SD 7847: Machine Learning for Statistics
An Outline of "An Introduction to Statistical Learning" with Applications in R (ISLR)

Darshan Patel

Spring 2019

Contents

1 Linear Regression

1.1 Simple Linear Regression

- Simple linear regression is a straightforward approach for predicting a quantitative response Y on the basis of a single predictor variable X
- It assumes that there is approximately a linear relationship between X and Y

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

- β_0 and β_1 are two unknown constants that represent the intercept and slope terms in the linear model; together, β_0 and β_1 are known as the model coefficients or parameters
- The most common approach to minimizing closeness involves minimizing the least squares criterion
- Let $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$ be the prediction for Y based on the ith value of X; then $e_i = y_i - \hat{y}_i$ represents the ith residual - the difference between the ith observed response value and the ith response value that is predicted by the linear model
- Residual Sum of Squares (RSS)

$$\text{RSS} = e_1^2 + e_2^2 + \cdots + e_n^2$$

or equivalently as

$$\text{RSS} = (y_1 - \hat{\beta}_{00} - \hat{\beta}_1 x_1)^2 + (y_2 - \hat{\beta}_0 - \hat{\beta}_1 x_2)^2 + \cdots + (y_n - \hat{\beta}_0 - \hat{\beta}_1 x_n)^2$$

- The least squares approach chooses $\hat{\beta}_0$ and $\hat{\beta}_1$ to minimize the RSS
- It can be shown that the minimizers are

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$
$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

  where $\bar{y} \equiv \frac{1}{n}\sum_{i=1}^{n} y_i$ and $\bar{x} \equiv \frac{1}{n}\sum_{i=1}^{n} x_i$ are the sample means
- The above equations for $\hat{\beta}_0$ and $\hat{\beta}_1$ defines the least squares coefficient estimates for simple linear regression
- Assume that the true relationship between X and Y takes the form $Y = f(X) + \varepsilon$ for some unknown function f, where ε is a mean-zero random error term; if f is to be estimated by a linear function, then

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

- The error term is a catch-all for what is missed by the simple model: the true relationship is probably not linear, there may be other variables that cause variation in Y and there may be measurement error
- It is assumed that the error term is independent of X
- The population regression line is the best linear approximation to the true relationship between X and Y; the least squares regression coefficient estimates characterize the least squares line
- The difference between the population regression line and the least squares line is subtle; the unknown coefficients β_0 and β_1 in linear regression define the population regression line and we seek to estimate these unknown coefficients using $\hat{\beta}_0$ and $\hat{\beta}_1$ - these coefficient estimates define the least squares line
- An unbiased estimators does not systematical over- or under-estimate the true parameter
- If β_0 and β_1 are estimated based on a particular dataset, then the estimates won't be exactly equal to β_0 and β_1; but if the estimates are averaged over a large number of data points, then the averages of the estimates would be close to the true coefficients
- To measure how far off a single estimate of $\hat{\mu}$ would be, compute the standard error of $\hat{\mu}$, or $\mathrm{SE}(\hat{\mu})$:

$$\mathrm{Var}[\hat{\mu}] = \mathrm{SE}[\hat{\mu}]^2 = \frac{\sigma^2}{n}$$

  where σ is the standard deviation of each of the realizations y_i of Y

- The standard error tells the average deviation of the estimate $\hat{\mu}$ from the actual value of μ; it also tells how the deviations shrinks with n (the more observations, the smaller the standard error)
- Likewise, the standard errors associated with $\hat{\beta}_0$ and $\hat{\beta}_1$ are

$$\text{SE}[\hat{\beta}_0]^2 = \sigma^2 \left[\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \right]$$
$$\text{SE}[\hat{\beta}_1]^2 = \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

  where $\sigma^2 = \text{Var}[\varepsilon]$
- $\text{SE}[\hat{\beta}_1]$ is smaller when the x_i are more spread out; this gives more leverage to estimating the slope
- $\text{SE}[\hat{\beta}_0]$ would be the same as $\text{SE}[\hat{\mu}]$ if $\bar{x}$ were zero, and so $\hat{\beta}_0$ would be equal to $\bar{y}$
- Residual Standard Error - the error of σ^2

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

- Standard errors can be used to compute confidence intervals
- A 95% confidence interval is a range of values such that with 95% probability, the true unknown value of the parameter is captured within the range
- For linear regression, the 95% confidence interval for β_1 is

$$\hat{\beta}_1 \pm 2 \cdot \text{SE}[\hat{\beta}_1]$$

  or, there is a 95% probability that the interval

$$[\hat{\beta}_1 - 1 \cdot \text{SE}[\hat{\beta}_1], \hat{\beta}_1 + 2 \cdot \text{SE}[\hat{\beta}_1]]$$

  will contain the true value of β_1
- Similarly, a confidence interval for β_0 is of the form

$$\hat{\beta}_0 \pm 2 \cdot \text{SE}[\hat{\beta}_0]$$

- Standard errors can be used to perform hypothesis tests on the coefficients
- Null Hypothesis:

$$H_0 : \text{ There is no relationship between } X \text{ and } Y$$

  or

$$H_0 : \beta_1 = 0$$

- Alternative Hypothesis:

$$H_A: \text{ There is some relationship between } X \text{ and } Y$$

or

$$H_A : \beta_1 \neq 0$$

- If $\beta_1 = 0$, then the model reduces to $Y = \beta_0 + \varepsilon$ and X is not associated with Y
- To test the null hypothesis, determine whether $\hat{\beta}_1$ is sufficiently far from zero - this depends on the accuracy of $\hat{\beta}_1$
- If $\text{SE}[\hat{\beta}_1]$ is small, then relatively small values of $\hat{\beta}_1$ will show evidence that $\beta_1 \neq 0$ and so there is a relationship between X and Y; if $\text{SE}[\hat{\beta}_1]$ is large, hen $\hat{\beta}_1$ is large and so the null hypothesis is rejected
- To measure the number of standard deviations that $\hat{\beta}_1$ is away from 0, compute the t-statistic

$$t = \frac{\hat{\beta}_1 - 0}{\text{SE}[\hat{\beta}_1]}$$

- If there is no relationship between X and Y, then the above statistics will have a t-distribution with $n - 2$ degrees of freedom
- The p-value is the probability of observing any value equal to $|t|$ or larger, assuming $\beta_1 = 0$
- A small p-value indicates that there is no association between the predictor and the response due to chance, in the absence of any real association between the two and so there is there an association between X and Y and thus the null hypothesis is rejected
- Typical p-value cutoffs for rejecting the null hypothesis are 5% and 1%
- The quality of a linear regression fit is assessed using the residual standard error (RSE) and the R^2 statistic
- The RSE is an estimate of the standard deviation of ε; it is the average amount that the response will deviate from the true regression line

$$RSE = \sqrt{\frac{\text{RSS}}{n-2}} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-2}}$$

- The RSE measures the lack of fit of the model to the data; if the RSE is small, that means the predictions are very close to the true outcome values ($\hat{y}_i \approx y_i$ for $i = 1, \ldots, n$); if the RSE is large, then $\hat{y}_i$ is very far from y_i for some observations, indicating that the model does not fit the data well
- Since the RSE is measured in the units of Y, it is not always clear what is a good RSE value and so the R^2 statistic can help

- The R^2 statistic measures the proportion of variance explained and is thus bounded on $[0, 1]$ and is independent of the scale of Y

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

where $\text{TSS} = \sum(y_i - \bar{y})^2$ is the total sum of squares

- TSS measures the total variance in Y, before the regression is performed, and the RSS measures the amount of variability that is left unexplained after performing the regression
- Thus R^2 measures the proportion of variability in Y that can be explained using X
- An R^2 value close to 1 means that a large proportion of the variability in the response has been explained by the regression; a value close to 0 means otherwise
- If R^2 is close to 0, it could mean that the linear model is wrong or the error σ^2 is high
- Recall that correlation, defined as

$$r = \text{Cor}[X, Y] = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

is also a measure of the linear relationship between X and Y

- It can be shown that $R^2 = r^2$ for the simple linear regression setting but this is not true in the multiple linear regression setting where R^2 will have a greater meaning

1.2 Multiple Linear Regression

- Suppose there are p distinct predictors, then the multiple linear regression model takes the form

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon$$

where X_j represents the jth predictor and β_j quantifies the association between that variable and the response

- Interpret β_j as the average effect on Y of a one unit increase in X_j, holding all other predictors fixed
- To make predictions using the formula

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$

estimate the beta values by minimizing the sum of squared residuals

$$\begin{aligned}\text{RSS} &= \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \\ &= \sum_{i=1}^{n}(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \hat{\beta}_2 x_{i2} - \cdots - \hat{\beta}_p x_{ip})^2\end{aligned}$$

- The values $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ that minimize the RSE are the multiple least squares regression coefficient estimates
- Questions to Consider When Performing Multiple Linear Regression
  1. Is at least one of the predictors $X_1, X_2, \ldots, X_p$ useful in predicting the response?
  2. Do all the predictors help to explain Y, or is only a subset of the predictors useful?
  3. How well does the model fit the data?
  4. Given a set of predictor values, what response value should we predict, and how accurate is our prediction?
- In the multiple LR setting, a hypothesis test will ask whether all of the regression coefficients are zero; the null hypothesis is

  $$H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$$

  whereas the alternative hypothesis is

  $$H_A : \text{ at least one } \beta_j \text{ is non-zero}$$
- This hypothesis test is performed by computing the F-statistic

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

  where $\text{TSS} = \sum(y_i - \bar{y})^2$ and $\text{RSS} = \sum(y_i - \hat{y}_i)^2$
- If the linear model assumptions are correct, then it can be shown that

  $$E\left[\frac{\text{RSS}}{n-p-1}\right] = \sigma^2$$

  and that, provided H_0 is true,

  $$E\left[\frac{\text{TSS - RSS}}{p}\right] = \sigma^2$$
- When there is no relationship between the response and the predictors, then the F-statistic should be close to 1; if H_A is true, then $E\left[\frac{\text{TSS - RSS}}{p}\right] > \sigma^2$ and so F is greater than 1
- When n is large, an F-statistic that is just a slightly larger than 1 will provide evidence against H_0; a large F-statistic is needed to reject H_0 if n is small
- Sometimes, only a subset of q of the coefficients are tested to be equal to zero; here the null hypothesis is

  $$H_0 : \beta_{p-q+1} = \beta_{p-q+2} = \cdots = \beta_p = 0$$

  Here a new model is fitted that uses all of the variables except the ones in the null hypothesis

- The residual sum of squares for this model is RSS_0 and the appropriate F-statistic is

$$F = \frac{(\text{RSS}_0 - \text{RSS})/q}{\text{RSS}/(n-p-1)}$$

- When building a model as so, for each individual predictor a t-statistic and a p-value is reported; these provide information about whether each individual predictor is related to the response, after adjusting for the other predictors
- It reports the partial effect of adding that variable to the model
- It seems likely that if any one of the p-values for the individual variables is very small, then at least one of the predictors is related to the response; this logic is flawed, especially when the number of predictors p is large
- The approach of using an F-statistic to test for any association between the predictors and the response works when p is relatively small compared to n; if $p > n$, then there are more coefficients β_j to estimate than there are observations to estimate from
- When p is large, use forward selection or another higher-dimensional approach
- The task of determining which predictors are associated with the response is called variable selection
- Variable selection can be performed by brute force by creating 2^p models containing subsets of p variables; this is not feasible for large values of p variables
- Alternative Approaches for Variable Selection
  - Forward Selection: Beginning with the null model with no predictors, fit p simple linear regressions and add to the null model the variable that results in the lowest RSS; add the variable that results in the lowest RSS to the model for the new two-variable model; continue doing this until some stopping rule is fulfilled
  - Backward Selection: Begin with all the variables in the model and remove the variable with the largest p-value (the one that is least statistically significant); build a model from these $p-1$ variables and continue removing least significant variables until some threshold is fulfilled
  - Mixed Selection: Start with no variables in the mode and add the predictor that provides the best fit, one by one; if any any point the p-value for one of the variables in the model rises above a certain point, remove that variable from the model; continue doing this until all variables in the model have a low p-value and all variables outside the model would have a large p-value if added to the model
- Note: Backward selection cannot be used if $p > n$ while forward selection can always be used
- In simple regression, R^2 is the square of the correlation of the response and the variable; in multiple linear regression, it equals $\text{Cor}[Y.\hat{Y}]^2$, the square of the correlation between the response and the fitted linear model

- One property of the fitted linear model is that it maximizes this correlation among all possible linear models
- R^2 will always increase when more variables are added to the model, even if those variables are only weakly associated with the response
- Models with more variables can have higher RSE if the decrease in RSS is small relative to the increase in p

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

- In addition to looking at RSE and R^2 statistic, graphical summaries can reveal problems with a model that are not visible from numerical statistics
- Uncertainties associated with Predictions from a Multiple LR Model:
  - Reducible Error: the least squares plane

    $$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_p X_p$$

    is an estimate for the true population regression plane

    $$f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

    The inaccuracy in the coefficient estimate is related to reducible error and computing a confidence interval will show how close $\hat{Y}$ is to $f(X)$
  - Model Bias: assuming a linear model for $f(X)$ is an approximation of reality
  - Irreducible Error: the response value cannot be predicted perfectly because of random error ε in the model; use prediction intervals to determine how much Y varies from $\hat{Y}$ with the error in the estimate for $f(X)$

1.3 Other Considerations in the Regression Model

- When dealing with problems involving a qualitative predictor with two levels (two possible values), create an indicator that takes two possible values

$$x_i = \begin{cases} 1 & \text{if ...} \\ 0 & \text{if ...} \end{cases}$$

and use this variable as a predictor in the regression equation

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i = \begin{cases} \beta_0 + \beta_1 + \varepsilon_i & \text{if ...} \\ \beta_0 + \varepsilon_i & \text{if ...} \end{cases}$$

- Another choice of coding scheme is

$$x_i = \begin{cases} 1 & \text{if ...} \\ -1 & \text{if ...} \end{cases}$$

  The only difference this would make is how the coefficients are interpreted

- When dealing with problems involving a qualitative predictor with more than two levels, create several dummy variables to represent all possible values where each variable represents either being a certain value or not

- In this case, the level with no dummy variable is known as the baseline; there will always be one fewer dummy variable than the number of levels; the choice of this baseline value is arbitrary

- Two of the most important assumptions of LR modeling is that the relationship between the predictors and response are additive and linear

  - The additive assumption means that the effect of changes in a predictor X_j on the response Y is independent of the values of the other predictors
  - The linear assumption stats that the change in the response Y due to a one unit change in X_j is constant, regardless of the value of X_j

- Synergy/Interaction Effect: when two (or more) predictor variables together impact the response Y

- To allow for interaction effects in the linear model, include interaction terms as follows

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

  This relaxes the additive assumption

$$\begin{aligned} Y &= \beta_0 + (\beta_1 + \beta_2 X_2) X_1 + \beta_2 X_2 + \varepsilon \\ &= \beta_0 + \tilde{\beta}_1 X_1 + \beta_2 X_2 + \varepsilon \end{aligned}$$

  where $\tilde{\beta}_1 = \beta_1 + \beta_3 X_2$l adjusting X_2 will change the impact of X_1 on Y

- Models that include interaction terms are superior to models that contain only the main effects; the p-value for the interaction term being low shows strong evidence for $H_A : \beta_3 \neq 0$, or that the true relationship is not additive

- Hierarchical Principle: if an interaction in a model is included, also include the main effects, even if the p-values associated with their coefficients are not significant

- To accommodate non-linear relationships, use polynomial regression to capture the true relationship between the response and the predictors

- A model such as
$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2^2 + \varepsilon$$
is still a linear model where X_2 is still numerical
- Including higher degrees of polynomials can allow for better fits to the model but lead to overfitting
- Problems with Fitting a LR Model to a Data Set
  - Non-linearity of the response-predictor relationships: if the true relationship between the predictors and the response is not linear, then all the conclusions drawn from the fit are suspect; use residual plots to identify non-linearity - if there is a pattern in the residual plot, there may be a problem in the linear model and so, consider using other non-linear transformations of the predictors in the regression model
  - Correlation of error terms: if the errors terms are correlated, then the estimated standard errors will underestimate the true standard errors and so confidence/prediction intervals will be narrower than they should be and so there is an unwarranted sense of confidence in the model - this in common in time series data, identify by plotting residuals against time
  - Non-constant variance of error terms: the variances of the error terms may increase with the value of the response; identify non-constant variances in the errors, or heteroscedasticity, from the presence of a funnel shape in the residual plot
  - Outliers: an outlier is a point for which y_i is far from the value predicted by the model; removing outliers can cause drastic decreases in the RSE and R^2 - use residual plots to identify outliers
  - High-leverage points: observations with high leverage have an unusual value for x_i; removing them can have a higher impact on the LS line than removing an outlier because it has a sizable impact on the estimated regression line - look for observations for which the predictor value is outside of the normal range of the observations in a simple linear regression but for multiple linear regression with multiple predictors, this is impossible to look at; to look at an observation's leverage, compute the leverage statistic
$$h_i = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{i'=1}^{n}(x_{i'} - \bar{x})^2}$$
for the simple linear regression case
  - Collinearity: the situation in which two or more predictor variables are closely related to one another; it can be difficult to separate out the individual effects of collinear variables on the response and reduces the accuracy of the estimates of the regression coefficients; it increases the standard error and so decreases the t-statistic and so the null hypothesis could be failed to reject, thus the power of the hypothesis test - the probability of correctly detecting a non-zero coefficient;

is reduced by collinearity - to identify collinearity, look at the correlation matrix of the predictors; for identifying collinearity between three or more variables (multicollinearity), compute the variable inflation factor - the ratio of the variance of $\hat{\beta}_j$ when fitting the full model divided by the variance of $\hat{\beta}_j$ if fit on its own

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

for each variable, where $R^2_{X_j|X_{-j}}$ is the $R * 2$ from a regression of X_j onto all of the other predictors; if this R^2 value is close to one, then collinearity is present and the VIF value will be large; a VIF value that is greater than 5 or 10 is usually associated with collinearity

1.4 Comparison of Linear Regression with K-Nearest Neighbors

- Linear regression is an example of a parametric approach because it assumes a linear functional form for $f(X)$; non-parametric methods do not explicitly assume a parametric form for $f(X)$ and thus are more flexible for preforming regression
- One non-parametric method is K-nearest neighbors regression (KNN regression)
- Given a value for K and a prediction point x_0, KNN regression first identifies the K training observations that are closest to x_0, represented by N_0; it then estimates $f(x_0)$ using the average of all the training responses in N_0

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

- The optimal value for K will depend on the bias-variance tradeoff; a small value for K provides the most flexible fit with low bias but high variance due to the prediction in a given region is entirely dependent on just one observation; larger values of K provide a smoother and less variable fit with high bias and low variance due to smoothing by several points and so masking some of the structure in $f(X)$
- 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
- Curse of Dimensionality: when p is large and thus produces no clear consensus on the prediction of $f(x_0)$
- As a general rule, parametric methods will tend to outperform non-parametric approaches when there is a small number of observations per predictor

2 Classification

2.1 An Overview of Classification

- Predicting a qualitative response for an observation can be referred to as classifying that observation, since it involves assigning the observation to a category, or class
- Three of the most widely-used classifiers are logistic regression, linear discriminant analysis and K-nearest neighbors
- In the classification setting, there is a set of training observations $(x_1, y_1), \ldots, (x_n, y_n)$ that is used to build a classifier
- The classifier should perform well on the training data as well as on test observations that were not used to train the classifier

2.2 Why Not Linear Regression

- Linear regression is not appropriate in the case of a qualitative response
- This is because it implies an ordering on the outcomes when the outcomes are dummified for linear regression
- If the response variable's values did take on a natural ordering, then a $1, 2, 3$ coding would be reasonable
- In general, there is no natural way to convert a qualitative response variable with more than two levels into a quantitative response that is ready for linear regression
- For a binary response with a 0/1 coding, regression by least squares does make sense

2.3 Logistic Regression

- Logistic regression models the probability that Y belongs to a particular category
- To avoid the problem of having probability less than 0 and greater than 1, logistic regression uses the logistic function to give outputs between 0 and 1

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

- To fit this model, use maximum likelihood
- With some manipulation, it can be seen that

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

  This quantity is called the odds and can take on any value between 0 and ∞

- Values of the odds close to 0 indicate very low probabilities while values close to ∞ indicate very high probabilities
- By taking the logarithm of both side,

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$$

  This is called the log-odds, or logit. What this shows is that, in a logistic regression model, increasing X by one unit changes the log odds by β_1, or equivalently it multiplies the odds by e^{β_1}
- Since the relationship between $p(X)$ and X is not linear, β_1 does not correspond to the change in $p(X)$ associated with a one-unit increase in X; the amount that $p(X)$ changes due to a one-unit change in X will depend on the current value of X
- If β_1 is positive then increasing X will be associated with increasing $p(X)$ and if β_1 is negative, then increasing X will be associated with decreasing $p(X)$
- To find $\hat{\beta}_0$ and $\hat{\beta}_1$ such that plugging these estimates into the model for $p(X)$ yields a number close to 1 or 0 can be formalized using a a likelihood function

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

  The estimates $\hat{\beta}_0$ and $\hat{\beta}_1$ are chosen to maximize this likelihood function.
- To measure the accuracy of the coefficient estimates, use the z-statistic calculated and its respective p-value and interpret similarly as done with linear regression
- A large (absolute) value of the z-statistic indicates evidence against the null hypothesis

$$H_0 : \beta_1 = 0$$

  This null hypothesis indicates that

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

  The estimated intercept is typically not of interest; its main purpose is to adjust the average fitted probabilities to the proportion of ones in the data
- To make predictions, calculate $\hat{p}(X)$

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

- To fit qualitative predictors of two levels, create a dummy variable that takes on 0/1 values which can then be inputted into the calculation for $\hat{p}(X)$

- In multiple logistic regression, the model becomes

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

  where $X = (X_1, \ldots, X_p)$ are p predictors

- The equation for $p(X)$ becomes

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

  where the maximum likelihood method is used to estimate $\beta_0, \beta_1, \ldots, \beta_p$

- The results obtained using one predictor may be different rom those obtained using multiple predictors, especially when there is correlation among the predictors

- For classifying a response variable with more than two classes, use discriminant analysis, although it is possible to do so with logistic regression except it is not used oftenly

2.4 Linear Discriminant Analysis

- Logistic regression involves directly modeling $P(Y = k \mid X = x)$ using the logistic function for the case of two response classes; this means it models the conditional distribution of the response Y given the predictor(s) X

- In an alternative approach, try to model the distribution of the predictors X separately in each of the response classes (given Y) and then use Bayes' theorem to flip those into estimates for $P(Y = k \mid X = x)$

- When the classes are well-separated, the parameter estimates for the logistic regression model are surprisingly unstable; linear discriminant analysis does not suffer from this problem

- If n is small and the distribution of the predictors X is approximately normal in each of the classes, the linear discriminant model is again more stable than the logistic regression model

- LDA is popular when there are more than two response classes

- Let a qualitative response variable take on K possible distinct (and unordered) values. Let π_k represent the overall or prior probability that a randomly chosen observation comes from the kth cass. Let $f_k(X) = P(X = x \mid Y = k)$ denote the density function of X for an observation that comes from the kth class. Then Bayes' theorem states that

$$P(Y = k \mid X = x) = \frac{\pi_k f_k(x)}{\sum_{i=1}^{K} \pi_i f_i(x)}$$

- Let $p_k(X) = P(Y = k \mid X)$; this suggests that instead of directly computing $p_k(X)$, simply plug in estimates of π_k and $f_k(X)$ into the above equation

- $p_k(x)$ will be referred to as the posterior probability that an observation $X = x$ belongs to the kth class
- Recall that the Bayes classifier, which classifies an observation to the class for which $p_k(X)$ is largest, has the lowest possible error rate of all classifiers
- Suppose $p = 1$ (one predictor), then to make estimate $f_k(x)$, several assumptions have to be made about its form
  - Assume that $f_k(x)$ is normal or Gaussian

    $$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{1}{2\sigma_k^2}(x - \mu_k)^2\right)$$

    where μ_k and σ_k^2 are the mean and variance parameters for the kth class
  - Assume that $\sigma_1^2 = \cdots = \sigma_k^2$; this means that there is a shared variance term across of K classes
- What this shows is that

  $$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(x - \mu_k)^2\right)}{\sum_{i=1}^{K} \pi_i \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(x - \mu_i)^2\right)}$$

- The Bayes classifier involves assigning an observation $X = x$ to the class for which $p_k(x)$ is largest; taking the log of this and rearranging the terms show that it is the same s assigning the observation to the class for which the following is largest

  $$\delta_k(x) = x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\mu_k)$$

- If $K = 2$ and $\pi_1 = \pi_2$, then the Bayes classifier assigns an observation to class 1 if $2x(\mu_1 - \mu_2) > \mu_1^2 - \mu_2^2$, and to class 2 otherwise; here the Bayes decision boundary corresponds to the point where

  $$x = \frac{\mu_1^2 - \mu_2^2}{2(\mu_1 - \mu_2)} = \frac{\mu_1 + \mu_2}{2}$$

- The linear discriminant analysis method approximates the Bayes classifier by plugging estimates fir π_k, μ_k and σ^2 into the equation for δ_k where the following estimates are used

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

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

  where n is the total number of training observations and n_k is the number of training observations in the kth class. The estimate for μ_k is the average of all the training observations from the kth class while the estimate for σ^2 is an weighted average of the sample variances for each of the K classes

- In the absence of any additional information, LDA estimates π_k using the proportion of the training observations that belong to the kth class

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

- The LDA classifier plugs the estimates into $\delta_k(x)$ assigns an observation $X = x$ to the class for which

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

  is largest

- The word linear in LDA comes from the fact that the discriminant functions $\hat{\delta}_k$ are linear functions of x

- When using the LDA classifier for multiple predictors, assume that $X = (X_1, \ldots, X_)$ is drawn from a multivariate Gaussian (or multivariate normal)distribution with a class-specific mean vector and a common covariance matrix

  - To indicate that a p-dimensional random variable X has a multivariate Gaussian distribution, write $X \sim N(\mu, \sigma)$ where $\mathrm{E}[X] = \mu$ is the mean of X (a vector with p components) and $\mathrm{Cov}[X] = \sigma$ is the $p \times p$ covariance matrix of X
  - The multivariate Gaussian density is defined as

$$f(x) = \frac{1}{(2\pi)^{p/2}|\sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\sigma^{-1}(x-\mu)\right)$$

- 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 $N(\mu_k, \sigma)$ where μ_k is a class-specific mean vector and σ is a covariance matrix that is common to all K classes

- The Bayes classifier will assign an observation $X = x$ to the class for which the following is largest

$$\delta_k(x) = x^T\sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\sigma^{-1}\mu_k + \log \pi_k$$

- Bayes decision boundaries are lines that represent the set of values x for which $\delta_k(x) = \delta_l(x)$, or

$$x^T\sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\sigma^{-1}\mu_k = x^t\sigma^{-1}\mu_l - \frac{1}{2}\mu_l^T\sigma^{-1}\mu_l$$

  for $k \neq l$

- For k classes, there are k lines that represent the Bayes decision boundaries because there are k pairs of classes which divide the predictor space into k regions

- To estimate the unknown parameters $\mu_1, \ldots, \mu_k$, $\pi_1, \ldots, \pi_k$ and σ, use the formulas that were used in the one-dimensional case

- To assign a new observation $X = x$, LDA plugs the estimates into $\delta_k(x)$ and classifies to the class for which $\hat{\delta}_k(x)$ is largest
- A confusion matrix can be used to display errors made from a classifier
- Sensitivity/specificity characterize the performance of a classier by determining how many of the positives were identified and how many of the negatives were identified, respectively
- When the sensitivity metric is low, it makes sense to lower the threshold for classification
- In the general case, the Bayes classifier works by assigning an observation to the class for which the posterior probability $p_k(X)$ is greatest; in the two-class case, this is assigning an observation to the k_i class if

  $$P(Y = k_i \mid X = x) > 0.5$$

  Here, the Bayes classifier (and by extension LDA) uses a threshold of 0.5 for the posterior probability of Y to assign an observation to the k_i class
- When sensitivity is high, lower this threshold to below 0.5
- The ROC (receiver operating characteristics) curve is a graph that displays the two types of errors for all possible thresholds; the overall performance of a classifier, summarized over all possible thresholds, is given by the area under the curve (AUC)
- An ideal ROC curve will come close to the top left corner, so the larger the AUC, the better the classifier
- Sensitivity is also called the true positive rate while the specificity is also called the false positive rate
- Quadratic discriminate analysis (QDA) provides an alternative approach that does not assume the observations within each class are drawn from a multivariate Gaussian distribution with a class-specific mean vector and a covariance matrix that is common to all K classes
- Unlike LDA, QDA assumes that the observations from each class are drawn from a Gaussian distribution where each class has its own covariance matrix; that is, it assumes that an observation from the kth class is of the form $X \sim N(\mu_k, \sigma_k)$, where σ_k is a covariance matrix for the kth class
- Under this assumption, the Bayes classifier assigns an observation $X = x$ to the class for which

  $$\begin{aligned}\delta_k(x) &= -\frac{1}{2}(x-\mu_k)^T\sigma_k^{-1}(x-\mu_k) = \log \pi_k \\ &= -\frac{1}{2}x^T\sigma_k^{-1}x + x^T\sigma_k^{-1}\mu_k - \frac{1}{2}\mu_k^T\sigma_k^{-1}\mu_k + \log \pi_k\end{aligned}$$

  is largest

- The QDA classifier involves plugging estimates for σ_k, μ_k and π_k into the above equation and then assigning an observation $X = x$ to the class for which the quantity is largest; the quantity x appears as a quadratic function
- To determine whether to prefer LDA to QDA, look at the bias-variance trade-off; when there are p predictors, then estimating a covariance matrix requires estimating $p(p+1)/2$ parameters; QDA estimates a separate covariance matrix for each class, for a total of $Kp(p+1)/2$ parameters
- By assuming that the K classes share a common covariance matrix, the LDA model becomes linear in x, which means there are Kp linear coefficients to estimate; consequently, LDA is a much less flexible classifier than QDA and so has a substantially lower variance
- This comes at a tradeoff: if LDA's assumption that the K classes share a common covariance matrix that is badly off, then LDA can suffer from high bias - this means that LDA tends to be a better bet than QDA if there are relatively few training observations and so reducing variance is crucial; in contrast, QDA is recommended if the training set is very large so that the variance of the classifier si not a major concern, or if the assumption of a common covariance matrix for the K classes is clearly untenable

2.5 A Comparison of Classification Methods

- Between KNN, logistic regression, LDA and QDA, there is no clear classification method that dominates the others in all cases
- Assume $k = 2$ with $p = 1$ predictor and let $p_1(x)$ and $p_2(x) = 1 - p_1(x)$ be the probabilities that the observation $X = x$ belongs to class 1 and class 2 respectively
  - In the LDA framework, it can be seen that the log odds is given by
    $$\log\left(\frac{p_1(x)}{1-p_1(x)}\right) = \log\left(\frac{p_1(x)}{p_2(x)}\right) = c_0 + c_1 x$$
    where c_0 and c_1 are functions of μ_1, μ_2 and σ^2
  - In logistic regression,
    $$\log\left(\frac{p_1}{1-p_1}\right) = \beta_0 + \beta_1 x$$
  - Both of these are linear functions of x and so both logistic regression and LDA produce linear decision boundaries; the only difference between them lies in the fact that β_0 and β_1 are estimated using maximum likelihood whereas c_0 andc_1 are computed using the estimated mean and variance from a normal distribution (this same connection also holds for $p > 1$)
  - LDA assumes that the observations are drawn from a Gaussian distribution with a common covariance matrix in each class and so can do better than logistic regression when this holds

  - On the other hand, logistic regression can do better than LDA if these Gaussian assumptions are not met

- Between KNN and logistic regression and LDA, KNN is a non-parametric approach: no assumptions are made about the shape of the decision boundary

- Therefore it can be expected that KNN will dominate LDA and logistic regression when the decision boundary is highly non-linear; on the other hand, KNN does not tell which predictors are important

- QDA serves as a compromise between the non-parametric KNN method and the linear LDA and logistic regression; it assumes a quadratic decision boundary and can accurately model a wider range of problems than can the linear methods

- Although not as flexible as KNN, QDA can perform better in the presence of a limited number of training observations because it does make some assumptions about the form of the decision boundary

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

- For increased flexibility, a non-linear relationship between the predictors and the response can be utilized by transforming the predictors for classification; this means including terms such as X^2, X^3 and more as predictors in logistic regression; this may or may not increase logistic regression performance, depending on whether the increase in variance due to the added flexibility is offset by a sufficiently large reduction in bias

- The same can be done for LDA; if all possible quadratic terms and cross-products were added to LDA, the form of the model would be same as the QDA model, although the parameter estimates would be different; this creates a model that is somewhere between a LDA model and a QDA model

3 Resampling Methods

3.1 Cross-Validation

- The test error is the average error that results from using a statistical learning method to predict the response on a new observation; the training error rate often underestimates the test error rate
- The validation set approach involves diving the available set of observations into two parts a training set and a validation set or hold-out set; the model is fit on the training set and the fitted model is used to predict the responses for the observations in the validation set; the resulting validation set error rate, usually MSE, provides an estimate of the test error rate
- The validation set approach has two drawbacks:
  - The validation estimate of the test error rate can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the validation set
  - In the validation approach, only a subset of the observations are used to fit the model; since statistical methods tend to perform worse when trained on fewer observations, this suggests that the validation set error rate may tend to overestimate the test error rate for the model fit on the entire data set
- Leave-One-Out-Cross-Validation (LOOCV) involves splitting the set of observations into two parts but instead of creating two subsets of comparable size, a single observation (x_1, y_1) is used for the validation set and the remaining $\{(x_2, y_2), \ldots, (x_n, y_n)\}$ make up for the training set; the statistical learning method is fit on the $n-1$ training observations and a prediction $\hat{y}_1$ is made for the excluded observation, using its value x_1 and computing $\text{MSE}_1 = (y_1 - \hat{y}_1)^2$
- This procedure is repeated $n-1$ times on the remaining $n-1$ observations to produce n squared errors
- The LOOCV estimate for the test MSE is the average of these n test error estimates

$$\text{CV}_{(n)} = \frac{1}{n} \sum_{i=1}^{n} \text{MSE}_i$$

- LOOCV has far less than bias than the validation set approach because the method is fit n times on $n-1$ different collections of observations whereas in the validation set approach, the model is only fit on, usually, half the size of the entire data set
- This means that the LOOCV approach tends not to overestimate the test error rate as much as the validation set approach
- Performing LOOCV multiple times will always yield the same results because there is not randomness in the training/validation set splits

- LOOCV has the potential to be expensive to implement when n is large
- With least squares linear or polynomial regression, the following formula holds

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

  where $\hat{y}_i$ is the ith fitted value from the original least squares fit and h_i is the leverage; the leverage lies between $\frac{1}{n}$ and 1 and reflects the amount that an observation influences its own fit

- K-fold Cross-Validation involves randomly dividing the set of observations into k groups, or folds, of approximately equal size; the first fold is treated as a validation set and the method is fit on the remaining $k-1$ folds; the mean squared error MSE_1 is computed on the observations in the held-out fold; this procedure is repeated k times where each time, a different group of observations is treated as a validation set
- The k-fold CV estimate is computed by averaging the MSEs

$$\text{CV}_{(k)} = \frac{1}{k}\sum_{i=1}^{k}\text{MSE}_i$$

- Note that LOOCV is a special case of k-fold CV where $k = n$
- An important advantage of k-fold CV is that it often gives more accurate estimates of the test error rate than does LOOCV due to the bias-variance trade-off
- LOOCV has higher variance than does k-fold CV with $k < n$ since the mean of many highly correlated quantities (from LOOCV) have higher variance than does the mean of many quantities that are not as highly correlated (from k-fold CV)
- There is a bias-variance trade-off associated with the choice of k in k-fold CV; typically, one performs k-fold CV using $k = 5$ or $k = 10$ since these values have been shown to yield test error rate estimates that suffer neither from very high bias nor very high variance
- In the classification setting, LOOCV can still be used where the LOOCV error rate is

$$\text{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\text{Err}_i$$

  where $\text{Err}_i = I(y_i \neq \hat{y}_i)$

- The k-fold CV error rate and validation set error rates for classification are defined similarly
- The training error tends to decrease as the flexibility of the fit increases; in contrast, the test error displays a characteristic U-shape

3.2 The Bootstrap

- The bootstrap can be used to estimate the standard errors of the coefficients from many statistical learning methods, including some for which a measure of variability is otherwise difficult to obtain and is not automatically output by statistical software
- Suppose we want to invest a fixed sum of money in two financial assets that yield returns of X and Y, respectively, where X and Y are random quantities; we will invest a fraction α of the money in X and invest the remaining $1 - \alpha$ in Y
- Since there is variability associated with the returns on the two assets, choose α that minimizes the total risk, or variance, of the investment, or $\text{Var}[\alpha X + (1 - \alpha)Y]$
- The value of α that best minimizes the risk is given by

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

  where $\sigma_X^2 = \text{Var}[X]$, $\sigma_Y^2 = \text{Var}[Y]$ and $\sigma_{XY} = \text{Cov}[X, Y]$
- These three quantities can be estimated using a data set that contains past measurements for X and Y and then estimate α using

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}$$

- The procedure for estimating $\text{SE}[\hat{\alpha}]$, by simulating paired observations of X and Y to estimate α thousands of times cannot be applied to real data because for real data, new samples cannot be generated from the original population
- The bootstrap approach will emulate the process of obtaining new sample sets so that the variability of $\hat{\alpha}$ can be estimated by obtaining distinct data sets by repeatedly sampling (with replacement) observations from the original data set
- The bootstrap method performs sampling with replacement, meaning the same observation can occur more than once in the bootstrap data set
- Randomly select n observations from a simple data set Z and produce a new bootstrap estimate for α; repeat this procedure B times for some large values of B, in order to produce B different bootstrap data sets and B corresponding α estimates
- The standard error of these bootstrap estimates is computed as

$$\text{SE}_B[\hat{\alpha}] = \sqrt{\frac{1}{B-1}\sum_{i=1}^{B}\left(\hat{\alpha}^{r*} - \frac{1}{B}\sum_{r'=1}^{B}\hat{\alpha}^{r'*}\right)^2}$$

  which serves as an estimate of the standard error of $\hat{\alpha}$ estimated from the original data set

4 Linear Model Selection and Regularization

4.1 Subset Selection

- Alternative fitting procedures, instead of just least squares, can yield better prediction accuracy and model interpretability
  - If n is not much larger than p, then there can be a lot of variability in the least squares fit, resulting in overfitting and consequently poor predictions on future observations; if $p > n$, then there is no longer a unique LS coefficient estimate: the variance is infinite and so the method cannot be used at all
  - By constraining or shrinking the estimated coefficients, the variance can be reduced at the cost of a negligible increase in bias leading to substantial improvements in the accuracy of the model
  - Including irrelevant variables lead to unnecessary complexity in the resulting model; by removing such variables, a model can be obtained that more easily interpretable
  - LS is extremely unlikely to yield any coefficient estimates that are exactly zero; look for other approaches that can do that such as feature selection or variable selection
- Alternatives to Least Squares
  - Subset Selection
  - Shrinkage
  - Dimension Reduction
- To perform best subset selection, it a separate least squares regression for each possible combination of the p predictors; that is, fit all p models that contain exactly one predictor, all $\binom{p}{2} = \frac{p(p-1)}{2}$ models that contain exactly two predictors, and so forth, then look at all the of the resulting models, with the goal of identifying the one that is best
- Algorithm: Best Subset Selection
  1. Let M_0 denote the null model, which contains no predictors; this model simply predicts the sample mean for each observation
  2. For $k = 1, 2, \ldots, p$
     (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors
     (b) Pick the best among these $\binom{p}{k}$ models, and call it M_k; here best is defined as having the smallest RSS, or equivalently, largest R^2
  3. Select a single best model from among $M_0, \ldots, M_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2

- In order to select a single best model, simply choose amongst the $p+1$ options; note that the RSS of these $p+1$ models decrease monotonically and the R^2 increases monotonically, as the number of features included in the models increases
- Thus by using these statistics to pick the best model, you'll end up with a model involving all of the variables; the problem with this is is that the model will have a low training error but not a low test error necessarily
- Therefore, use cross-validated prediction error, C_p, BIC, or adjusted R^2 to pick the best model
- When applying best subset selection for logistic regression, instead of ordering models by RSS, use the deviance, a measure that plays the role of RSS for a broader class of models; the deviance is negative two times the maximized log-likelihood; the smaller the deviance, the better the fit
- Best subset selection suffers from computational limitations because the number of possible models that must be considered grows rapidly as p increases; in general, there are 2^p models that involve subsets of p predictors
- Stepwise methods is a good alternative to best subset selection for when p is very large
- Forward stepwise selection begins with a model containing no predictors, and then adds predictors to the model, one at a time, until all of the predictors are in the model; in particular, at each step, the variable that gives the greatest additional improvement to the fit is added to the model
- Algorithm: Forward Stepwise Selection
  1. Let M_0 denote the null model, which contains no predictors
  2. For $k = 0, \ldots, p-1$:
     (a) Consider all $p-k$ models that augment the predictors in M_k with one additional predictor
     (b) Choose the best among these $p-k$ models, and call it M_{k+1}; here, best is defined as having the smallest RSS or highest R^2
  3. Select a single best model from among $M_0, \ldots, M_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2
- Unlike best subset selection, forward stepwise selection involves fitting one null model, along with $p-k$ models in the kth iteration, for $k = 0, \ldots, p-1$; this is a total of $1 + \sum_{k=0}^{p-1}(p-k) = 1 + \frac{p(p+1)}{2}$ models, a huge deduction from the amount needed for best subset selection
- Forward stepwise selection can be applied even in the high-dimensional setting where $n < p$; however in this case, it is possible to construct sub-models, $M_0, \ldots, M_{n-1}$ only, since each sub-model is fit using least squares, which will not yield a unique solution if $p \geq n$

- Unlike forward stepwise selection, backward stepwise selection begins with full least squares model containing all p predictors and then iteratively remove the least useful predictor one at a time

- Algorithm: Backward Stepwise Selection

  1. Let M_p denote the full model, which contains all p predictors
  2. For $k = p, p-1, \ldots, 1$:
     (a) Consider all k models that contain all but one of the predictors in M_k, for a total of $k-1$ predictors
     (b) Choose the best among these k models, and call it M_{k-1}; here, best is defined as having the smallest RSS or highest R^2
  3. Select a single best model from among $M_0, \ldots, M_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2

- Like forward stepwise selection, the backward selection approach searches through only $1 + \frac{p(p+1)}{2}$ models and so can be used when p is too large to try best subset selection; it is also not guaranteed to yield the best model containing a subset of the p predictors

- Backward selection requires that n is larger than p; on the other hand, forward selection can be used even when $n < p$ and so is the only viable subset method when p is very large

- Hybrid versions of forward and backward stepwise selection are available, in which variables are added to the model sequentially; however, after adding each new variable, the method may also remove any variables that no longer provide an improvement in the model fit

- For a fitted least squares model containing d predictors, the C_p estimate of test MSE is computed as follows:

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

  where $\hat{\sigma}^2$ is an estimate of the variance of the error ε associated with each response measurement in the linear model equation

- Essentially, the C_p statistic adds a penalty of $2d\hat{\sigma}^2$ to the training RSS in order to adjust for the fact that the training error tends to underestimate the test error; the penalty increases as the number of predictors in the model increases; this is intended to adjust for the corresponding decrease in training RSS

- If $\hat{\sigma}^2$ is an unbiased estimator fo σ^2, then C_p is an unbiased estimator of test MSE; as a consequence, the C_p statistic tends to take on a small value for models with a low test error, so when determining which of a set of a models is best, choose the model with the lowest C_p value

- The AIC (Akaike Information Criterion) is defined for a large class of models fit by maximum likelihood; in the case of the linear model with Gaussian errors, maximum likelihood and least squares are the same thing and so AIC is defined as follows:

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

- For least squares models, C_p and AIC are proportional to each other
- BIC is derived from a Bayesian point of view but looks similar to C_p (and AIC); for the least squares model with d predictors, the BIC is defined as follows:

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

- Like C_p, the BIC will take on a small value for a model with a low test error
- Note that BIC replaces the $2d\hat{\sigma}^2$ in C_p with a $\log(n)d\hat{\sigma}^2$ term; since $\log(n) > 2$ for any $n > 7$, the BIC statistic generally places a heavier penalty on models with many variables and hence results in the selection of smaller models than C_p
- Since RSS always decreases as more variables are added to the model, the R^2 always increases as more variables are added; for a least squares model with d variables, the adjusted R^2 statistic is defined as follows:

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

  where TSS is defined as the total sum of squares for the response, or $\sum(y_i - \bar{y})^2$

- Unlike C_p, AIC and BIC, a large value of adjusted R^2 indicates a model with a small test error
- Maximizing the adjusted R^2 is equivalent to minimizing $\frac{\text{RSS}}{n-d-1}$; while RSS always decreases as the number of variables in the model increases, $\frac{\text{RSS}}{n-d-1}$ may increase or decrease due to the presence of d int the denominator
- The intuition behind the adjusted R^2 is that once all of the correct variables have been included in the model, adding noise variables will lead to only a very small decrease in RSS; in theory, the model with the largest adjusted R^2 will have only correct variables and no noise variables
- Unlike the R^2 statistic, the adjusted R^2 statistic pays a price for the inclusion of unnecessary variables in the model
- As an alternative to all these approaches, the test error can be estimated using a validation set and cross-validation methods where the validation set error or cross-validation error is computed for each model under consideration and then selecting the model for which the resulting estimated test error is smallest

- The advantage of this over the other statistics is 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 σ^2
- In this setting, select a model using the one-standard-error rule: first calculate the standard error of the estimated test MSE for each model size, and then select the smallest model for which the estimated test error is within one standard error of the lowest point on the curve

4.2 Shrinkage Methods

- The subset selection methods involve using least squares to fit a linear model that contains a subset of the predictors; as an alternative, a model can be fit containing all p predictors using a technique that constrains or regularizes the coefficient estimates, or equivalently, that shrinks the coefficient estimates towards zero
- Shrinking the coefficient estimates can significantly reduce their variance
- Ridge regression is very similar to least squares, except that the coefficients are estimated by minimizing the following equation

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

  where $\lambda \geq 0$ is a tuning parameter
- Ridge regression tries to make the RSS small; the second term, $\lambda\sum_k \beta_k^2$, is called a shrinkage penalty, is small when $\beta_1, \ldots, \beta_p$ are close to zero and so it has the effect of shrinking the estimates of β_k close to zero
- When $\lambda = 0$, the penalty term has no effect an ridge regression will produce the least squares estimate; however as $\lambda \to \infty$, the impact of the shrinkage penalty grows and the ridge regression coefficient estimates will approach zero
- Ridge regression will produce a different set of coefficient estimates, $\hat{\beta}_\lambda^R$ for each value of λ
- Note that the shrinkage penalty is not applied to the intercept β_0 which is simply a measure of the mean value of the response when $x_{i1} = x_{i2} = \cdots = x_{ip} = 0$; if the variables are assumed to be centered to have mean zero before ridge regression is performed, then the estimated intercept will be $\hat{\beta}_0 = \bar{y} = \sum_{i=1}^{n} y_i/n$
- To measure the distance of β from zero, calculate the l_2 norm:

$$\|\beta\|_2 = \sqrt{\sum_{j=1}^{p}\beta_j^2}$$

- As λ increases, the l_2 norm of $\hat{\beta}_\lambda^R$ will always decrease and so will $\left\|\hat{\beta}_\lambda^R\right\|_2 / \left\|\hat{\beta}\right\|_2$; this last quantity ranges from 1 when $\lambda = 0$ to 0 when $\lambda = \infty$
- It is best to apply ridge regression after standardizing the predictors, using the formula

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

  so that they are all on the same scale
- Ridge regression's advantage over least squares is rooted in the bias-variance trade-off; as λ increases, the flexibility of the ridge regression fit decreases, leading to decreased variance but increased bias
- In general, 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; this means that a small change in the training data can cause a large change in the least squares coefficient estimates
- Ridge regression works best in situations where the least squares estimates have high variance
- Ridge regression has substantial computational advantages over best subset selection since for any fixed value of λ, ridge regression only fits a single model
- The disadvantage of ridge regression is that it will include all p predictors in the final model; the penalty $\lambda \sum \beta_j^2$ will shrink all the coefficients towards zero but it will not set any of them to exactly zero (unless $\lambda = \infty$)
- The lasso is an alternative to ridge regression that overcomes this disadvantage; the lasso coefficients, $\hat{\beta}_\lambda^L$, minimize the quantity

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

- The only difference between the lasso and ridge regression is that the β_j^2 term in the ridge regression penalty has been reduced to $|\beta_j|$ in the lasso penalty
- The lasso uses an l_1 penalty instead of an l_2 penalty; the l_1 norm of a coefficient vector β is given by $\|\beta\|_1 = \sum|\beta_j|$
- As with ridge regression, the lasso shrinks the coefficient estimates towards zero; however in the case of the lasso, the l_1 penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when λ is sufficiently large
- The lasso, thus, performs variable selection just like best subset selection; thus models generated from the lasso are generally much easier to interpret than those produced by ridge regression

- The lasso yields sparse models (models that involve only a subset of the variables)
- When $\lambda = 0$, the lasso simply gives the least squares fit, and when λ is sufficiently large, the lasso gives the null model where all coefficient estimates equal zero
- Depending on the value of λ, the lasso can produce a model involving any number of variables; in contrast, ridge regression will always include all of the variables in the model, although the magnitude of the coefficient estimates will depend on λ
- The lasso regression coefficient estimates solve the problem

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

  subject to $\sum_{j=1}^{p} |\beta_j| \leq s$ and the ridge regression coefficient estimates solve the problem

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

  subject to $\sum_{j=1}^{p} \beta_j^2 \leq s$
- When s is extremely large, then the solution is not very restricted in its choice and so the coefficient estimates can be large; in fact, if s is large enough that the least squares solution falls within the boundary, then the least squares solution will come up
- When s is small, then $\sum_{j=1}^{p} |\beta|_j$ must be small in order to stay in the boundary
- The lasso leads to qualitatively similar behavior to ridge regression, in that as λ increases, the variance decreases and the bias increases
- In general, the lasso will 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
- Cross-validation can be used in order to determine which approach is better on a particular dataset; choose a grid of λ values and compute the cross-validation error for each value of λ; 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
- As with ridge regression, when the least squares estimates have excessively high variance, the lasso solution can yield a reduction in variance at the expense of a small increase in bias and consequently can generate more accurate predictions; unlike ridge regression, the lasso performs variable selection and hence results in models that are easier to interpret

4.3 Dimension Reduction Methods

- Dimension reduction methods transform the predictors and then fit a least squares model using the transformed variables
- Let $Z_1, \ldots, Z_M$ represent $M < p$ linear combinations of the original p predictors, that is,

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

for some constants $\phi_{1m}, \phi_{2m}, \ldots, \phi_{pm}$, $m = 1, \ldots, M$; then fit the linear regression model

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \varepsilon_i \quad i = 1, \ldots, n$$

using least squares; the regression coefficients are given by $\theta_0, \theta_1, \ldots, \theta_M$

- If the constants $\phi_{1m}, \phi_{2m}, \ldots, \phi_{pm}$ are chosen wisely, then such dimension reduction approaches can often outperform least squares regression
- This method is called dimension reduction because the dimension of the problem has been reduced from $p + 1$ to $M + 1$ where $M < p$
- Dimension reduction serves to constrain the estimated β_j coefficients, since now they must take the form

$$\beta_j = \sum_{m=1}^{M} \theta_m \phi_{jm}$$

- This constraint has the potential to bias the coefficient estimates
- In situations where p is large relative to n, selecting a value of $M << p$ can significantly reduce the variance of the fitted coefficients; if $M = p$, and all the X_m are linearly independent, then the above constraint for β_j poses no constraint; in this case, no dimension reduction occurs
- All dimension reduction methods work in two steps
  1. First, the transformed predictors $Z_1, Z_2, \ldots, Z_M$ are obtained
  2. Second, the model is fit using these M predictors
- The choice of $Z_1, Z_2, \ldots, Z_M$, or equivalently, the selection of the $\phi_{jm}s$ can be achieved in different ways such as the principal components and partial least squares
- PCA (Principal Components Analysis) is a technique for reducing the dimension of a $n \times p$ data matrix X
- The first principal component direction of the data is that along which the observations vary the most, or in another interpretation, the first principal component vector defines the line that is as close as possible to the data

- The first principal component score for the ith observation is the distance in the x-direction of the ith cross from zero
- In general, one can construct up to p distinct principal components; the second principal component Z_2 is a linear combination of the variables that is uncorrelated with Z_1 and has the largest variance subject to this constraint
- It turns out that the zero correlation condition of Z_1 with Z_2 is equivalent to the condition that the direction must be perpendicular, or orthogonal, to the first principal component direction
- The principal components regression (PCR) approach involves constructing the first M principal components, $Z_1, \ldots, Z_M$ and then using these components as the predictors in a linear regression model that is fit using least squares
- Assume that the directions in which $X_1, \ldots, X_p$ show the most variation are the directions that are associated with Y; which this assumption is not guaranteed to be true, it often turns out be a reasonable enough approximation to give good results
- If the assumption underlying PCR holds, then fitting a least squares model to $Z_1, \ldots, Z_M$ will lead to better results than fitting a least squares model to $X_1, \ldots, X_p$, since most or all of the information in the data that relates to the response is contained in $Z_1, \ldots, Z_M$ and by estimating only $M << p$ coefficients, overfitting can be mitigated
- As more principal components are used in the regression model, the bias decreases but the variance increases; this results in a typical U-shape for the mean squared error
- PCR will tend to do well in cases where the first few principal components are sufficient to capture most of the variation in the predictors as well as the relationship with the response
- Even though PCR provides a simple way to perform regression using $M < p$ predictors, it is not a feature selection method; this is because each of the M principal components used in the regression is a linear combination of all p of the original features
- Therefore, while PCR often performs well in many practical settings, it does not result in the development of a model that relies upon a small set of the original features; in this sense, PCR is most closely related to ridge regression than to the lasso
- Ridge regression can be thought of a continuous version of PCR
- In PCR, the number of principal components, M, is typically chosen by cross-validation
- When performing PCR, do standardize each predictor prior to generating the principal components; this standardization ensures that all variables are on the same scale; in the absence of standardization, the high-variance variables will tend to play a larger role in the principal components obtained, and the scale on which the variables are measured will ultimately have an effect on the final PCR model

- If the variables are all measured in the same unit (kg, inches, etc.), then they may not be a need to standardize them
- The drawback of the PCR approach is that since the directions are identified in an unsupervised way (since the response Y is not used to help determine the principal component directions), there is no guarantee that the directions that best explain the predictors will also be the best directions to use for predicting the response
- Partial Least Squares (PLS) is a supervised alternative to PCR; it is a dimension reduction method which first identifies a new set of features $Z_1, \ldots, Z_M$ that are linear combinations of the original features and then fits a linear model via least squares using these M new features
- Unlike PCR, PLS identifies these new features in a supervised way, that is, it makes use of the response Y in order to identify new features that not only approximate the old features well, but also that are related to the response
- The PLS approach attempts to find directions that help explain both the response and the predictors
- After standardizing the p predictors, PLS computes the first direction Z_1 by setting each ϕ_{j1} in

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

  equal to the coefficient from the simple linear regression of Y onto X_j; this coefficient is proportional to the correlation between Y and X_j; hence, in computing $Z_1 = \sum_{j=1}^{p} \phi_{j1} X_j$, PLS places the highest weight on the variables that are most strongly related to the response
- The PLS direction does not fit the predictors as closely as does PCA, but it does a better job explaining the response
- To identify the second PLS direction, first adjust each of the variables for Z_1, by regressing each variable on Z_1 and taking the residuals; these residuals can be interpreted as the remaining information that has not been explained by the first PLS direction; then compute Z_2 using this orthogonalized data in exactly the same fashion as Z_1 was computed based on the original data
- This iterative approach can be repeated M times to identify multiple PLS components $Z_1, \ldots, Z_M$
- At the end of this procedure, use least squares to fit a linear model to predict Y using $Z_1, \ldots, Z_M$ in exactly the same fashion as for PCR
- Like PCR, the number M of partial least squares directions to use in PLS is a tuning parameter that is chosen by cross-validation
- The predictors and response are normally standardized before performing PLS

- PLS often performs no better than ridge regression or PCR; while the supervised dimension reduction of PLS can reduce bias, it also has the potential to increase variance, so that the overall benefit of PLS relative to PCR is a wash

4.4 Considerations in High Dimensions

- Datasets containing more features than observations are often referred to as high-dimensional; classical approaches such as least squares linear regression are not appropriate in this setting
- When the number of features p is as large as, or larger than, the number of observations n, least squares cannot be performed; regardless of whether or not there truly is a relationship between the features and the response, least squares will yield a set of coefficient estimates that result in a perform fit to the data, such that the residuals are zero
- When $p > n$ or $p \approx n$, a simple least squares regression line is too flexible and hence overfits the data
- The C_p, AIC and BIC approaches for adjusting the training set RSS or R^2 are not appropriate in the high-dimensional setting because estimating $\hat{\sigma}^2$ is problematic
- Methods such as forward stepwise selection, ridge regression, the lasso and principal components regression are useful for performing regression in the high-dimensional setting; these approaches avoid overfitting by using a less flexible fitting approach than least squares
- Regularization or shrinkage plays a key role in high-dimensional problems
- Appropriate tuning parameter selection is crucial for good predictive performance
- The test error tends to increase as the dimensionality of the problem (i.e., the number of features of predictors) increases, unless the additional features are truly associated with the response
- In general, adding additional signal features that are truly associated with the response will improve the fitted model, in the sense of leading to a reduction in test set error; however, adding noise features that are not truly associated with the response will lead to a deterioration in the fitted model and consequently an increased test set error
- Noise features increase the dimensionality of the problem, exacerbating the risk of overfitting without any potential upside in terms of improved test set error; this is the curse of dimensionality
- In the high-dimensional setting, the 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

- This means that the variables (if any) that are truly predictive of the outcome can never be identified and that the best coefficients for use in the regression can also never be identified
- At best, it can be hoped that large regression coefficients are assigned to variables that are correlated with the variables that truly are predictive of the outcome
- It is important to be careful in reporting errors and measures of model fit in the high-dimensional setting
- When $p > n$, it is easy to obtain a useless model that has zero residuals; so, never use SSE, p-values, R^2 statistics or any other traditional measures of model fit on the data as evidence of a good model fit in the high-dimensional setting
- Instead, report results on an independent test set, or cross-validation errors, such as the MSE or R^2 on an independent test set which are valid measures of model fit

5 Moving Beyond Linearity

5.1 Polynomial Regression

- Polynomial regressions extends the linear model by adding extra predictors obtained by raising each of the original predictors to a power; this provides a non-linear fit to data
- The polynomial regression model has the polynomial function

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_d x_i^d + \varepsilon_i$$

  where ε_i is the error term
- The above coefficients can be easily estimated using least squares linear regression because this is just a standard linear model with predictors x_i, x_i^2, x_i^3, $\ldots$, x_i^d
- When d is very large, the polynomial curve can become overly flexible and can take on some very strange shapes, especially true near the boundary of the X variable

5.2 Step Functions

- Step functions cut the range of a variable into K distinct regions in order to produce a qualitative variable; this has the effect of fitting a piecewise constant function
- Using polynomial functions of the features as predictors in a linear model imposes a global structure on the non-linear function of X; use step functions to avoid this

- To use step functions, create cutpoints $c_1, c_2, \ldots, c_K$ in the range of X and then construct $K+1$ new variables

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

  where $I(\cdot)$ is an indicator function that returns a 1 if the condition is true and returns a 0 otherwise

- Notice for that for any value of X,

$$C_0(X) + C_1(X) + \cdots + C_K(X) = 1$$

  since X must be in exactly one of the $K+1$ intervals

- Use least squares to fit a linear model using $C_1(X), C_2(X), \ldots, C_K(X)$ as predictors

$$y_i = \beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + \cdots + \beta_K C_K(x_i) + \varepsilon_i$$

- For a given value of X, at most one of $C_1, C_2, \ldots, C_K$ can be non-zero

- Note that when $X < c_1$, all of the predictors are zero and so β_0 can be interpreted as the mean value of Y for $X < c_1$

- By comparison, a response of $\beta_0 + \beta_j$ for $c_j \leq X < c_{j+1}$ represents the average increase in the response for X in $c_j \leq X < c_{j+1}$ relative to $X < c_1$

- Unless there are natural breakpoints in the predictors, piecewise-constant functions can miss the action

5.3 Basis Functions

- Polynomial and piecewise-constant regression models are special cases of a basis function approach

- The idea is to have a family a functions of transformations that can be applied to a variable X: $b_1(X), b_2(X), \ldots, b_K(X)$; then the following model is fit

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \beta_3 b_3(x_i) + \cdots + \beta_K b_k(x_i) + \varepsilon$$

  Note that the basis functions $b_1(\cdot), b_2(\cdot), \ldots, b_K(\dot{})$ are fixed and known

- For polynomial regression, the basis functions are $b_j(x_i) = x_i^j$, and for piecewise constant functions, they are $b_j(x_i) = I(c_j \leq x_i < c_{j+1})$

5.4 Regression Splines

- Regression splines are more flexible than polynomials and step functions; it involve dividing the range of X into K distinct regions, and within each region, a polynomial function is fit to the data; however, these polynomials are constrained so that they join smoothly at the region boundaries
- Piecewise polynomial regression involves fitting separate low-degree polynomials over different regions of X
- A piecewise cubic polynomial works by fitting a cubic regression model of the form

  $$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \varepsilon_i$$

  where the coefficients $\beta_0, \beta_1, \beta_2$ and β_3 different in different parts of the range of X; the points where the coefficients change are called knots
- A piecewise polynomial function with no knows is a standard polynomial
- In the case of the piecewise cubic polynomial with a single knot at a point c takes the form

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

  Here two different polynomial functions are fitted to the data, one on the subset of the observations with $x_i < c$ and one on the subset of the observations with $x_i \geq c$
- Using more knots can lead to a more flexible piecewise polynomial
- Using constraints so that the fitted curve is continuous is done by incorporating the first and second derivatives of the piecewise polynomials
- Each constraint imposed on a piecewise polynomial effectively frees up one degree of freedom, by reducing the complexity of the resulting piecewise polynomial fit
- A degree-d spline is a piecewise degree-d polynomial, with continuity in derivatives up to degree $d - 1$ at each knot
- A cubic spline with K knots can be modeled as

  $$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \cdots + \beta_{K+3} b_{K+3}(x_i) + \varepsilon_i$$

  for an appropriate choice of basis functions $b_1, b_2, \ldots, b_{K+3}$
- The most direct way to represent a cubic spline using the above equation is to start off with a basis for a cubic polynomial, x, x^2, x^3 and then adding one truncated power basis function per knot

- A truncated power basis function is defined as

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

  where ξ is the knot

- In order to fit a cubic spline to a data set with K knots, perform least squares regression with an intercept and $3+K$ predictors, of the form $X, X^2, X^3, h(X, \xi_1), h(X, \xi_2), \ldots, h(X, \xi_K)$, where $\xi_1, \ldots, \xi_K$ are the knots; this results to estimating a total of $K + 4$ regression coefficients and so, fitting a cubic spline with K knots uses $K + 4$ degrees of freedom

- At a disadvantage, splines can have high variance at the outer range of the predictors, when X takes on either a very small or very large value

- A natural spline is a regression spline with additional boundary constraints: the function is required to be linear at the boundary (in the region where X is smaller than the smallest knot, or larger than the largest knot); this additional constraint means that natural splines generally produce more stable estimates at the boundaries

- It is common to place knots in a uniform fashion, by specifying the desired degrees of freedom and then having software automatically place the corresponding number of knots at uniform quantiles of the data

- To find the optimal number of knots to use, or how many degrees of freedom a spline should contain, use cross validation

- Regression splines give superior results to polynomial regression because there is flexibility in the number of knots but keeps the degree fixed, producing more stable estimates

- Splines also allow to place more knots and hence flexibility over regions where the function appears to be changing rapidly and fewer knots where the function appears to be stable

5.5 Smoothing Splines

- Smoothing splines are similar to regression splines except that it results from minimizing a residual sum of squares criterion subject to a smoothness penalty

- To ensure that a function g is smooth, a natural approach is to find the function g that minimizes

$$\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int g''(t)^2 \, dt$$

  where λ is a nonnegative tuning parameter; the function g that minimizes this is called a smoothing spline

- The first term of this is a loss function that encourages g to fit the data well while the second term is a penalty term that penalizes the variability in g
- The larger the value of λ, the smoother g will be
- When $\lambda = 0$ then the penalty term has no effect and the function g will be very jumpy and will exactly interpolate the training observations; when $\lambda \to \infty$, g will be perfectly smooth - essentially a straight line that passes as closely as possible to the training points, the linear least squares line
- λ controls the bias-variance trade-off of the smoothing spline
- The function $g(x)$ that minimizes the above equation is a natural cubic spline with knots at $x_1, \ldots, x_n$; it differs from the natural cubic spline where the tuning parameter λ controls the level of shrinkage as well as the roughness of the smoothing spline and hence the effective degrees of freedom; as λ increases from 0 to ∞, the effective degrees of freedom, df_λ, decreases from n to 2
- Although a smoothing spline has n parameters and hence n nominal degrees of freedom, these n parameters are heavily constrained or shrunk down and so df_λ 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
- Let

  $$\hat{g}_\lambda = S_\lambda y$$

  where $\hat{g}$ is the optimal g for a particular choice of λ; $\hat{g}$ is a n-vector containing the fitted values of the smoothing spline at the training points $x_1, \ldots, x_n$, which so indicates that the vector of fitted values when applying a smoothing spline to the data can be written as a $n \times n$ matrix S_λ times the response vector y; then the effective degrees of freedom is

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

  the sum of the diagonal elements of the matrix S_λ
- In fitting a smoothing spline, the number or location of the knots doesn't need to be selected (there is a knot at each training observation) but an optimal value of λ needs to be found using cross validation
- The LOOCV error can be computed very efficiently for smoothing splines using the following formula:

  $$\text{RSS}_{\text{CV}}(\lambda) = \sum_{i=1}^{n} (y_i - \hat{g}_\lambda^{(-i)}(x_i))^2 = \sum_{i=1}^{n} \left[\frac{y_i - \hat{g}_\lambda(x_i)}{1 - \{S_\lambda\}_{ii}} \right]^2$$

  The notation $\hat{g}_\lambda^{(-1)}(x_i)$ indicates the fitted value for this smoothing spline evaluated at x_i where the fit uses all of the training observations except for the ith observation (x_i, y_i)

- This says that each of the love-one-out fits can be computed using only $\hat{g}_\lambda$, the original fit to all of the data

5.6 Local Regression

- Local regression is similar to splines but the regions are allowed to overlap (in a smooth way)
- Local regression involves computing the fit at a target point x_0 using only the nearly training observations
- Algorithm: 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

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

  4. The fitted value at x_0 is given by $\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0$
- Note that the weight K_{i0} will differ for each value of x_0; so in order to obtain the local regression fit at a new point, a new weighted least squares regression model is fit
- The span s is the most important choice to make; it controls the flexibility of the non-linear fit; the smaller the value of s, the more local and wiggly the fit will be; alternatively, a very large value of s will lead to a global fit of the data using all of the observations; use cross-validation to choose s
- In a setting with multiple features $X_1, X_2, \ldots, X_p$, one very useful generalization involves fitting a multiple linear regression model that is global in some variables but local in another; such varying coefficient models are a useful way of adapting a model to the most recently gathered data
- Local regression also generalizes very naturally when fitting models that are local in a pair of variables X_1 and X_2, rather than one
- Local regression can perform poorly if p is much larger than about 3 or 4 because there will generally be very few training observations close to x_0

5.7 Generalized Additive Models

- Generalized additive models extends all the above methods to deal with multiple predictors while maintaining additivity
- Just like linear models, GAMs can be applied with both quantitative and qualitative responses
- To extend the multiple linear regression model in order to allow for non-linear relationships between each feature and the response is to replace each linear component $\beta_j x_{ij}$ with a (smooth) nonlinear function $f_j(x_{ij})$; then the model becomes

$$\begin{aligned} y_i &= \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) + \varepsilon_i \\ &= \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + \varepsilon_i \end{aligned}$$

- This is an example of a GAM; it is called an additive model because a separate f_j can be calculated for each X_j and then added together
- With GAMs, all of the above methods for fitting functions can be put together as building blocks for fitting an additive model
- GAMs can be fit using smoothing splines via an approach called backfitting where a model is fit using multiple predictors by repeatedly updating the fit for each predictor in turn, holding the others fixed
- Each time the function is updated, the fitting method for that variable is applied to a partial residual
- In most situations, the differences in the GAMs obtained using smoothing splines vs natural splines are small
- Pros of GAMs
  - GAMs allow fitting a non-linear f_j to each X_j so that non-linear relationships can be modeled that standard linear regression will miss
  - The non-linear fits can potentially make more accurate predictions for the response Y
  - Because the model is additive, the effect of each X_j on Y individually while holding all of the other variables fixed can be examined; thus GAMs are useful in inference
  - The smoothness of the function f_j for the variable X_j can be summarized via degrees of freedom
- The main limitation of GAMs is that the model is restricted to be additive and so, with many variables, important interactions can be missed

- However, interaction terms can be manually added to the GAM model by including predictors of the form $X_j \times X_k$ as well as low-dimensional interaction functions of the form $f_{jk}(X_j, X_k)$ into the model
- GAMs can also be used in situations where Y is qualitative
- The logistic regression model is

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

This logit is the log of the odds of $P(Y = 1 \mid X)$ versus $P(Y = 0 \mid X)$; to allow for non-linear relationships, use

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + f_1(X_1) + f_2(X_2) + \cdots + f_p(X_p)$$

This is a logistic regression GAM

6 Tree-Based Models

6.1 The Basics of Decision Trees

- Tree-based methods for regression (and classification) involve stratifying or segmenting the predictor space into a number of simple regions
- Tree-based methods are simple and useful for interpretation; however, they typically are not competitive with the best supervised learning approaches in terms of prediction accuracy
- Decision trees are typically drawn upside down, as in the sense that the leaves are at the bottom of three; the points along the tree where the predictor space is split are referred to as internal nodes; the segments of the trees that connect the modes are called branches
- The factors are the top of the trees are considered more important than factors at the bottom
- To build a regression tree,

1. Divide the predictor space - that is, the set of possible values for $X_1, X_2, \ldots, X_p$ into J distinct and non-overlapping regions, $R_1, R_2, \ldots, R_J$
2. For every observation that falls into the region R_j, make the same prediction, which is simply the mean of the response values for the training observations in R_j

- To divide the predictor space, find high-dimensional rectangles, or boxes, $R_1, \ldots, R_J$ that minimize the RSS, given by

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

  where $\hat{y}_{R_j}$ is the mean response for the training observations within the jth box

- It is computationally infeasible to consider every possible partition of the feature space into J boxes; therefore, take a top-town, greedy approach, known as recursive binary splitting

- In order to perform recursive binary splitting, first select the predictor X_j and the cutpoint s such that splitting the predictor space into the regions $\{X \mid X_j < s\}$ and $\{X \mid X_k \geq s\}$ leads to the greatest possible reduction in RSS; that is, consider all predictors $X_1, \ldots, X_p$ and all possible values of the cutpoint s for each of the predictors and then choose the predictor and cutpoint such that the resulting tree has the lowest RSS

- In greater detail, for any j and s, define the pair of half-planes

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

  and see the value of j and s that minimize the equation

$$\sum_{i: x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i: x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

  where $\hat{y}_{R_1}$ is the mean response for the training observations in $R_1(j, s)$ and $\hat{y}_{R_2}$ is the mean response for the training observations in $R_2(j, s)$

- Next, the process is repeated where instead of splitting the entire predictor space, split one of the two previously identified regions; again, look to split one of those three regions further, so as to minimize the RSS

- The process continues until a stopping criterion is reached

- Once the region $R_1, \ldots, R_J$ have been created, the response for a given test observation is predicted using the mean of the mean of the training observations in the region to which the test observation belongs in

- This process can tend to overfit the data because the resulting tree might be too complex

- A smaller tree with fewer splits might lead to lower variance and better interpretation at the cost of a little bias

- A good strategy is o grow a very large tree T_0 and then prune it back in order to obtain a subtree; the goal is to select a subtree that leads to the lowest test error rate; given a subtree, estimate its test error using cross-validation or the validation set approach
- Cost complexity pruning, also known as weakest link pruning, considers a sequence of trees indexed by a nonnegative tuning parameter α to select only a small set of subtrees of consideration
- For each value of α¡ there corresponds a subtree $T \subset T_0$ such that

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

  is as small as possible; here, $|T|$ refers to the number of terminal nodes of the tree T, R_m is the rectangle (i.e., the subset of predictor space) corresponding to the mth terminal node and $\hat{y}_{R_m}$ is the predicted response associated with R_m (the mean of the training observations in R_m)
- When $\alpha = 0$, the subtree T will equal T_0 and thus only measures the training error; as α increases, there is a price to pay for having a tree with many terminal nodes and so the quantity above will tend to have minimized for a smaller subtree
- It turns out that as α increases from zero, branches get pruned from the tree in a nested and predictable fashion, so obtaining the whole sequence of subtrees as a function of α is easy - select α using a validation set or cross-validation, then return to the full data set and obtain the subtree corresponding to α
- Algorithm: 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 subtrees, as a function of α
  3. Use k-fold cross-validation to choose α; that is, divide the training observations in to K folds. For each $k : 1, \ldots, K$:
     (a) Repeat steps 1 and 2 on all but the kth fold of the training data
     (b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α
     (c) Average the results for each value of α and pick α to minimize the average error
  4. Return the subtree from step 2 that corresponds to the chosen value of α
- A classification tree is very similar to a regression, expect that it is used to predict a qualitative response rather than a quantitative one

- For a classification tree, predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs
- Use recursive binary splitting to grow a classification tree where the alternative to the RSS as a criterion for making the binary splits is the classification error rate
- The classification error rate is the fraction of the training observations in that region hat do not belong to the most common class

$$E = 1 - \max_k(\hat{p}_{mk})$$

  where $\hat{p}_{mk}$ represents the proportion of training observations in the mth region that are from the kth class
- Two other measures are preferable to this error rate: the Gini index and cross-entropy
- The Gini index is defined as

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

  which is a measure of total variance across the K classes; it takes on a small value if all of the $\hat{p}_{mk}$'s are close to zero or one
- The Gini index is referred to as a measure of node purity - a small value indicates that a node contains predominately observations from a single class
- The cross-entropy is given by

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

  The cross-entropy will take on a value near zero if the $\hat{p}_{mk}$'s are all near zero or near one
- Like the Gini index, the cross-entropy will take on a small value if the mth node is pure
- When building a classification tree, either the Gini index or the cross-entropy are typically used to evaluate the quality of a particular split
- When pruning the three, any three approaches (including classification error rate) can be used but the classification error rate is preferable if prediction accuracy of the final pruned tree is the goal
- Decision trees can be constructed even in the presence of qualitative predictor variables where a split on one of these variables amounts to assigning some of the qualitative values to one branch and assigning the remaining to the other branch

- Linear regression assumes a model of the form

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

whereas regression trees assume a model of the form

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

where $R_1, \ldots, R_M$ represent a partition of feature space

- If there is a nonlinear / complex relationship between the features and the response variable, then decision trees will outperform linear regression and other classical approaches
- Advantages and Disadvantages of Trees
  - Trees are very easy to explain to people, easier to explain than linear regression
  - Some people believe that decision trees more closely mirror human decision-making than do the regression and classification approaches
  - Trees can be be displayed graphically and are easily interpreted even by a non-expert
  - Trees can easily handle qualitative predictors without the need to create dummy variables
  - Unfortunately, trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches

6.2 Bagging, Random Forests, Boosting

- Bagging, random forests, and boosting are ways to improve predictive performances of trees
- The decision tree suffers from high variance
- Bootstrap aggregation, or bagging, is a general purpose procedure for reducing the variance of a statistical learning method
- Given a set of n independent observations $Z_1, \ldots, Z_n$, each with variance σ^2, the variance of the mean $\bar{Z}$ of the observations is given by σ^2/n; in other words, averaging a set of observations reduces variance
- In bootstrapping, generate B different bootstrapped training datasets by taking repeated samples from a single training data set and then train the method on the bth

bootstrapped training set in order to get $\hat{f}^{\cdot b}(x)$ and then average all of the predictions to obtain

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

This is called bagging.

- To apply bagging to regression trees, construct B regression trees using B bootstrapped training sets and average the resulting predictions; these trees are grown deep and are not pruned; hence each individual tree has high variance but low bias; averaging these B trees reduces the variance
- For classification trees: for a given test observation, record the class predicted by each of the B trees and take a majority vote (the overall prediction is the most commonly occurring class among the B predictions)
- The number of trees B is not a critical parameter with bagging; using a very large value of B will not lead to overfitting
- To estimate the test error of a bagged model, perform out of bag error estimation
- An out of bag prediction can be obtained by averaging the predicted responses or taking a majority vote for each ith observation; then an overall OOB MSE or classification error can be computed
- The resulting OOB error is a valid estimate of the test error for the bagged model since the response for each observation is predicted only using the trees that were not fit using that observation
- With B sufficiently large, OOB error is equivalently equivalent to LOOCV error
- The OOB approach for estimating the test error is convenient when performing bagging on large datasets for which cross-validation would be computationally expensive
- Bagging improves accuracy but hinders interpretation of the resulting model
- An overall summary of the importance of each predictor can be obtained using the RSS (for bagging regression trees) or the Gini index (for bagging classification trees)
- In the case of bagging regression trees, record the total amount that the RSS is decreased due to splits over a given predictor, averaged over all B trees; a large value indicates an important predictor; similarly, in the context of bagging classification trees, add up the total amount that the Gini index is decreased by splits over a given predictor, averaged over all B trees
- Random forests provide an improvement over bagged trees by way of a small tweak that decorrelates the trees

- When building a number of decision trees, each a 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; a fresh sample of m predictors is taken at each split
- In building a random forest, at each split in the tree, the algorithm is not even allowed to consider a majority of the available predictors; on average, $\frac{p-m}{p}$ of the splits will not even consider the strong predictor and so other predictors will have more of a chance - this can be thought of as decorrelating the trees, thereby making the average of the resulting trees less variable and hence more reliable
- The main difference between bagging and random forests is the choice of predictor subset size m; if a random forest is built using $m = p$, then this amounts simply to bagging
- Rule of thumb: $m = \sqrt{p}$
- As with bagging, random forests will not overfit when B increases; therefore, use a value of B sufficiently large for the error rate to have settled down
- Boosting is another approach for improving the predictions resulting from a decision tree
- Boosting works similarly to bagging except that the trees are grown sequentially: each tree is grown using information from previously grown trees; it does not involve bootstrap sampling but instead each tree is fit on a modified version of the original dataset
- Algorithm: Boosting for Regression Trees
  1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set
  2. For $b = 1, \ldots, B$, 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
     $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$
     (c) Update the residuals
     $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$
  3. Output the boosted model,
     $$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$$
- Unlike fitting a single large decision tree to the data, which amounts to fitting the data hard and potentially overfitting, the boosting approach instead learns slowly

- A tree is fit using the current residuals, rather than the outcome Y, as the response, which is then added to the new decision tree into the fitted function in order to update the residuals
- Each of these trees can be rather small, with just a few terminal nodes, determined by the parameter d in the algorithm; by fitting small trees to the residuals, $\hat{f}$ is slowly improved in areas where it does not perform well
- The shrinkage parameter λ slows the process down even further, allowing more and differently shaped trees to attack the residuals
- In general, statistical learning approaches that learn slowly tend to perform well
- Boosting classification trees are performed in a similar but slightly more complex way
- Boosting has three tuning parameters:
  - the number of trees B - boosting can overfit if B is too large; thus use cross-validation to select B
  - the shrinkage parameter λ - controls the rate at which boosting learns; very small λ can require a very large value of B to achieve good performance
  - the number d of splits in each trees - controls the complexity of the boosted ensemble; often $d = 1$ works well, in which case each tree is a stump consisting of a single split; in this case, the boosted ensemble is fitting an additive model, since each term involves only a single variable; more generally, d is the interaction depth and controls the interaction order of the boosted model, since d splits can involve at most d variables
- In boosting, because the growth of a particular tree takes into account the other trees that have already been grown, smaller trees are typically sufficient; using smaller trees can aid in interpretability as well; for instance, using stumps leads to an additive model

7 Support Vector Machines

7.1 Maximal Margin Classifier

- In a p-dimensional space, a hyperplane is a flat affine subspace of dimension $p - 1$; for example, in two dimensions, a hyperplane is a flat 1 dimension subspace (a line); in three dimensions, a hyperspace is a flat 2 dimension space (a plane); for $p > 3$ a hyperplane is hard to visualize
- A hyperplane can be defined mathematically; in the p-dimensional setting:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p = 0$$

  defines a p-dimensional hyperplane, in the sense that if a point $X = (X_1, X_2, \ldots, X_p)^T$ in p-dimensional space (i.e. a vector of length p) satisfies the above equation, then X lies on the hyperplane

- Suppose
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p > 0$$
Then X lies on one side of the hyperplane; on the other hand, if
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p < 0$$
then X lies on the other side of the hyperplane
- The hyperplane can be thought of as dividing a p-dimensional space into two halves
- Suppose data matrix X has dimensions $n \times p$, where there are n training observations in p-dimensional space, and that these observations fall into two classes, or, $y_1, \ldots, y_n \in \{-1, 1\}$ where -1 represents one class and 1 is the other class
- The goal is to develop a classifier based on the training data that will correctly classify the test observations $x^* = (x_1^*, \ldots, x_p^*)^\top$, a p-vector of observed features
- If a hyperplane that separates the training observations perfectly according to their class labels can be made, then the separating hyperplane has the properties that
$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} > 0 \text{ if } y_i = 1$$
$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} < 0 \text{ if } y_i = -1$$
Equivalently, a separating hyperplane with the property
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) > 0$$
for all $i = 1, \ldots, n$
- If this separating hyperplane exists, it can be used to construct a classifier: a test observation is assigned a class depending on which side of the hyperplane it is located, based on the sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \cdots + \beta_p x_p^*$
- If $f(x^*)$ is positive, then the test observation is assigned to class 1; if $f(x^*)$ is negative, then the test observation is assigned to class -1
- If $f(x^*)$ is far from zero, then x^* lies far from the hyperplane and so the class assignment for x^* can be trusted; on the other hand, if $f(x^*)$ is close to zero, then x^* is located near the hyperplane and so there is less certainty of the class assignment for x^*
- In general, if the data can be perfected separated using a hyperplane, then there will exist an infinite number of such hyperplanes because a given separating hyperplane can usually be shifted a tiny but up or down, or rotated, without coming into contact with any of the observations
- A natural choice in deciding which of the infinite possible separating hyperplanes to use is the maximal margin hyperplane (or optimal separating hyperplane), which is the separating hyperplane that is farthest from the training observations; it is the hyperplane that has the farthest minimum distance to the training observations

- A maximum margin classifier is a classifier that uses the maximum margin hyperplane to classify test observations
- If $\beta_0, \beta_1, \ldots, \beta_p$ are the coefficients of the maximum margin hyperplane, then the maximum margin classifier classifies the test observation x^* based on the sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \cdots + \beta_p x_p^*$
- The maximum margin hyperplane depends on only a small subset of the observations
- The maximum margin hyperplane constructed by solving the following optimization problem

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

  subject to

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

  and

  $$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M \quad \forall i = 1, \ldots, n$$

- The last constraint guarantees that each observation will be on the correct side of the hyperplane, provided that M is positive
- The first "constraint" allows the perpendicular distance from the ith observation to the hyperplane to be given by

  $$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip})$$

- M represents the margin of the hyperplane and the optimization problem chooses the β's to maximize M
- When a separating hyperplane does not exist, a soft margin must be used that can almost separate the classes; this generalization is known as the support vector classifier

7.2 Support Vector Classifiers

- The maximal margin hyperplane is extremely sensitive to a change in a single observation and so it can lead to overfitting the training dat a
- The support vector classifier, or soft margin classifier, seeks to create a hyperplane that does not perfectly separate the two classes so that there is greater robustness to individual observations and better classification of most of the training observations
- The support vector classifier allows some observations to be on the incorrect side of the margin, or even the incorrect side of the hyperplane
- An observation can be not only on the wrong side of the margin, but also on the wrong side of the hyperplane; when there is no separating hyperplane, such a situation is inevitable; observations on the wrong side of the hyperplane correspond to training observations that are misclassified by the support vector classifier

- In the support vector classifier, the hyperplane is chosen by solving the following optimization problem

$$\max_{\beta_0,\beta_1,\ldots,\beta_p,\varepsilon_1,\ldots,\varepsilon_n} M$$

subject to

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

and

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M(1 - \varepsilon_i)$$

where

$$\varepsilon_i \geq 0, \text{ and } \sum_{i=1}^{n} \varepsilon_i \leq C$$

where C is a nonnegative tuning parameter and M is the width of the margin to maximize

- The variables $\varepsilon_1, \ldots, \varepsilon_n$ are slack variables that allow individual observations to be on the wrong side of the margin or the hyperplane; the slack variable ε_i tells where the ith observation is located, relative to the hyperplane and relative to the margin

- If $\varepsilon_i = 0$, then the ith observation is on the correct side of the margin; if $\varepsilon_i > 0$, then the ith observation is on the wrong side of the margin (and thus this observation has violated the margin); if $\varepsilon_i > 1$, then it is on the wrong side of the hyperparameter

- The tuning parameter C bounds the sum o the ε_i's, thus determining the number and severity of the violations to the margin (and to the hyperplane); if $C = 0$, then no violations can be made and $\varepsilon_1 = \cdots = \varepsilon_n = 0$ (this is the maximal margin hyperplane problem); if $C > 0$, no more than C observations can be on the wrong side of the hyperplane

- As C increases, there is more tolerance of the violations to the margin and so the margin will widen; as C decreases, there is less tolerance of the violations to the margin and so the margin narrows

- C can be chosen via cross-validation

- C controls the bias-variance tradeoff of the statistical learning technique; when C is small, then the margins are narrow with little violations to training data thus having low bias but high variance; when C is larger, the margin is wider and more violations are allowed thus fitting the training data less hard thus having high bias but may have lower variance

- It turns out that only observations that either lie on the margin or that violate the margin will affect the hyperplane and hence the classifier obtained; this means that an observation that lies strictly on the correct side of the margin does not affect the support vector classifier

- Observations that lie directly on the margin or on the wrong side of the margin for their class are known as support vectors and they do affect the support vector classifier
- When C is large, then there are many support vectors; when C is small, there are fewer support vectors
- The support vector classifier is robust to the behavior of observations that far away from the hyperplane since the classifier's decision rule is based only on a potentially small subset of the training observations (the support vectors)

7.3 Support Vector Machines

- When there are non-linear class boundaries, the support vector classifier can't make a linear boundary
- In the case of the supper vector classifier, the problem of non-linear boundaries can be solved by enlarging the feature space to using polynomial functions of the predictors; rather than fitting a support vector classifier using p features $(X_1, \ldots, X_p)$, instead fit a support vector classifier using $2p$ features

  $$X_1, X_1^2, X_2, X_2^2, \ldots, X_p, X_p^2$$

  The optimization problem then becomes

  $$\max_{\beta_0, \beta_{11}, \beta_{12}, \ldots, \beta_{p1}, \beta_{p2}, \varepsilon_1, \ldots, \varepsilon_n} M$$

  subject to

  $$y_i \left(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{p} \beta_{j2} x_{ij}^2 \right) \geq M(1 - \varepsilon_i)$$

  and

  $$\sum_{i=1}^{n} \varepsilon_i \leq C \text{ and } \sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1$$

- This leads to a non-linear decision boundary because it is of the form $q(x) = 0$ where q is a quadratic polynomial
- The feature space can be enlarged to include higher-order polynomial terms or interaction terms or other functions by the use of support vector machine (SVM) that enlarges the feature space using kernels
- The solutions to the support vector classifier problem involves only the inner products of the observations, where if x_i and x_i' are two observations, then the inner product of them is

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

- It can be shown that the linear support vector classifier can be represented as

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

where there are n parameters α_i, $i = 1, \ldots, n$, one per training observations; To estimate the parameters $\alpha_1, \ldots, \alpha_n$ and β_0, only the $\binom{n}{2}$ inner products $\langle x_i, x_i' \rangle$ between all pairs of training observations are needed

- Note that α_i is nonzero only for the support vectors in the solution; that is, if a training observation is not a support vector, then its α_i equals zero; so if S is the collection of indices of these support points, then the solution function can be written as

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i \langle x, x_i \rangle$$

which has far less terms than the above equation for $f(x)$

- Suppose that each inner product is replaced by a generalization of the inner product of the form $K(x_i, x_i')$ where K is some kernel function that quantifies the similarity of two observations

- The linear kernel is

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

which returns the support vector classifier which is linear in the features

- The polynomial kernel of degree d ($d > 1$) is

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

which leads to a more flexible decision boundary; it amounts of fitting a support vector classifier in a higher-dimensional space involving polynomials of degree d

- When the support vector classifier is combined with a non-linear kernel such as the polynomial kernel, the resulting classifier is a support vector machine where the non-linear function has the form

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

- Another popular choice of a nonlinear kernel is the radial kernel, which takes the form

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

where γ is a positive constant

- The radial kernel has very local behavior, in the sense that only nearby training observations have an effect on the class label of a test observation; if a given test observation x^* is far from a training observation x_i in terms of Euclidean distance, then $\sum_{j=1}^{p}(x_j^* - x_{ij})^2$ will be large and so $K(x^*, x) = \exp\Big(-\gamma \sum_{j=1}^{p}(x_j^* - x_{ij})^2\Big)$ will be very tiny and so x_i will play virtually no role in $f(x^*)$

- The advantage of using a kernel rather than simply enlarging the feature space using functions of the original features is computational, that it amounts to the fact that when using kernels, only $K(x_i, x_i^*)$ needs to be computed for all $\binom{n}{2}$ distinct pairs i, i'; this can be done without explicitly working in the enlarged feature space

7.4 SVMs with More than Two Classes

- There are two popular approaches for handling multiple classes for SVMs: one vs one and one vs all

- A one-versus-one, or all-pairs, approach constructs $\binom{K}{2}$ SVMs (where there are $K > 2$ classes), each of which compares a pair of classes; a test observation is classified using each of the $\binom{K}{2}$ classifiers and then the number of times the test observation is assigned to each of the K classes is noted; the final classification is performed by assigning the test observation to the class to which it was most frequently assigned in these $\binom{K}{2}$ pairwise classifications

- The one-versus-all approach fits K SVMs, each time comparing one of the K classes to the remaining $K-1$ classes; let $\beta_{0k}, \beta_{1k}, \ldots, \beta_{pk}$ denote the parameters that result from fitting an SVM comparing the kth class (coded as $+1$) to the others (coded as -1); then test observation x^* is assigned to the class for which $\beta_{0k} + \beta_{1k}x_1^* + \beta_{2k}x_2^* + \cdots + \beta_{pk}x_p^*$ is largest, as this amounts to a high level of confidence that the test observation belongs to the kth class rather than to any of the other classes

7.5 Relationship to Logistic Regression

- The criterion for fitting the support vector classifier $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$ can be rewritten as

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

  where λ is a nonnegative tuning parameter

- When λ is large, then the β's are small, more violations to the margin are made and a low variance high bias classifier is made; when λ is small, then only a few violations are made, resulting in high variance but low bias classifier; a small value of λ amounts to a small value of C; note also that the $\lambda \sum_{j=1}^{p} \beta_j^2$ term is the ridge penalty term from ridge regression and plays a similar role in controlling the bias-variance trade-off for the support vector classifier

- Hence the criterion fitting the support vector classifier takes the "loss + penalty" form

$$\min_{\beta_0,\beta_1,\ldots,\beta_p} \{L(X, y, \beta) + \lambda P(\beta)\}$$

  where $L(X, y, \beta)$ is some function quantifying the extent to which the model, parameterized by β, fits the data (X, y) and $P(\beta)$ is a penalty function on the parameter vector β whose effect is controlled by a nonnegative tuning parameter λ

- The loss function takes the form of

$$L(X, y, \beta) = \sum_{i=1}^{n} \max[0, 1 - y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip})]$$

  This is known as the hinge loss, and the hinge loss function is closely related to the loss function used in logistic regression

- The loss function is exactly zero for observations for which $y_i(\beta_0+\beta_1 x_{i1}+\cdots+\beta_p x_{ip}) \geq 1$, when observations are on the correct side of the margin; this is why only support vectors play a role in the classifier obtained and the observations on the correct side of the margin do not affect it

- In contrast, the loss function for logistic regression is not exactly zero anywhere; but it is very small for observations that are far from the decision boundary

- Due to the similarity between their loss functions, logistic regression and the support vector classifier often give very similar results

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

- An extension of the SVM for regression (i.e. for a quantitative rather than a qualitative response) is called support vector regression

- In least squares regression, β's coefficients were calculated such that the sum of squared residuals was as small as possible; 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; this is an extension of the margin used in support vector classifiers to the regression setting

8 Unsupervised Learning

8.1 The Challenge of Unsupervised Learning

- In the supervised learning setting, there is access to a set of p features $X_1, \ldots, X_p$, measured on n observations, and a response Y also measured on those same n observations; the goal is to predict Y using $X_1, \ldots, X_p$

- In unsupervised learning, there is only a set of features $X_1, \ldots, X_p$ measured on n observations; there is no interest in predictions because there is no associated response variable Y - the goal is to discover interesting things about the measurements on $X_1, \ldots, X_p$
- Unsupervised learning is often more challenging than supervised learning because it tends to be more subjective and there is no simple goal for the analysis, such as prediction of a response
- Unsupervised learning is often performed as part of an exploratory data analysis
- It can be hard to assess the results obtained from unsupervised learning methods because there is no universally accepted mechanism for performing cross validation or validating results on an independent dataset
- In unsupervised learning, there is no way to check our work because the true answer is not known - the problem is unsupervised

8.2 Principal Components Analysis

- When faced with a large se of correlated variables, principal components allows us to summarize this set with a smaller number of representative variables that collectively explain most of the variability in the original set
- Principal component analysis (PCA) refers to the process by which principal components are computed and the subsequent use of these components in understanding the data
- PCA is an unsupervised approach because it involves only a set of features that do not have an associated response Y
- PCA finds a low-dimensional representation of a dataset that contains as much as possible of the variation; the idea is that each of the n observations lives in p-dimensional space but not all of these dimensions are equally interesting
- PCA 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
- Each of the dimensions found by PCA is a linear combination of the p features
- The first principal component of a set of features $X_1, \ldots, X_p$ is the normalized linear combination of the features

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

that has the largest variance; normalized means that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$

- The elements $\phi_{11}, \ldots, \phi_{p1}$ are referred to as the loadings of the first principal component; together, the loadings make up the principal components loading vector, $\phi_1 = (\phi_{11}, \phi_{21}, \ldots, \phi_{p1})^\top$; the loadings are constrained so they don't get arbitrarily large which could result in an arbitrarily large variance
- Given a $n \times p$ data set X, assume that each of the variables in X have been centered to have mean zero, then look for the linear combination of the sample feature values of the form

  $$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \cdots + \phi_{p1} x_{ip}$$

  that has the largest sample variance, subject to the constraint $\sum_{j=1}^{p} \phi_{j1}^2 = 1$
- The first principal component loading vector solves the optimization problem

  $$\max_{\phi_{11}, \ldots, \phi_{p1}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1} x_{ij} \right)^2 \right\} \text{ subject to } \sum_{j=1}^{p} \phi_{j1}^2 = 1$$

- The average of the $z_{11}, \ldots, z_{n1}$ will be zero since $\frac{1}{n} \sum_{i=1}^{n} x_{ij} = 0$; therefore the objective being maximized above is just the sample variance of the n values of z_{i1}
- The $z_{11}, \ldots, z_{n1}$ are referred to as the scores of the first principal component
- The above optimization problem can be solved by eigendecomposition
- The loading vector ϕ_1 with elements $\phi_{11}, \ldots, \phi_{p1}$ defines a direction in feature space along which the data vary the most; if the n data points $x_1, \ldots, x_n$ are projected onto this direction, the projected values are the principal component scores $z_{11}, \ldots, z_{n1}$ themselves
- After the first principal component Z_1 of the features has been determined, the second principal component Z_2 can be found
- The second principal component is the linear component of $X_1, \ldots, X_p$ that has maximal variance out of all linear combinations that are uncorrelated with Z_1; the second principal component scores $z_{12}, z_{22}, \ldots, z_{n2}$ take the form

  $$z_{i2} = \phi_{12} x_{i1} + \phi_{22} x_{i2} + \cdots + \phi_{p2} x_{ip}$$

  where ϕ_2 is the second principal component loading vectors, with elements $\phi_{12}, \phi_{22}, \ldots, \phi_{p2}$
- Constraining Z_2 to be uncorrelated with Z_1 is equivalent to constraining the direction ϕ_2 to be orthogonal to the direction ϕ_1
- To find ϕ_2, a similar optimization problem as above is solved with ϕ_2 replacing ϕ_1 and with the additional constraint that ϕ_2 is orthogonal to ϕ_1
- In a larger dataset with $p > 2$ variables, there are multiple distinct principal components and they are defined in a similar manner

- Once the principal components are computed, it can be plotted against each other in order to produce a low-dimensional view of he data; geometrically, this amounts to projecting the original data down into the subspace spanned by $\phi_1, \phi_2, \ldots$ and plotting the projected points
- Principal component loading vectors can be described as the directions in feature space along which the data vary the most; an alternative interpretation for principal components is that it provides low-dimensional linear surfaces that are closest to the observations
- The first principal component loading vector 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 notion of principal components as the dimensions that are closest to the N observations extends beyond just the first principal component; for instance, the first two principal components of a dataset span the plane that is closest to the n observation, in terms of average squared Euclidean distance; the first three principal components of a dataset span the three-dimensional hyperplane that is closest to the n observations, and forth
- Together, using this interpretation, 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 ith observation x_{ij}; this representation can be written as
$$x_{ij} \approx \sum_{m=1}^{M} z_{im}\phi_{jm}$$
assuming the original data matrix X is column-centered
- Together, the M principal component score vectors and M principal component loading vectors can give a good approximation to the data when M is sufficiently large; when $M = \min(n-1, p)$, then the representation is exact: $x_{ij} = \sum_{m=1}^{M} z_{im}\phi_{jm}$
- Before PCA is performed, the variables should be centered to have mean zero; furthermore, the results obtained when performing 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, variables are typically scaled to have a standard deviation of one before performing PCA
- In certain settings, variables may be measured in the same units and so it may not be needed to scale the variables to have standard deviation of one before performing PCA
- Each principal component loading vector is unique, up to a sign flip - different softwares can give the same vectors but with different signs; the signs may differ because each principal component loading vector specifies a direction in p-dimensional space; flipping the sign has no effect as the direction does not change

- 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 total variance present in a dataset (assuming that the variables have been centered to have mean zero) is defined as

$$\sum_{j=1}^{p} \text{Var}[X_j] = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2$$

  and the variance explained by the mth principal component is

$$\frac{1}{n} \sum_{i=1}^{n} z_{im}^2 = \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2$$

  Therefore, the PVE, or proportion of variance explained, by the mth principal component is given by

$$\frac{\sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{jm} x_{ij} \right)^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2}$$

- The PVE of each principal component is a positive quantity; in order to compute the cumulative PVE of the first M principal components, simply sum the above quantity over each of the first M PVEs; in total, there are $\min(n-1, p)$ principal components and their PVEs sum to one
- A scree plot is a plot that shows the proportion of variance explained by each of the principal components
- There is no simple answer in knowing what is the smallest number of principal components required to get a good understanding of the data
- A reasonable way to estimate this is by looking at a scree plot and looking for a point at which the proportion of variance explained by each subsequent principal component drops off; this is referred to as an elbow in the scree plot
- This type of visual analysis is ad hoc; unfortunately, there is no well-adapted objective way to decide how many principal components are enough
- If no interesting patterns are found in the first few principal components, then further principal components are unlikely to be of interest; conversely, if the first few principal components are interesting, then continue to look at subsequent principal components until no further interesting patterns are found
- If principal components are computed in a supervised analysis, such as in regression, there is a simple and objective way to determine how many principal components to use: 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

- Many statistical techniques, such as regression, classification and clustering, can be easily adapted to use the $n \times M$ matrix whose columns are the first $M \ll p$ principal component score vectors, rather than using the full $n \times p$ data matrix; this leads to less noisy results since it is often the case that the signal (not noise) in a dataset is concentrated in its first few principal components

8.3 Clustering Methods

- Clustering refers to a broad set of techniques for finding subgroups, or clusters, in a dataset; when observations are clustered in a dataset, the goal is 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
- Clustering is an unsupervised problem because we are trying to discover structure - in this case, distinct clusters - on the basis of a dataset
- Both clustering PCA try to simplify the data using a small number of summaries but their mechanisms are different: PCA looks to find a low-dimensional representation of the observations that explain a fraction of the variance; clustering looks to find homogeneous subgroups among the observations
- Since clustering is popular in many fields, there are many clustering methods, such as K-means clustering and hierarchical clustering
- In general, observations can be clustered on the basis of he features in order to identify subgroups among the observations, or features can be clustered on the basis of the observations in order to discover subgroups among the features
- K-means clustering is a simple approach for partitioning a dataset into K distinct, non-overlapping clusters
- To perform K-means clustering, first specify the desired number of clusters K; then the K-means algorithm will assign each observation to exactly one of the K clusters
- Let $C_1, \ldots, C_K$ denote sets containing the indices of the observations in each cluster; these sets satisfy two properties: (1) $C_1 \cup C_2 \cup \cdots \cup C_K = \{1, \ldots, n\}$; in other words, each observation belongs to at least one of the K clusters, and (2) $C_k \cap C_{k'} = \emptyset$ for all $k \neq k'$; in other words, the clusters are non-overlapping (no observation belongs to more than one cluster)
- 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 within-cluster variation for cluster C_k is a measure $W(C_k)$ of the amount by which the observations within a cluster differ from each other
- K-means clustering solves the problem

$$\min_{C_1,\ldots,C_K} \left\{ \sum_{k=1}^{K} W(C_k) \right\}$$

This says that the observations are partitioned into K clusters such that the total within-cluster variation, summed over all K clusters is as small as possible

- One popular way to define within-cluster variation involves 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 observations in the kth cluster; this formula gives the sum of all the pairwise squared Euclidean distances between the observations in the kth cluster, divided by the total number of observations in the kth cluster

- This creates the following optimization problem for K-means clustering

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

- There are almost K^n ways to partition n observations into K clusters; the algorithm below gives a good solution to the K-means clustering problem

- Algorithm: 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 kth cluster centroid is the vector of the p feature means for observations in the kth cluster
     (b) Assign each observation to the cluster whose centroid is closest (where closest is defined using Euclidean distance)

- This algorithm is guaranteed to decrease h value of the objective function at each step; to understand why, note that

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

where $\bar{x}_{kj} = \frac{1}{|C_k|} \sum_{i \in C_k} x_{ij}$ is the mean for feature j in cluster C_k; in step 2(a) the cluster means for each feature are the constants that minimize the sum-of-squared deviations and in step 2b, relocating the observations can only improve the above objective identity

- This means that as the algorithm is run, the clustering obtained will continually improve until the result no longer changes; the objective function can never increase; when the result no longer changes, a local optimum has been reached

- 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; thus it is important to run the algorithm multiple times from different random initial configurations from which a best solution is selected, i.e. the one for which the objective function is the smallest
- To perform K-means clustering, the number of clusters to expect must be defined beforehand; the problem of selecting K is not simple
- Hierarchical clustering is an alternative approach which does not require specifying a particular choice of K
- Hierarchical clustering has an added advantage over K-means clustering in that it results in a tree-based representation of the observations, called a dendrogram
- The most common type of hierarchical clustering is bottom-up, or agglomerative, clustering, which refers to the fact that a dendrogram (typically depicted as an upside-down tree) is build starting from the leaves and combining clusters up to the trunk
- To interpret a dendrogram, each leaf represents an observation; when moving up the tree, some leaves start to fuse into branches; this means that the observations are similar to each other; when moving further up, the branches themselves fuse, either with leaves or other branches; the earlier (lower in the tree) fusions occur, the more similar the groups of observations are to each other; on the other hand, observations that fuse later (near the top of the tree) can be quite different
- For any two observations, look for the point in the tree where branches containing these two observations are first fused; the height of this fusion, as measured on the vertical axis, indicates how different the two observations are
- 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 very different
- There are 2^{n-1} possible reorderings of the dendrogram, where n is the number of leaves; this is because at each of the $n-1$ points where fusions occur, the positions of the two fused branches could be swapped without affecting the meaning of the dendrogram; therefore, conclusions about the similarity of two observations cannot be drawn based on their proximity along the horizontal axis; conclusions about the similarity of two observations can only be drawn based on the location on the vertical axis where branches containing those two observations first are fused
- To identify clusters on a dendrogram, make a horizontal cut across the dendrogram; the distinct set of observations beneath the cut can be interpreted as clusters
- Cuts can be made at any level; further cuts can be made as one descends the dendrogram in order to obtain any number of cluster, between 1 (corresponding to no cut) and n (corresponding to a cut at height 0, so that each observation is in its own cluster); in other words, the height of the cut to the dendrogram serves the same role as the K in K-means clustering: it controls the number of clusters obtained

- One single dendrogram can be used to obtain any number of clusters; in practice, people often look at the dendrogram and select by eye a sensible number of clusters, based on the heights of the fusion and the number of clusters desired
- 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; on an arbitrary dataset, this assumption of hierarchical structure may be unrealistic
- Hierarchical clustering can sometimes give worse (i.e. less accurate) results than K-means clustering for a given number of clusters in situations where splitting up into groups do not result in nested clusters
- Algorithm: 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.
- The concept of dissimilarity between a pair of groups of observations is defined by linkage; the four most common types of linkage are given below

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.

- Average, complete and single linkage are most popular among statisticians
- Average and complete linkage are generally preferred over single linkage as they tend to yield more balanced dendrograms; centroid linkage is often used in genomics, but

suffers from a major drawback in that an inversion can occur, whereby two clusters are fused a height below either of the individual clusters in the dendrogram, leading to difficulties in visualization and interpretation of the dendrogram

- The dissimilarities computed in step 2b of the hierarchical clustering algorithm will depend on the type of linkage used, as well as on the choice of dissimilarity measure; hence, the resulting dendrogram typically depends quite strongly on the type of linkage used
- Correlation-based distance is another dissimilarity measure; it considers two observations to be similar if their features are highly correlated, even though the observed values may be far apart in the terms of Euclidean distance; it is computed between the observation profiles for each pair of observations
- Correlation-based distance focuses on the shapes of observation profiles rather than their magnitudes
- 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; if the variables are scaled to have standard deviation one before the inter-observation dissimilarities are computed, then each variable will in effect be given equal importance in the hierarchical clustering performed; it may also be beneficial to scale the variables to have standard deviation one if they are measured on different scales
- In order to perform clustering, some decisions should be made
  - Should the observations or features first be standardized in some way? For instance, maybe the variables should be centered to have mean zero and scaled to have standard deviation one
  - In the case of hierarchical clustering,
    - What dissimilarity measure should be used?
    - What type of linkage should be used?
    - Where should the dendrogram be cut in order to obtain clusters?
  - In the case of K-means clustering, how many clusters should be looked for in the data?
- In practice, try different choices and look for the one with the most useful or interpretable solution
- There are a number of techniques for assigning a p-value to a cluster in order to assess whether there is more evidence for the cluster than one would expect due to chance;

however, there has been no consensus on a single best approach for knowing whether the clusters that have been found represent true subgroups in the data, or whether they are simply a result of clustering the noise

- Both K-means and hierarchical clustering will assign each observation to a cluster; however, sometimes this might not be appropriate
- Suppose that most of the observations truly belong to a small number of subgroups and a small subset of the observations are different from each other and from all other observations; then since the clustering algorithms force each 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 alternative approach for accommodating the presence of such outliers; these amount to a soft version of K-means clustering
- Clustering methods are generally not very robust to perturbations to the data; for instance, suppose n observations are clustered and then clustered again after removing a subset of the n observations at random; the two sets of clusters obtained are expected to be the same but in fact won't be
- Clustering can be a very useful and valid statistical tool if used properly; perform clustering with different choices of parameters and look at the full set of results in order to see what patterns consistently emerge
- Since clustering can be non-robust, cluster subsets of the data in order to get a sense of the robustness of the clusters obtained
- Be careful about how the results of a clustering analysis are reported; these results should be taken as the absolute truth about a dataset; rather, they should constitute a starting point for the development of a scientific hypothesis and further study, preferably on an independent dataset