Matt Dickenson `mcd31`

STA561/CS571 — Fall 2013 **Homework 2** Due: 23 September, 2013

Homework Notes: I did not work with anyone else on this homework or refer to resources other than the course notes, textbook, and course Piazza page. This homework took me $X+2$ hours, where X is the amount of time it took to complete the original assignment and submit it on Friday, and 2 additional hours revising when homework problems were changed several days after being assigned.

Problem 1

A Using the normal equation $\hat{\beta} = (X^TX)^{-1}(X^TY)$, we can calculate $\hat{\beta}$ in $\mathcal{R}$ with the following program (after loading the data as `lindata`):

Listing 1: R Code for 1A

```
X = as.matrix(lindata[1:1000 , 1:2])
X = cbind(1, X)
Y = as.matrix(lindata[1:1000 , 3])

normaleqn = function(x, y){
  x.prime.x.inverse = solve(t(x) %*% x)
  x.prime.y = t(x) %*% y
  beta = x.prime.x.inverse %*% x.prime.y
  return(beta)
}

beta.hat = normaleqn(X, Y)
beta.hat #=> 69.942167, 15.119109, 0.321028
```

Letting β_i correspond to X_i and β_0 represent the intercept, $\hat{\beta}_0 = 69.94, \hat{\beta}_1 = 15.12, \hat{\beta}_2 = 0.32$.

B We can also estimate β using online stochastic gradient descent with $\mathcal{R}$'s `optim` function for minimization:

Listing 2: R Code for 1B

```
linreg = function(X, Y){
  RSS = function(b, x, y){
    residuals = y - (x %*% b )
    sq.resid = residuals^2
    rss = sum(sq.resid)
    return(rss)
  }
  results = optim(rep(0, ncol(X)), RSS,
    hessian=TRUE, method="BFGS", x=X, y=Y)
  list(beta=results$par,
    vcov=solve(results$hessian),
    converged=results$convergence==0)
}
model = linreg(X, Y)
model$beta #=> 69.942167, 15.119109, 0.321028
```

Using this method we reach the same results as above: $\hat{\beta}_0 = 69.94, \hat{\beta}_1 = 15.12, \hat{\beta}_2 = 0.32$.

C A third method we can use to estimate β is ridge regression, using the following $\mathcal{R}$ code:

Listing 3: R Code for 1C

```
ridgereg = function(X, Y, sigma.sq=1, tau.sq=1){
  lambda = sigma.sq / tau.sq
  D = ncol(X)
  lambda.id = lambda * diag(D)
  X.prime.X = t(X) %*% X
  lambda.X.prime.X.inverse = solve(lambda.id + X.prime.X)
  X.prime.Y = t(X) %*% Y
  beta.hat = lambda.X.prime.X.inverse %*% X.prime.Y
  return(beta.hat)
}
beta.ridge = ridgereg(X, Y)
beta.ridge #=> 65.9129285, 14.3521921, 0.3478858
```

From this approach we get the estimates $\hat{\beta}_1 = 3.00, \hat{\beta}_2 = 0.78$.

D Table 1 presents the residual sum of squares (RSS) for the training data using each of the methods above.

Table 1: Training set RSS for linear regression methods

Method	RSS
Normal equations	98,729.3
Online stochastic gradient descent	98,729.3
Ridge regression	99,005.9

E Table 2 presents the residual sum of squares (RSS) for the test data using each of the methods above.

Table 2: Test set RSS for linear regression methods

Method	RSS
Normal equations	11,573.2
Online stochastic gradient descent	11,573.2
Ridge regression	11,523.5

F We can now take the predicted blood pressure for a hypothetical female weighing 135 pounds for the three different methods (Table 3):

Table 3: Predicted values

Method	$\mathbb{E}[Y \mid X = [1, 135]^T, \hat{\beta}]$
Normal equations	128.4
Online stochastic gradient descent	128.4
Ridge regression	127.2

G Bivariate plots are given in Figures 1 and 2 below, along with regression lines for the three methods used above. The y-axis values are $Y - \hat{\beta}_i X_i - \hat{\beta}_0$ and the x-axis values are $X_j, j \neq i$. (Note that indices in the y-axis label do not correspond to the X and $\hat{\beta}$ subscripts because $\mathcal{R}$ uses 1-based indexing.) Regression lines in the plots for X_i use $\hat{\beta}_i X_i$

H Of the three methods used above, ridge regression is the least likely to overfit the data. This is because the λ term (using the notation in the MLAPP textbook) helps to regularize the coefficients. This leads to higher RSS in the training data but also makes the coefficients less susceptible to outliers, as evidenced by the lower RSS using the test set data. The other two methods provide no such protection against outliers, and are thus more likely to overfit the training data.

I The researchers should use the ridge regresion results. One reason for this is its robustness (relative to the other two methods) to overfitting. This is especially pertinent given the small n of the training data. However, the similarity in the coefficients should help allay any concerns the researchers may have over the differences in the three methods.

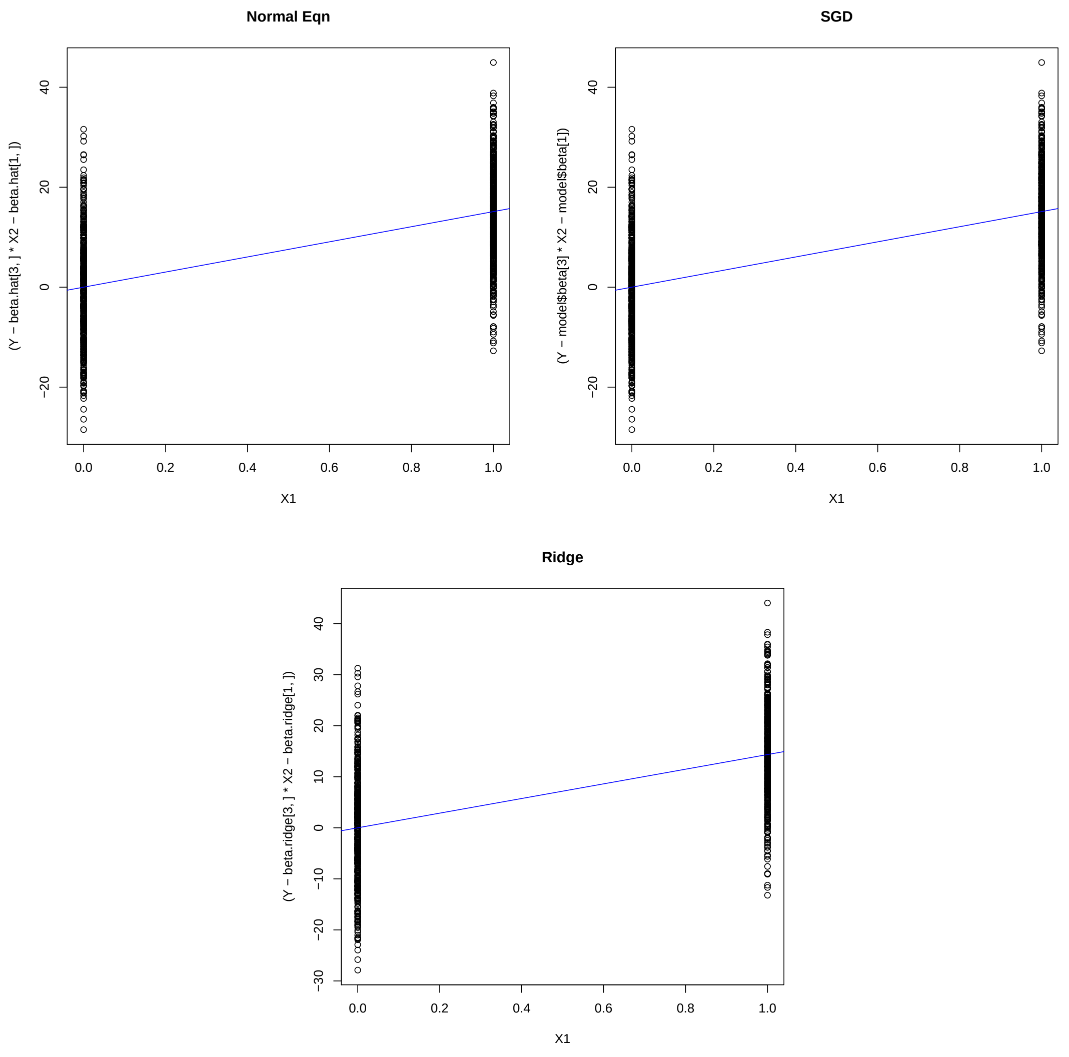

Figure 1: X_1 and Y with Regression Lines

Problem 2

A The following $\mathcal{R}$ code implements the IRLS algorithm as presented in the MLAPP textbook (after loading the data):

Listing 4: R Code for 2A

```
X = as.matrix(logdata[1:1000 , 1:2])
X = cbind(1, X)
Z = as.matrix(logdata[1:1000 , 3])

sigm = function(eta){
```

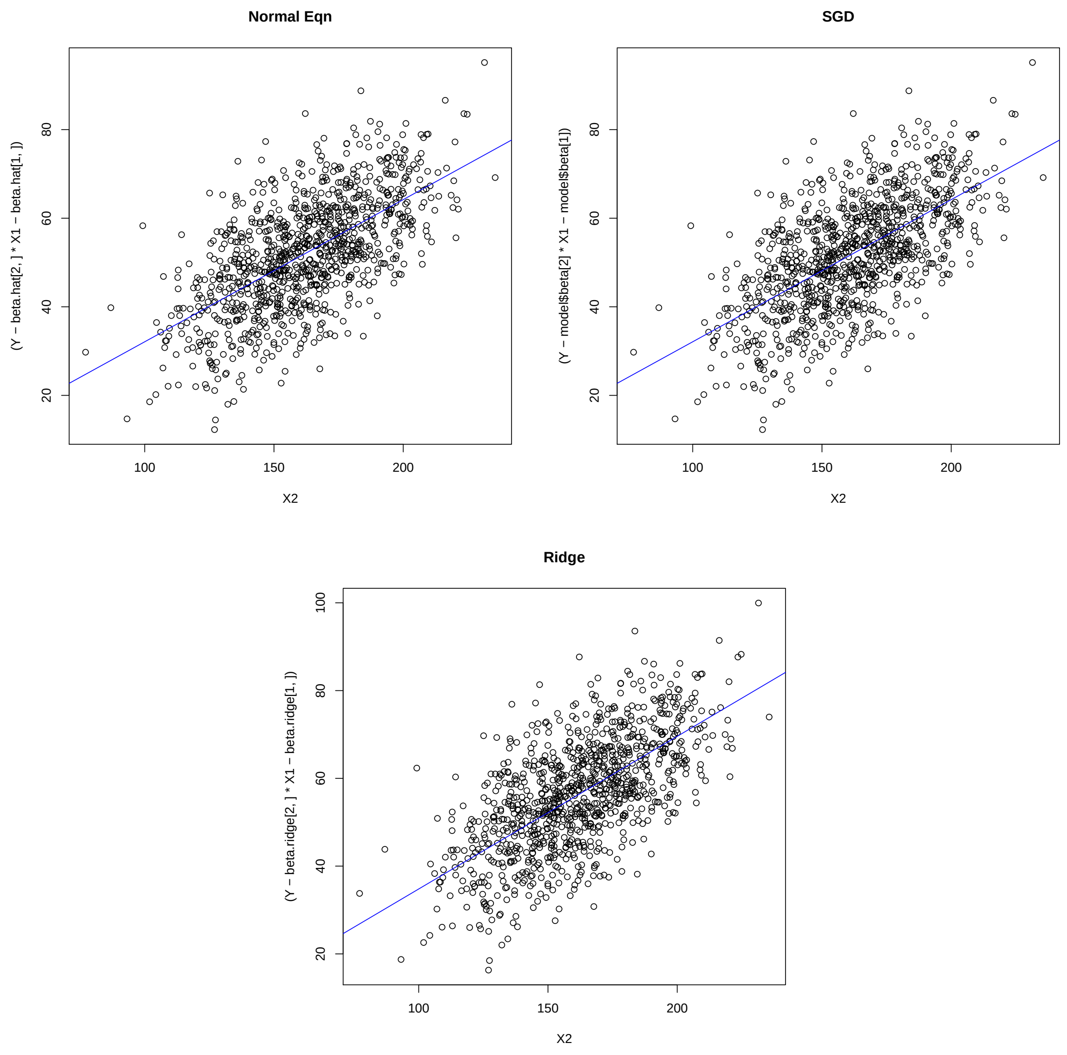

Figure 2: X_2 and Y with Regression Lines

```
  out = exp(eta)/( exp(eta) + 1)
  return(out)
}

irls = function(X, Y, epsilon=1/1e10){
  D = ncol(X)
  N = nrow(X)

  w = matrix(0, nrow=D, ncol=D)
  w0 = log(mean(Y)/(1-mean(Y)))
  eta = mu = s = z = matrix(NA, nrow=N, ncol=D)

  converged = FALSE
  numiters = 0
  while(!converged){
    last.w = w
    for(i in 1:N){
      xi = matrix(X[i, ], ncol=1)
      eta[i, ] = w0 + (t(w) %*% xi)
      mu[i, ] = sigm(eta[i, ])
      s[i, ] = mu[i, ] * (1-mu[i, ])
      z[i, ] = eta[i, ] + ((Y[i, ] - mu[i, ])/s[i, ])
    }
    S = matrix(0, nrow=N, ncol=N)
    for(i in 1:N){
      S[i, i] = s[i]
    }
    first.term = solve(t(X) %*% S %*% X)
    second.term = t(X) %*% S %*% z
    w =  first.term %*% second.term
    numiters = numiters + 1
    diff = w - last.w
    smalldiff = TRUE
    for(i in 1:D){
      smalldiff = smalldiff & (diff[i, i] < epsilon)
    }
    if(smalldiff){
      converged = TRUE
      output = paste("converged after ", numiters, " iterations", sep="")
      print(output)
    }
    if(numiters > 100){
      print("reached max iterations")
      break
    }
  }
  return(w)
}

w = irls(X, Z) #=> -13.52371957, 0.24204760, 0.08109647
```

Note that the results do not change (in this instance, at least) when the data is permuted between steps.

B The RSS($\hat{\beta}$) for the training data is 141.47.

C The RSS($\hat{\beta}$) for the test data is 15.73.

D $\mathbb{E}[Z|X = [1, 135]^T, \hat{\beta}] = 0.088$

E Bivariate plots are given in Figures 3 and 4 below, along with regression lines from the IRLS results. The y-axis values are $Z - \hat{\beta}_i X_i - \hat{\beta}_0$ and the x-axis values are $X_j, j \neq i$. (Note that indices in the y-axis label do not correspond to the X and $\hat{\beta}$ subscripts because $\mathcal{R}$ uses 1-based indexing.) Regression lines in the plots for X_i use $\hat{\beta}_i X_i$

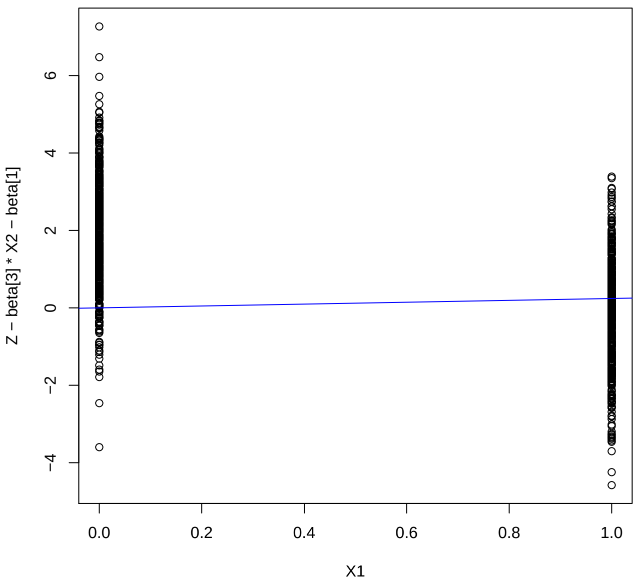

Figure 3: X_1 and Z with Regression Line

F The correlation between the features X_1 and X_2 is high (0.63). This causes our estimates of β to be biased, since it is difficult to isolate the impact of each feature. If the features are gender and weight, this is because the mean of the male weight distribution is higher than that for females. However, both features are still important to the analysis.

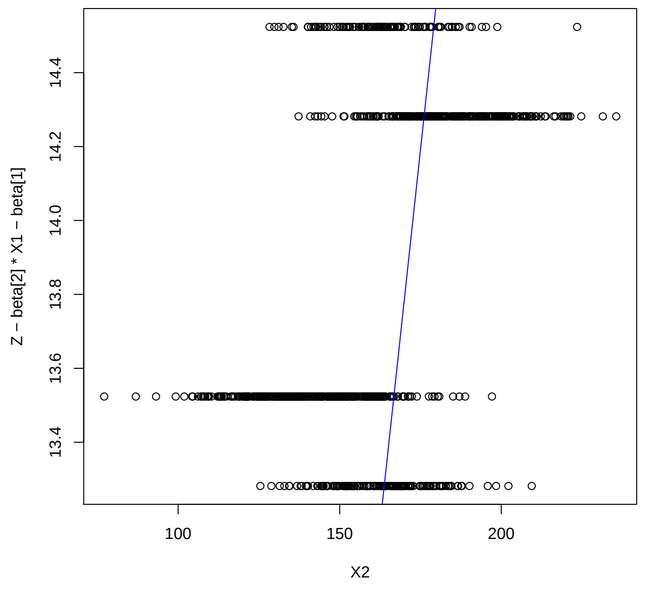

Figure 4: X_2 and Z with Regression Line

G Although X_1 and X_2 are correlated, both features are important for the decision rule used to predict Z^*. Assuming X_1 is gender (female=1) and X_2 is weight, the decision rule I would recommend (based on Figure 4) is that females with a weight over about 145 pounds and males with a weight over about 160 pounds have a high risk of Z. These patients should be counseled to seek additional diagnostic tests.