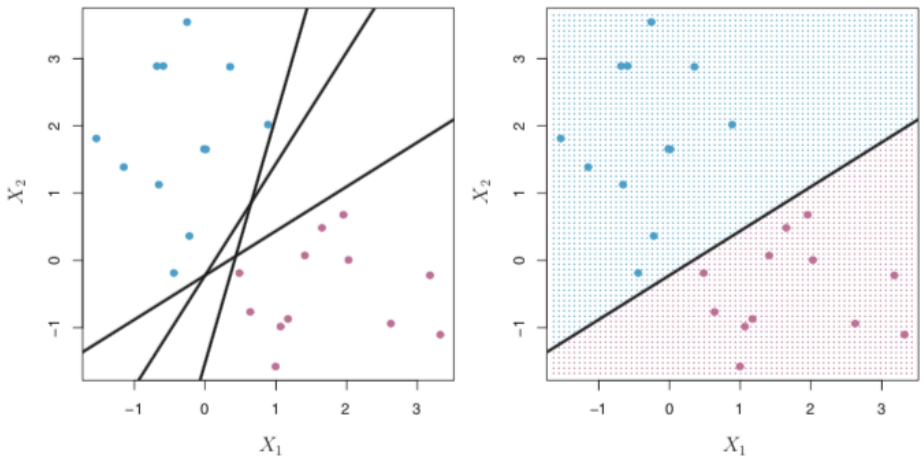


**FIGURE 9.2.** Left: *There are two classes of observations, shown in blue and in purple, each of which has measurements on two variables. Three separating hyperplanes, out of many possible, are shown in black.* Right: *A separating hyperplane is shown in black. The blue and purple grid indicates the decision rule made by a classifier based on this separating hyperplane: a test observation that falls in the blue portion of the grid will be assigned to the blue class, and a test observation that falls into the purple portion of the grid will be assigned to the purple class.*

If we label one set of observations that are red = 1 and the blue = -1, then we can utilize the equations above as,

$$\beta_0 + \beta_1 X_1 + ... + \beta_p X_p > 0 \text{if} y_i = 1$$

and

$$\beta_0 + \beta_1 X_1 + ... + \beta_p X_p > 0 \text{if} y_i = -1$$

or, more closely, a separating hyperplane has the property,

$$y_i(\beta_0 + \beta_1 X_1 + ... + \beta_p X_p) > 0$$

We can also quantify the magnitude based on how far the results are from the hyperplane; that is, how far away from zero (pos or neg).
**Note:** Hyperplane leads to a linear decision boundary.

### 8.1.3 The Maximal Margin Classifier

In general, if our data can be perfectly separated using a hyperplane, then there will in fact exist an infinite number of such hyperplanes. This is because a given separating hyperplane can usually be shifted a tiny bit up or down, or rotated, without coming into contact with any of the observations.

The *maximal margin hyperplane* (also known as the optimal separating hyperplane) separates the hyperplane that is farthest from the training observations. The goal is to compute the (perpendicular) distance from each training observation to a given separating hyperplane. The smallest distance is the minimal distance from the observations to the hyperplane (margin).

**Note:** The maximal margin hyperplane is the separating hyperplane for which the margin is larger – that is, it is the hyperplane that has the farthest minimum distance to the training observations.

It is important that there is a clear separation between training and test set in order to classify correctly.

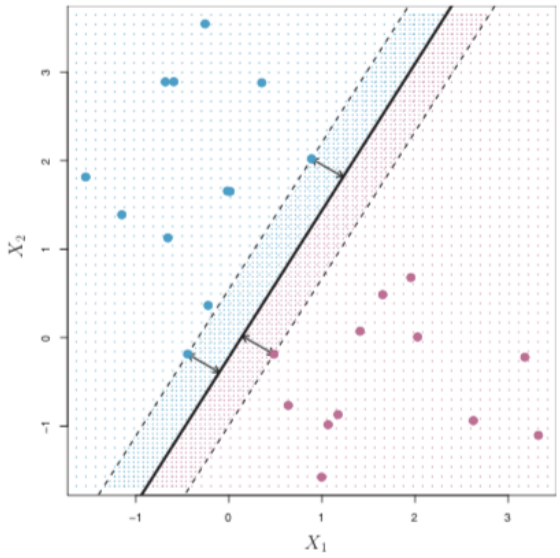


**FIGURE 9.3.** *There are two classes of observations, shown in blue and in purple. The maximal margin hyperplane is shown as a solid line. The margin is the distance from the solid line to either of the dashed lines. The two blue points and the purple point that lie on the dashed lines are the support vectors, and the distance from those points to the hyperplane is indicated by arrows. The purple and blue grid indicates the decision rule made by a classifier based on this separating hyperplane.*

Observations that lie on the hyperplane are known as *support vectors* and "support" the maximal margin hyperplane such that if these points shift, so does the maximal margin hyperplane.
**Note:** The maximal margin hyperplane depends directly on a small subset of the observations and is an important property in support vector machines.

### 8.1.4 Construction of the Maximal Margin Classifier

The maximal margin hyperplane is the solution that optimizes the following problem,

$$\text{maximize}_{\beta_0,\beta_1,...,\beta_p,M} M$$

$$s.t. \sum_{j=1}^{p} \beta_j^2 = 1 \text{and} y_i(\beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}) \geq M \forall i = 1, .., n$$

The problem is fairly simple to solve:

- The constraints ensure each observation is on the correct side of the hyperplane and at least a distance $M$ from the hyperplane.

- $M$ represents the margin of our hyperplane and the optimization problem choose $\beta_0, ..., \beta_p$ to maximize $M$.

### 8.1.5 The Non-separable Case

If no separating hyperplane exists there is no maximal margin classifier. Instead, we can estimate a hyperplane that almost separates the class, known as a soft margin.

## 8.2 Support Vector Classifiers

### 8.2.1 Overview of the Support Vector Classifier

A classifier based on a separating hyperplane will necessarily perfectly classify all of the training observations; this can lead to sensitivity to individual observations. However, adding additional observations that are not classified correctly results in a tiny margin. This creates a problem because the distance of an observation from the hyperplane can be seen as a measure of our confidence that the obs. is classified correctly. This can also result in overfitting.

It might be worthwhile to misclassify a few training observations in order to do a better job in classifying the remaining observations. A support vector (soft margin classifier) does exactly this.

**Support Vector Classifier Intuition:** Rather than seeking the largest possible margin so that every observation is not only on the correct side of the hyperplane but also on the correct side of the margin, we instead allow some observations to be on the incorrect side of the margin, or even the incorrect side of the hyperplane.

### 8.2.2 Details of the Support Vector Classifier

The support vector classifier classifies a test observation depending on which side of a hyperplane it lies. The optimization problem is,

$$maximize_{\beta_0,...,\beta_p,\epsilon_1,....,\epsilon_n,M} M$$

$$s.t. \sum_{j=1}^{p} \beta_j^2 = 1$$

$$y_i(\beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$$

$$\epsilon_i \geq 0, \sum_{i=1}^{n} \epsilon_i \leq C$$

where $C$ is a nonnegative tuning parameter and $M$ is the width of the margin where we seek to increase the margin as large as possible. $\epsilon$ are slack variables that allow individual observations to be on the wrong side of the margin.

$\epsilon$ tells us where the $i_{the}$ observation is located relative to the hyperplane and margin.

- $\epsilon = 0$, then the $i_{the}$ observation is on the correct side of the margin.
- $\epsilon > 0$ then it is on the wrong side and is said to violate the margin.
- $\epsilon > 1$ then it is on the wrong side of the hyperplane

$C$ bounds the sum of the $\epsilon$s and determines the number and severity of the violations to the margin and can be thought of as a budget for the amount that the margin can be violated to the margin.

- $C = 0$, then there is no budget or tolerance within the margin
- $C > 0$ no more than $C$ obs. can be on the wrong side of the hyperplane
- As $C$ increases become more tolerant of the violations (margin increases) and as $C$ decreases becomes less tolerant (margin shrinks).

$C$ is generally tuned via cross-validation.

$C$ controls the bias-variance trade-off: (1) Small $C$ narrows the margin and highly fits the data with lower bias but higher variance (2) large $C$ is wider and all more violations classifier leads to more bias but lower variance.

Note: Changing observations on either side will not affect the model, but obs. directly on the margin (support vectors) will affect the model.

Note: The fact that the support vector classifier's decision rule is based only on a potentially small subset of the training observations (the support vectors) means that it is quite robust to the behavior of observations that are far away from the hyperplane.

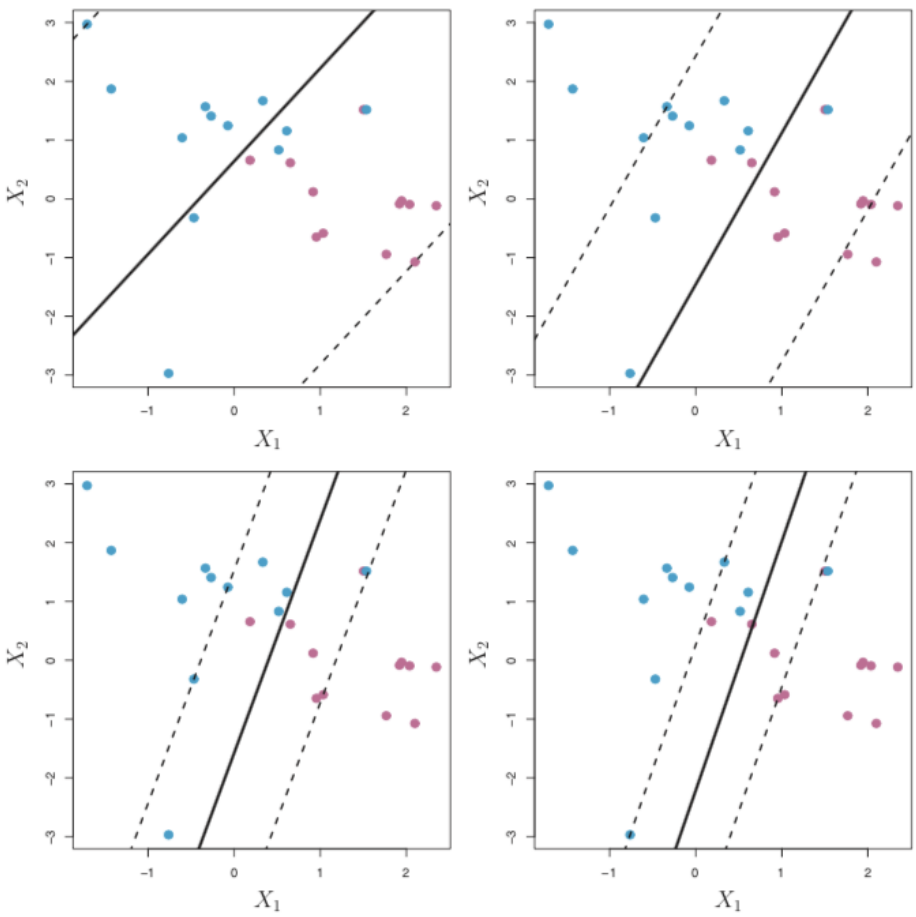


**FIGURE 9.7.** *A support vector classifier was fit using four different values of the tuning parameter $C$ in (9.12)–(9.15). The largest value of $C$ was used in the top left panel, and smaller values were used in the top right, bottom left, and bottom right panels. When $C$ is large, then there is a high tolerance for observations being on the wrong side of the margin, and so the margin will be large. As $C$ decreases, the tolerance for observations being on the wrong side of the margin decreases, and the margin narrows.*

## 8.3 Support Vector Machines

The support vector classifier is a natural approach for classification in the two-class setting, if the boundary between the two classes is linear. How- ever, in practice we are sometimes faced with non-linear class boundaries.

### 8.3.1 Classification with Non-linear Decision Boundaries

we could address the problem of possibly non-linear boundaries between classes in a similar way, by enlarging the feature space using quadratic, cubic, and even higher-order polynomial functions of the predictors,

$$X_1, X_2, ..., X_p$$

and could instead fit a support vector classifier using $2p$ features,

$$X_1, X_1^2, X_2, X_2^2, ..., X_p, X_p^2$$

which can be used to solve the optimization routine. However, increasing the feature space increases the complexity and computation. Support Vector Machines allow for increasing the feature space that leads to efficient computations.

### 8.3.2 The Support Vector Machine

The support vector machine (SVM) is an extension of the support vector classifier that results from enlarging the feature space in a specific way, using kernels.

To compute the support vector involves only the inner products of the observations,

$$< x_i, x_{i'} = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

The linear support vector classifier can be represented as,

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i < x, x_i >$$

in order to evaluate the function f(x), we need to compute the inner product between the new point x and each of the training points xi. However, it turns out that $\alpha_i$ is nonzero only for the support vectors in the solution—that is, if a training observation is not a support vector, then its $\alpha_i$ equals zero. So, if $S$ is the collection of indices of these support points, we can rewrite any solution function of the form ,

$$f(x) = \beta_0 + \sum_{ii \in S} \alpha_i < x, x_i >$$

We can generalize the inner product using a (linear) kernel,

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

which gives us back the support vector classifier.

The nonlinear (polynomial kernel) form can be estimated as,

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

When the support vector classifier is combined with a non-linear kernel, the resulting classifier is known as a support vector machine,

$$f(x) = \beta_0 + \sum_{ii \in S} \alpha_i K(x, x_i)$$

A radial kernel is also another popular choice,

$$K(x_i, x_{i'}) = exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

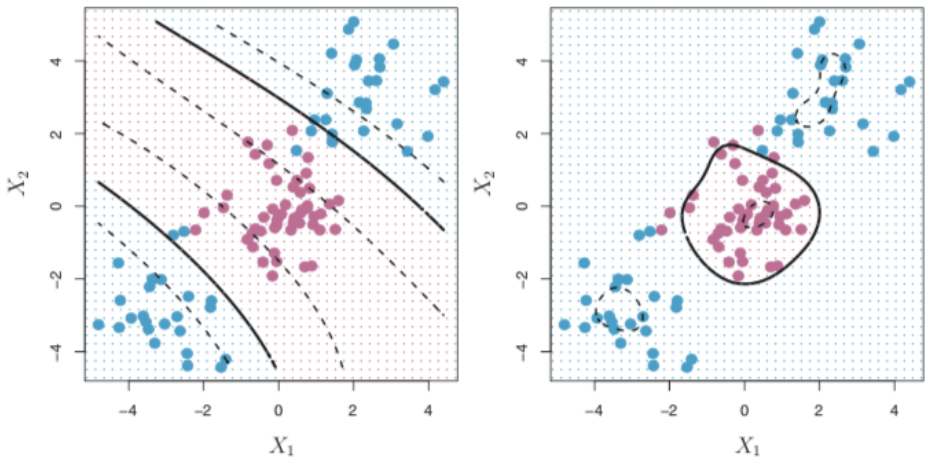


**FIGURE 9.9.** Left: *An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from Figure 9.8, resulting in a far more appropriate decision rule.* Right: *An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.*

Advantage to using a kernel rather than an enlarging feature space: (1) computationally faster by only computing the kernel instead of the space, which helps with large feature spaces.

## 8.4 SVMs with More than Two Classes

It turns out that the concept of separating hyperplanes upon which SVMs are based does not lend itself naturally to more than two classes. However, two approaches have been proposed to extend SVMs (1) one-versus-one and (2) one-versus-all

### 8.4.1 One-Versus-One Classification

A one-versus-one or all pairs approach constructs SVM's as a pair of classes. The test observation is classified to each pair of SVMs and the most frequent classification is assigned.

### 8.4.2 One-Versus-All Classification

One-Versus-All fit $K$ SVM's, each time comparing one of the K classes to the remaining K-1 classes. We assign the observation to the class where $(\beta_{0k} + \beta_{1k}x_1^* + ... + \beta_{pk}x_p^*)$ is the largest.

## 8.5 Relationship to Logistic Regression

There are similarities between SVM and other classical methods. It turns out the criterion can be written as,

$$minimize_{\beta_0,\beta_1,...,\beta_p}\{\sum_{i=1}^{n}\max[0, 1 - y_i f(x_i)] + \lambda\sum_{j=1}^{p}\beta_j^2\}$$

where $\lambda$ is a nonnegative tuning parameter. When $\lambda$ is large, $\beta_1, ..., \beta_p$ are small with more violations to the margin. When $\lambda$ is small, few violations occur thus high-variance, but low bias.

Note that $\lambda\sum_{j=1}^{p}\beta_j^2$ is the ridge penalty term. The loss and penalty functions between logistic and SVM are very similar.

When the classes are well separated, SVMs tend to behave better than logistic regression; in more overlapping regimes, logistic regression is often preferred.

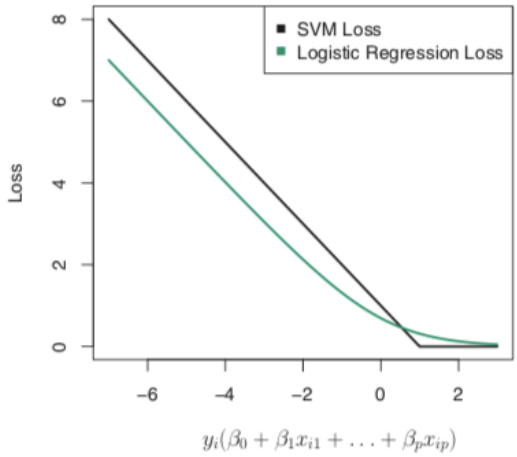


**FIGURE 9.12.** *The SVM and logistic regression loss functions are compared, as a function of* $y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip})$. *When* $y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip})$ *is greater than 1, then the SVM loss is zero, since this corresponds to an observation that is on the correct side of the margin. Overall, the two loss functions have quite similar behavior.*

We can just as well perform logistic regression or other classifications using a non-linear kernel; however, the use of nonlinear kernels in SVM's is more widespread.

**Support Vector Regression**: Extended SVM to quantitative rather than classification.

Support vector regression instead seeks coefficients that minimize a different type of loss, where only residuals larger in absolute value than some positive constant contribute to the loss function.

# 9 Unsupervised Learning

In unsupervised learning, the goal is to discover interesting things about the measurements on $X_1, X_2, ..., X_p$. Is there an informative way to visualize the data? Can we discover subgroups among the variables or among the observations?

## 9.1 The Challenge of Unsupervised Learning

Unsupervised learning is often performed as part of an exploratory data analysis. However, in unsupervised learning, there is no way to check our work because we don't know the true answer—the problem is unsupervised.

## 9.2 Principal Components Analysis

When faced with a large set of correlated variables, principal components allow us to summarize this set with a smaller number of representative variables that collectively explain most of the variability in the original set. These directions also define lines and subspaces that are as close as possible to the data cloud.

Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization (visualization of the observations or visualization of the variables).

### 9.2.1 What Are Principal Components?

We can start EDA by looking at scatterplots, but with many variables the number of scatter plots produced becomes large. A better method is to visualize $n$ observations when $p$ is large. In particular, we would like to find a low-dimensional representation of the data that captures as much of the information as possible. For instance, if we can obtain a two-dimensional representation of the data that captures most of the information, then we can plot the observations in this low-dimensional space.

PCA finds a low-dimensional representation of a data set that contains as much as possible of the variation. It seeks a small number of dimensions that are as interesting as possible, where the concept of interesting is measured by the amount that the observations vary along each dimension. The dimensions are a linear combination of $p$ features.

The first principal component is the normalized linear combination of the features,

$$Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + ... + \phi_{p1}X_p$$

that has the largest variance. $\phi$ are the loadings where together they make up the principal component loading vector $\phi_1 = (\phi_{11}\phi_{21}...\phi_{p1}$. The loadings are constrained so their sum of squares is equal to one.

The first principal component loading vector solves the optimization problem,

$$maximize_{\phi_{11},...,\phi_{p1}}\{\frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{p}\phi_{j1}x_{ij})^2\} s.t. \sum_{j=1}^{p}\phi_{j1}^2 = 1$$

The objective is to maximize the sample variance of the $n$ values of $z_{i1}$. $z_{11}, ..., z_{n1}$ are the scores of the first principal component.

The loading vectors $\phi$ define a direction in the feature space along which the data vary the most.

To calculate the second principal components $Z_2$ it is the linear combinations of $X_1, ..., X_p$ that has maximal variance out of all linear combinations that are uncorrelated with $Z_1$. The second principal component is uncorrelated with the first PC.

Once the PC are calculated, they can be plotted against each other to produce low-dimensional views of the data using a biplot.

In the plot below, the PC scores and loading vectors are displayed in a single plot. Equal weight for Rape, Assault, and Murder are displayed for the first PC; thus, this component corresponds to a measure of overall rates of serious crimes. The second PC places more weight on UrbanPop; thus, this component roughly corresponds to level of urbanization of the state. Further, Murder, Assault, and Rape are located class to one another, thus correlated. UrbanPop is less correlated with the other three. Additionally, states with large positive scores on the first PC, such as California, Nevada, and Florida, have high crime rates while North Dakota how low crime rates. California has a high score on the second PC which indicates a high level of urbanization while the opposite is true for Mississippi. States close to zero on both components, such as Indiana, have approx. average levels of both crime and urbanization.

Take-aways from biplot:

- Distinguishes which variables correspond most to each PC.
- Correlation between variables; those closest and in direction are most correlated
- Groups (states) that relate to rates of variables.

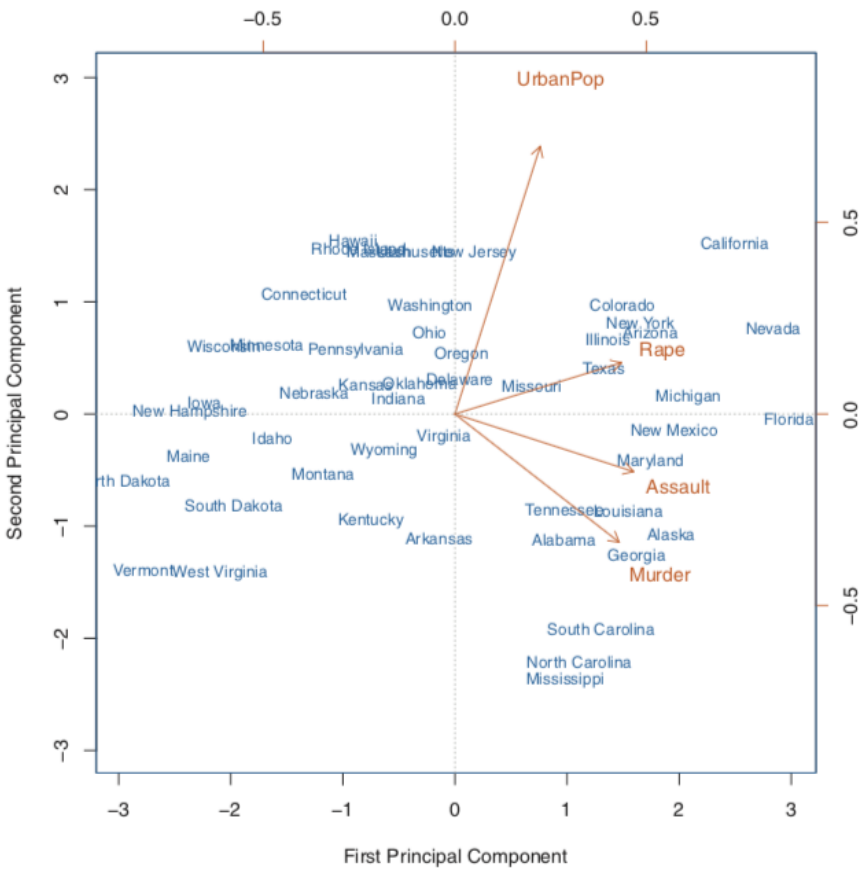


**FIGURE 10.1.** *The first two principal components for the* `USArrests` *data. The blue state names represent the scores for the first two principal components. The orange arrows indicate the first two principal component loading vectors (with axes on the top and right). For example, the loading for* `Rape` *on the first component is* 0.54*, and its loading on the second principal component* 0.17 *(the word* `Rape` *is centered at the point* $(0.54, 0.17)$*). This figure is known as a biplot, because it displays both the principal component scores and the principal component loadings.*

### 9.2.2 Another Interpretation of Principal Components

An alternative interpretation for principal components can also be useful: principal components provide low-dimensional linear surfaces that are closest to the observations.

**Note:** The first principal component loading vector has a very special property: it is the line in p-dimensional space that is closest to the n observations (using average squared Euclidean distance as a measure of closeness).

The appeal of this interpretation is clear: we seek a single dimension of the data that lies as close as possible to all of the data points, since such a line will likely provide a good summary of the data.

Together the first M principal component score vectors and the first M principal component loading vectors provide the best M-dimensional approximation (in terms of Euclidean distance) to the $i$th observation $x_{ij}$.

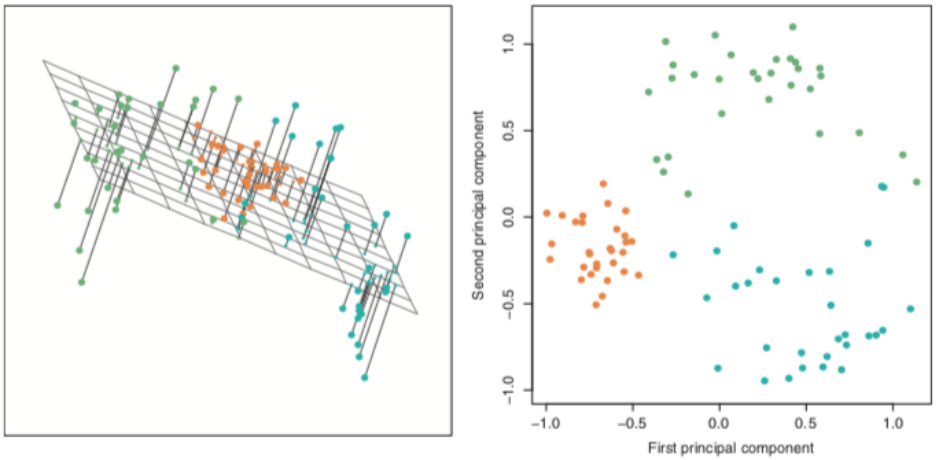


**FIGURE 10.2.** *Ninety observations simulated in three dimensions.* Left: *the first two principal component directions span the plane that best fits the data. It minimizes the sum of squared distances from each point to the plane.* Right: *the first two principal component score vectors give the coordinates of the projection of the 90 observations onto the plane. The variance in the plane is maximized.*

### 9.2.3 More on PCA

**Scaling the Variables** We have already mentioned that before PCA is performed, the variables should be centered to have mean zero. Furthermore, the results obtained when we perform PCA will also depend on whether the variables have been individually scaled (each multiplied by a different constant). Because it is undesirable for the principal components obtained to depend on an arbitrary choice of scaling, we typically scale each variable to have standard deviation one before we perform PCA.

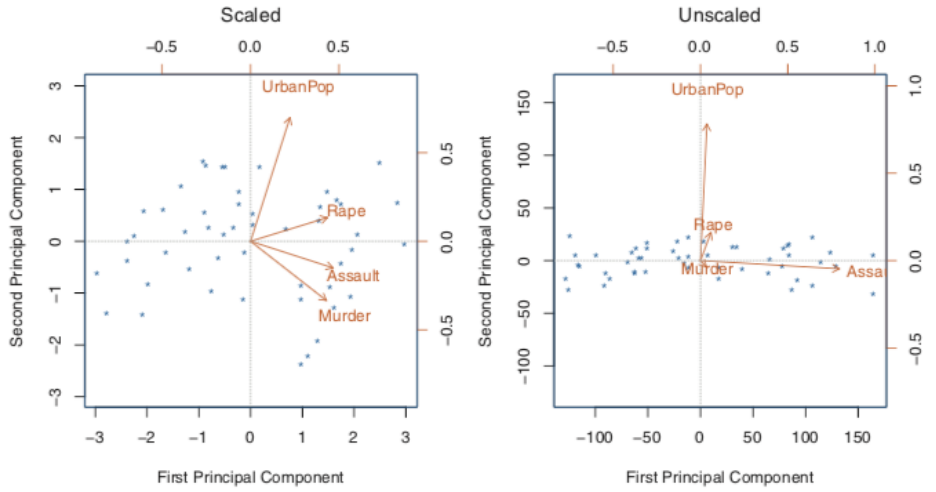


**FIGURE 10.3.** *Two principal component biplots for the* `USArrests` *data.* Left: *the same as Figure 10.1, with the variables scaled to have unit standard deviations.* Right: *principal components using unscaled data.* `Assault` *has by far the largest loading on the first principal component because it has the highest variance among the four variables. In general, scaling the variables to have standard deviation one is recommended.*

In certain settings, however, the variables may be measured in the same units. In this case, we might not wish to scale the variables to have standard deviation one before performing PCA. For instance, suppose that the variables in a given data set correspond to expression levels for p genes. Then since expression is measured in the same "units" for each gene, we might choose not to scale the genes to each have standard deviation one.

**Uniqueness of the Principal Components** Each principal component loading vector is unique, up to a sign flip. Similarly, the score vectors are unique up to a sign flip, since the variance of $Z$ is the same as the variance of $-Z$.

**The Proportion of Variance Explained** How much of the variance in the data is not contained in the first few principal components? More generally, we are interested in knowing the proportion of variance explained (PVE) by each principal component.

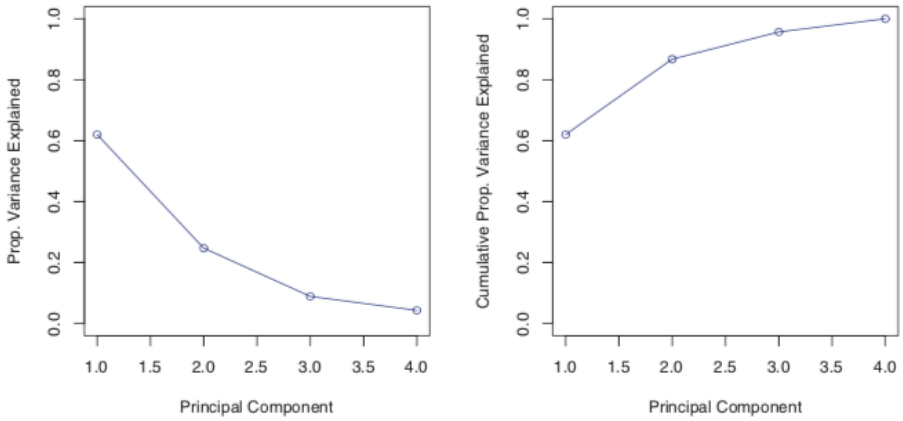


**FIGURE 10.4.** Left: *a scree plot depicting the proportion of variance explained by each of the four principal components in the* `USArrests` *data.* Right: *the cumulative proportion of variance explained by the four principal components in the* `USArrests` *data.*

**Deciding How Many Principal Components to Use** We would like to use just the first few principal components in order to visualize or interpret the data. In fact, we would like to use the smallest number of principal components required to get a good understanding of the data. We typically decide on the number of principal components required to visualize the data by examining a scree plot,

The number of PC to use is determined by looking for a point at which the proportion of variance explained by each subsequent principal component drops off. This is known as an elbow.

if we compute principal components for use in a supervised analysis, such as the principal components regression presented in Section 6.3.1, then there is a simple and objective way to determine how many principal components to use: we can treat the number of principal component score vectors to be used in the regression as a tuning parameter to be selected via cross-validation or a related approach.

### 9.2.4 Other Uses for Principal Components

Many statistical techniques, such as regression, classification, and clustering, can be easily adapted to use the $nM$ matrix whose columns are the first $M << p$ principal component score

vectors, rather than using the full $np$ data matrix. This can lead to less noisy results, since it is often the case that the signal (as opposed to the noise) in a data set is concentrated in its first few principal components.

## 9.3 Clustering Methods

Clustering refers to a very broad set of techniques for finding subgroups, or clusters, in a data set. When we cluster the observations of a data set, we seek to partition them into distinct groups so that the observations within each group are quite similar to each other, while observations in different groups are quite different from each other.

Differences between PCA and Clustering:

- PCA looks to find a low-dimensional representation of the observations that explain a good fraction of the variance.
- Clustering looks to find homogenous subgroups among the observations.

K-means clustering, we seek to partition the observations into a pre-specified number of clusters. On the other hand, in hierarchical clustering, we do not know in advance how many clusters we want; in fact, we end up with a tree-like visual representation of the observations, called a dendrogram.

In general, we can cluster observations on the basis of the features in order to identify subgroups among the observations, or we can cluster features on the basis of the observations in order to discover subgroups among the features.

### 9.3.1 K Means Clustering

K-means clustering is a simple and elegant approach for partitioning a data set into K distinct, non-overlapping clusters. The idea behind K-means clustering is that a good clustering is one for which the within-cluster variation is as small as possible. The goal of K-means clustering is to minimum the clusters such that the total within-cluster variation, summed over all $K$ clusters, is as small as possible,

$$minimize_{C_1,\ldots,C_K}\{\sum_{k=1}^{K} W(C_K)\}$$

In words, this formula says that we want to partition the observations into K clusters such that the total within-cluster variation, summed over all K clusters, is as small as possible. To solve, we need to define the within-cluster variation. This can be accomplished with squared Euclidean distance,

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

where $|C_k|$ denotes the number of obs in the kth cluster. In other words, the within-cluster variation for the kth cluster is the sum of all of the pairwise squared Euclidean distances between the observations in the kth cluster, divided by the total number of observations in the kth cluster.

The optimization problem is then defined as,

$$minimize_{C_1,\ldots,C_K}\{\sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2\}$$

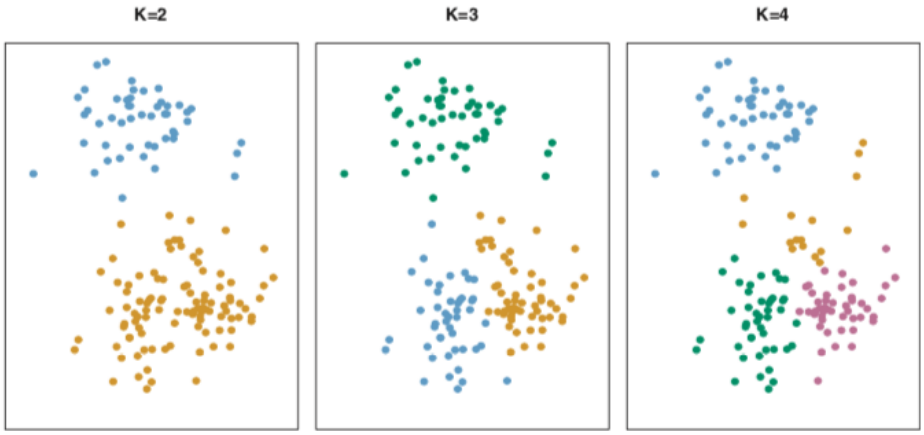


**FIGURE 10.5.** *A simulated data set with 150 observations in two-dimensional space. Panels show the results of applying K-means clustering with different values of K, the number of clusters. The color of each observation indicates the cluster to which it was assigned using the K-means clustering algorithm. Note that there is no ordering of the clusters, so the cluster coloring is arbitrary. These cluster labels were not used in clustering; instead, they are the outputs of the clustering procedure.*

---

**Algorithm 10.1** *K-Means Clustering*

---

1. Randomly assign a number, from 1 to $K$, to each of the observations. These serve as initial cluster assignments for the observations.

2. Iterate until the cluster assignments stop changing:

   (a) For each of the $K$ clusters, compute the cluster *centroid*. The $k$th cluster centroid is the vector of the $p$ feature means for the observations in the $k$th cluster.

   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

---

Because the K-means algorithm finds a local rather than a global optimum, the results obtained will depend on the initial (random) cluster assignment of each observation. For this reason, it is important to run the algorithm multiple times from different random and choose one for which the optimization problem is the smallest.

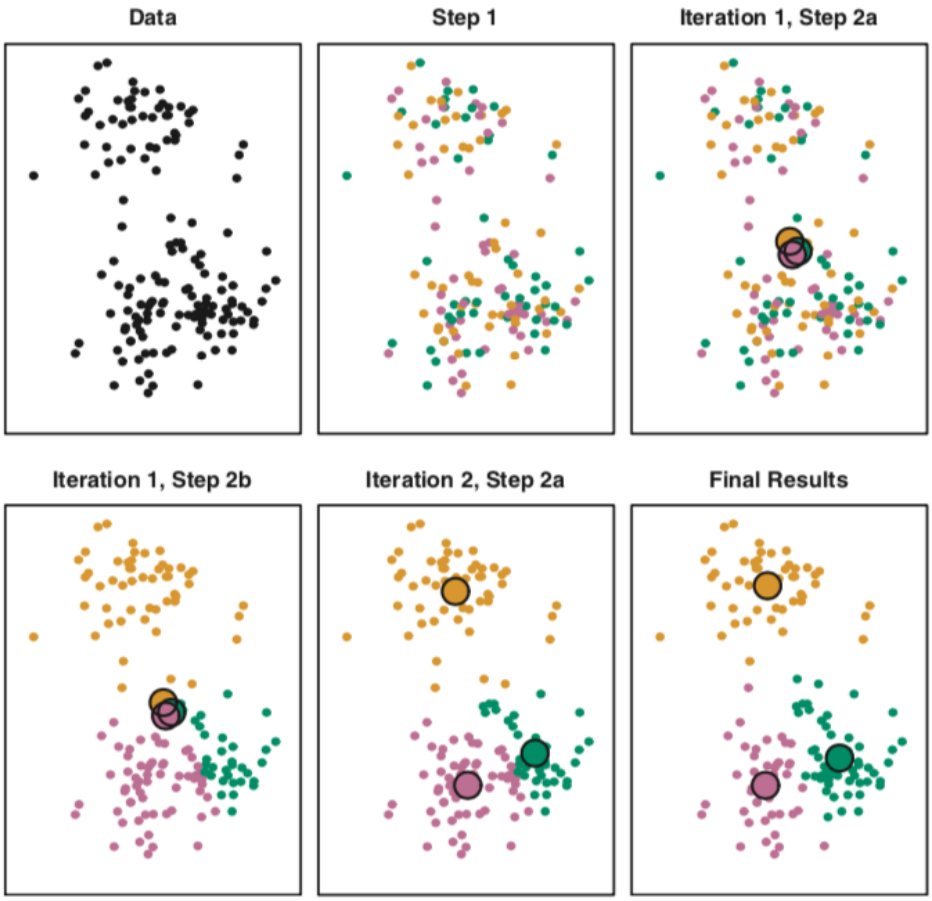


**FIGURE 10.6.** *The progress of the K-means algorithm on the example of Figure 10.5 with K=3.* Top left: *the observations are shown.* Top center: *in Step 1 of the algorithm, each observation is randomly assigned to a cluster.* Top right: *in Step 2(a), the cluster centroids are computed. These are shown as large colored disks. Initially the centroids are almost completely overlapping because the initial cluster assignments were chosen at random.* Bottom left: *in Step 2(b), each observation is assigned to the nearest centroid.* Bottom center: *Step 2(a) is once again performed, leading to new cluster centroids.* Bottom right: *the results obtained after ten iterations.*

### 9.3.2 Hierarchical Clustering

One potential disadvantage of K-means clustering is that it requires us to pre-specify the number of clusters K. Hierarchical clustering is an alternative approach which does not require that we commit to a particular choice of K. Hierarchical clustering has an added advantage over K-means clustering in that it results in an attractive tree-based representation of the observations, called a dendrogram.

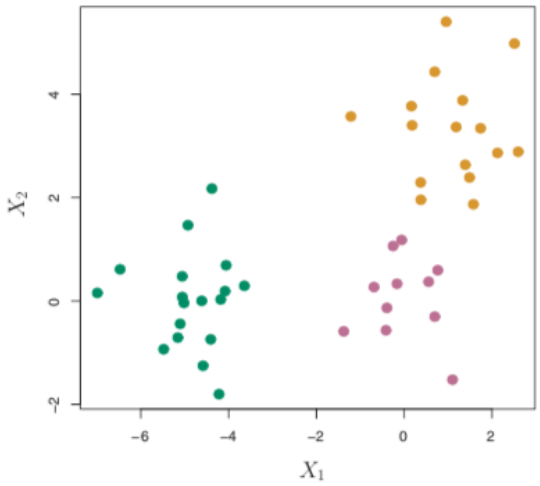


**FIGURE 10.8.** *Forty-five observations generated in two-dimensional space. In reality there are three distinct classes, shown in separate colors. However, we will treat these class labels as unknown and will seek to cluster the observations in order to discover the classes from the data.*

The main difference between Hierarchical Clustering and K-Means Clustering is the former utilizes a dendrogram that is built starting from the leaves and combining clusters up to the trunk.

**Interpreting a Dendrogram** Suppose the data are observed without class labels. In a dendrogram, each leaf represents a single observation. Branches are leaves that have fused because the observations are similar to each other.

**Note:** We look for points in the tree where branches contain those two observations are first fused. The height represents how different the two observations are. Thus, observations that fuse at the very bottom of the tree are quite similar to each other, whereas observations that fuse close to the top of the tree will tend to be quite different.

**Interpreting Dendrogram:** we cannot draw conclusions about the similarity of two observations based on their proximity along the horizontal axis. Rather, we draw conclusions about the similarity of two observations based on the location on the vertical axis where branches containing those two observations first are fused.

The term hierarchical refers to the fact that clusters obtained by cutting the dendrogram at a given height are necessarily nested within the clusters obtained by cutting the dendrogram at any greater height. However, on an arbitrary data set, this assumption of hierarchical structure might be unrealistic.

A very attractive aspect of hierarchical clustering: one single dendrogram can be used to obtain any number of clusters.

A problem arises when using hierarchical clustering when the split is greater than the realistic split, such as dividing men and women into three groups. Therefore, in this situation, hierarchical clustering can sometimes yield worse results than K-means clustering.

**The Hierarchical Clustering Algorithm** The hierarchical clustering dendrogram is obtained via an extremely simple algorithm. We begin by defining some sort of dissimilarity measure between each pair of observations. Most often, Euclidean distance is used; we will discuss the choice of dissimilarity measure later in this chapter.

**Algorithm 10.2** *Hierarchical Clustering*

1. Begin with $n$ observations and a measure (such as Euclidean distance) of all the $\binom{n}{2} = n(n-1)/2$ pairwise dissimilarities. Treat each observation as its own cluster.

2. For $i = n, n-1, \ldots, 2$:

   (a) Examine all pairwise inter-cluster dissimilarities among the $i$ clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.

   (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters.

| *Linkage* | *Description* |
|---|---|
| Complete | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *largest* of these dissimilarities. |
| Single | Minimal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *smallest* of these dissimilarities. Single linkage can result in extended, trailing clusters in which single observations are fused one-at-a-time. |
| Average | Mean intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *average* of these dissimilarities. |
| Centroid | Dissimilarity between the centroid for cluster A (a mean vector of length $p$) and the centroid for cluster B. Centroid linkage can result in undesirable *inversions*. |

**TABLE 10.2.** *A summary of the four most commonly-used types of linkage in hierarchical clustering.*

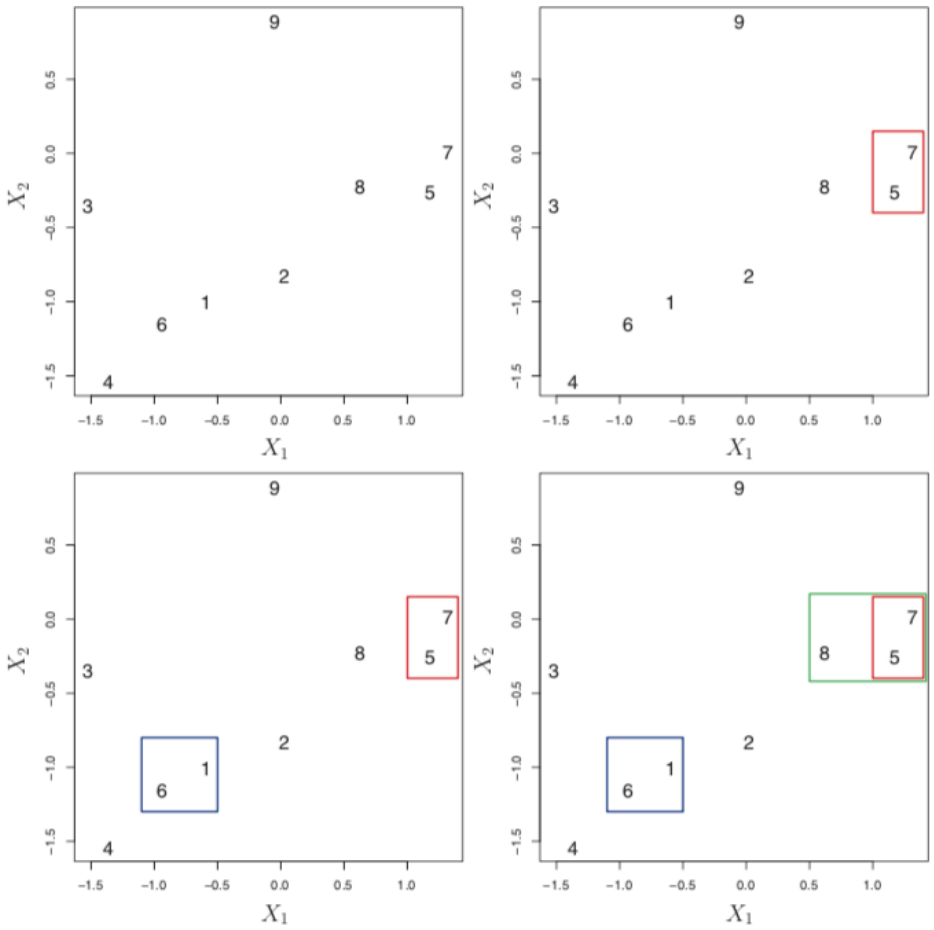


**FIGURE 10.11.** *An illustration of the first few steps of the hierarchical clustering algorithm, using the data from Figure 10.10, with complete linkage and Euclidean distance.* Top Left: *initially, there are nine distinct clusters,* $\{1\}, \{2\}, \ldots, \{9\}$. Top Right: *the two clusters that are closest together,* $\{5\}$ *and* $\{7\}$*, are fused into a single cluster.* Bottom Left: *the two clusters that are closest together,* $\{6\}$ *and* $\{1\}$*, are fused into a single cluster.* Bottom Right: *the two clusters that are closest together using* complete linkage*,* $\{8\}$ *and the cluster* $\{5, 7\}$*, are fused into a single cluster.*

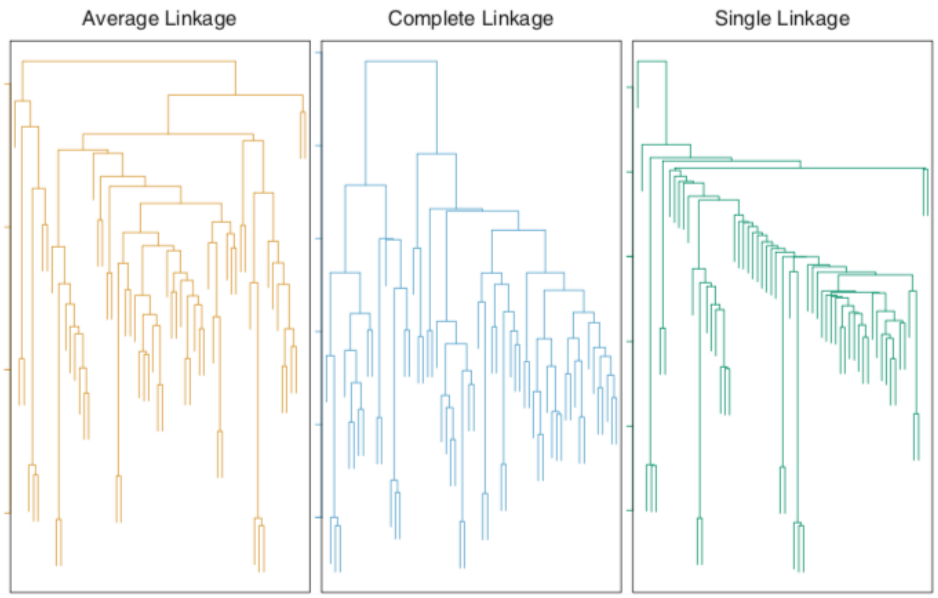


**FIGURE 10.12.** *Average, complete, and single linkage applied to an example data set. Average and complete linkage tend to yield more balanced clusters.*

**Choice of Dissimilarity Measure** Thus far, the examples in this chapter have used Euclidean distance as the dissimilarity measure. But sometimes other dissimilarity measures might be preferred. For example, correlation-based distance considers two observations to be similar if their features are highly correlated, even though the observed values may be far apart in terms of Euclidean distance.

The choice of dissimilarity measure is very important, as it has a strong effect on the resulting dendrogram. In general, careful attention should be paid to the type of data being clustered and the scientific question at hand. These considerations should determine what type of dissimilarity measure is used for hierarchical clustering.

In addition to carefully selecting the dissimilarity measure used, one must also consider whether or not the variables should be scaled to have standard deviation one before the dissimilarity between the observations is computed.

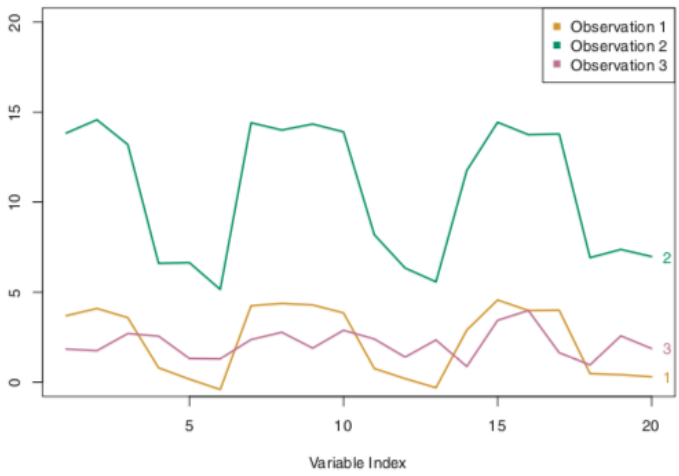


**FIGURE 10.13.** *Three observations with measurements on 20 variables are shown. Observations 1 and 3 have similar values for each variable and so there is a small Euclidean distance between them. But they are very weakly correlated, so they have a large correlation-based distance. On the other hand, observations 1 and 2 have quite different values for each variable, and so there is a large Euclidean distance between them. But they are highly correlated, so there is a small correlation-based distance between them.*

### 9.3.3 Practical Issues in Clustering

There are a number of issues that arise in performing clustering:

#### Small Decisions with Big Consequences

- Should obs or features be standardized?
- Hierarchical clustering: (1) Choice of dissimilarity measure? (2) Type of linkage? (3) Where should dendrogram be cut
- K-means clustering: how many clusters should we look for in the data?

Each of these decisions can have a strong impact on the results obtained. In practice, we try several different choices, and look for the one with the most useful or interpretable solution. With these methods, there is no single right answer—any solution that exposes some interesting aspects of the data should be considered.

**Validating the Clusters Obtained** Any time clustering is performed on a data set we will find clusters. But we really want to know whether the clusters that have been found represent true subgroups in the data, or whether they are simply a result of *clustering the noise*. Techniques exist for assigning a p-value to a cluster, but no consensus has been reached.

**Other Considerations in Clustering** Since K- means and hierarchical clustering force every observation into a cluster, the clusters found may be heavily distorted due to the presence of outliers that do not belong to any cluster. Mixture models are an attractive approach for accommodating the presence of such outliers. Such as soft version of K-means clustering.

Clustering methods generally are not very robust to instability to the data.

**A Tempered Approach to Interpreting the Results of Clustering** We mentioned that small decisions in how clustering is performed, such as how the data are standardized and what type of linkage is used, can have a large effect on the results. Therefore, we recommend performing clustering with different choices of these parameters, and looking at the full set of results in order to see what patterns consistently emerge. Since clustering can be non-robust, we recommend clustering subsets of the data in order to get a sense of the robustness of the clusters obtained. Most importantly, we must be careful about how the results of a clustering analysis are reported. These results should not be taken as the absolute truth about a data set. Rather, they should constitute a starting point for the development of a scientific hypothesis and further study, preferably on an independent data set.