A GUIDE AND SOLUTION MANUAL TO THE ELEMENTS OF STATISTICAL LEARNING

by

JAMES CHUANBING MA

Under the direction of WILLIAM MCCORMICK

ABSTRACT

This Master's thesis will provide R code and graphs that reproduce some of the figures in the book *Elements of Statistical Learning.* Selected topics are also outlined and summarized so that it is more readable. Additionally, it covers some of the solutions to the problems for chapters 2, 3, and 4.

INDEX WORDS: Elements of Statistical Learning, Solution Manual, Guide, ESL Guide

A GUIDE AND SOLUTION MANUAL TO THE ELEMENTS OF STATISTICAL LEARNING

By

JAMES CHUANBING MA

B.S., Emory University, 2008

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2014

A GUIDE AND SOLUTION MANUAL TO THE ELEMENTS OF STATISTICAL LEARNING

by

JAMES CHUANBING MA

Major Professor: William Mccormick

Committee: Jaxk Reeves
Kim Love-Myers

Electronic Version Approved:

Julie Coffield Interim Dean of the Graduate School
The University of Georgia
December 2014

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The *Elements of Statistical Learning* is a popular book on data mining and machine learning written by three statistics professors at Stanford. The book is intended for researchers in the field and for people that want to build robust machine learning libraries and thus is inaccessible to many people that are new into the field.

On Amazon, roughly 1 out of 5 people[1] find the book too difficult to read and went as far as to call the book a "disaster". There are many instances of the expressions "it is easy to show" or "the exercise is left to the reader". Often times for the novice reader, these problems are not so easy to show. My goal in writing the thesis is to provide clarity to the book for novice readers. Therefore, my goal is not to reproduce the book, but to act as a supplement that covers parts of the book the author overlooks. In particular, my contributions are as follows:

- Derive problems that are overlooked ("it is easy to show").
- Provide solutions to exercises that can be understood to the novice reader.
- Provide R code that the reader can copy and paste.

With that said, there is a level requirement for reading this thesis. Specifically, I assume that the reader is has taken courses in calculus, probability, linear algebra, and linear regression. I will assume that the reader is comfortable with calculus topics such as gradients, derivatives, and vector spaces. For probability, an understanding of random variables, probability mass functions, cumulative distribution functions, expectation, variance and covariance, and basic

multivariate distributions is required. For linear algebra, the reader should have a solid understanding of basic matrix operations, inverses, norms, determinants, eigenvalues and eigenvectors, and definiteness of matrices. And for linear regression, the reader should be familiar with sum-of-squared errors, least squares estimation of parameters, and general hypothesis testing.

This thesis is an introduction and covers Chapters 2 (Overview of Supervised Learning), 3 (Linear Regression), and 4 (Classification). An updated copy with Chapters 7(Model Validation), 8 (Model Inference), 9 (Additive Models), and 10 (Boosted Trees) is intended by mid-January.

This manual is in no way complete and will be an ongoing project after graduating. Please refer to the site[1] below if you are interested in new updates or contributing to the project. Suggestions and comments for improvement are always welcome!

[1] http://eslsolutionmanual.weebly.com

CHAPTER 2

OVERVIEW OF SUPERVISED LEARNING

2.1 Introduction

This section goes over *mathematical notation, least squares and nearest neighbors, statistical decision theory, and the bias-variance decomposition*.

2.2 Mathematical Notation

The mathematical notation adopted in this guide is identical to the one used in the book and is summarized below.

- We bold matrices: $\mathbf{X} \in R^{n \times p}$ is a matrix with *n* rows and *p* columns.
- We denote the *j*th column of matrix $\mathbf{X}$ as $\mathbf{x}_j$:

$$\mathbf{X} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_p \\ | & | & & | \end{bmatrix}.$$

- We denote the *i*th row of matrix $\mathbf{X}$ as $\mathbf{x}_i^T$:

$$\mathbf{X} = \begin{bmatrix} - & \mathbf{x}_1^T & - \\ - & \mathbf{x}_2^T & - \\ & \vdots & \\ - & \mathbf{x}_n^T & - \end{bmatrix}$$

- Generic vectors are capitalized X and observed vectors are in lowercase x.

- The *i*th observation of vector x is denoted x_i:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

- We denote Y as a quantitative output and takes values in the real numbers.
- We denote G as a qualitative output.

In the text, there are many terms that are not explained so we will try to define them here. We denote x_i as the *input* variable (also called features) and y_i the output variable (also called target) that we are trying to predict. A training example is the pair (x_i, y_i) and the *training set* is a list of n training examples $\{(x_i, y_i) : i = 1,2, \dots, n\}$ often denoted $\boldsymbol{T}$ in the text.

Given a training set, our goal is to learn a function (also called model) $f: X \to Y$ so that our function is "good" at mapping the inputs to the corresponding outputs. We will define what "good" means in Section 2.4. *Supervised learning* refers to these types of functions with labeled data.

2.3 Least Squares and Nearest Neighbors

On page 12 in Equation 2.6, the author provides the unique solution to the coefficient vector β as follows

$$\hat{\beta} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{T}y. \tag{1}$$

Recall that in linear regression, we find the solution to the parameter vector β by minimizing the sum-of-squared-errors written as follows

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

Now define the vector $\beta = \langle \beta_0, \beta_1, \dots, \beta_p \rangle$ so that $\beta \in R^{p+1}$. It is important to note that we have included the intercept term in the vector and thus the corresponding observed values x_i have an implicit 1 as its first element so that we get

$$x_i = \langle x_{i1}, x_{i2}, \dots, x_{ip} \rangle \rightarrow \langle 1, x_{i1}, x_{i2}, \dots, x_{ip} \rangle \quad (3)$$

after including the intercept term.

Then we can rewrite the above quantity in vector form as follows

$$RSS(\beta) = \sum_{i=1}^{N} (y_i - x_i^T \beta)^2 . \quad (4)$$

To get the above quantity in matrix form, we will introduce the design matrix $\mathbf{X} \in R^{N \times (p+1)}$ that contains the training input vectors x_i along the rows:

$$\mathbf{X} = \begin{bmatrix} - & x_1^T & - \\ - & x_2^T & - \\ & \vdots & \\ - & x_n^T & - \end{bmatrix}$$

and define the vector $y \in R^N$ to be the vector with the training labels:

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}.$$

Then we can easily show that

$$y - \mathbf{X}\beta = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} x_1^T\beta \\ x_2^T\beta \\ \vdots \\ x_n^T\beta \end{bmatrix}$$

$$= \begin{bmatrix} y_1 - x_1^T\beta \\ y_2 - x_2^T\beta \\ \vdots \\ y_n - x_n^T\beta \end{bmatrix}$$

so that we can rewrite the residual-sum-of-squared errors in matrix form as

$$RSS(\beta) = (y - \mathbf{X}\beta)^T(y - \mathbf{X}\beta) \tag{5}$$

$$= y^T y - 2\beta^T \mathbf{X}^T y + \beta^T \mathbf{X}^T \mathbf{X}\beta \tag{6}$$

Now, to minimize the function, set the derivative to zero

$$\frac{\partial RSS(\beta)}{\partial \beta} = -2\mathbf{X}^T y + 2\mathbf{X}^T \mathbf{X}\beta = 0 \tag{7}$$

$$\Rightarrow \quad \mathbf{X}^T \mathbf{X}\beta = \mathbf{X}^T y \tag{8}$$

$$\Rightarrow \quad \hat{\beta} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^T y \tag{9}$$

and if **X** is full rank, we get the unique solution in β shown in (1).

It is important to point out that for linear regression, our model here is $f(x_i) = x_i^T \beta$ so that the prediction at an arbitrary point x_0 is $\hat{y}_0 = x_0^T \beta$.

On page 14, Equation (2.8) states that the *k*-nearest neighbors model has the form

$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i. \tag{10}$$

Where $N_k(x)$ is the neighborhood of *x* defined by the *k* closest points x_i in the training sample. The word *closest* implies a metric and here it is the Euclidean distance. More formally, define the $D = \{d(x_i, x): x_i \in T\}$ where $d(x_i, x)$ is any metric where for Euclidean distance it is $d(x_i, x) = ||x_i - x||_2$. Then the neighborhood is

$$N_k(x) = \{x_i: d(x_i, x) \leq d_k \text{ where } \mathrm{d_k} \text{is the } k\text{th smallest element of } D, x_i \in T\}.$$

2.4 Statistical Decision Theory

On page 18, Equation (2.9) of the book defines the squared error loss function as

$$L\big(Y, f(X)\big) = \big(Y - f(X)\big)^2 \tag{11}$$

which gives us the following expected prediction error

$$EPE(f) = E\left[\big(Y - f(X)\big)^2\right] \tag{12}$$

$$= \int \int [y - f(x)]^2 \Pr(y, x) dy dx \tag{13}$$

Now, we will fill in the steps that are skipped in the book. Recall that we can factor the joint density as $\Pr(X, Y) = \Pr(Y|X) \Pr(X)$.

Then

$$= \int_x \int_y [y - f(x)]^2 \Pr(y|x) \Pr(x)\, dydx$$

$$EPE(f) = E_X E_{Y|X}([Y - f(X)]^2|X). \tag{14}$$

Notice that by conditioning on X, we have freed the dependency of the function f on X and since the quantity $[Y - f]^2$ is convex, there is a unique solution. We can now minimize to solve for f

$$f(x) = \arg\min_f E_{Y|X}([Y - f]^2|X = x) \tag{15}$$

$$\Rightarrow \frac{\partial}{\partial f} \int [Y - f]^2 \Pr(y|x)\, dy = 0 \tag{16}$$

$$= \int \frac{\partial}{\partial f} [y - f]^2 \Pr(y|X)\, dy = 0 \tag{17}$$

$$= 2\int yPr(y|x)dy = 2f \int \quad Pr(y|x)\, dy = 0 \tag{18}$$

$$\Rightarrow 2E[Y|X] = 2f \tag{19}$$

$$\Rightarrow f = E[Y|X = x]. \tag{20}$$

Thus we get the Equation (2.13) from the book.

Lets continue down this path and work out the problem from page 20, Equation (2.18) the author replaces the squared loss function with the absolute loss function: $E[|Y - f(X)|]$. The equation from the book is as follows

$$\hat{f}(x) = \text{median}(Y|X = x) \tag{21}$$

then the expected prediction error for absolute loss is almost identical to that of squared loss. If you become confused with this example, refer back to the squared loss derivation for the first few steps. We can write the expected prediction error for absolute loss as follows

$$EPE(f) = E[|Y - f(X)|] \tag{22}$$

$$= \int \int |y - f(x)| \Pr(y, x) dy dx \tag{23}$$

then recall that by factoring joint density, we can rewrite the above quantity as

$$EPE(f) = E_X E_{Y|X}(|Y - f(X)||X). \tag{24}$$

and we free the dependency of *f* on *x* again so that it suffices to minimize the conditional density of *y*

$$f(x) = \arg\min_f E_{Y|X}(|Y - f||X = x) \tag{25}$$

$$= \frac{\partial}{\partial f} \int |Y - f| \Pr(y|X)\, dy = 0 \tag{26}$$

From here, the integration is a little sophisticated and requires a branch of analysis known as measure theory. For this reason, we will provide an approximation that does not address the smaller details. By using the law of large numbers, we get the following

$$\int |Y - f| \Pr(y|X) dy = Lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} |Y_i - f| \approx \frac{1}{n} \sum_{i=1}^{n} |Y_i - f| \tag{27}$$

and so that we can get an approximation that converges when *n* is large. For this example, we will use the approximation, the last term in the above equation. Notice here that the absolute function is piecewise

$$|Y_i - f| = \begin{cases} Y_i - f, & Y_i - f > 0 \\ f - Y_i, & Y_i - f < 0 \\ 0, & Y_i = f \end{cases} \tag{28}$$

so that taking the derivative is not continuous at zero

$$\frac{\partial}{\partial f}|Y_i - f| = \begin{cases} -1, & Y_i - f > 0 \\ 1, & Y_i - f < 0 \\ 0, & Y_i = f \end{cases}. \tag{29}$$

Now, we can introduce a new function, the sign function to make the definition clearer

$$sign(x) = \begin{cases} 1, & x > 0 \\ -1, & x < 0 \\ 0, & otherwise \end{cases} \tag{30}$$

so that substituting it back in we get

$$\approx \frac{\partial}{\partial f}\frac{1}{n}\sum_{i=1}^{n}|Y_i - f| = 0 \tag{31}$$

$$= \frac{1}{n}\sum_{i=1}^{n} -sign(Y_i - f) = 0 \tag{32}$$

$$= \sum_{i=1}^{n} sign(Y_i - f) = 0. \tag{33}$$

At what value of *f* does the above quantity hold? It holds when there is an equal number of positive and negative values; that is, where

$$card(Y_i - f > 0) = card(Y_i - f < 0). \tag{34}$$

The value of f where that is true is the median. Recall that the median can be found by sorting a finite list of numbers from lowest value to highest value and picking the middle one. When there is an odd number of observations, there is a single number that divides the set (e.g. the median of {2,3,4,5,6} is 4). When there is an even number, there is a range of values that can divide the set (e.g. the median of {2,4,6,8} is the any value in $f \in (4,6)$). In conclusion, we have shown that $\hat{f}(x) = \text{median}(Y|X = x)$.

On page 24, Equation (2.25) is nested in Equation (2.27) so we will derive Equation (2.27). The derivation is a little tedious and algebraic but it is an important one. The expected (squared) prediction error at an arbitrary point x_0 is

$$EPE(x_0) = E_{y_0|x_0} E_T (y_0 - \hat{y}_0)^2. \tag{35}$$

The ***T*** here is the training set that we defined earlier, $y_0|x_0$ and y_0 are identical quantities that represent a random variable conditioned on x_0. Additionally, $\hat{y}_0$ is our prediction at x_0, that is $\hat{y}_0 = \hat{f}(x_0)$ where $\hat{f}$ is our model estimate of the true model f fitted on the training data. We assume the model

$$Y = f(X) + \epsilon \tag{36}$$

where $\epsilon \sim N(0, \sigma^2)$ and independently distributed. To start the proof, we use a little trick by inserting the true function $f(x_0)$ as follows

$$\begin{aligned} EPE(x_0) &= E_{y_0|x_0} E_T (y_0 - \hat{y}_0)^2 \\ &= E_{y_0|x_0} E_T ((y_0 - f(x_0)) + (f(x_0) - \hat{y}_0))^2. \end{aligned} \tag{37}$$

Notice that by subtracting and adding $f(x_0)$, the value of the function does not change. It is also important to point out here that the model $\hat{f}(x_0) = \hat{y}_0$ is fitted on the training set by our definition in the introduction. Thus, the only term in the above function that depends on $\boldsymbol{T}$ is $\hat{y}_0$. We then continue with the problem and factor out the squared term as

$$= E_{y_0|x_0}E_T\left[\left(y_0 - f(x_0)\right)^2\right] + 2E_{y_0|x_0}E_T\left[\left(y - f(x_0)\right)\cdot\left(f(x_0) - \hat{y}_0\right)\right]$$

$$+E_{y_0|x_0}E_T[(f(x_0) - \hat{y}_0)^2]. \quad (38)$$

The first quantity $E_{y_0|x_0}E_T\left[\left(y_0 - f(x_0)\right)^2\right]$ is independent of the training set so we can reduce it to

$$E_{y_0|x_0}\left[\left(y_0 - f(x_0)\right)^2\right] = \int \left(y_0 - f(x_0)\right)^2 \Pr(y_0|x_0)\, dy \quad (39)$$

where $E[y_0] = f(x_0)$, the true mean. Then notice that this is just the function for the conditional variance that we specified as σ^2.

For the second quantity, we can factor out the middle term:

$$E_{y_0|x_0}E_T\left[\left(y_0 - f(x_0)\right)\cdot\left(f(x_0) - \hat{y}_0\right)\right] = E_{y_0|x_0}E_T[y_0 f(x_0) - y_0\hat{y}_0 - f(x_0)^2 + f(x_0)\hat{y}_0] \quad (40)$$

and by linearity of expectation,

$$= E_{y_0|x_0}E_T[y_0 f(x_0)] - E_{y_0|x_0}E_T[y_0\hat{y}_0] - E_{y_0|x_0}E_T[f(x_0)^2] + E_{y_0|x_0}E_T[f(x_0)\hat{y}_0]. \quad (41)$$

We stated that only $\hat{y}_0$ is dependent on $\boldsymbol{T}$. Additionally, notice that $f(x_0) = E[y_0]$ so it is a constant term so that we can reduce the above quantity to

f是我们对函数的估计，我们希望f(x)=E(y)

$$= f(x_0)E_{y_0|x_0}[y_0] - E_{y_0|x_0}[y_0 E_T[\hat{y}_0]] - f(x_0)^2 + f(x_0)E_{y_0|x_0}E_T[\hat{y}_0]. \quad (42)$$

Notice that for the first term, the relationship

$$E_{y_0|x_0}[y_0] = \int y_0 \Pr(y_0|x_0)\, dy = f(x_0). \tag{43}$$

For the second term, the relationship is

$$E_{y_0|x_0}\left[y_0 E_T[\hat{y}_0]\right] = E_T[\hat{y}_0] E_{y_0|x_0}[y_0] = E_T[\hat{y}_0] f(x_0) \tag{44}$$

and the last term, we get

$$f(x_0) E_{y_0|x_0} E_T[\hat{y}_0] = f(x_0) E_T[\hat{y}_0] \tag{45}$$

and thus adding all of these terms together, the first and third quantities cancel, the second and last quantities cancel so that the entire middle term equals 0.

Now, we are left with the last term $E_{y_0|x_0} E_T[(f(x_0) - \hat{y}_0)^2]$ which we can factor out as follows

$$= E_{y_0|x_0} E_T[f(x_0)^2] - 2E_{y_0|x_0} E_T[f(x_0)\hat{y}_0] + E_{y_0|x_0} E_T[\hat{y}_0^2]. \tag{46}$$

We are going to use another property here, which is that

$$E_{y_0|x_0} E_T[(f(x_0) - E_T[\hat{y}_0])^2] = E_{y_0|x_0} E_T[f(x_0)^2] - 2E_{y_0|x_0} E_T[f(x_0)\hat{y}_0] + E_{y_0|x_0} E_T[\hat{y}_0]^2 \tag{47}$$

and then substitute it back into the original equation to get

$$= E_{y_0|x_0} E_T[(f(x_0) - E_T[\hat{y}_0])^2] - E_{y_0|x_0} E_T[\hat{y}_0^2] + E_{y_0|x_0} E_T[\hat{y}_0^2] \tag{48}$$

and we can reduce it as we did in the previous quantities as

$$= [(f(x_0) - E_T[\hat{y}_0])^2] + E_T[\hat{y}_0^2] - E_T[\hat{y}_0^2] \quad (49)$$

$$= Bias(\hat{f})^2 + Var(\hat{f}). \quad (50)$$

Recall that $\hat{y}_0 = \hat{f}(x_0)$ and so we use them interchangeably. The first quantity is called the squared bias of an estimator (not to be confused with the vernacular bias meaning being partial to something). Our estimator here is our approximating function $\hat{f}$ and the bias is the difference between the expected value of our estimator and the true function. Often times, it is desirable to have an unbiased estimator but, as we will soon see, this is not always the case. The variance arises from our basic probability class $Var(\hat{y}_0) = E[\hat{y}_0^2] - E[\hat{y}_0]^2$ and tells us how much our function varies across different training sets. Notice also that the third quantity above is also the same one in Equation (2.5) so that we have solved for the $MSE(\hat{f})$ here.

So recapping the expected squared prediction error at x_0 equals

$$EPE(x_0) = \sigma^2 + Bias(\hat{f})^2 + Var(\hat{f}) \quad (51)$$

where σ^2 is the irreducible error. This is a famous result in statistics and is also known as the bias-variance decomposition. This completes Equations (2.25) and (2.27).

On page 31, Equation (2.35) gives us the equation for the likelihood of the data under the least squares model as

$$L(\theta) = -\frac{n}{2}\log(2\pi) - n\log\sigma - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - f_\theta(x_i))^2. \quad (52)$$

We will derive that here. Recall that for least squares, we assume the model

$$Y = X^T\beta + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$. Notice that the density of ϵ_i is

$$\Pr(\epsilon_i) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{{\epsilon_i}^2}{2\sigma^2}\right) \tag{53}$$

then $\epsilon_i = y_i - {x_i}^T\beta$ so that we get the density

$$\Pr(y_i|x_i, \beta) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - x_i^T\beta)^2}{2\sigma^2}\right) \tag{54}$$

where here we assume that x_i and β are fixed and y_i is random. We can also write the distribution of $y_i|x_i \sim N(x_i^T\beta, \sigma^2)$. Now suppose we have our training data, $\boldsymbol{T}$, and we want to estimate the parameters. As stated in the text, we would like to find the value of β that maximizes the probability of the data. When we write the quantity as $\Pr(y|\mathbf{X}, \beta)$, this assumes that β is fixed. Instead, we will call this the likelihood and rewrite it

$$L(\theta) = P(y|\mathbf{X}, \beta) \tag{55}$$

so that now β is a function of the likelihood. Since we assume that our training examples our drawn independently, we can write the likelihood of the data as

$$L(\beta) = \prod_{i=1}^{n} p(y_i|x_i, \beta) \tag{56}$$

$$= \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_i - x_i^T\beta)^2}{2\sigma^2}\right) \tag{57}$$

Maximizing any monotonic transformation of the likelihood function is also the maximum. That is, given a monotonic function $g(x)$, we have that for the likelihood $L(x_1) > L(x_2)$ for any x_1, x_2 in the domain of L implies that $g(x_1) > g(x_2)$. A common transformation used in

statistics is the log transformation since it turns the products of the likelihood into sums for the log likelihood. By applying the log transformation, we get

$$\log\big(L(\beta)\big) = \ell(\beta) = \sum_{i=1}^{n} \log\left(\frac{1}{\sqrt{2\pi\sigma}} exp\left(-\frac{(y_i - x_i^T\beta)^2}{2\sigma^2}\right)\right) \tag{58}$$

$$= n\log\frac{1}{\sqrt{2\pi\sigma}} - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - x_i^T\beta)^2 \tag{59}$$

$$= -\frac{n}{2}\log 2\pi - n\log\sigma - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - x_i^T\beta)^2 \tag{60}$$

which gives us Equation (2.35) from the book and maximizing the above quantity is the same as minimizing the quantity $\sum_{i=1}^{n}(y_i - x_i^T\beta)^2$since it is again a monotonic transformation and multiplying by (-1) turns a maximization problem into a minimization. Notice the surprising fact here; we get the least squares loss function from linear regression from maximum likelihood estimation.

To conclude this section, we illustrate the bias-variance tradeoff by reproducing Figure 2.11 from the book using nearest neighbors.

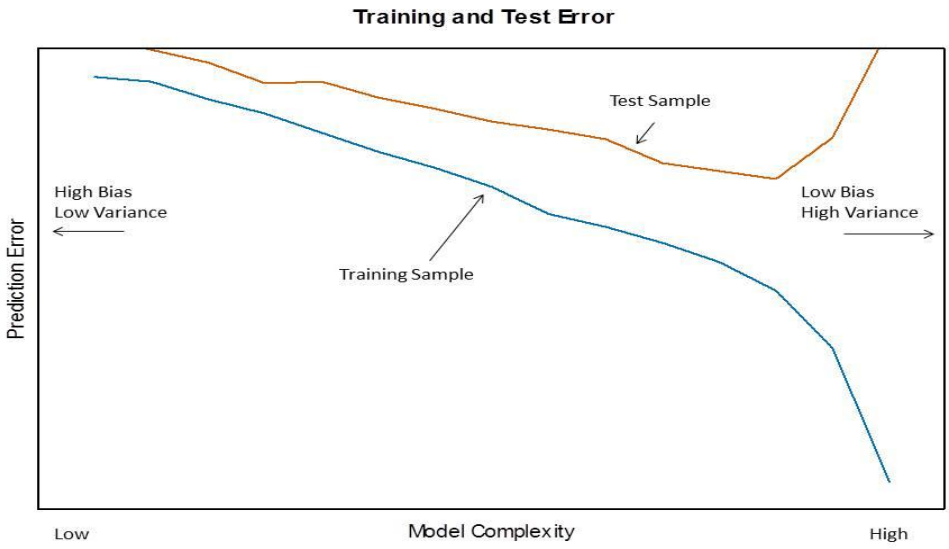

Figure 2.1 Bias-Variance Tradeoff.

The left side of Figure 2.1 represents a large value of k using nearest neighbors which has low variance in the test set but a high bias. The right side shows that as $k \to 1$, the prediction error on the training set falls to zero but that the error on the testing set is high and thus the model fails to generalize well.

Table 2.1. R Code for Figure 2.1.

```
library(FNN) # For nearest neighbor
library(lattice)
library(latticeExtra)
library(gridExtra)

# Color palette # http://www.cookbook-r.com/Graphs/Colors_%28ggplot2%29/
cbPalette <- c("#999999", "#E69F00", "#56B4E9", "#009E73", "#F0E442",
"#0072B2", "#D55E00", "#CC79A7")

# Define mean squared error
mse=function(x,y){return(mean((y-x)^2))}

# Generate random data
x1=rnorm(1000)
x2=x1+rnorm(1000)
x3=rnorm(1000)
```

```
y=x1+rnorm(1000)
data=data.frame(x1,x2,x3,y)

# Train/test (50/50) split
train=data[1:500,]
test=data[500:1000,]

# Instantiate error vectors
train_err=NULL
test_err=NULL

# Find train/test error for varying values of k
for(i in 15:1)
{
    # Nearest Neighbor model
    nearest_train <- knn.reg(train=train, test=train, y=y, k = i)
    nearest_test <- knn.reg(train=train, test=test, y=y, k = i)

    # MSE
    train_err=append(train_err,mse(nearest_train$pred,train$y))
    test_err=append(test_err,mse(nearest_test$pred,test$y))
}


# Plot
a1=xyplot(train_err~1:15,
    xlab=list(label='Model Complexity',cex=0.7),
    ylab=list(label='Prediction Error',cex=0.7),
    main=list(label='Training and Test Error',cex=0.75),
    scales = list(x = list(draw = FALSE),y=list(draw=FALSE)),
    panel=function(...){
        panel.lines(train_err,col=cbPalette[6])
})


a2=xyplot(test_err~1:15,
    panel=function(...){
        panel.lines(test_err,col=cbPalette[7])
})


grid.arrange(a1+a2,ncol=1)
```

2.5 Problems and Solutions

Exercise 2.1. *Suppose each of K-classes has an associated target t_k, which is a vector of all zeros, except a one in the kth position. Show that classifying to the largest element of $\hat{y}$ amounts to choosing the closest target,$min_k ||t_k - \hat{y}||$, if the elements of $\hat{y}$ sum to one.*

Proof: This is a classification problem defined in the book. Suppose we have a set T with K-elements (or classes) that hold the standard basis in R^K. That is, $T = \{(1,0,\dots,0)^T, (0,1,\dots,0)^T \dots, (0,0,\dots,1)^T\}$ and $t_k \in T$. In this problem, $\hat{y}$ is a K-dimensional vector with the element $\hat{y}_i$ being the probability that $\Pr(y_i = t_k)$. It is not clear what model $\hat{y}$ is generated from but presumably it is regression since $\hat{y}$ is continuous. We will not place any assumptions on the model.

To solve the equation, we can write it as

$$\min_k \left|\left|t_k - \hat{y}\right|\right| = \min_k \sum_{i=1}^{K} \left(t_{k,i} - y_i\right)^2$$

$$= \min_k \sum_{i=1}^{K} t_{k,i}^2 - 2t_{k,i}y_i + y_i^2$$

Then notice here that for the first term, when $k = i$, the quantity equals 1 else it is 0. Thus, $\sum_i t_{k,i}^2 = 1$ for all values of k. Likewise, the last term $\sum_i y_i^2$ is independent of k so that it is constant with respect to k. Finally, the middle term $\sum_i -2t_{k,i}y_i = -2y_i$ when $k = i$ and is 0 otherwise. Note that is also varies across different values of k so that it is a function of k. Then, we can rewrite the above function as a function of only the middle term as follows

$$= \min_k \sum_{i=1}^{K} -2t_{k,i}y_i.$$

And as we stated above, this is only non-zero at

$$= \min_k -2t_k y_k$$

$$\Leftrightarrow \min_k -y_k$$

and multiplying the above quantity by (-1), we can change the min to a max problem as follows

$$= \max_k y_k$$

and so we have shown that classifying the largest element $\hat{y}$ amounts to choosing the closest target.

Exercise 2.2. Show how to compute the Bayes decision boundary for the simulation example in Figure 2.5.

Proof: For this example, the data has two classes and each was generated by a separate mixture of Gaussians. That is, our generating density is $N(m_k, \frac{I}{5})$ is a weighted sum of 10 Gaussians generated from $N((0,1)^T, I)$.

The generation process is described in the table below. They are identical except for the first step.

Class 1	Class 2
1. 10 means generated from a bivariate Gaussian $N((0,1)^T, I)$. 2. 100 Samples selected as follows a. For each observation, m_k was selected with probability $\frac{1}{10}$. b. Then a sample was generated from the bivariate Gaussian $N(m_k, \frac{I}{5})$.	1. 10 means generated from a bivariate Gaussian $N((1,0)^T, I)$. 2. 100 Samples selected as follows a. For each observation, n_i was selected with probability $\frac{1}{10}$. b. Then a sample was generated from the bivariate Gaussian $N(n_i, \frac{I}{5})$.

Recall that the Bayes classifier says that we classify to the most probable class using the conditional distribution $\Pr(G|X)$. Hence, the decision boundary is the set of points that partitions the vector space into two sets; one for each class. On the decision boundary itself, the output label is ambiguous. Thus, the optimal Bayes decision boundary is defined as

$$Boundary = \left\{x: \max_{g \in G} \Pr(g|X = x) = \max_{k \in G} \Pr(k|X = x)\right\}.$$

That is, it is the set of points where the most probable class is tied between two or more classes. In our example, there are only two classes so that $card(G) = 2$.

Hence, we can rewrite the above quantity as

$$Boundary = \{x: \Pr(g|X = x) = \Pr(k|X = x)\}$$

$$= \left\{x: \frac{\Pr(g|X = x)}{\Pr(k|X = x)} = 1\right\}.$$

We can then rewrite the above quantity by using Bayes rule as follows

$$\frac{\Pr(g|X = x)}{\Pr(k|X = x)} = \frac{\Pr(X = x|g)\Pr(g)\,/\Pr(X = x)}{\Pr(X = x|k)\,\Pr(k)\,/\Pr(X = x)} = \frac{\Pr(X = x|g)\,\Pr(g)}{\Pr(X = x|k)\,\Pr(k)} = 1$$

because we have 100 points in each class, $\Pr(g) = \Pr(k)$ so we can remove those from the above equation. Then the boundary is $\{x: \Pr(X = x|g) = \Pr(X = x|k)\}$ which in this example, since we know the generating density to be Gaussian, we can rewrite

$$\Pr(X = x|g) = \prod_{k=1}^{10} \frac{1}{5\sqrt{2\pi}} \exp\left(-\frac{(x - m_k)^2}{2 \cdot 25}\right)$$

and since we know the log transformation is monotonic and preserves the ordering, we can log it and write it as

$$\log(\Pr(X = x|g)) = \sum_{k=1}^{10} \log\left(\frac{1}{5\sqrt{2\pi}}\right) - \frac{(x - m_k)^2}{2 \cdot 25}.$$

Now, equating class g and k to get the decision boundary, we get the following

$$Boundary = \left\{x: \sum_{k=1}^{10} \log\left(\frac{1}{5\sqrt{2\pi}}\right) - \frac{(x - m_k)^2}{2 \cdot 25} = \sum_{i=1}^{10} \log\left(\frac{1}{5\sqrt{2\pi}}\right) - \frac{(x - n_i)^2}{2 \cdot 25}\right\}$$

$$= \left\{x: \sum_{k=1}^{10} (x - m_k)^2 = \sum_{i=1}^{10} (x - n_i)^2\right\}$$

The exact boundary for figure 2.5 would depend on our generated means.

Exercise 2.4. The edge effect problem discussed on page 23 is not peculiar to uniform sampling from bounded domains. Consider inputs drawn from a spherical multinormal distribution $X \sim N(0, I_p)$. *The squared distance from any sample point to the origin has a* χ_p^2 *distribution with mean p. Consider a prediction point* x_0 *drawn from this distribution, and let* $a = x_0/||x_0||$ *be an associated unit vector. Let* $z_i = a^T x_i$ *be the projection of each of the training points on this direction.*

1. *Show that the* z_i *are distributed* $N(0,1)$.
2. *Show that the target point has expected squared distance* p *from the origin.*

Hence for p *=10, a randomly drawn test point is about 3.1 standard deviations from the origin, while all the training points are on average one standard deviation along direction a. So most prediction points see themselves as lying on the edge of the training set.*

Proof: There is a well-known property that states that a linear combination of mutually independent normal random variables is itself normal. We will not derive that proof but it can be easily found on the internet or in a probability textbook. That is, if

$$z = \sum_{i=1}^{p} a_i x_i$$

where each $x_i \sim N(0,1)$ then the mean is

$$E[z] = \sum_{i=1}^{p} a_i x_i$$

and the variance is

$$Var(z) = \sum_{i=1}^{p} a_i^2 Var(x_i)$$

and $z_i \sim N\big(E[z], Var(z)\big)$.

For (1), $z_i = a^T x_i$ so that the expectation is

$$E[z_i] = E[a^T x_i] = a^T E[x_i] = a^T 0$$

by linearity since a is constant and 0 is the $p \times 1$-dimensional vector of zeros. Notice here that a is constant although it is randomly drawn since once we draw it, we are using the same value and hence it is no longer random.

For the variance,

$$Var(a^T x_i) = a^T Var(x_i) a = a^T I_p a = ||a||_2^2 = \frac{x_0^T x_0}{||x_0||_2^2} = \frac{||x_0||_2^2}{||x_0||_2^2} = 1$$

by the property of variance where constant terms are squared. This leaves that the resulting distribution z_i to have mean 0 and variance 1.

Using that property, $z_i = a^T x$ states that z_i is a linear combination of normal distributions implies that z_i is itself normally distributed. We have shown that $z_i \sim N(0,1)$ and so that this completes the problem.

For (2), it is not clear from the problem what the "target" point is, but presumably it is X since it can be shown X has a squared expected distance p. Since X is a $p \times 1$-dimensional random vector generated from $N(0, \mathbf{I}_p)$, then the squared distance of X can be written conveniently in vector form as $X^T X = \sum_{i=1}^{p} x_i^2$. Notice that the covariance between x_i and x_j is 0 for all i, j but this is not imply the independence required by the chi-squared distribution. However, since the multivariate distribution is spherical, we will assume that the author implies that they are independently drawn. Then assuming independence, each x_i has mean 0 and variance of 1 and

$$x_i^T x_i = \sum_{i=1}^{p} x_i^2 \sim \chi_p^2$$

and so is distributed χ_p^2 with mean p.

Exercise 2.5. Derive Equation (2.27) where the expected prediction error at a point x_0 is as follows

$$\begin{aligned} EPE(x_0) &= E_{y_0|x_0} E_T (y_0 - \hat{y}_0)^2 \\ &= Var(y_0|x_0) + E_T[\hat{y}_0 - E_T\hat{y}_0]^2 + [E_T\hat{y}_0 - x_0^T\beta]^2 \\ &= Var(y_0|x_0) + Var_T(\hat{y}_0) + Bias^2(\hat{y}_0) \\ &= \sigma^2 + E_T x_0^T (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1} x_0 \sigma^2 + 0^2 \end{aligned} \qquad (2.27)$$

Proof : On page 11 of this guide, we showed the derivation of lines 1-3. For the last line 4, we assume that the relationship between Y and X is linear,

$$Y = X^T\beta + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$ and the model is fit by least squares. Then the first term $Var(y_0|x_0)$ is the conditional variance and since is distributed $N(0, \sigma^2)$, it equals σ^2. The least squares is unbiased under for the linear assumption so the last term is 0.

This leaves us with the second term which we can write as follows

$$Var_T(\hat{y}_0) = Var_T\left(x_0^T\hat{\beta}\right) = x_0^T Var_T\left(\hat{\beta}\right)x_0.$$

The multivariate covariance is a generalization of the scalar case for squaring constant terms under variance properties. That is, we used the property of covariance which is $Cov(\mathbf{A}x + a) = \mathbf{A}Cov(x)\mathbf{A}^T$ for $x \in R^p$. Keep in mind that we are assuming the x_i's are held fixed and y is random. Now, we only need to find the $Var_T(\hat{\beta})$. Recall that

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y$$

and

$$Y = X\beta + \epsilon$$

then this implies that

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{X}\beta + \epsilon)$$

$$\Rightarrow \hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\beta + (\mathbf{X}^T\mathbf{X})^{-1}X^T\epsilon$$

by doing some matrix operations. Notice that the left term $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\beta$ is non-random and so this leaves that the variance is

$$Var_T(\hat{\beta}) = Var((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon)$$

$$= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T Var_T(\epsilon) X (\mathbf{X}^T\mathbf{X})^{-1}$$

Since $Var_T(\epsilon) = \sigma^2$

$$= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\sigma^2$$

$$= (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2$$

then this implies that

$$Var_T(\hat{y}_0) = x_0^T(\mathbf{X}^T\mathbf{X})^{-1}x_0^T\sigma^2$$

and since none of the remaining variables are random with respect to the training data, this is equivalent to

$$= E_T x_0^T(\mathbf{X}^T\mathbf{X})^{-1}x_0^T\sigma^2$$

which concludes the problem.

2. Derive equation (2.28), making use of the cyclic property of the trace operator $[trace(AB) = trace(BA)]$*, and its linearity (which allows us to interchange the order of trace and expectation).*

$$E_{x_0}EPE(x_0) \sim E_{x_0}x_0^T Cov(X)^{-1}x_0\sigma^2/N + \sigma^2$$

$$= \text{trace}[Cov(X)^{-1}Cov(x_0)]\sigma^2/N + \sigma^2$$

$$= \sigma^2 \left(\frac{p}{N}\right) + \sigma^2 \tag{2.28}$$

which is the formula for expected prediction error.

Our assumption here is that the matrix X has mean 0 along the columns. Then, by the definition of covariance matrices, the covariance of a random vector X is

$$Cov(X) = E[(X-\mu)(X-\mu)^T] = E[XX^T] - \mu\mu^T$$

where $\mu = 0$ here since our matrix is centered. Then we get our sample covariance matrix to be

$$\widehat{Cov}(X) = \frac{\mathbf{X}^T\mathbf{X}}{N}.$$

To see this, recall that one way to view matrix products is

$$\frac{\mathbf{X}^T\mathbf{X}}{\mathrm{N}} = \begin{bmatrix} - & \mathbf{x}_1^T & - \\ - & \mathbf{x}_2^T & - \\ & \vdots & \\ - & \mathbf{x}_n^T & - \end{bmatrix} \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_p \\ | & | & & | \end{bmatrix} /N$$

$$= \begin{bmatrix} \frac{\mathbf{x}_1^T\mathbf{x}_1}{N} & \frac{\mathbf{x}_1^T\mathbf{x}_2}{N} & \cdots & \frac{\mathbf{x}_1^T\mathbf{x}_p}{N} \\ \frac{\mathbf{x}_2^T\mathbf{x}_1}{N} & \frac{\mathbf{x}_2^T\mathbf{x}_2}{N} & \cdots & \frac{\mathbf{x}_2^T\mathbf{x}_p}{N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\mathbf{x}_p^T\mathbf{x}_1}{N} & \frac{\mathbf{x}_p^T\mathbf{x}_1}{N} & \cdots & \frac{\mathbf{x}_p^T\mathbf{x}_p}{N} \end{bmatrix}$$

$$= \begin{bmatrix} \widehat{Cov}(X_1, X_1) & \widehat{Cov}(X_1, X_2) & \cdots & \widehat{Cov}(X_1, X_p) \\ \widehat{Cov}(X_2, X_1) & \widehat{Cov}(X_2, X_2) & \cdots & \widehat{Cov}(X_2, X_p) \\ \vdots & \vdots & \ddots & \vdots \\ \widehat{Cov}(X_p, X_1) & \widehat{Cov}(X_p, X_2) & \cdots & \widehat{Cov}(X_p, X_p) \end{bmatrix}$$

So that in the first line, if N is large and assuming $E(X) = 0$, then $\mathbf{X}^T\mathbf{X} \to NCov(X)$.

For the second line, $E_{x_0} x_0^T Cov(X)^{-1} x_0$ is scalar, thus we can use the linearity property where the expectation distributes over the trace sum as follows

$$E_{x_0}\text{trace}[x_0^T Cov(X)^{-1} x_0] = \text{trace}\left[E_{x_0} x_0^T Cov(X)^{-1} x_0\right]$$

and then use the cyclical property of the trace operator to rearrange terms to get

$$= \text{trace}[E_{x_0} x_0 x_0^T Cov(X)^{-1}]$$

and thus

$$\text{trace}[E_{x_0}[x_0 x_0^T] Cov(X)^{-1}] = \text{trace}[Cov(X) \cdot Cov(X)^{-1}]$$

$$= \text{trace}\left[\mathbf{I}_p\right]$$

$$= p$$

which gives us that $\text{trace}[Cov(X)^{-1} Cov(x_0)]\sigma^2/N = \sigma^2\left(\frac{p}{N}\right)$ and thus we have finished the problem. It is important to understand the motivation behind the problem – that is, for linear regression models, the expected squared prediction error grows by a factor of p so that it is relatively small when N is large.

Exercise 2.7. Suppose we have a sample of N pairs x_i, y_i drawn i.i.d. from the distribution characterized as follows:

$$x_i \sim h(x), the\ design\ density$$

$$y_i = f(x_i) + \epsilon_i, f\ is\ the\ regression\ function$$

$$\epsilon_i \sim (0, \sigma^2), (mean\ zero, variance\ \sigma^2)$$

We construct an estimator for f linear in the y_i,

$$\hat{f}(x_0) = \sum_{i=1}^{N} l_i(x_0; X) y_i,$$

where the weights $l_i(x_0; X)$ do not depend on the y_i but do depend on the entire training sequence of x_i denoted here by X.

(a). Show that linear regression and k-nearest-neighbor regression are members of this class of estimators. Describe explicitly the weights $l_i(x_0; X)$ in each of these cases.

For linear regression, the coefficient vector $\hat{\beta} = \langle \hat{\beta}_0, \dots, \hat{\beta}_p \rangle$ depends on the entire training set **X** through its derivation

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y$$

since each training example is a row in the design matrix

$$\mathbf{X} = \begin{bmatrix} - & \mathbf{x}_1^T & - \\ - & \mathbf{x}_2^T & - \\ & \vdots & \\ - & \mathbf{x}_n^T & - \end{bmatrix}.$$

Then, to make a prediction at a point x_0, we would need to solve the following

$$\hat{f}(x_0) = x_0^T\hat{\beta} = x_0^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y.$$

Define $[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T]_j^T$ to be the *j*th row of $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$. Then the weight vector is

$$l_i(x_0; X) = x_{01}[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T]_1^T + x_{02}[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T]_2^T + \cdots + x_{0p}[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T]_p^T$$

$$= \sum_{j=1}^{p} (x_0^T (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T)_j$$

so that we have shown that the weight vector depends on the entire training set but not on y_i.

For *k*-nearest-neighbor, the function is as follows

$$\hat{f}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i.$$

Then the weight is an indicator function $l_i(x_0; X) = 1[x_i \in N_k(x)]$ where recall we defined a set $D = \{d(x_i, x_0): x_i \in T\}$ where $d(x_0, x_i)$ is any metric. We also defined

$$N_k(x_0) = \{x_i: d(x_i, x_0) \leq d_k \text{ where } \mathrm{d_k} \text{is the } k\text{th smallest element of } D, x_i \in T\}.$$

Notice here that to construct the set D for any new point x_0, we have to search through the entire training set and thus the weight depends on the entire set *X*.

(b). Decompose the conditional mean-squared error

$$E_{y|x} \left(f(x_0) - \hat{f}(x_0) \right)^2$$

into a conditional squared bias and a conditional variance component. Let X, Y represent the entire training sequence of y_i.

<u>*Proof:*</u> For readability, assume the following: $f = f(x_0)$ and $\hat{f}(x_0) = \hat{f}$, and recall that only $\hat{f}$ depends on the training set. Also, refer back to page 11 of the thesis for clarity, since we solved the same problem there. Then

$$E_{y|x}(f - \hat{f})^2 = f^2 - 2f E_{y|x}(\hat{f}) + E_{y|x}(\hat{f}^2)$$

$$= E_{y|x}\left(f - E_{y|x}(\hat{f})\right)^2 + E_{y|x}(\hat{f}^2) - E_{y|x}(\hat{f})^2$$

$$= Bias^2_{y|x}(\hat{f}) + Var_{y|x}(\hat{f})$$

(c). Decompose the unconditional mean-squared error.

Proof: Using the same notation above where $f = f(x_0)$ and $\hat{f}(x_0) = \hat{f}$, then again, only $\hat{f}$ depends on the training data and we get

$$E_{y,x}(f - \hat{f})^2 = f^2 - 2fE_{y,x}(\hat{f}) + E_{y,x}(\hat{f}^2)$$

$$= E_{y,x}\left(f - E_{y,x}(\hat{f})\right)^2 + E_{y,x}(\hat{f}^2) - E_{y,x}(\hat{f})^2$$

$$= Bias^2_{y,x}(\hat{f}) + Var_{y,x}(\hat{f})$$

CHAPTER 3

LINEAR METHODS FOR REGRESSION

3.1 Introduction

This section goes over *linear regression models, subset selection, shrinkage methods, and methods using derived input directions*.

3.2 Linear Regression Models and Least Squares

On page 47, Equation (3.11) states that linear regression model

$$Y = X\beta + \epsilon$$

where the error $\epsilon \sim N(0, \sigma^2)$ implies that $\hat{\beta} \sim N(\beta, (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2)$. We will show that this statement holds. Consider the following estimate of $\hat{\beta}$

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y \tag{61}$$

where y is defined above, then this implies that

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(X\beta + \epsilon) \tag{62}$$

$$= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\beta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon \tag{63}$$

$$= \beta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon \tag{64}$$

Then we can use the property that a linear transformation of a multivariate normal random vector also has a multivariate normal distribution. We won't prove this fact here but it states that given $X \sim N(\mu, V)$ and suppose we have a linear transformation of X

$$y = A + \mathbf{B}X$$

then Y also has a multivariate normal distribution with mean

$$E[y] = A + \mathbf{B}\mu$$

and covariance matrix

$$Var[y] = \mathbf{B}\mathbf{V}\mathbf{B}^T.$$

Substituting this above, this implies that $\hat{\beta} \sim N(\beta, (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\sigma^2)$

$$E[\hat{\beta}] = \beta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T 0$$

$$= \beta \tag{65}$$

$$Var(\hat{\beta}) = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\sigma^2$$

$$= (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2 \tag{66}$$

$$\Rightarrow \hat{\beta} \sim N(\beta, (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2) \tag{67}$$

and so we have shown Equation 3.10.

3.3 Shrinkage Methods

On page 54, Algorithm 3.1 has the Gram-Schmidt procedure for multiple regression. We will use the Gram-Schmidt procedure to show how this leads to the QR decomposition Equation

(3.31). Then we will perform the QR decomposition on a small 3×3 matrix in R and show how it is beneficial in describing least squares.

The QR decomposition is

$$\mathbf{X} = \mathbf{QR}. \tag{68}$$

If $\mathbf{X} \in R^{n \times p+1}$ then $\mathbf{Q} \in R^{n \times p+1}$ and is an orthogonal matrix (i.e.$\mathbf{Q}^T\mathbf{Q} = I$) and $\mathbf{R} \in R^{p \times p}$ where **R** is an upper triangular matrix. It is important to remember that an orthogonal matrix is a matrix with orthogonal unit column vectors rather than just orthogonal columns as the name suggests. Often times orthogonal matrices are assumed to be square, however, it is not the case in the book.

Recall the algorithm is as follows:

Table 3.1. Algorithm for Gram-Schmidt Orthogonalization.

1. We initialize $z_0 = x_0 = 1, e_0 = \frac{z_0}{||z_0||}$
2. For $j = 1,2, \dots, p$
    a. Dot product x_j and $e_0, e_1, \dots, e_{j-1}$ to produce coefficients $\hat{\gamma}_{lj} = e_l x_j, \ell = 0, \dots, j-1$ and residual vector $z_j = x_j - \sum_{k=1}^{j-1} \hat{\gamma}_{kj} e_k$
3. Regress y on the residual z_p to give the estimate for $\hat{\beta}_p$.

Suppose we have $\mathbf{X} \in R^{n \times p+1}$ as follows

$$\mathbf{X} = \begin{bmatrix} | & | & \dots & | \\ x_0 & x_1 & \dots & x_p \\ | & | & \dots & | \end{bmatrix}$$

then

$$z_0 = x_0 = 1 \qquad e_0 = \frac{z_0}{||z_0||_2}$$

$$z_1 = x_1 - (x_1 \cdot e_0)e_0 \qquad e_1 = \frac{z_1}{||z_1||_2}$$

$$z_2 = x_2 - (x_2 \cdot e_0)e_0 - (x_2 \cdot e_1)e_1 \qquad e_2 = \frac{z_2}{||z_2||_2}$$

$$z_p = x_p - (x_p \cdot e_0)e_0 - (x_p \cdot e_1)e_1 - \cdots - (x_p \cdot e_{p-1})e_{p-1} \qquad e_p = \frac{z_p}{||z_p||_2}$$

and so that the resulting matrix is

$$\mathbf{X} = \begin{bmatrix} | & | & \dots & | \\ x_0 & x_1 & \dots & x_p \\ | & | & \dots & | \end{bmatrix} = \begin{bmatrix} | & | & \dots & | \\ e_0 & e_1 & \dots & e_p \\ | & | & \dots & | \end{bmatrix} \begin{bmatrix} x_0 \cdot e_0 & x_1 \cdot e_0 & \dots & x_p \cdot e_0 \\ 0 & x_1 \cdot e_1 & \dots & x_p \cdot e_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & x_p \cdot e_p \end{bmatrix} = \mathbf{QR}$$

For example, consider the matrix

$$X = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}, y = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

We will use this matrix to show that solving least squares by the QR decomposition is equal to the normal equation.

Table 3.2. R Code for Gram-Schmidt Orthogonalization.

```
# Create Matrix
x0=c(1,1,1)
x1=c(1,1,0)
x2=c(0,1,1)
y=c(1,2,3)

X=data.frame(x0,x1,x2,y)
```

```
# Create empty matrices Q and R
Q=R=matrix(0,nrow=3,ncol=3)


# Perform Operations
# First iteration
z0=x0
e0=z0/(sqrt(z0%*%z0))
Q[,1]=e0

# Second iteration
z1=x1-x1%*%e0*e0
e1=z1/(sqrt(z1%*%z1))
Q[,2]=e1

# Third iteration
z2=x2-x2%*%e0*e0-x2%*%e1*e1
e2=z2/(sqrt(z2%*%z2))
Q[,3]=e2

# Fill in R matrix
R[1,1]=x0%*%e0
R[1,2]=x1%*%e0
R[1,3]=x2%*%e0
R[2,2]=x1%*%e1
R[2,3]=x2%*%e1
R[3,3]=x2%*%e2

# Check the matrix
Q%*%R

# Beta Hat
solve(R)%*%t(Q)%*%y

# Check using linear regression
lm(y~x0+x1+x2-1,data=X)
```

Where the solution for $\hat{\beta}$ under the QR Decomposition is

$$\hat{\beta} = \mathbf{R}^{-1}\mathbf{Q}^{T}y \tag{69}$$

and gives the coefficient vector identical to the one fit by the R function lm()

$$\hat{\beta} = \begin{bmatrix} 2 \\ -1 \\ -1 \end{bmatrix}. \tag{70}$$

In order to solve from Equation (3.31) given as

$$\mathbf{X} = \mathbf{QR} \tag{71}$$

to equation (3.32)

$$\hat{\beta} = \mathbf{R}^{-1}\mathbf{Q}^T y \tag{72}$$

we can just use basic matrix operations. That is $\mathbf{X} = \mathbf{QR}$, and $\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y$

then,

$$\begin{aligned}\hat{\beta} &= (\mathbf{R}^T\mathbf{Q}^T\mathbf{QR})^{-1}\mathbf{R}^T\mathbf{Q}^T y \\ &= (\mathbf{R}^T\mathbf{R})^{-1}\mathbf{R}^T\mathbf{Q}^T y \\ &= \mathbf{R}^{-1}\mathbf{R}^{T^{-1}}\mathbf{R}^T\mathbf{Q}^T y \\ &= \mathbf{R}^{-1}\mathbf{Q}^T y \end{aligned} \tag{73}$$

and so we get Equation (3.32).

By decomposing **X** this way, we can save a lot on computation. Recall that the column space of the matrix **X** is the set of all possible linear combinations of its column vectors. More formally, suppose $\mathbf{X} \in R^{n\times p}$, then the column space of **X** is $C(\mathbf{X}) = \{v: \mathbf{X}a = v, \forall a \in R^p\}$. **Q** is in the column space of **X**, as seen by $\mathbf{X} = \mathbf{QR} \Rightarrow \mathbf{XR}^{-1} = \mathbf{Q}$, and it provides an orthonormal basis in the column space of **X**. Recall that the geometric interpretation of least squares fitted vector was to orthogonally project y into the column space of **X** by using the projection matrix of **X** (also known as hat matrix). Now with the QR decomposition, we can just project y onto **Q** to get the fitted vector $\hat{y}$ which turns out to be

$$\mathbf{Q}(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T y = \mathbf{QQ}^T y \tag{74}$$

and so we can save considerably on computation by not inverting the matrix $(\mathbf{X}^T\mathbf{X})^{-1}$. We can check this is true by using Equation (3.32)

$$\hat{\beta} = \mathbf{R}^T\mathbf{Q}^T y \tag{75}$$

$$\Rightarrow X\hat{\beta} = \mathbf{Q}\mathbf{R}\mathbf{R}^T\mathbf{Q}^T y \tag{76}$$

$$= \mathbf{Q}\mathbf{Q}^T y \tag{77}$$

which is the same as the projection. We will also show in Exercise 3.9 that we can use the QR decomposition to implement the forward selection algorithm efficiently.

On page 64, Equation (3.44) gives the solution to the ridge regression coefficient as

$$\hat{\beta}^{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda I)^{-1}\mathbf{X}^T y. \tag{78}$$

We will derive this here so that we can show other properties of ridge regression. Equation (3.43) shows that the loss function for ridge regression is as follows

$$RSS(\lambda) = (y - \mathbf{X}\beta)^T(y - \mathbf{X}\beta) + \lambda\beta^T\beta \tag{79}$$

$$= y^T y - 2\beta^T\mathbf{X}^T y + \beta^T\mathbf{X}^T\mathbf{X}\beta + \lambda\beta^T\beta \tag{80}$$

and then set the gradient to 0 and solve

$$\nabla_\beta RSS(\lambda) = -2\mathbf{X}^T y + 2\mathbf{X}^T\mathbf{X}\beta + 2\lambda\beta = 0 \tag{81}$$

$$\Rightarrow (\mathbf{X}^T\mathbf{X} + \lambda I)\beta = \mathbf{X}^T y \tag{82}$$

$$\Rightarrow \hat{\beta}^{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda I)^{-1}\mathbf{X}^T y \tag{83}$$

and so we have shown Equation (3.44).

Now, we will introduce the singular value decomposition of the centered matrix **X**. The author does not define centered so we will make explicit here what he means. Centering a matrix means that we normalize the columns to have mean 0 and variance 1. That is we use the following algorithm,

Table 3.2. Algorithm for Centering.

1. Let $\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$
2. Replace each $x_j \in \mathbf{X}$ with $x_j - \mu_j$
3. Let $\sigma_j^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(x_{ij}\right)^2$
4. Replace each x_j with x_j/σ_j.

Once we do that, we can decompose matrix **X** to its singular value decomposition

$$\mathbf{X} = \mathbf{UDV^T}. \tag{84}$$

The actual decomposition is quite complicated so we will not go over derive it here. Here, **U** and **V** are $n \times p$ and $p \times p$ orthogonal matrices with the columns of **U** spanning the column space of **X** and the columns of **V** spanning the row space. **D** is a $p \times p$ diagonal matrix, with diagonal entries $d_1 \geq d_2 \geq \cdots \geq d_p \geq 0$ and if one or more $d_j = 0$, then **X** is singular.

We can use the singular value decomposition to represent the matrix $\mathbf{X}^T\mathbf{X}$.

$$\begin{aligned}\mathbf{X^TX} &= (\mathbf{UDV^T})^\mathbf{T}\mathbf{UDV} \\ &= \mathbf{VDU^TUDV^T} \\ &= \mathbf{VD^2V^T}\end{aligned} \tag{85}$$

and use this as in page 66, Equation (3.48) which states that the eigen decomposition of $\mathbf{X}^T\mathbf{X}$ is

$$\mathbf{X}^T\mathbf{X} = \mathbf{V}\mathbf{D}^2\mathbf{V}^T. \tag{86}$$

To show that this is the eigen decomposition, recall that the sample covariance matrix $\widehat{Cov}(X) = \frac{\mathbf{X}^T\mathbf{X}}{N}$ since the matrix has been pre-centered. Then by the definition of covariance matrices, $\frac{\mathbf{X}^T\mathbf{X}}{N}$ is symmetric which implies that $\mathbf{X}^T\mathbf{X}$ is symmetric. One property of symmetric matrices we will use is that the eigen vectors of symmetric matrices are orthonormal. We will not go over the proof but it can be found in a linear algebra textbook. Since the inverse of an orthogonal matrix is its transpose (for an orthogonal matrix $\mathbf{A}$, by definition $\mathbf{A}^T\mathbf{A} = \mathbf{I}$ then taking the inverse, $\mathbf{A}^T\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}$ implies that $\mathbf{A}^T = \mathbf{A}^{-1}$), we can write it in eigen form as

$$\mathbf{X}^T\mathbf{X}\mathbf{V} = \mathbf{V}\mathbf{D}^2 \tag{87}$$

$$\Rightarrow \mathbf{X}^T\mathbf{X}\mathbf{V}\mathbf{V}^{-1} = \mathbf{V}\mathbf{D}^2\mathbf{V}^{-1} \tag{88}$$

$$= \mathbf{X}^T\mathbf{X} = \mathbf{V}\mathbf{D}^2\mathbf{V}^T \tag{89}$$

so we get Equation 3.48. Here, the matrix $\mathbf{V}$ holds the eigenvectors of $\mathbf{X}^T\mathbf{X}$ and the matrix $\mathbf{D}^2$ are the eigenvalues.

What happens if we multiply $\mathbf{X}$ and $\mathbf{V}$? Recall that the d_i's in $\mathbf{D}$ are ordered $d_1 \geq d_2 \geq \cdots \geq d_p \geq 0$ and we multiply $\mathbf{X}v_i$ as in Equation (3.49). Then we can look at its variance as follows

$$Var(\mathbf{X}v_i) = v_i^T Var(\mathbf{X}) v_i \tag{90}$$

$$= v_i^T \left(\frac{\mathbf{X}^T\mathbf{X}}{N}\right) v_i \tag{91}$$

$$= v_i^T \mathbf{V}\mathbf{D}^2\mathbf{V}^T v_i / N \tag{92}$$

recall that **V** is an orthogonal matrix which implies that $(v_i \cdot v_j) = 0, \forall i \neq j$, we get

$$= v_i^T v_i d_i^2 v_i^t v_i / N \tag{93}$$

and since V has orthonormal columns, $v_i^T v_i = 1$ and the above implies

$$Var(\mathbf{X}v_i) = \frac{d_i^2}{N} \tag{94}$$

which completes Equation (3.49). This equation is stating that by taking the eigenvectors of matrix **V** and multiplying it by the design matrix **X**, the variance is proportional to the eigenvalues. This variance is sorted since the d_i's of **D** are sorted. In fact, let $z_i = \mathbf{X}v_i$, then z_i has the ith largest variance among all normalized linear combinations of the columns of **X** subject to being orthogonal to the earlier ones. We will prove this in the next section. Since we have this nice property, the matrix **V** gets a special name and is called the *principal components* of the variables in **X**.

To tie this together, we will derive Equation (3.47) of ridge regression using the singular value decomposition. Showing this requires a lot of matrix operations and can be seen as follows

$$\mathbf{X}\hat{\beta}^{ridge} = \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T y \tag{95}$$

$$= \mathbf{U}\mathbf{D}\mathbf{V}^T((\mathbf{U}\mathbf{D}\mathbf{V}^T)^{-1}\mathbf{U}\mathbf{D}\mathbf{V}^T + \lambda\mathbf{I})^{-1}(\mathbf{U}\mathbf{D}\mathbf{V}^T)^T y$$

$$= \mathbf{U}\mathbf{D}\mathbf{V}^T(\mathbf{V}\mathbf{D}^2\mathbf{V}^T + \lambda\mathbf{I})^{-1}\mathbf{V}\mathbf{D}\mathbf{U}^T y$$

$$= \mathbf{U}\mathbf{D}\mathbf{V}^T V^{T^{-1}}(\mathbf{D}^{-2})\mathbf{V}^{-1}\mathbf{V}\mathbf{D}\mathbf{U}^T y + \mathbf{U}\mathbf{D}\mathbf{V}^T(\lambda^{-1}I)\mathbf{V}\mathbf{D}\mathbf{U}^T y$$

$$= \mathbf{UD}(\mathbf{D}^{-2})\mathbf{DU}^T y + \mathbf{UD}(\lambda^{-1}\mathbf{I})\mathbf{DU}^T y$$

$$= \mathbf{UD}(\mathbf{D}^2 + \lambda\mathbf{I})^{-1}\mathbf{DU}^T y$$

$$= \sum_{j=1}^{p} u_j \frac{d_j^2}{d_j^2 + \lambda} u_j^T y \tag{96}$$

so that we have shown equation (3.47). By looking at the middle term $\frac{d_j^2}{d_j^2+\lambda}$ we can see what ridge regression does. When $\lambda = 0$, we get the least squares estimates $\frac{d_j^2}{d_j^2} = 1$. Ridge regression gives a value $\frac{d_j^2}{d_j^2+\lambda} < 1$. Recall what the d_j's mean here; they are the eigenvalues of $\mathbf{X}^T\mathbf{X}$ and we they are the variance of $\mathbf{X}$ in the direction of its principal components $z_i = \mathbf{X}v_i$. Thus for ridge regression, smaller values of d_j and thus the smaller variances in the direction of its principal components are shrunken most. To prove this, we have the original equation

$$\frac{d_j^2}{d_j^2 + \lambda}. \tag{97}$$

Since $d_1 > d_2$ implies that $d_1^2 > d_2^2$ since the square function is monotonic for non-negative values, then we need to show that

$$\frac{d_2^2}{d_2^2 + \lambda} = \frac{d_1^2 - c}{d_1^2 - c + \lambda} < \frac{d_1^2}{d_1^2 + \lambda}. \tag{98}$$

Where c is some small constant $0 < c \leq d_1^2$ and λ is the parameter in ridge regression $\lambda > 0$. By cross multiplying the function above, we get the following

$$(d_1^2 - c)(d_1^2 + \lambda) < d_1^2(d_1^2 - c + \lambda)$$

$$d_1^4 + \lambda d_1^2 - c d_1^2 - c\lambda < d_1^4 - c d_1^2 + \lambda d_1^2$$

$$\Rightarrow -c\lambda < 0 \tag{99}$$

and since $\lambda, c > 0$ this is last inequality is true and so the statement that smaller directions of the principal components are shrunken the most holds.

3.4 Methods Using Derived Input Directions

On page 81, Equation (3.63) we see that the *m*th principal component direction v_m solves

$$\max_{\alpha} Var(\mathbf{X}\alpha)$$

$$\text{subject to } ||\alpha|| = 1, \alpha^T \mathbf{S} v_\ell = 0, \ell = 1, \ldots, m-1. \tag{100}$$

We will show that α here are the eigenvalues; that is, we will show $a = v_m$. Recall from the text that our design matrix $\mathbf{X}$ is standardized so that that we can rewrite the function $Var(\mathbf{X}\alpha^T) = a^T Var(\mathbf{X}) a = a^T \mathbf{X}^T \mathbf{X} a$. A standard way of solving optimization problems with equality constraints is by including the constraints in the objective function; this is known as the Lagrangian and it can often be found in a calculus book. So rewriting the above quantity in Lagrangian form, we get the following

$$L(X, \lambda) = a^T \mathbf{X}^T \mathbf{X} a - \lambda a^T a \tag{101}$$

where λ is called the Lagrange multiplier. Then, to find the optimal point, we have to set the gradient of the Lagrangian to zero and solve for x.

$$\nabla_a L(X, \lambda) = 2\mathbf{X}^T \mathbf{X} a - 2\lambda a = 0. \tag{102}$$

Notice here that we get the standard eigen form $\mathbf{X}^T\mathbf{X}a = \lambda a$. This shows that the only points which can maximize the functions are the eigenvectors of $\mathbf{X}^T\mathbf{X}$ and we showed in the previous section that this is the matrix $\mathbf{V}$ under the SVD decomposition. Since $\mathbf{V}$ is an orthogonal matrix, we know that the second constraint ($\alpha^T\mathbf{S}v_\ell = 0$) holds so we have solved Equation (3.63).

3.5 Problems and Solutions

Exercise 3.2. Given data on two variables X and Y, consider fitting a cubic polynomial regression model $f(X) = \sum_{j=0}^{3}\beta_j X^j$. In addition to plotting the fitted curve, you would like a 95% *confidence band about the curve. Consider the following two approaches:*

(1). At each point x_0, form a 95% *confidence interval for the linear function at $a^T\beta = \sum_{j=0}^{3}\beta_j x_0^j$.*

Proof: For this problem, the data matrix is $n \times 4$ matrix as follows

$$\mathbf{X} = \begin{bmatrix} 1 & x_{i1} & x_{i1}^2 & x_{i1}^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n1}^2 & x_{n1}^3 \end{bmatrix}$$

and $\hat{\beta}$ is estimated in the usual least-squares way by setting up the normal equations. Then, to generate confidence intervals around an estimated mean $\hat{f}(x_0) = E[\widehat{y_0|x_0}]$, use the Wald confidence interval as follows

$$95\%\ \text{Confidence Interval}(x_0) = x_0^T\hat{\beta} \pm 1.96 \cdot \sqrt{Var\left(x_0^T\hat{\beta}\right)}$$

where the variance from Exercise 2.5 is as follows

$$Var\left(x_0^T\hat{\beta}\right) = x_0^T Var\left(\hat{\beta}\right)x_0 = \sigma^2 x_0^T(\mathbf{X}^T\mathbf{X})^{-1}x_0$$

then the interval is

$$95\%\ \text{Confidence Interval}(x_0) = x_0^T\hat{\beta} \pm 1.96 \cdot \sigma\sqrt{x_0^T(\mathbf{X}^T\mathbf{X})^{-1}x_0}\blacksquare$$

(2). Form a 95% *confidence set for* β *as in (3.15), which in turn generates confidence intervals for* $f(x_0)$.

Proof: Equation (3.15) is as follows

$$\mathrm{C}_\beta = \left\{\beta|\left(\hat{\beta}-\beta\right)^T\mathbf{X}^T\mathbf{X}\left(\hat{\beta}-\beta\right) \le \hat{\sigma}^2{\chi_{p+1}^2}^{(1-a)}\right\}$$

where C_β is the confidence set and ${\chi_{p+1}^2}^{(1-a)}$ is the $(1-a)$-percentile of the chi-squared distribution on $(\mathrm{p}+1)$-degrees of freedom. Then, at $\alpha = 0.05, \chi^2 = 3.84$ so that the interval for this particular set β is

$$\mathrm{C}_\beta = \left\{\beta|(\hat{\beta}^T-\beta)(\mathbf{X}^T\mathbf{X})(\hat{\beta}-\beta) \le \hat{\sigma}^2 \cdot 3.84\right\}$$

which implies that the interval for the conditional mean is

$$\Rightarrow C_{x_0^T\widehat{\beta}} = \left\{x_0^T\hat{\beta}|\beta \in \mathrm{C}_\beta\right\}.$$

Exercise 3.3. (1). Prove the Gauss–Markov theorem: the least squares estimate of a parameter $a^T\beta$ *has variance no bigger than that of any other linear unbiased estimate of* $a^T\beta$ *(Section 3.2.2).*

Proof: Recall that the least squares estimator for $\hat{\beta}$ is

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y.$$

Suppose we have another unbiased linear estimator $\tilde{\beta}$ so that $\tilde{\beta} = \mathbf{A}y$ and let $\mathbf{A} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T + \mathbf{C}$. Then we get the following

$$\tilde{\beta} = ((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T + \mathbf{C})y$$

$$= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y + \mathbf{C}y.$$

Since least squares is unbiased and we assume the model $Y = X^T\beta$, the following holds $E[y] = X^T\beta$. Then, use this below to find the expectation on $\tilde{\beta}$ so that

$$E[\tilde{\beta}] = E[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y] + E[\mathbf{C}y]$$

$$= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\beta + \mathbf{C}\mathbf{X}\beta$$

$$= (\mathbf{I}_p + \mathbf{C}\mathbf{X})\beta$$

then $\tilde{\beta}$ is unbiased if and only if $\mathbf{C}\mathbf{X} = 0$.

Then the variance of $\tilde{\beta}$ is as follows

$$Var(\tilde{\beta}) = Var(((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T + \mathbf{C})y)$$

Let $\mathbf{D} = ((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T + \mathbf{C})$, then the above is equal to

$$= \mathbf{D}Var(y)\mathbf{D}^T$$

$$= \sigma^2 \cdot \mathbf{D}\mathbf{D}^T$$

$$= \sigma^2 \cdot ((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T + \mathbf{C})(\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1} + \mathbf{C}^T)$$

$$= \sigma^2 \cdot [(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1} + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{C}^T + \mathbf{C}\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1} + \mathbf{C}\mathbf{C}^T]$$

And above we stated that $\mathbf{C}\mathbf{X} = 0$ so we remove the 2nd and 3rd terms to reduce it to the following

$$= \sigma^2 \cdot [(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1} + \mathbf{C}\mathbf{C}^T]$$

$$= Var(\hat{\beta}) + \sigma^2[\mathbf{C}\mathbf{C}^T]$$

and $\mathbf{C}\mathbf{C}^T$ is a positive semidefinite matrix since $a^T C C^T a = (a^T C \cdot C^T a) \geq 0$, so that

$Var(\tilde{\beta}) > Var(\beta)$ holds and we complete the proof.

Exercise 3.4. (1). Show how the vector of least squares coefficients can be obtained from a single pass of the Gram–Schmidt procedure (Algorithm 3.1). Represent your solution in terms of the QR decomposition of ***X****.*

Proof: After a single pass of the Gram-Schmidt process, we get the following decomposition

$$\mathbf{X} = \mathbf{Q}\mathbf{R}$$

where $\mathbf{Q}$ is an $N \times (p+1)$ orthogonal matrix and R is an $(p+1) \times (p+1)$ upper triangular matrix equal. Then by least squares, we get the following equation

$$\mathbf{X}^T\mathbf{X}\beta = \mathbf{X}^T y$$

$$\Rightarrow (\mathbf{Q}\mathbf{R})^T\mathbf{Q}\mathbf{R}\beta = (\mathbf{Q}\mathbf{R})^T y$$

$$\Rightarrow \quad \mathbf{R}^T\mathbf{R}\beta = \mathbf{R}^T\mathbf{Q}^T y$$

$$\Rightarrow \quad \mathbf{R}\beta = \mathbf{Q}^T y$$

and we can solve the last equation by back-substitution since $\mathbf{R}$ is an upper-triangular matrix.

For example,

$$\begin{bmatrix} r_{00} & r_{01} & \cdots & r_{0p} \\ 0 & r_{11} & \cdots & r_{1p} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{pp} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} = \begin{bmatrix} q_0^T y \\ q_1^T y \\ \vdots \\ q_p^T y \end{bmatrix}$$

the last element is $\hat{\beta}_p = \frac{q_p^T y}{r_{pp}}$, and solving the next elements in order by back-substitution $\hat{\beta}_p, \hat{\beta}_{p-1}, \ldots \hat{\beta}_0$.

Exercise 3.5. (1). Consider the ridge regression problem (3.41). Show that this problem is equivalent to the problem

$$\hat{\beta}^c = \arg\min_{\beta^c} \left\{ \sum_{i=1}^{N} [y_i - \beta_0^c - \sum_{j=1}^{p} (x_{ij} - \bar{x}_j)\beta_j^c]^2 + \lambda \sum_{j=1}^{p} {\beta_j^c}^2 \right\}$$

Give the correspondence between β^c and the original β in (3.41).

$$\hat{\beta}^{ridge} = arg\, min_{\beta} \left\{ \sum_{i=1}^{N} [y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j]^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \quad (3.41)$$

Proof: For the first equation above, we can expand out the middle summation so that we get the following

$$\sum_{j=1}^{p} (x_{ij} - \bar{x}_j)\beta_j^c = \sum_{j=1}^{p} x_{ij}\beta_j^c - \sum_{j=1}^{p} \bar{x}_j\beta_j^c$$

and we get the correspondence between β_0^c and β_0

$$\beta_0 = \beta_0^c + \sum_{j=1}^{p} \bar{x}_j\beta_j^c.$$

That is, β_0 is equivalent to β_0^c except it is shifted. At this point, the two functions are equal and thus all other coefficients will stay the same

$$\beta_1^c = \beta_1, \beta_2^c = \beta_2 \ldots, \beta_p^c = \beta_p$$

so that β^c has the same slopes as the β's so we have shown the correspondence.

Exercise 3.6. Show that the ridge regression estimate is the mean (and mode) of the posterior distribution, under a Gaussian prior $\beta \sim N(0, \tau \boldsymbol{I})$, and Gaussian sampling model $y \sim N(\boldsymbol{X}\beta, \sigma^2 \boldsymbol{I})$. Find the relationship between the regularization parameter λ in the ridge formula, and the variances τ and σ^2.

Proof: From Bayes rule, we can write the posterior distribution of β as follows

$$\Pr(\beta|D) = \frac{\Pr(D|\beta) \cdot \Pr(\beta)}{\Pr(D)} \propto \Pr(D|\beta) \cdot \Pr(\beta)$$

where $P(D|\beta)$ is the likelihood; that is, the probability of the data given β. The last term states that the posterior distribution is proportional to $\Pr(D|\beta) \cdot \Pr(\beta)$. Since the denominator is the marginal density of D, this will be constant for all β, we can leave it out when solving for β. Next, notice that each density of the last two terms are

$$\Pr(D|\beta) \sim N(X^T\beta, \sigma^2 I)$$

$$\Pr(\beta) \sim N(0, \tau \mathbf{I})$$

The first comes from the likelihood of the data under regression assumptions and the second is stated in the problem. The maximum likelihood with respect to β under the log transformation, we get the following

$$\log(Pr(D|\beta) \cdot Pr(\beta)) = -\frac{(y - \mathbf{X}\beta)^T(y - \mathbf{X}\beta)}{2\sigma^2} - \frac{\beta^T\beta}{2\tau}$$

$$= \frac{1}{2\sigma^2}\sum_{i=1}^{N}[y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j]^2 + \frac{1}{2\tau}\sum_{j=1}^{p}\beta_j^2$$

and if we multiply the equation by $2\sigma^2$, then we get the ridge equation (3.41) with $\lambda = \frac{\sigma^2}{\tau}$ thus we have showed the relationships between λ and the two variance terms. Finally, by plugging λ into the above equation, we get the following,

$$\hat{\beta}^{ridge} = \arg\max_{\beta} \sum_{i=1}^{N}[y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j]^2 + \lambda\sum_{j=1}^{p}\beta_j^2.$$

which is the ridge regression form. Since we assumed our data were generated from a normal distribution and we used a normal prior (conjugate prior), from statistics, we know that the posterior will be normal. This implies that the maximum β under the posterior distribution gives the mean. Since the mean and mode are equal for normal distributions (due to symmetry) we have shown the relationship between them and answered the problem.

Exercise 3.9. Forward stepwise regression. Suppose we have the QR decomposition for the $N \times q$ matrix $\boldsymbol{X}_1$ in a multiple regression problem with response y, and we have an additional $p - q$ predictors in the matrix $\boldsymbol{X}_2$. Denote the current residual by $\boldsymbol{r}$. We wish to establish which one of these additional variables will reduce the residual-sum-of squares the most when included with those in $\boldsymbol{X}_1$. Describe an efficient procedure for doing this.

Proof: Recall from linear algebra that the nullspace is $N(X) = \{v: Xv = 0\}$. This implies that the residual vector **r** lives in the nullspace of $\mathbf{X}^T$ since

$$\mathbf{X}^T r = \mathbf{X}^T(y - \hat{y})$$

$$= \mathbf{X}^T(y - \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y)$$

$$= \mathbf{X}^T y - \mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y$$

$$= \mathbf{X}^T y - \mathbf{X}^T y = 0.$$

Since the QR decomposition uses the Gram-Schmidt procedure, we have seen that this means that this forms an orthonormal basis for **X**. This implies that the current fitted values have been projected onto the subspace spanned by $\mathbf{X}_1$ and so that the remaining $p - q$ variables in $\mathbf{X}_2$ (assuming linear independence) live in the nullspace of $\mathbf{X}_1$. Since the residual sum-of-squares is defined as $||y - \hat{y}||_2^2$ which is the squared distance between y and its projected vector, we would like to find the vector $v \in R^{p-q}$ where such that we decrease the length of the residual vector. This means that reducing the residual sum-of-square error the most amounts to finding the largest projection of *r* onto the nullspace. That is, we need to find the following

$$X_j = \arg\max_{X_j \in \mathbf{X}_2} \frac{\langle X_j, r \rangle}{\langle X_j, X_j \rangle}$$

So that completes the first part of the problem.

For the second part, define a new matrix $\mathbf{X}^*$ that is $\mathbf{X}_1$ with an appended vector X_j as follows

$$\mathbf{X}^* = \begin{bmatrix} & & | \\ & \mathbf{X}_1 & X_j \\ & & | \end{bmatrix}$$

we can use the QR decomposition for a more efficient fit. Recall that the QR decomposition fits each row iteratively and from page 35 of this thesis, we have the last row

$$z_j = x_j - (x_j \cdot e_0)e_0 - (x_j \cdot e_1)e_1 - \cdots - (x_j \cdot e_p)e_p \qquad e_j = \frac{z_j}{||z_j||_2}$$

and then we get the QR matrix by

$$\mathbf{X} = \begin{bmatrix} | & | & \dots & | \\ x_0 & x_1 & \dots & x_p \\ | & | & \dots & | \end{bmatrix} = \begin{bmatrix} | & | & \dots & | \\ e_0 & e_1 & \dots & e_j \\ | & | & \dots & | \end{bmatrix} \begin{bmatrix} x_0 \cdot e_0 & x_1 \cdot e_0 & \dots & x_j \cdot e_0 \\ 0 & x_1 \cdot e_1 & \dots & x_j \cdot e_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & x_j \cdot e_j \end{bmatrix}$$

And then we can solve for the coefficients as in Exercise 3.4 by back substitution

$$\mathbf{R}\beta = \mathbf{Q}^T y$$

so that we have completed the problem. Solving this problem using the QR decomposition is much faster solving for $\hat{\beta}$ using the normal equations and X^*.

Exercise 3.10. Backward stepwise regression. Suppose we have the multiple regression fit of y on ***X****, along with the standard errors and Z-scores as in Table 3.2. We wish to establish which variable, when dropped, will increase the residual sum-of-squares the least. How would you do this?*

Exercise 3.1 established that the F statistic for dropping a single coefficient is equal to the square of the corresponding z-score so that we get the following relationship

$$F = \frac{RSS_0 - RSS_1}{\frac{RSS_1}{N-p-1}} = \frac{RSS_0 - RSS_1}{\hat{\sigma}^2} = z_j^2$$

and then shuffling terms around, we get

$$\Rightarrow RSS_0 - RSS_1 = \hat{\sigma}^2 z_j^2$$

$$\Rightarrow RSS_0 = \hat{\sigma}^2 z_j^2 + RSS_1$$

which states that the smallest increase in the residual sum-of-squares corresponds to dropping the smallest z-score (i.e. smallest z_j^2 value) and we have concluded the problem.

Exercise 3.13. Derive the expression (3.62), and show that $\hat{\beta}^{pcr}(p) = \hat{\beta}^{ls}$.

Proof: Using the singular value decomposition of $\mathbf{X} = \mathbf{UDV}^T$ we can represent $\hat{\beta}^{ls}$ as follows

$$\mathbf{X}^T\mathbf{X}\beta = \mathbf{X}^T y$$

$$\mathbf{VD}^T\mathbf{U}^T\mathbf{UDV}^T\beta = \mathbf{VD}^T\mathbf{U}^T y$$

$$\mathbf{VD}^2\mathbf{V}^T\beta = \mathbf{VDU}^T y$$

where $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ since $\mathbf{U}$ is an orthogonal matrix and $\mathbf{D}^T = \mathbf{D}$ because it is a diagonal matrix. Then we can solve for $\hat{\beta}$ and get the following result

$$\hat{\beta}^{ls} = \mathbf{VD}^{-1}\mathbf{U}^T y$$

Equation (3.62) on principal component regression has the form

$$\hat{\beta}^{pcr}(M) = \sum_{m=1}^{M} \hat{\theta}_m v_m$$

where $\hat{\theta}_m = \langle z_m, y\rangle / \langle z_m, z_m\rangle$ and where z_m are the principal components and is the mth column in the matrix $\mathbf{Z} = \mathbf{XV} = \mathbf{UD}$. v_m is the column in the singular value decomposition matrix $\mathbf{V}$.

When $M = p$, the vector $\hat{\theta}$ is the coefficient vector when regressing y onto the principal components and is solved as follows

$$\hat{\theta} = (\mathbf{Z}^T\mathbf{Z})^{-1}\mathbf{Z}^T y = (\mathbf{D}^T\mathbf{U}^T\mathbf{U}\mathbf{D})^{-1}\mathbf{D}^T\mathbf{U}^T y = \mathbf{D}^{-1}\mathbf{U}^T y$$

and to get the vector $\hat{\beta}^{pcr}$ we do the following

$$\hat{\beta}^{pcr} = \mathbf{V}\hat{\theta} = \mathbf{V}\mathbf{D}^{-1}\mathbf{U}^T y$$

which is equivalent to that of least squares when $M = p$■

Exercise 3.16. Derive the entries in Table 3.4, the explicit forms for estimators in the orthogonal case.

Estimator	Formula
Best subset (size M)	$\hat{\beta}_j \cdot I(\lvert\hat{\beta}_j\rvert \geq \lvert\hat{\beta}_M\rvert)$
Ridge	$\hat{\beta}_j/(1+\lambda)$
Lasso	$\text{sign}(\hat{\beta}_j)(\lvert\hat{\beta}_j\rvert - \lambda)_+$

Proof: Table 3.4 Estimators of β_j in the case of orthonormal columns of $\mathbf{X}$. M and λ are constants chosen by the corresponding techniques; sign denotes the sign of its argument (± 1).

For *least-squares*

$$\hat{\beta}^{ls} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y = \mathbf{X}^T y$$

For *best-subset selection* we get the following

$$\hat{\beta}^{ls} = \hat{\beta}^{bs}(M = p) = \mathbf{X}^T y$$

where the coefficients are identical even if we take $M \leq p$ since the design matrix is orthogonal.

For *ridge regression* we get the following estimates

$$\hat{\beta}^{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T y = (\mathbf{I} + \lambda\mathbf{I})^{-1}\mathbf{X}^T y = \hat{\beta}^{ls}/(1+\lambda)$$

For *lasso regression*, we get the following

$$L(\beta) = (y - \mathbf{X}\beta)^T(y - \mathbf{X}\beta) + \lambda|\beta|$$

$$\Rightarrow \frac{\partial L(\beta)}{\partial \beta} = -\mathbf{X}^T y + \mathbf{X}^T\mathbf{X}\beta + \lambda \cdot \text{sign}(\beta)$$

and by setting the gradient to zero and solving with respect to β we get

$$\hat{\beta}^{lasso} = \mathbf{I}(\mathbf{X}^T y - \lambda \cdot \text{sign}(\beta))$$

$$= \text{sign}(\beta)(|\mathbf{X}^T y| - \lambda) \blacksquare$$

Exercise 3.17. Repeat the analysis of Table 3.3 on the spam data discussed in Chapter 1 where Table 3.3 compares the coefficient estimates of least-squares, best subset, ridge, lasso, principal component regression, and partial least-squares.

This dataset has 57 variables with a binary response indicating spam or not spam. Fitting each model and using cross validation to generate the mean squared errors, we get the following table

	LS	Best Subset	Ridge	Lasso	PCR	PLS
Parameters			$\lambda = .08$	$\lambda = 0.0017$	ncomp = 56	ncomp = 12
Test Error	0.334		0.334	0.334	0.335	0.333
Std Error	0.0169		0.0115	0.0182	0.0177	0.0158

Best subset regression did not complete as it took too long. The author mentions that it can work when the number of variables p up to 30 or 40 but is not feasible since it is an exponential time search time.

Exercise 3.19. Show that $||\hat{\beta}^{ridge}||$ increases as its tuning parameter $\lambda \to 0$. Does the same property hold for the lasso and partial least squares estimates? For the latter, consider the "tuning parameter" to be the successive steps in the algorithm.

Proof: The solution to $\hat{\beta}^{ridge}$ can be found using equation (3.47)

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

and using the singular value decomposition on the centered matrix $\mathbf{X}$

$$= (\mathbf{D}^2 + \lambda\mathbf{I})^{-1}\mathbf{V}\mathbf{D}\mathbf{U}^T y \tag{3.47}$$

so that

$$||\hat{\beta}^{ridge}||^2 = y^T\mathbf{U}\mathbf{D}\mathbf{V}^T(\mathbf{D}^2 + \lambda\mathbf{I})^{-2}\mathbf{V}\mathbf{D}\mathbf{U}^T y$$

$$= y^T\mathbf{U}\mathbf{D}(\mathbf{D}^2 + \lambda\mathbf{I})^{-2}\mathbf{D}\mathbf{U}^T y$$

$$= \sum_{j=1}^{p} y u_j \frac{d_j^2}{\left(d_j^2 + \lambda\right)^2} u_j^T y$$

and as $\lambda \to 0$, the quantity $\frac{d_j^2}{\left(d_j^2+\lambda\right)^2}$ is increasing and thus the vector $||\hat{\beta}^{ridge}||^2$ increases and so

we have concluded the exercise.

CHAPTER 4

LINEAR METHODS FOR CLASSIFICATION

4.1 Introduction

This section goes over topics including *the classification task and linear discriminant analysis*.

4.2 Linear Discriminant Analysis

This chapter is an extension of Chapter 3 from a regression point a view to a classification one. On page 113, we will derive the equations listed in the center of the page. Notice that the second equation is

$$\log|\widehat{Cov}_k| = \sum_{l=1}^{K} \log d_{kl}. \tag{103}$$

This is the determinant of the covariance matrix and uses the property of determinants where $\det\left(\widehat{Cov}_k\right) = \prod_{l=1}^{K} d_{kl}$. That is, the determinant is equal to the product of the eigenvalues which can be obtained from matrix **D** of the eigen decomposition of the covariance matrix.

The next bullet point on the same page states that we can sphere the data with respect to the common covariance matrix by applying

$$X^* \leftarrow \mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T X. \tag{104}$$

This occurs by first decomposing the covariance of X by its eigen decomposition; that is, $Cov(X) = \mathbf{UDU}^T$. We can take the negative square root of a matrix by performing each operation 1-by-1. That is, we take the square root and then invert it as follows

$$(\mathbf{UDU}^T)^{-\frac{1}{2}} = \left(\mathbf{UD}^{\frac{1}{2}}\mathbf{D}^{\frac{1}{2}}\mathbf{U}^T\right)^{-\frac{1}{2}}$$

$$= (\mathbf{UD}(\mathbf{UD})^T)^{-\frac{1}{2}}$$

$$= (\mathbf{UD})^{-1}$$

$$= \mathbf{D}^{-\frac{1}{2}}\mathbf{U}^{-1} = \mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T \tag{105}$$

and then by sphering the data, we change the covariance of X^* to be the identity matrix. This can be shown as follows

$$Cov(X^*) = Cov\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T\mathbf{X}\right) \tag{106}$$

$$= \mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T Cov(X)\mathbf{U}\mathbf{D}^{-\frac{1}{2}}$$

$$= \mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{U}\mathbf{D}^{-\frac{1}{2}}$$

$$= \mathbf{I} \tag{107}$$

so that we get the identity matrix as the covariance of X^* and complete the proof.

4.3 Problems and Solutions

Exercise 4.1. Show how to solve the generalized eigenvalue problem max $\alpha^T \boldsymbol{B}\alpha$ subject to $\alpha^T \boldsymbol{W}\alpha = 1$ by transforming to a standard eigenvalue problem where $\boldsymbol{B}$ is the between-class covariance matrix and $\boldsymbol{W}$ is the within-class covariance matrix.

Proof: Since this is an equality constraint, we can set it up in Lagrangian form and solve using lagrangian multipliers. The problem is of the form

$$\max_{\alpha} \alpha^T \mathbf{B}\alpha$$

$$\text{subject to } \alpha^T \mathbf{W}\alpha = 1$$

Then, in Lagrangian form, this is

$$\mathrm{L}(a, \lambda) = a^T \mathbf{B}a + \lambda(a^T \mathbf{W}a - 1)$$

We can take partials with respect to a and λ so that

$$\frac{\partial \mathrm{L}(a, \lambda)}{\partial a} = 2\mathbf{B}a + 2\lambda \mathbf{W}a = 0 \qquad (1)$$

$$\frac{\partial \mathrm{L}(a, \lambda)}{\partial \lambda} = a^T \mathbf{W}a - 1 = 0 \qquad (2)$$

And so for the first equation,

$$\Rightarrow -\mathbf{B}a = \lambda \mathbf{W}a$$

$$\Rightarrow -\mathbf{W}^{-1}\mathbf{B}a = \lambda a$$

Notice that this is in eigen decomposition form and since we want to maximize the original quantity, we know that a must be the first eigenvector and λ the corresponding eigenvalue to the matrix $-\mathbf{W}^{-1}\mathbf{B}$.

REFERENCES

[1] Agresti, Alan. (2012). Categorical Data Analysis, 3rd edition.

[2] Hastie, Tibshirani, & Friedman. (2009). The Elements of Statistical Learning, 2nd edition.

[3] Ng, Andrew. CS229 Lecture Notes on Learning Theory.

[4] Schmidt, Mark. (2005). Least Squares Optimization with L1-Norm Regularization.

[5] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[6] Weatherwax, John & Epstein, David. (2013). A Solution Manual and Notes for: The Elements of Statistical Learning by Jerome Friedman, Trevor Hastie, and Robert Tibshirani.

[7] Wolfram Alpha LLC. (2009). Wolfram|Alpha. http://www.wolframalpha.com/input/?i=x^2%2By^2 (access October 20, 2014).