Lab 4

Jia Yu Lin

11:59PM March 11, 2021

Load up the famous iris dataset. We are going to do a different prediction problem. Imagine the only input x is Species and you are trying to predict y which is Petal.Length. A reasonable prediction is the average petal length within each Species. Prove that this is the OLS model by fitting an appropriate `lm` and then using the predict function to verify.

```
data(iris)
mod =lm( Petal.Length ~ Species, iris)
mod
```

```
##
## Call:
## lm(formula = Petal.Length ~ Species, data = iris)
##
## Coefficients:
##       (Intercept)  Speciesversicolor   Speciesvirginica
##             1.462              2.798              4.090
```

```
mean(iris$Petal.Length[iris$Species == 'setosa'])
```

```
## [1] 1.462
```

```
mean(iris$Petal.Length[iris$Species == 'versicolor'])
```

```
## [1] 4.26
```

```
mean(iris$Petal.Length[iris$Species == 'virginica'])
```

```
## [1] 5.552
```

```
predict(mod, data.frame(Species = c("setosa")))
```

```
##     1
## 1.462
```

```
predict(mod, data.frame(Species = c("versicolor")))
```

```
##    1
## 4.26
```

```
predict(mod, data.frame(Species = c("virginica")))
```

```
##     1
## 5.552
```

Construct the design matrix with an intercept, X, without using `model.matrix`.

```
#TO-DO
X = cbind(1,iris$Species == 'versicolor', iris$Species == 'virginica')
View(X)
head(X)
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    1    0    0
## [3,]    1    0    0
## [4,]    1    0    0
## [5,]    1    0    0
## [6,]    1    0    0
```

Find the hat matrix H for this regression.

```
#TO-DO
H = X %*% solve(t(X) %*% X) %*% t(X)
Matrix::rankMatrix(H)
```

```
## [1] 3
## attr(,"method")
## [1] "tolNorm2"
## attr(,"useGrad")
## [1] FALSE
## attr(,"tol")
## [1] 3.330669e-14
```

```
#head(H)
```

Verify this hat matrix is symmetric using the `expect_equal` function in the package `testthat`.

```
#TO-DO
pacman::p_load(testthat)
expect_equal(H, t(H))
```

Verify this hat matrix is idempotent using the `expect_equal` function in the package `testthat`.

```
#TO-DO
expect_equal(H , H%*%H)
```

Using the `diag` function, find the trace of the hat matrix.

```r
#TO-DO
#trace is the sum of diagonal
sum(diag(H))
```

```
## [1] 3
```

```r
#sum of trace is rank
```

It turns out the trace of a hat matrix is the same as its rank! But we don't have time to prove these interesting and useful facts..

For masters students: create a matrix $X_\perp$.

```r
#TO-DO
```

Using the hat matrix, compute the $\hat{y}$ vector and using the projection onto the residual space, compute the e vector and verify they are orthogonal to each other.

```r
#TO-DO
y = iris$Petal.Length
y_hat = H %*% iris$Petal.Length
#we are supposed to see y bars for setosa, versicolor, or virignica
#table(y_hat)
I = diag(nrow(iris))
e = (I - H) %*% y
head(e)
```

```
##        [,1]
## [1,] -0.062
## [2,] -0.062
## [3,] -0.162
## [4,]  0.038
## [5,] -0.062
## [6,]  0.238
```

```r
Matrix::rankMatrix(I-H)
```

```
## [1] 147
## attr(,"method")
## [1] "tolNorm2"
## attr(,"useGrad")
## [1] FALSE
## attr(,"tol")
## [1] 3.330669e-14
```

Compute SST, SSR and SSE and R^2 and then show that SST = SSR + SSE.

```r
#TO-DO
SSE = t(e) %*% e  #same thing is sum(e^2)
y_bar = mean(y)
SST = t(y - y_bar)  %*% (y - y_bar)

Rsq = 1 - SSE/SST
Rsq
```

```
##            [,1]
## [1,] 0.9413717
```

```
SSR = t(y_hat - y_bar) %*% (y_hat - y_bar)
SSR
```

```
##           [,1]
## [1,] 437.1028
```

```
expect_equal(SSE+SSR, SST)
#this would mean each species would have similar petal length
```

Find the angle θ between y - $\bar{y}1$ and $\hat{y} - \bar{y}1$ and then verify that its cosine squared is the same as the R^2 from the previous problem.

```
#TO-DO
#231 formula
theta = acos(t(y - y_bar) %*% (y_hat - y_bar) / sqrt(SST * SSR))
#Rsq was pretty large so theta should be pretty small
#theta is 14 degrees
theta  * (180/pi)
```

```
##           [,1]
## [1,] 14.01245
```

```
cos(theta)^2
```

```
##            [,1]
## [1,] 0.9413717
```

```
expect_equal(cos(theta)^2, Rsq)
```

Project the y vector onto each column of the X matrix and test if the sum of these projections is the same as yhat.

```
#TO-DO
proj1= (X[,1] %*% t(X[,1]) / as.numeric(t(X[,1]) %*% X[,1])) %*% y #H on to X1
proj2= (X[,2] %*% t(X[,2]) / as.numeric(t(X[,2]) %*% X[,2])) %*% y #H on to X2
proj3= (X[,3] %*% t(X[,3]) / as.numeric(t(X[,3]) %*% X[,3])) %*% y #H on to X3

expect_equal(proj1+proj2+proj3, y_hat)
#this will fail which is what we want
```

Construct the design matrix without an intercept, X, without using `model.matrix`.

```
#TO-DO
X = cbind( 1e-4, as.numeric(iris$Species == 'versicolor'), as.numeric(iris$Species == 'virginica'))
head(X)
```

```
##        [,1] [,2] [,3]
## [1,] 1e-04    0    0
## [2,] 1e-04    0    0
## [3,] 1e-04    0    0
## [4,] 1e-04    0    0
## [5,] 1e-04    0    0
## [6,] 1e-04    0    0
```

```
#iris
```

Find the OLS estimates using this design matrix. It should be the sample averages of the petal lengths within species.

```
#TO-DO
mod_X = lm(Petal.Length ~ X, iris)
mod_X
```

```
##
## Call:
## lm(formula = Petal.Length ~ X, data = iris)
##
## Coefficients:
## (Intercept)           X1           X2           X3
##       1.462           NA        2.798        4.090
```

Verify the hat matrix constructed from this design matrix is the same as the hat matrix constructed from the design matrix with the intercept. (Fact: orthogonal projection matrices are unique).

```
#TO-DO
H_new = X %*% solve(t(X) %*% X) %*% t(X)

expect_equal(H_new, H, tol= 1e-4)
```

Project the y vector onto each column of the X matrix and test if the sum of these projections is the same as yhat.

```
#TO-DO
proj1= (X[,1] %*% t(X[,1]) / as.numeric(t(X[,1]) %*% X[,1])) %*% y #H on to X1
proj2= (X[,2] %*% t(X[,2]) / as.numeric(t(X[,2]) %*% X[,2])) %*% y #H on to X2
proj3= (X[,3] %*% t(X[,3]) / as.numeric(t(X[,3]) %*% X[,3])) %*% y #H on to X3

expect_equal(proj1+proj2+proj3, y_hat)
```

Convert this design matrix into Q, an orthonormal matrix.

```
#TO-DO
qrX = qr(X)
Q = qr.Q(qrX)
```

Project the y vector onto each column of the Q matrix and test if the sum of these projections is the same as yhat.

```r
#TO-DO

proj1= (Q[,1] %*% t(Q[,1]) / as.numeric(t(Q[,1]) %*% Q[,1])) %*% y #H on to X1
proj2= (Q[,2] %*% t(Q[,2]) / as.numeric(t(Q[,2]) %*% Q[,2])) %*% y #H on to X2
proj3= (Q[,3] %*% t(Q[,3]) / as.numeric(t(Q[,3]) %*% Q[,3])) %*% y #H on to X3

expect_equal(proj1+proj2+proj3, y_hat)
```

Find the $p = 3$ linear OLS estimates if Q is used as the design matrix using the `lm` method. Is the OLS solution the same as the OLS solution for X?

```r
#TO-DO

mod_Q = lm(Petal.Length ~ Q, iris)
mod_Q
```

```
##
## Call:
## lm(formula = Petal.Length ~ Q, data = iris)
##
## Coefficients:
## (Intercept)           Q1           Q2           Q3
##       3.758           NA        4.347       20.450
```

```r
#It is not the same as the OLS solution for X
```

Use the predict function and ensure that the predicted values are the same for both linear models: the one created with X as its design matrix and the one created with Q as its design matrix.

```r
#TO-DO

pred_X = predict(mod_X)
pred_Q = predict(mod_Q)

expect_equal(pred_X, pred_Q)
```

Clear the workspace and load the boston housing data and extract X and y. The dimensions are $n = 506$ and $p = 13$. Create a matrix that is $(p + 1) \times (p + 1)$ full of NA's. Label the columns the same columns as X. Do not label the rows. For the first row, find the OLS estimate of the y regressed on the first column only and put that in the first entry. For the second row, find the OLS estimates of the y regressed on the first and second columns of X only and put them in the first and second entries. For the third row, find the OLS estimates of the y regressed on the first, second and third columns of X only and put them in the first, second and third entries, etc. For the last row, fill it with the full OLS estimates.

```r
#TO-DO
rm(list = ls())

boston = MASS::Boston
#?Boston
X = cbind(1,as.matrix(boston[, 1:13]))
Y = boston[, 14]
```

```r
X_matrix = matrix(NA, nrow= 14, ncol = 14)

colnames(X_matrix) <- c(colnames(X))
X_matrix
```

```
##          crim zn indus chas nox rm age dis rad tax ptratio black lstat
##  [1,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
##  [2,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
##  [3,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
##  [4,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
##  [5,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
##  [6,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
##  [7,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
##  [8,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
##  [9,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
## [10,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
## [11,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
## [12,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
## [13,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
## [14,] NA   NA NA    NA   NA  NA NA  NA  NA  NA  NA       NA    NA    NA
```

```r
for( i in 1:ncol(X)){
  mod_coef = coef(lm(Y ~ X[,1:i], data= as.data.frame(boston)))
  count = 1
  for( j in 2:(length(mod_coef))){
    X_matrix[i, count] = mod_coef[j]
    X_matrix[i, 1] = mod_coef[1]
    count = count+1
  }
}
X_matrix
```

```
##                         crim         zn       indus     chas        nox
##  [1,]  22.5328063         NA         NA          NA       NA         NA
##  [2,]  24.0331062 -0.4151903         NA          NA       NA         NA
##  [3,]  22.4856281 -0.3520783 0.11610909          NA       NA         NA
##  [4,]  27.3946468 -0.2486283 0.05850082 -0.41557782       NA         NA
##  [5,]  27.1128031 -0.2287981 0.05928665 -0.44032511 6.894059         NA
##  [6,]  29.4899406 -0.2185190 0.05511047 -0.38348055 7.026223  -5.424659
##  [7,] -17.9546350 -0.1769135 0.02128135 -0.14365267 4.784684  -7.184892
##  [8,] -18.2649261 -0.1727607 0.01421402 -0.13089918 4.840730  -4.357411
##  [9,]   0.8274820 -0.1977868 0.06099257 -0.22573089 4.577598 -14.451531
## [10,]   0.1553915 -0.1780398 0.06095248 -0.21004328 4.536648 -13.342666
## [11,]   2.9907868 -0.1795543 0.07145574 -0.10437742 4.110667 -12.591596
## [12,]  27.1523679 -0.1840321 0.03909990 -0.04232450 3.487528 -22.182110
## [13,]  20.6526280 -0.1599391 0.03887365 -0.02792186 3.216569 -20.484560
## [14,]  36.4594884 -0.1080114 0.04642046  0.02055863 2.686734 -17.766611
##              rm           age        dis          rad          tax     ptratio
##  [1,]        NA            NA         NA           NA           NA          NA
##  [2,]        NA            NA         NA           NA           NA          NA
##  [3,]        NA            NA         NA           NA           NA          NA
##  [4,]        NA            NA         NA           NA           NA          NA
```

```
## [5,]       NA            NA        NA          NA          NA         NA
## [6,]       NA            NA        NA          NA          NA         NA
## [7,] 7.341586            NA        NA          NA          NA         NA
## [8,] 7.386357 -0.0236248493        NA          NA          NA         NA
## [9,] 6.752352 -0.0556354540 -1.760312          NA          NA         NA
## [10,] 6.791184 -0.0562612189 -1.748296 -0.04529059          NA         NA
## [11,] 6.664084 -0.0546675064 -1.727933  0.15926305 -0.01434060         NA
## [12,] 6.075744 -0.0451880522 -1.583852  0.25472196 -0.01221262 -0.9962062
## [13,] 6.123072 -0.0459320518 -1.554912  0.28157503 -0.01173838 -1.0142228
## [14,] 3.809865  0.0006922246 -1.475567  0.30604948 -0.01233459 -0.9527472
##            black      lstat
##  [1,]          NA         NA
##  [2,]          NA         NA
##  [3,]          NA         NA
##  [4,]          NA         NA
##  [5,]          NA         NA
##  [6,]          NA         NA
##  [7,]          NA         NA
##  [8,]          NA         NA
##  [9,]          NA         NA
## [10,]          NA         NA
## [11,]          NA         NA
## [12,]          NA         NA
## [13,] 0.013620833         NA
## [14,] 0.009311683 -0.5247584
```

Why are the estimates changing from row to row as you add in more predictors?

#TO-DO The estimates are changing as more predictors are added because it is trying to find a better fit.

Create a vector of length $p + 1$ and compute the R^2 values for each of the above models.

```r
#TO-DO
y <- c()
for( j in 1:14){
  mod_coef = lm(Y~X[,1:j])
  y <- append(y, summary(mod_coef)$r.squared)
}
y
```

```
##  [1] 0.0000000 0.1507805 0.2339884 0.2937136 0.3295277 0.3313127 0.5873770
##  [8] 0.5894902 0.6311488 0.6319479 0.6396628 0.6703141 0.6842043 0.7406427
```

Is R^2 monotonically increasing? Why?

#TO-DO R square increases because the amount of features we have went up