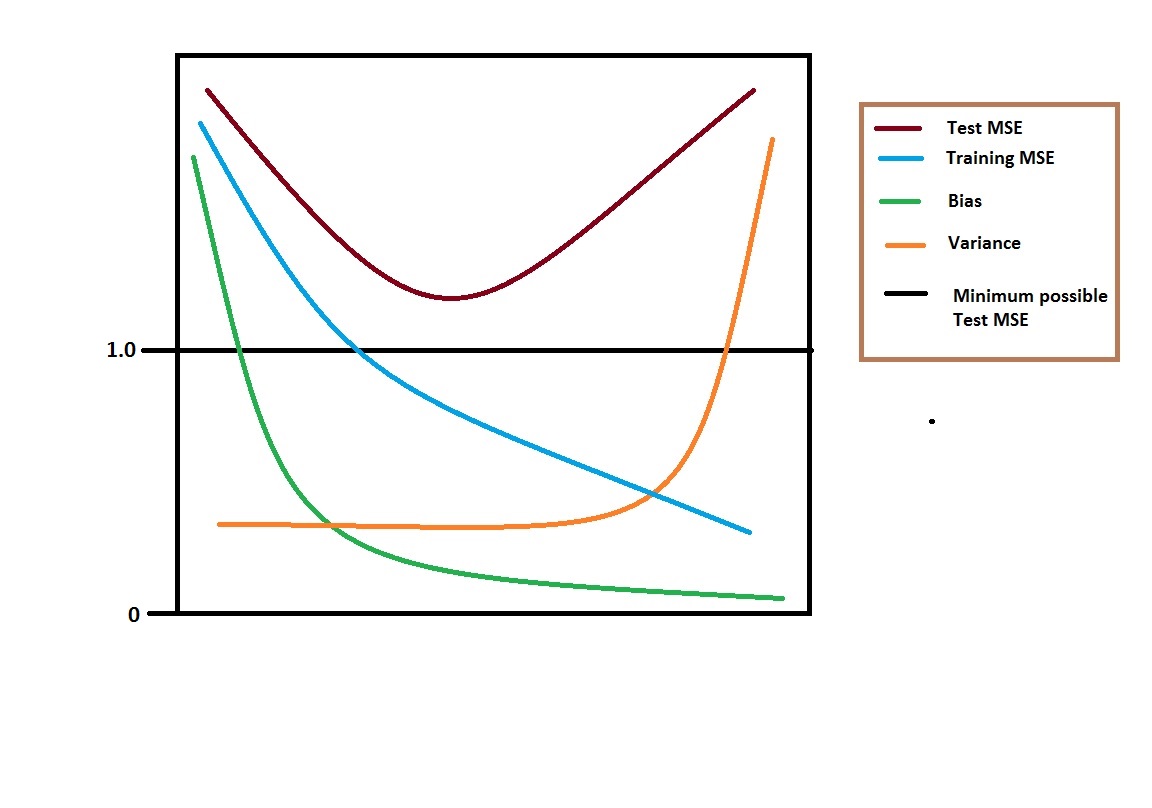
**Question 1:**

**Solution:**

**(a)**



**Errors**

**Flexibility**

**(b)**

1. **(Squared) bias** decreases consistently because increases in flexibility give a closer fit

2. **Variance** increases consistently because increases in flexibility give overfitted data

3. **Training error** decreases consistently because increases in flexibility give a closer fit

4. **Test error** shows concave up curve because increase in flexibility yields a closer fit before it overfits

5. **Bayes (irreducible) error** defines the lower limit, the test error is bounded below by the irreducible error due to variance in the error (epsilon) in the output values (0 <= value). When the training error is lower than the irreducible error, overfitting has taken place.

The Bayes error rate is defined for classification problems and is determined by the ratio of data points which lie at the 'wrong' side of the decision boundary, (0 <= value < 1).

**Question 2:**

**Solution:**

The advantages for a very flexible approach for regression or classification are obtaining a better fit for non-linear models, decreasing bias.

The disadvantages for a very flexible approach for regression or classification are requires estimating a greater number of parameters, follow the noise too closely (overfit), increasing variance.

A more flexible approach would be preferred to a less flexible approach when we are interested in prediction and not the interpretability of the results.

A less flexible approach would be preferred to a more flexible approach when we are interested in inference and the interpretability of the results.

**Question 3:**

**Solution:**

**R code:**

#Q3(a)

college= read.csv("College.csv")

#Q3(b)

fix(college)

rownames(college)=college[,1]

fix(college)

college=college[,-1]

fix(college)

#Q3(c)

summary(college)

pairs(college)

pairs(college[,1:10])

attach(college)

plot(Outstate,Private)

Private=as.factor(Private)

plot(Private, Outstate, xlab="Private", ylab="Outstate")

Elite=rep("No", nrow(college))

Elite[college$Top10perc>50]="Yes"

Elite=as.factor(Elite)

college=data.frame(college,Elite)

summary(college)

plot(Elite,Outstate,xlab="Elite", ylab="Outstate")

par(mfrow=c(3,3))

hist(Apps)

hist(Accept)

hist(Enroll)

hist(Top10perc)

hist(Top25perc)

hist(F.Undergrad)

hist(P.Undergrad)

hist(Outstate)

hist(Room.Board)

hist(Books)

hist(Personal)

hist(PhD)

hist(Terminal)

hist(S.F.Ratio)

hist(perc.alumni)

hist(Expend)

hist(Grad.Rate)

**Console:**

college= read.csv("College.csv")

> fix(college)

> rownames(college)=college[,1]

> fix(college)

> college=college[,-1]

> fix(college)

> summary(college)

Private Apps Accept Enroll Top10perc Top25perc F.Undergrad P.Undergrad

No :212 Min. : 81 Min. : 72 Min. : 35 Min. : 1.00 Min. : 9.0 Min. : 139 Min. : 1.0

Yes:565 1st Qu.: 776 1st Qu.: 604 1st Qu.: 242 1st Qu.:15.00 1st Qu.: 41.0 1st Qu.: 992 1st Qu.: 95.0

Median : 1558 Median : 1110 Median : 434 Median :23.00 Median : 54.0 Median : 1707 Median : 353.0

Mean : 3002 Mean : 2019 Mean : 780 Mean :27.56 Mean : 55.8 Mean : 3700 Mean : 855.3

3rd Qu.: 3624 3rd Qu.: 2424 3rd Qu.: 902 3rd Qu.:35.00 3rd Qu.: 69.0 3rd Qu.: 4005 3rd Qu.: 967.0

Max. :48094 Max. :26330 Max. :6392 Max. :96.00 Max. :100.0 Max. :31643 Max. :21836.0

Outstate Room.Board Books Personal PhD Terminal S.F.Ratio

Min. : 2340 Min. :1780 Min. : 96.0 Min. : 250 Min. : 8.00 Min. : 24.0 Min. : 2.50

1st Qu.: 7320 1st Qu.:3597 1st Qu.: 470.0 1st Qu.: 850 1st Qu.: 62.00 1st Qu.: 71.0 1st Qu.:11.50

Median : 9990 Median :4200 Median : 500.0 Median :1200 Median : 75.00 Median : 82.0 Median :13.60

Mean :10441 Mean :4358 Mean : 549.4 Mean :1341 Mean : 72.66 Mean : 79.7 Mean :14.09

3rd Qu.:12925 3rd Qu.:5050 3rd Qu.: 600.0 3rd Qu.:1700 3rd Qu.: 85.00 3rd Qu.: 92.0 3rd Qu.:16.50

Max. :21700 Max. :8124 Max. :2340.0 Max. :6800 Max. :103.00 Max. :100.0 Max. :39.80

perc.alumni Expend Grad.Rate

Min. : 0.00 Min. : 3186 Min. : 10.00

1st Qu.:13.00 1st Qu.: 6751 1st Qu.: 53.00

Median :21.00 Median : 8377 Median : 65.00

Mean :22.74 Mean : 9660 Mean : 65.46

3rd Qu.:31.00 3rd Qu.:10830 3rd Qu.: 78.00

Max. :64.00 Max. :56233 Max. :118.00

> pairs(college)

> pairs(college[,1:10])

> attach(college)

> plot(Outstate,Private)

> Private=as.factor(Private)

> plot(Private, Outstate, xlab="Private", ylab="Outstate")

> Elite=rep("No", nrow(college))

> Elite[college$Top10perc>50]="Yes"

> Elite=as.factor(Elite)

> college=data.frame(college,Elite)

> summary(college)

Private Apps Accept Enroll Top10perc Top25perc F.Undergrad P.Undergrad

No :212 Min. : 81 Min. : 72 Min. : 35 Min. : 1.00 Min. : 9.0 Min. : 139 Min. : 1.0

Yes:565 1st Qu.: 776 1st Qu.: 604 1st Qu.: 242 1st Qu.:15.00 1st Qu.: 41.0 1st Qu.: 992 1st Qu.: 95.0

Median : 1558 Median : 1110 Median : 434 Median :23.00 Median : 54.0 Median : 1707 Median : 353.0

Mean : 3002 Mean : 2019 Mean : 780 Mean :27.56 Mean : 55.8 Mean : 3700 Mean : 855.3

3rd Qu.: 3624 3rd Qu.: 2424 3rd Qu.: 902 3rd Qu.:35.00 3rd Qu.: 69.0 3rd Qu.: 4005 3rd Qu.: 967.0

Max. :48094 Max. :26330 Max. :6392 Max. :96.00 Max. :100.0 Max. :31643 Max. :21836.0

Outstate Room.Board Books Personal PhD Terminal S.F.Ratio

Min. : 2340 Min. :1780 Min. : 96.0 Min. : 250 Min. : 8.00 Min. : 24.0 Min. : 2.50

1st Qu.: 7320 1st Qu.:3597 1st Qu.: 470.0 1st Qu.: 850 1st Qu.: 62.00 1st Qu.: 71.0 1st Qu.:11.50

Median : 9990 Median :4200 Median : 500.0 Median :1200 Median : 75.00 Median : 82.0 Median :13.60

Mean :10441 Mean :4358 Mean : 549.4 Mean :1341 Mean : 72.66 Mean : 79.7 Mean :14.09

3rd Qu.:12925 3rd Qu.:5050 3rd Qu.: 600.0 3rd Qu.:1700 3rd Qu.: 85.00 3rd Qu.: 92.0 3rd Qu.:16.50

Max. :21700 Max. :8124 Max. :2340.0 Max. :6800 Max. :103.00 Max. :100.0 Max. :39.80

perc.alumni Expend Grad.Rate Elite

Min. : 0.00 Min. : 3186 Min. : 10.00 No :699

1st Qu.:13.00 1st Qu.: 6751 1st Qu.: 53.00 Yes: 78

Median :21.00 Median : 8377 Median : 65.00

Mean :22.74 Mean : 9660 Mean : 65.46

3rd Qu.:31.00 3rd Qu.:10830 3rd Qu.: 78.00

Max. :64.00 Max. :56233 Max. :118.00

> plot(Elite,Outstate,xlab="Elite", ylab="Outstate")

> par(mfrow=c(3,3))

> hist(Apps)

> hist(Accept)

> hist(Enroll)

> hist(Top10perc)

> hist(Top25perc)

> hist(F.Undergrad)

> hist(P.Undergrad)

> hist(Outstate)

> hist(Room.Board)

> hist(Books)

> hist(Personal)

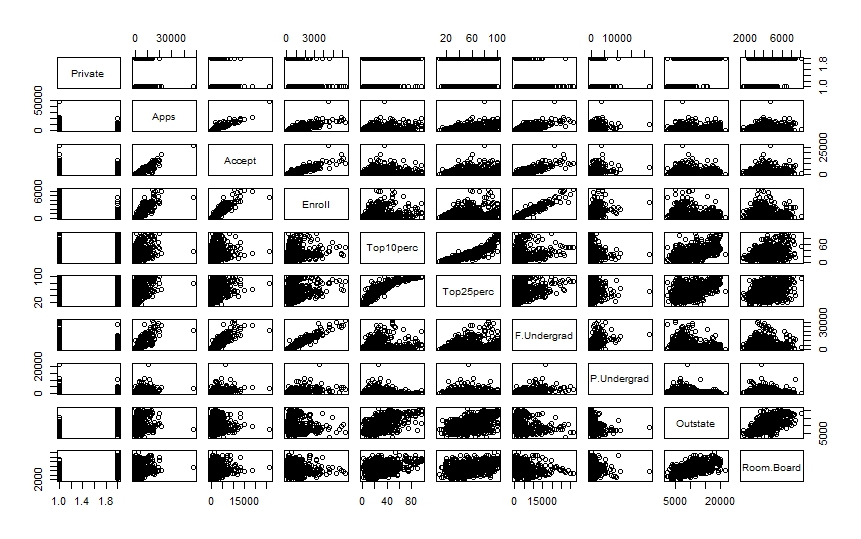
> hist(PhD)

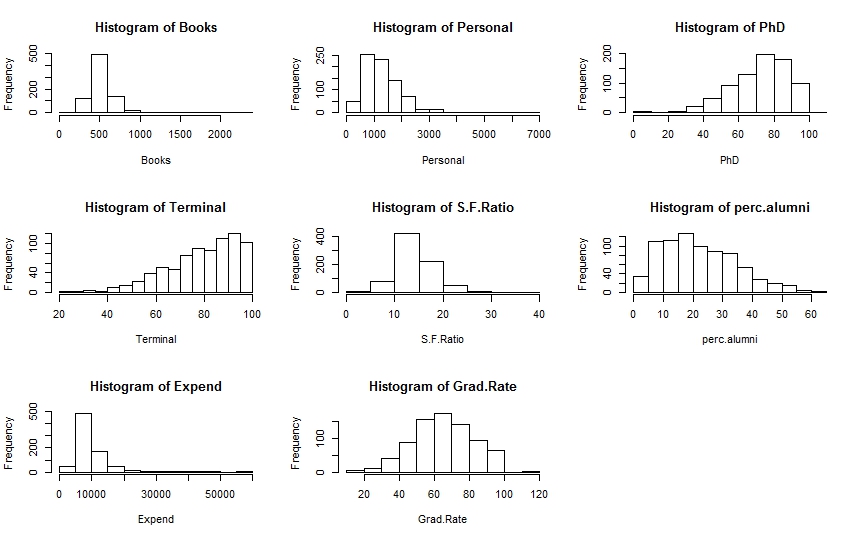
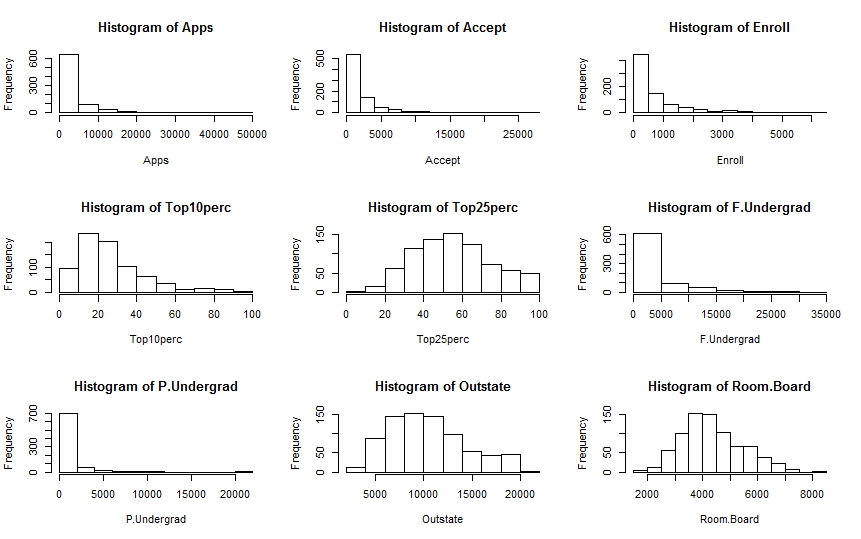
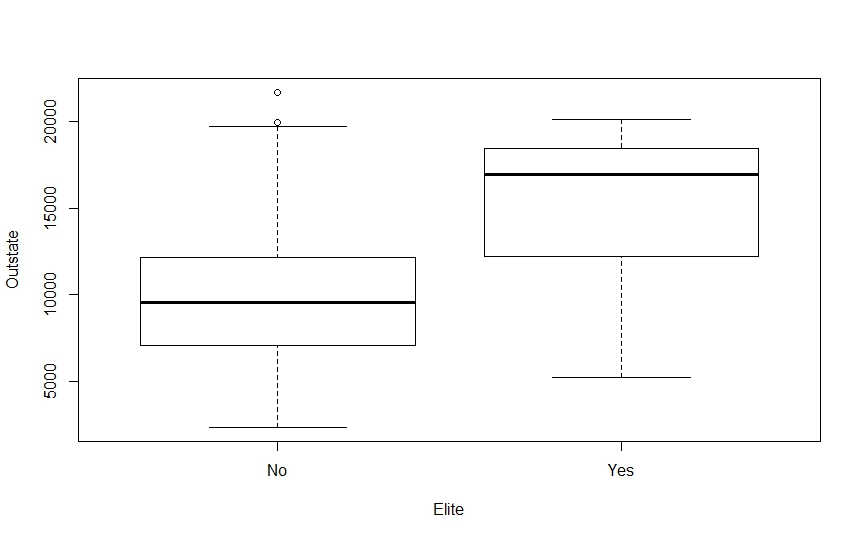
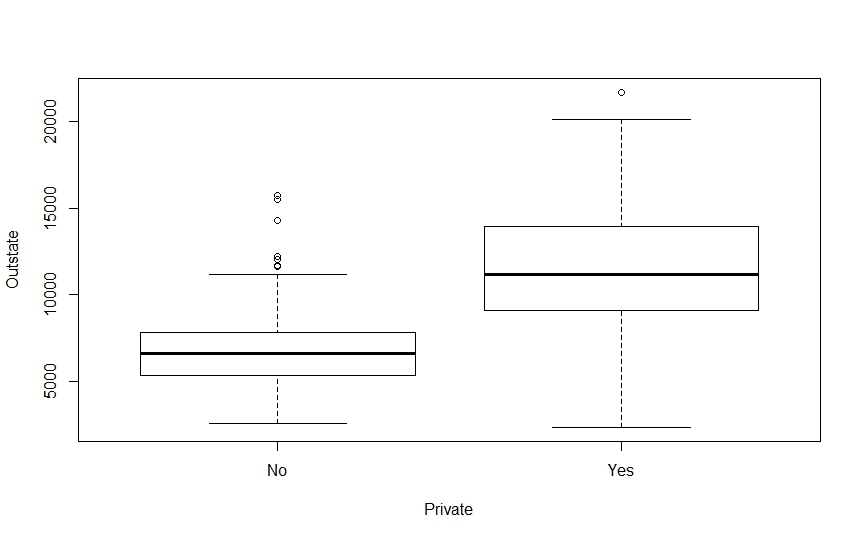
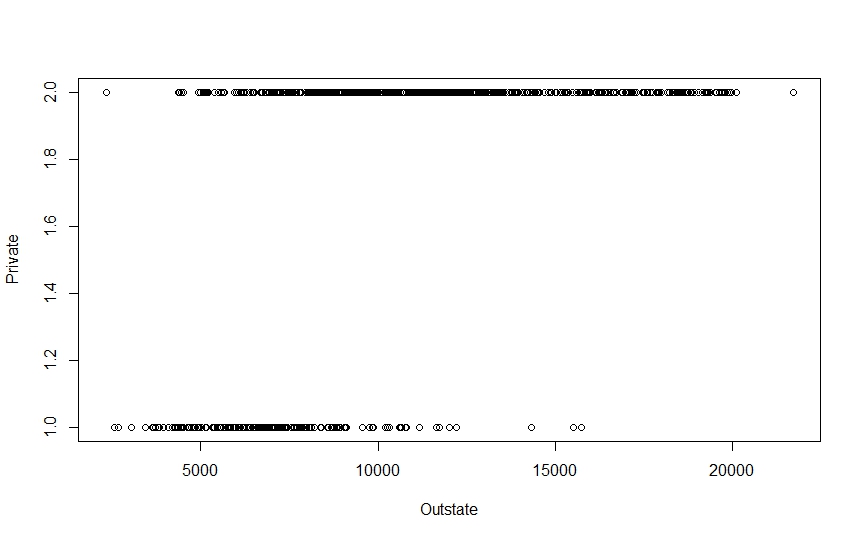
> hist(Terminal)

> hist(S.F.Ratio)

> hist(perc.alumni)

> hist(Expend)

> hist(Grad.Rate) 



Comments: Colleges with the most students from top 10% percent don't necessarily have the highest graduation rate.

**Question 4:**

**Solution:**

Y = 50 + 20(GPA) + 0.07(IQ) + 35(gender) + 0.01(GPA \* IQ) - 10 (GPA \* gender)

(a) Y = 50 + 20\*x1 + 0.07\*x2 + 35\*gender + 0.01(x1 \* x2) - 10 (x1 \* gender)

male: (gender = 0) 50 + 20\*x1 + 0.07\*x2 + 0.01(x1 \* x2)

female: (gender = 1) 50 + 20\*x1 + 0.07\*x2 + 35 + 0.01(x1 \* x2) - 10 (x1)

When GPA is high, more than 3.5, males earn more on average.

So, answer is (iii)

(b) Y (Gender = 1, IQ = 110, GPA = 4.0)

= 50 + 20 \* 4 + 0.07 \* 110 + 35 + 0.01 (4 \* 110) - 10 \* 4

= 137.1

(c) False.

p-value of the regression coefficient has to be considered to determine the statistical significance of the interaction term.

**Question 5:**

**Solution:**

**R code:**

#Q5(a)

Auto = read.csv("Auto.csv", header=T, na.strings="?")

Auto = na.omit(Auto)

summary(Auto)

attach(Auto)

lm.fit = lm(mpg ~ horsepower)

summary(lm.fit)

predict(lm.fit, data.frame(horsepower=c(98)), interval="confidence")

predict(lm.fit, data.frame(horsepower=c(98)), interval="prediction")

1. Yes, there is a relationship between horsepower and mpg.
2. The mean of mpg is 23.446. The RSE of the lm.fit was 4.096 which indicates a percentage error of 20.925%. The R2 of the lm.fit is 0.6059, i.e. 60.59% of the variance in mpg is explained by horsepower.
3. The relationship between mpg and horsepower is negative. It is shown in the mpg vs. horsepower plot
4. fit lwr upr

24.46708 23.97308 24.96108

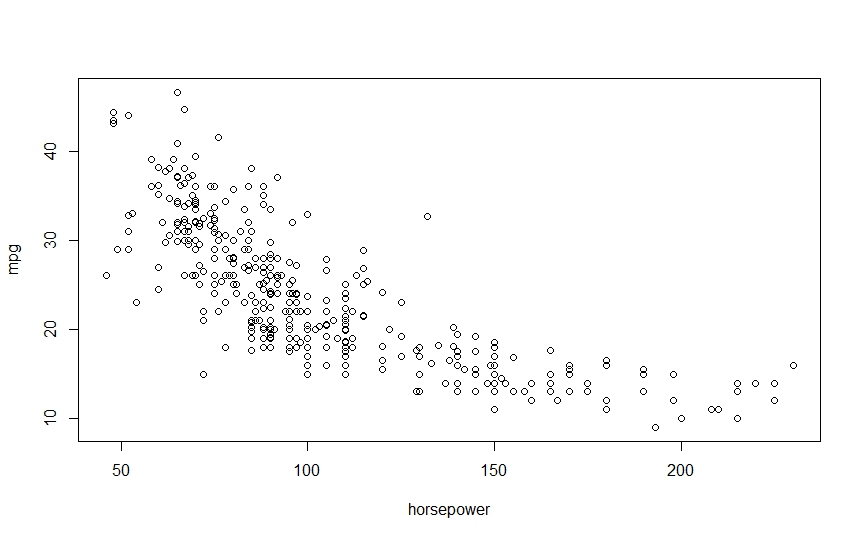
fit lwr upr

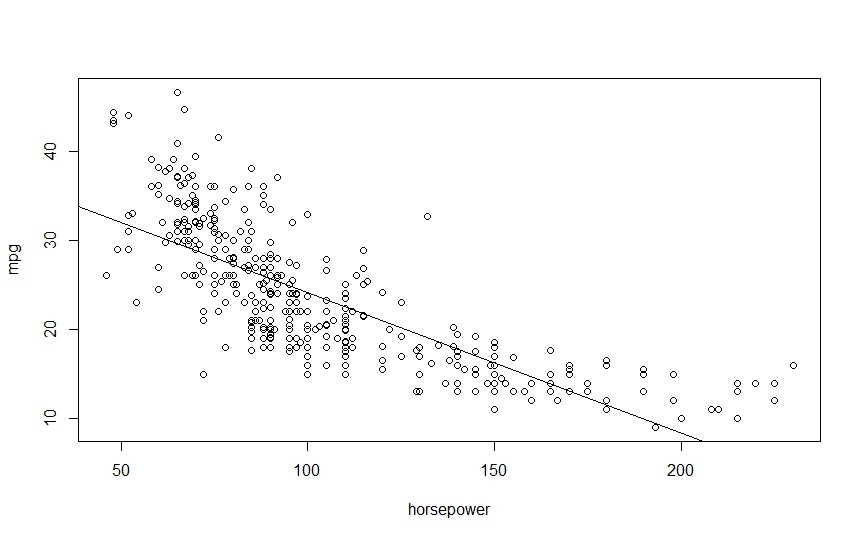
24.46708 14.8094 34.12476

#Q5(b)

plot(horsepower, mpg)

abline(lm.fit)

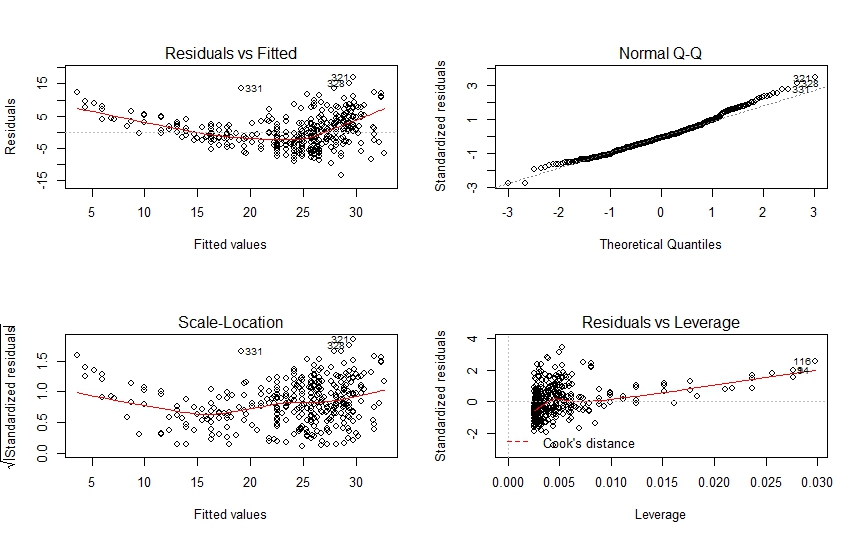




#Q5(c)

par(mfrow=c(2,2))

plot(lm.fit)



**Console:**

Auto = read.csv("Auto.csv", header=T, na.strings="?")

> Auto = na.omit(Auto)

> summary(Auto)

mpg cylinders displacement horsepower weight acceleration year

Min. : 9.00 Min. :3.000 Min. : 68.0 Min. : 46.0 Min. :1613 Min. : 8.00 Min. :70.00

1st Qu.:17.00 1st Qu.:4.000 1st Qu.:105.0 1st Qu.: 75.0 1st Qu.:2225 1st Qu.:13.78 1st Qu.:73.00

Median :22.75 Median :4.000 Median :151.0 Median : 93.5 Median :2804 Median :15.50 Median :76.00

Mean :23.45 Mean :5.472 Mean :194.4 Mean :104.5 Mean :2978 Mean :15.54 Mean :75.98

3rd Qu.:29.00 3rd Qu.:8.000 3rd Qu.:275.8 3rd Qu.:126.0 3rd Qu.:3615 3rd Qu.:17.02 3rd Qu.:79.00

Max. :46.60 Max. :8.000 Max. :455.0 Max. :230.0 Max. :5140 Max. :24.80 Max. :82.00

origin name

Min. :1.000 amc matador : 5

1st Qu.:1.000 ford pinto : 5

Median :1.000 toyota corolla : 5

Mean :1.577 amc gremlin : 4

3rd Qu.:2.000 amc hornet : 4

Max. :3.000 chevrolet chevette: 4

(Other) :365

> attach(Auto)

> lm.fit = lm(mpg ~ horsepower)

> summary(lm.fit)

Call:

lm(formula = mpg ~ horsepower)

Residuals:

Min 1Q Median 3Q Max

-13.5710 -3.2592 -0.3435 2.7630 16.9240

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 39.935861 0.717499 55.66 <2e-16 \*\*\*

horsepower -0.157845 0.006446 -24.49 <2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.906 on 390 degrees of freedom

Multiple R-squared: 0.6059, Adjusted R-squared: 0.6049

F-statistic: 599.7 on 1 and 390 DF, p-value: < 2.2e-16

> predict(lm.fit, data.frame(horsepower=c(98)), interval="confidence")

fit lwr upr

1 24.46708 23.97308 24.96108

> predict(lm.fit, data.frame(horsepower=c(98)), interval="prediction")

fit lwr upr

1 24.46708 14.8094 34.12476

> #Q5(b)

> plot(horsepower, mpg)

> abline(lm.fit)

> #Q5(c)

> par(mfrow=c(2,2))

> plot(lm.fit)

**Question 6:**

**Solution:**

**(a)**

**R code:**

#Q6(a)

library(ISLR)

summary(Carseats)

attach(Carseats)

lm.fit = lm(Sales~Price+Urban+US)

summary(lm.fit)

**Console:**

> library(ISLR)

Attaching package: ‘ISLR’

The following object is masked \_by\_ ‘.GlobalEnv’:

Auto

> summary(Carseats)

Sales CompPrice Income Advertising Population Price ShelveLoc

Min. : 0.000 Min. : 77 Min. : 21.00 Min. : 0.000 Min. : 10.0 Min. : 24.0 Bad : 96

1st Qu.: 5.390 1st Qu.:115 1st Qu.: 42.75 1st Qu.: 0.000 1st Qu.:139.0 1st Qu.:100.0 Good : 85

Median : 7.490 Median :125 Median : 69.00 Median : 5.000 Median :272.0 Median :117.0 Medium:219

Mean : 7.496 Mean :125 Mean : 68.66 Mean : 6.635 Mean :264.8 Mean :115.8

3rd Qu.: 9.320 3rd Qu.:135 3rd Qu.: 91.00 3rd Qu.:12.000 3rd Qu.:398.5 3rd Qu.:131.0

Max. :16.270 Max. :175 Max. :120.00 Max. :29.000 Max. :509.0 Max. :191.0

Age Education Urban US

Min. :25.00 Min. :10.0 No :118 No :142

1st Qu.:39.75 1st Qu.:12.0 Yes:282 Yes:258

Median :54.50 Median :14.0

Mean :53.32 Mean :13.9

3rd Qu.:66.00 3rd Qu.:16.0

Max. :80.00 Max. :18.0

> attach(Carseats)

> lm.fit = lm(Sales~Price+Urban+US)

> summary(lm.fit)

Call:

lm(formula = Sales ~ Price + Urban + US)

Residuals:

Min 1Q Median 3Q Max

-6.9206 -1.6220 -0.0564 1.5786 7.0581

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 13.043469 0.651012 20.036 < 2e-16 \*\*\*

Price -0.054459 0.005242 -10.389 < 2e-16 \*\*\*

UrbanYes -0.021916 0.271650 -0.081 0.936

USYes 1.200573 0.259042 4.635 4.86e-06 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.472 on 396 degrees of freedom

Multiple R-squared: 0.2393, Adjusted R-squared: 0.2335

F-statistic: 41.52 on 3 and 396 DF, p-value: < 2.2e-16

(b)

**Price:** When price increases, sales decreases.

**UrbanYes:** High p-value of t statistics defers that there is no relationship between urban store and sales figure

**UsYes:** If the UsYes increases, sales figure also increases

(c)

Model in equation:

Sales = 13.04 + (-0.05)\*Price + (-0.02)\*UrbanYes + (1.20)\*USYes

(d)

I can reject the null hypothesis for Price and USYes, because of their p-values and F-stat

(e)

**R code**

#Q6(e)

lm.fit1 = lm(Sales ~ Price + US)

summary(lm.fit1)

**Console:**

> lm.fit1 = lm(Sales ~ Price + US)

> summary(lm.fit1)

Call:

lm(formula = Sales ~ Price + US)

Residuals:

Min 1Q Median 3Q Max

-6.9269 -1.6286 -0.0574 1.5766 7.0515

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 13.03079 0.63098 20.652 < 2e-16 \*\*\*

Price -0.05448 0.00523 -10.416 < 2e-16 \*\*\*

USYes 1.19964 0.25846 4.641 4.71e-06 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.469 on 397 degrees of freedom

Multiple R-squared: 0.2393, Adjusted R-squared: 0.2354

F-statistic: 62.43 on 2 and 397 DF, p-value: < 2.2e-16

(f)

Both will fit the data similarly though model (e) should fit better.

(g)

**R code:**

#Q6(g)

confint(lm.fit1)

**Console:**

confint(lm.fit1)

2.5 % 97.5 %

(Intercept) 11.79032020 14.27126531

Price -0.06475984 -0.04419543

USYes 0.69151957 1.70776632

(h)

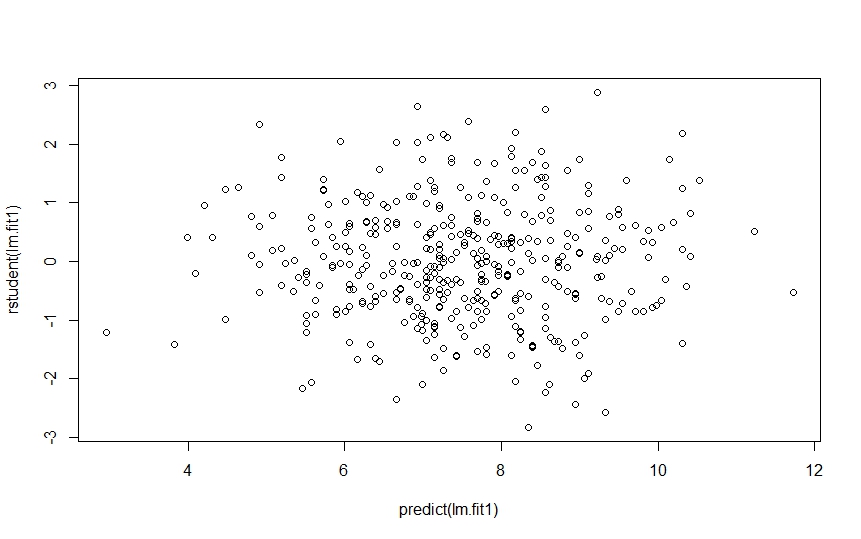
**R code:**

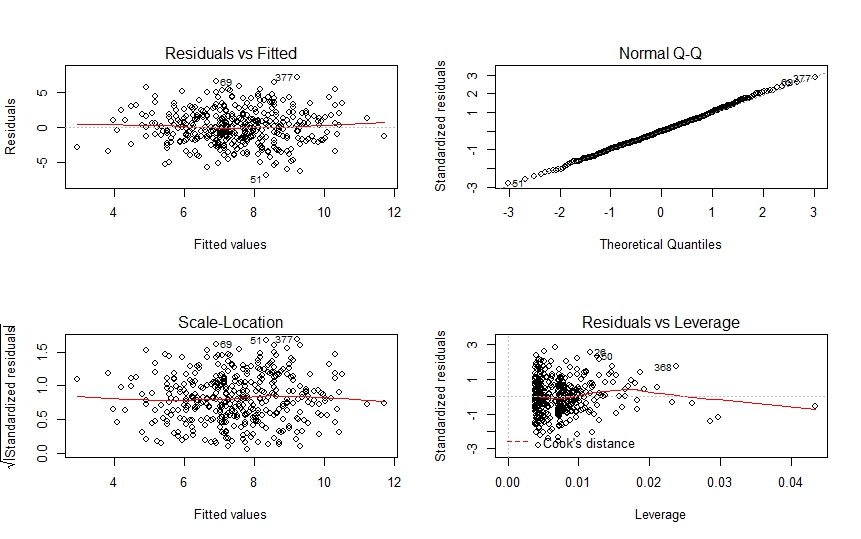
#Q6(h)

plot(predict(lm.fit1), rstudent(lm.fit1))

par(mfrow=c(2,2))

plot(lm.fit2)





From the Normal Q-Q curve, it is clear that all standardized residuals are lies between -3 to +3. So, there’s no evidence of outliers. From the leverage plot, it is also showed that there are some high values.

**Question 7:**

**Solution:**

(a)

A general form of Ridge regression optimization looks like

Minimize:



In this case,



So, the optimization looks like:

Minimize:



**(b)**

Now we are given that, x11=x12=x1 and x21=x22=x2, we take derivatives of above expression with respect to both β1^ and β2^ and setting them equal to zero find that,



Symmetry in these expressions suggests that



**(c)** Like Ridge regression,

Minimize:



**(d)** Here is a geometric interpretation of the solutions for the equation in c above. We use the alternate form of Lasso constraints 

The Lasso constraint take the form ,

which when plotted take the familiar shape of a diamond centered at origin (0,0). Next consider the squared optimization constraint



We use the facts x11=x12, x21=x22, x11+x21=0, x11 + x21=0 and y1+y2=0 to simplify it to Minimize:



This optimization problem has a simple solution:



This is a line parallel to the edge of Lasso-diamond β^1+β^2=s. Now solutions to the original Lasso optimization problem are contours of the function (y1−(β^1+β^2)x11)2 that touch the Lasso-diamond β^1+β^2=s.

Finally, as β^1 and β^2 very along the line β^1+β^2=y1x11, these contours touch the Lasso-diamond edge β^1+β^2=s at different points. As a result, the entire edge β^1+β^2=s is a potential solution to the Lasso optimization problem!

Similar argument can be made for the opposite Lasso-diamond edge: β^1+β^2=−s.

Thus, the Lasso problem does not have a unique solution. The general form of solution is given by two line segments:



**Question 8:**

**Solution:**

**R Code:**

#Q8(a)

set.seed(1)

X = rnorm(100)

eps = rnorm(100)

#Q8(b)

beta0 = 3

beta1 = 2

beta2 = -3

beta3 = 0.3

Y = beta0 + beta1 \* X + beta2 \* X^2 + beta3 \* X^3 + eps

#Q8(c)

library(leaps)

data.full = data.frame(y = Y, x = X)

mod.full = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10)

mod.summary = summary(mod.full)

# Find the model size for best cp, BIC and adjr2

which.min(mod.summary$cp)

which.min(mod.summary$bic)

which.max(mod.summary$adjr2)

# Plot cp, BIC and adjr2

plot(mod.summary$cp, xlab = "Subset Size", ylab = "Cp", pch = 20, type = "l")

points(3, mod.summary$cp[3], pch = 4, col = "red", lwd = 7)

plot(mod.summary$bic, xlab = "Subset Size", ylab = "BIC", pch = 20, type = "l")

points(3, mod.summary$bic[3], pch = 4, col = "red", lwd = 7)

plot(mod.summary$adjr2, xlab = "Subset Size", ylab = "Adjusted R2", pch = 20, type = "l")

points(3, mod.summary$adjr2[3], pch = 4, col = "red", lwd = 7)

coefficients(mod.full, id = 3)

#Q8(d)

mod.fwd = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10, method = "forward")

mod.bwd = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10, method = "backward")

fwd.summary = summary(mod.fwd)

bwd.summary = summary(mod.bwd)

which.min(fwd.summary$cp)

which.min(bwd.summary$cp)

which.min(fwd.summary$bic)

which.min(bwd.summary$bic)

which.max(fwd.summary$adjr2)

which.max(bwd.summary$adjr2)

par(mfrow = c(3, 2))

plot(fwd.summary$cp, xlab = "Subset Size", ylab = "Forward Cp", pch = 20, type = "l")

points(3, fwd.summary$cp[3], pch = 4, col = "red", lwd = 7)

plot(bwd.summary$cp, xlab = "Subset Size", ylab = "Backward Cp", pch = 20, type = "l")

points(3, bwd.summary$cp[3], pch = 4, col = "red", lwd = 7)

plot(fwd.summary$bic, xlab = "Subset Size", ylab = "Forward BIC", pch = 20, type = "l")

points(3, fwd.summary$bic[3], pch = 4, col = "red", lwd = 7)

plot(bwd.summary$bic, xlab = "Subset Size", ylab = "Backward BIC", pch = 20, type = "l")

points(3, bwd.summary$bic[3], pch = 4, col = "red", lwd = 7)

plot(fwd.summary$adjr2, xlab = "Subset Size", ylab = "Forward Adjusted R2", pch = 20, type = "l")

points(3, fwd.summary$adjr2[3], pch = 4, col = "red", lwd = 7)

plot(bwd.summary$adjr2, xlab = "Subset Size", ylab = "Backward Adjusted R2", pch = 20, type = "l")

points(4, bwd.summary$adjr2[4], pch = 4, col = "red", lwd = 7)

coefficients(mod.fwd, id = 3)

coefficients(mod.bwd, id = 3)

coefficients(mod.fwd, id = 4)

#Q8(e)

library(foreach)

library(Matrix)

library(glmnet)

xmat = model.matrix(y ~ poly(x, 10, raw = T), data = data.full)[, -1]

mod.lasso = cv.glmnet(xmat, Y, alpha = 1)

best.lambda = mod.lasso$lambda.min

best.lambda

plot(mod.lasso)

best.model = glmnet(xmat, Y, alpha = 1)

predict(best.model, s = best.lambda, type = "coefficients")

#Q8(f)

beta7 = 7

Y = beta0 + beta7 \* X^7 + eps

data.full = data.frame(y = Y, x = X)

mod.full = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10)

mod.summary = summary(mod.full)

which.min(mod.summary$cp)

which.min(mod.summary$bic)

which.max(mod.summary$adjr2)

coefficients(mod.full, id = 1)

coefficients(mod.full, id = 2)

coefficients(mod.full, id = 4)

xmat = model.matrix(y ~ poly(x, 10, raw = T), data = data.full)[, -1]

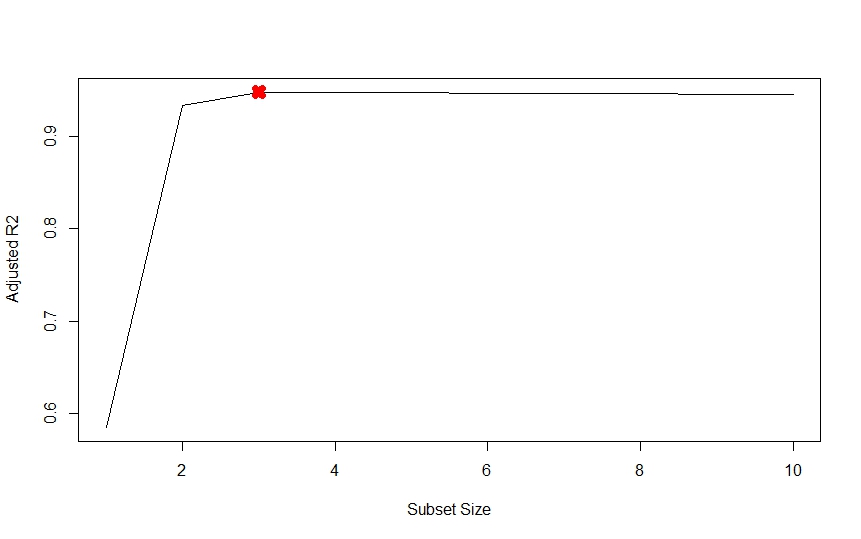
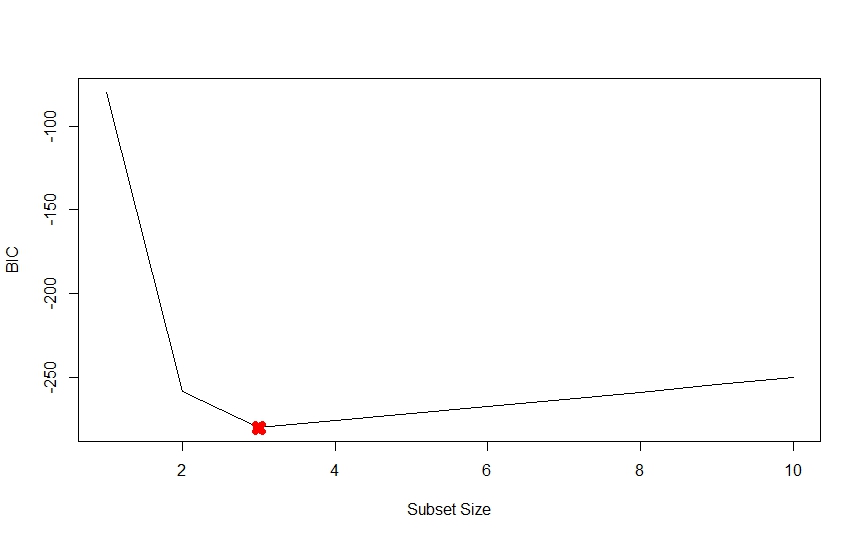
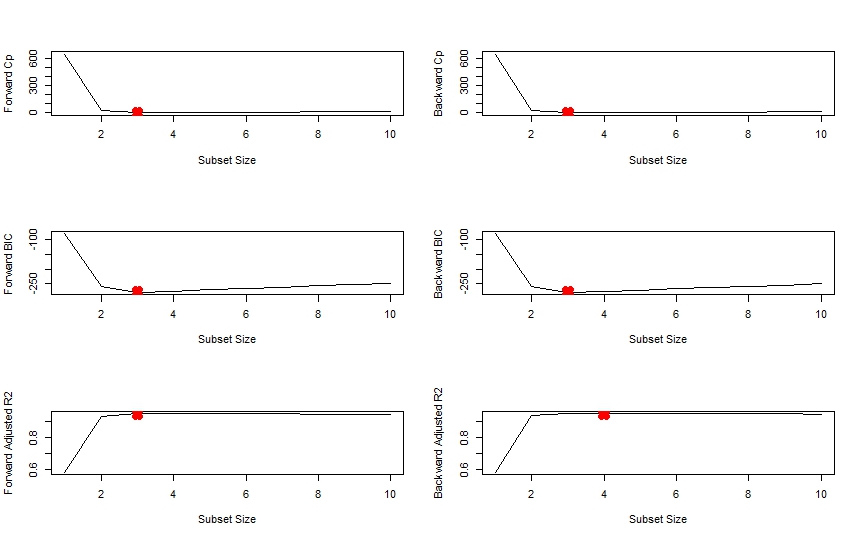
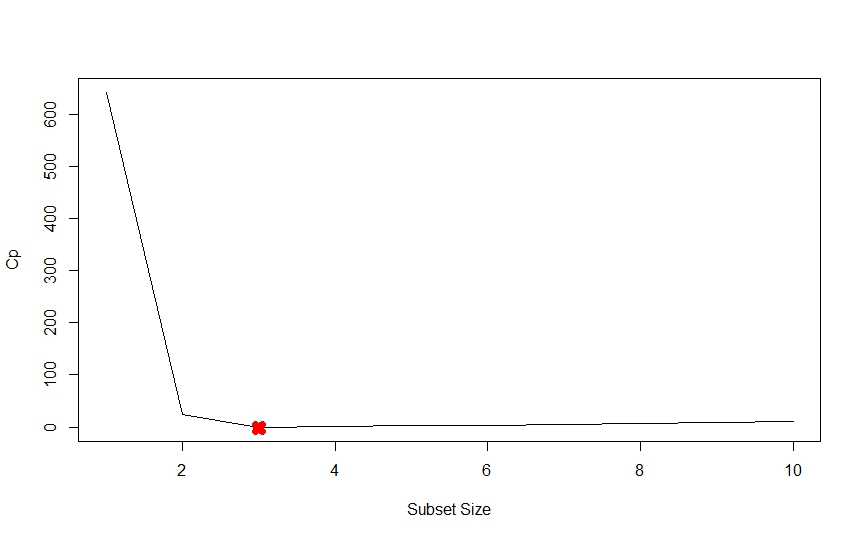
mod.lasso = cv.glmnet(xmat, Y, alpha = 1)

best.lambda = mod.lasso$lambda.min

best.lambda

best.model = glmnet(xmat, Y, alpha = 1)

predict(best.model, s = best.lambda, type = "coefficients")



**Console:**

> set.seed(1)

> X = rnorm(100)

> eps = rnorm(100)

> beta0 = 3

> beta1 = 2

> beta2 = -3

> beta3 = 0.3

> Y = beta0 + beta1 \* X + beta2 \* X^2 + beta3 \* X^3 + eps

> library(leaps)

> data.full = data.frame(y = Y, x = X)

> mod.full = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10)

> mod.summary = summary(mod.full)

> which.min(mod.summary$cp)

[1] 3

> which.min(mod.summary$bic)

[1] 3

> which.max(mod.summary$adjr2)

[1] 3

> plot(mod.summary$cp, xlab = "Subset Size", ylab = "Cp", pch = 20, type = "l")

> points(3, mod.summary$cp[3], pch = 4, col = "red", lwd = 7)

> plot(mod.summary$bic, xlab = "Subset Size", ylab = "BIC", pch = 20, type = "l")

> points(3, mod.summary$bic[3], pch = 4, col = "red", lwd = 7)

> plot(mod.summary$adjr2, xlab = "Subset Size", ylab = "Adjusted R2", pch = 20, type = "l")

> points(3, mod.summary$adjr2[3], pch = 4, col = "red", lwd = 7)

> coefficients(mod.full, id = 3)

(Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2 poly(x, 10, raw = T)7

3.07627412 2.35623596 -3.16514887 0.01046843

> mod.fwd = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10, method = "forward")

> mod.bwd = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10, method = "backward")

> fwd.summary = summary(mod.fwd)

> bwd.summary = summary(mod.bwd)

> which.min(fwd.summary$cp)

[1] 3

> which.min(bwd.summary$cp)

[1] 3

> which.min(fwd.summary$bic)

[1] 3

> which.min(bwd.summary$bic)

[1] 3

> which.max(fwd.summary$adjr2)

[1] 3

> which.max(bwd.summary$adjr2)

[1] 4

> par(mfrow = c(3, 2))

> plot(fwd.summary$cp, xlab = "Subset Size", ylab = "Forward Cp", pch = 20, type = "l")

> points(3, fwd.summary$cp[3], pch = 4, col = "red", lwd = 7)

> plot(bwd.summary$cp, xlab = "Subset Size", ylab = "Backward Cp", pch = 20, type = "l")

> points(3, bwd.summary$cp[3], pch = 4, col = "red", lwd = 7)

> plot(fwd.summary$bic, xlab = "Subset Size", ylab = "Forward BIC", pch = 20, type = "l")

> points(3, fwd.summary$bic[3], pch = 4, col = "red", lwd = 7)

> plot(bwd.summary$bic, xlab = "Subset Size", ylab = "Backward BIC", pch = 20, type = "l")

> points(3, bwd.summary$bic[3], pch = 4, col = "red", lwd = 7)

> plot(fwd.summary$adjr2, xlab = "Subset Size", ylab = "Forward Adjusted R2", pch = 20, type = "l")

> points(3, fwd.summary$adjr2[3], pch = 4, col = "red", lwd = 7)

> plot(bwd.summary$adjr2, xlab = "Subset Size", ylab = "Backward Adjusted R2", pch = 20, type = "l")

> points(4, bwd.summary$adjr2[4], pch = 4, col = "red", lwd = 7)

> coefficients(mod.fwd, id = 3)

(Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2 poly(x, 10, raw = T)7

3.07627412 2.35623596 -3.16514887 0.01046843

> coefficients(mod.bwd, id = 3)

(Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2 poly(x, 10, raw = T)9

3.078881355 2.419817953 -3.177235617 0.001870457

> coefficients(mod.fwd, id = 4)

(Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2 poly(x, 10, raw = T)4 poly(x, 10, raw = T)7

3.112358625 2.369858879 -3.275726574 0.027673638 0.009997134

> library(foreach)

> library(Matrix)

> library(glmnet)

> xmat = model.matrix(y ~ poly(x, 10, raw = T), data = data.full)[, -1]

> mod.lasso = cv.glmnet(xmat, Y, alpha = 1)

> best.lambda = mod.lasso$lambda.min

> best.lambda

[1] 0.03991416

> plot(mod.lasso)

> best.model = glmnet(xmat, Y, alpha = 1)

> predict(best.model, s = best.lambda, type = "coefficients")

11 x 1 sparse Matrix of class "dgCMatrix"

1

(Intercept) 3.0398151056

poly(x, 10, raw = T)1 2.2303371338

poly(x, 10, raw = T)2 -3.1033192679

poly(x, 10, raw = T)3 .

poly(x, 10, raw = T)4 .

poly(x, 10, raw = T)5 0.0498410763

poly(x, 10, raw = T)6 .

poly(x, 10, raw = T)7 0.0008068431

poly(x, 10, raw = T)8 .

poly(x, 10, raw = T)9 .

poly(x, 10, raw = T)10 .

> beta7 = 7

> Y = beta0 + beta7 \* X^7 + eps

> data.full = data.frame(y = Y, x = X)

> mod.full = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10)

> mod.summary = summary(mod.full)

> which.min(mod.summary$cp)

[1] 2

> which.min(mod.summary$bic)

[1] 1

> which.max(mod.summary$adjr2)

[1] 4

> coefficients(mod.full, id = 1)

(Intercept) poly(x, 10, raw = T)7

2.95894 7.00077

> coefficients(mod.full, id = 2)

(Intercept) poly(x, 10, raw = T)2 poly(x, 10, raw = T)7

3.0704904 -0.1417084 7.0015552

> xmat = model.matrix(y ~ poly(x, 10, raw = T), data = data.full)[, -1]

> mod.lasso = cv.glmnet(xmat, Y, alpha = 1)

> best.lambda = mod.lasso$lambda.min

> best.lambda

[1] 13.57478

> best.model = glmnet(xmat, Y, alpha = 1)

> predict(best.model, s = best.lambda, type = "coefficients")

11 x 1 sparse Matrix of class "dgCMatrix"

1

(Intercept) 3.904188

poly(x, 10, raw = T)1 .

poly(x, 10, raw = T)2 .

poly(x, 10, raw = T)3 .

poly(x, 10, raw = T)4 .

poly(x, 10, raw = T)5 .

poly(x, 10, raw = T)6 .

poly(x, 10, raw = T)7 6.776797

poly(x, 10, raw = T)8 .

poly(x, 10, raw = T)9 .

poly(x, 10, raw = T)10 .

**Question 9:**

**Solution:**

**R code:**

#Q9(a)

library(ISLR)

set.seed(11)

sum(is.na(College))

train.size = dim(College)[1] / 2

train = sample(1:dim(College)[1], train.size)

test = -train

College.train = College[train, ]

College.test = College[test, ]

#Q9(b)

lm.fit = lm(Apps~., data=College.train)

lm.pred = predict(lm.fit, College.test)

mean((College.test[, "Apps"] - lm.pred)^2)

#Q9(c)

library(glmnet)

train.mat = model.matrix(Apps~., data=College.train)

test.mat = model.matrix(Apps~., data=College.test)

grid = 10 ^ seq(4, -2, length=100)

mod.ridge = cv.glmnet(train.mat, College.train[, "Apps"], alpha=0, lambda=grid, thresh=1e-12)

lambda.best = mod.ridge$lambda.min

lambda.best

ridge.pred = predict(mod.ridge, newx=test.mat, s=lambda.best)

mean((College.test[, "Apps"] - ridge.pred)^2)

#Q9(d)

mod.lasso = cv.glmnet(train.mat, College.train[, "Apps"], alpha=1, lambda=grid, thresh=1e-12)

lambda.best = mod.lasso$lambda.min

lambda.best

lasso.pred = predict(mod.lasso, newx=test.mat, s=lambda.best)

mean((College.test[, "Apps"] - lasso.pred)^2)

mod.lasso = glmnet(model.matrix(Apps~., data=College), College[, "Apps"], alpha=1)

predict(mod.lasso, s=lambda.best, type="coefficients")

#Q9(e)

library(pls)

pcr.fit = pcr(Apps~., data=College.train, scale=T, validation="CV")

validationplot(pcr.fit, val.type="MSEP")

pcr.pred = predict(pcr.fit, College.test, ncomp=10)

mean((College.test[, "Apps"] - data.frame(pcr.pred))^2)

#Q9(f)

pls.fit = plsr(Apps~., data=College.train, scale=T, validation="CV")

validationplot(pls.fit, val.type="MSEP")

pls.pred = predict(pls.fit, College.test, ncomp=10)

mean((College.test[, "Apps"] - data.frame(pls.pred))^2)

#Q9(g)

test.avg = mean(College.test[, "Apps"])

lm.test.r2 = 1 - mean((College.test[, "Apps"] - lm.pred)^2) /mean((College.test[, "Apps"] - test.avg)^2)

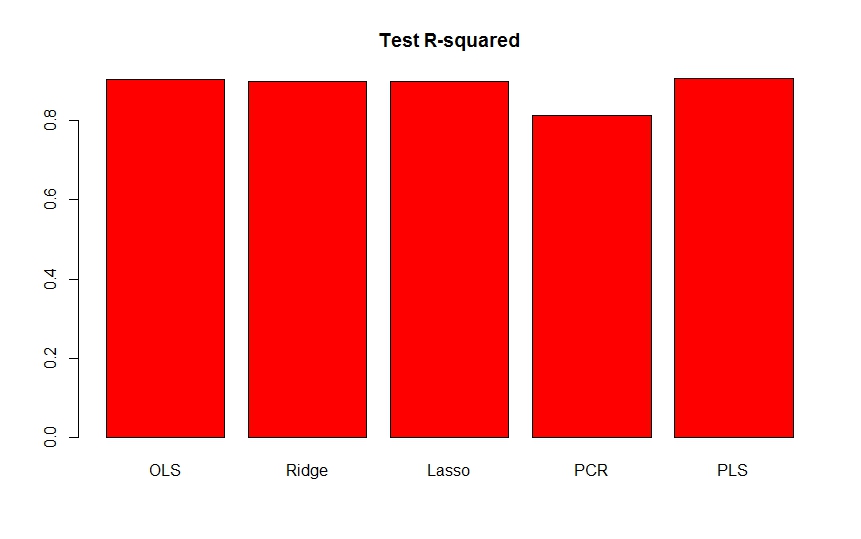
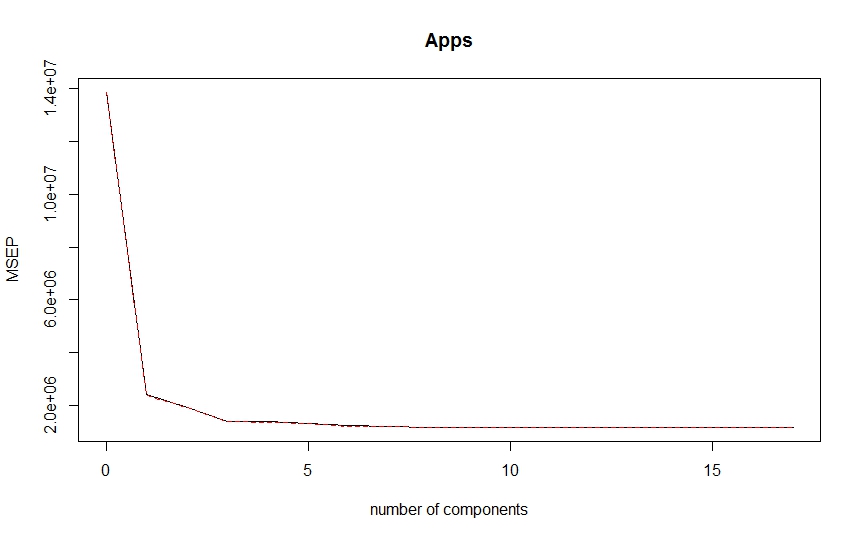
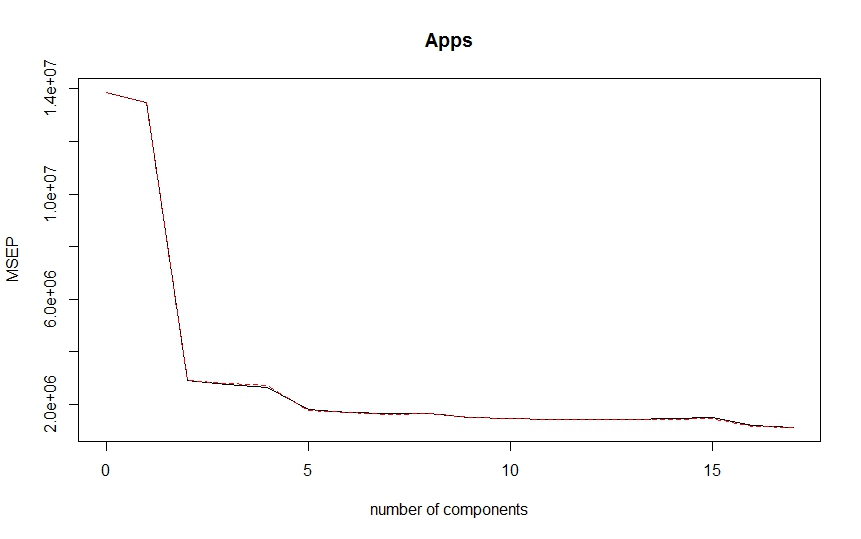
ridge.test.r2 = 1 - mean((College.test[, "Apps"] - ridge.pred)^2) /mean((College.test[, "Apps"] - test.avg)^2)

lasso.test.r2 = 1 - mean((College.test[, "Apps"] - lasso.pred)^2) /mean((College.test[, "Apps"] - test.avg)^2)

pcr.test.r2 = 1 - mean((College.test[, "Apps"] - data.frame(pcr.pred))^2) /mean((College.test[, "Apps"] - test.avg)^2)

pls.test.r2 = 1 - mean((College.test[, "Apps"] - data.frame(pls.pred))^2) /mean((College.test[, "Apps"] - test.avg)^2)

barplot(c(lm.test.r2, ridge.test.r2, lasso.test.r2, pcr.test.r2, pls.test.r2), col="red", names.arg=c("OLS", "Ridge", "Lasso", "PCR", "PLS"), main="Test R-squared")



**Console:**

> library(ISLR)

> set.seed(11)

> sum(is.na(College))

[1] 0

> train.size = dim(College)[1] / 2

> train = sample(1:dim(College)[1], train.size)

> test = -train

> College.train = College[train, ]

> College.test = College[test, ]

> View(College.test)

> View(College.test)

> lm.fit = lm(Apps~., data=College.train)

> lm.pred = predict(lm.fit, College.test)

> mean((College.test[, "Apps"] - lm.pred)^2)

[1] 1538442

> library(glmnet)

> train.mat = model.matrix(Apps~., data=College.train)

> train.mat = model.matrix(Apps~., data=College.train)

> test.mat = model.matrix(Apps~., data=College.test)

> grid = 10 ^ seq(4, -2, length=100)

> mod.ridge = cv.glmnet(train.mat, College.train[, "Apps"], alpha=0, lambda=grid, thresh=1e-12)

> lambda.best = mod.ridge$lambda.min

> lambda.best

[1] 24.77076

> ridge.pred = predict(mod.ridge, newx=test.mat, s=lambda.best)

> mean((College.test[, "Apps"] - ridge.pred)^2)

[1] 1632288

> mod.lasso = cv.glmnet(train.mat, College.train[, "Apps"], alpha=1, lambda=grid, thresh=1e-12)

> lambda.best = mod.lasso$lambda.min

> lambda.best

[1] 24.77076

> lasso.pred = predict(mod.lasso, newx=test.mat, s=lambda.best)

> mean((College.test[, "Apps"] - lasso.pred)^2)

[1] 1639664

> mod.lasso = glmnet(model.matrix(Apps~., data=College), College[, "Apps"], alpha=1)

> predict(mod.lasso, s=lambda.best, type="coefficients")

19 x 1 sparse Matrix of class "dgCMatrix"

1

(Intercept) -635.23797264

(Intercept) .

PrivateYes -408.11931578

Accept 1.43592777

Enroll -0.13872799

Top10perc 31.34261852

Top25perc -0.79389706

F.Undergrad .

P.Undergrad 0.01482400

Outstate -0.05320031

Room.Board 0.12039828

Books .

Personal .

PhD -5.12091278

Terminal -3.36566878

S.F.Ratio 2.73523904

perc.alumni -1.05561449

Expend 0.06836020

Grad.Rate 4.69112249

> library(pls)

Attaching package: ‘pls’

The following object is masked from ‘package:stats’:

loadings

> pcr.fit = pcr(Apps~., data=College.train, scale=T, validation="CV")

> validationplot(pcr.fit, val.type="MSEP")

> pcr.pred = predict(pcr.fit, College.test, ncomp=10)

> mean((College.test[, "Apps"] - data.frame(pcr.pred))^2)

[1] 3014496

> pls.fit = plsr(Apps~., data=College.train, scale=T, validation="CV")

> validationplot(pls.fit, val.type="MSEP")

> pls.pred = predict(pls.fit, College.test, ncomp=10)

> mean((College.test[, "Apps"] - data.frame(pls.pred))^2)

[1] 1508987

> test.avg = mean(College.test[, "Apps"])

> lm.test.r2 = 1 - mean((College.test[, "Apps"] - lm.pred)^2) /mean((College.test[, "Apps"] - test.avg)^2)

> ridge.test.r2 = 1 - mean((College.test[, "Apps"] - ridge.pred)^2) /mean((College.test[, "Apps"] - test.avg)^2)

> lasso.test.r2 = 1 - mean((College.test[, "Apps"] - lasso.pred)^2) /mean((College.test[, "Apps"] - test.avg)^2)

> pcr.test.r2 = 1 - mean((College.test[, "Apps"] - data.frame(pcr.pred))^2) /mean((College.test[, "Apps"] - test.avg)^2)

> pls.test.r2 = 1 - mean((College.test[, "Apps"] - data.frame(pls.pred))^2) /mean((College.test[, "Apps"] - test.avg)^2)

> barplot(c(lm.test.r2, ridge.test.r2, lasso.test.r2, pcr.test.r2, pls.test.r2), col="red", names.arg=c("OLS", "Ridge", "Lasso", "PCR", "PLS"), main="Test R-squared")

**Question 10:**

**Solution:**

X1=hours studied,

X2=undergrad GPA

β0=−6, β1=0.05, β2=1

1. Probability that a student who studies for 40 h and has an undergrad GPA of 3*.*5 gets an A in the class

**= 37.75%**

1. hours would the student in part (a) need to study to have a 50% chance of getting an A in the class

**Question 11:**

**Solution:**

**R code:**

#Q11(a)

library(ISLR)

summary(Weekly)

pairs(Weekly)

cor(Weekly[, -9])

#Q11(b)

attach(Weekly)

glm.fit = glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly, family = binomial)

summary(glm.fit)

#Q11(c)

glm.probs = predict(glm.fit, type = "response")

glm.pred = rep("Down", length(glm.probs))

glm.pred[glm.probs > 0.5] = "Up"

table(glm.pred, Direction)

#Q11(d)

train = (Year < 2009)

Weekly.0910 = Weekly[!train, ]

glm.fit = glm(Direction ~ Lag2, data = Weekly, family = binomial, subset = train)

glm.probs = predict(glm.fit, Weekly.0910, type = "response")

glm.pred = rep("Down", length(glm.probs))

glm.pred[glm.probs > 0.5] = "Up"

Direction.0910 = Direction[!train]

table(glm.pred, Direction.0910)

#Q11(e)

library(MASS)

lda.fit = lda(Direction ~ Lag2, data = Weekly, subset = train)

lda.pred = predict(lda.fit, Weekly.0910)

table(lda.pred$class, Direction.0910)

mean(lda.pred$class == Direction.0910)

#Q11(f)

qda.fit = qda(Direction ~ Lag2, data = Weekly, subset = train)

qda.class = predict(qda.fit, Weekly.0910)$class

table(qda.class, Direction.0910)

mean(qda.class == Direction.0910)

#Q11(g)

library(class)

train.X = as.matrix(Lag2[train])

test.X = as.matrix(Lag2[!train])

train.Direction = Direction[train]

set.seed(1)

knn.pred = knn(train.X, test.X, train.Direction, k = 1)

table(knn.pred, Direction.0910)

mean(knn.pred == Direction.0910)

#Q11(h)

# Logistic regression with Lag2:Lag1

glm.fit = glm(Direction ~ Lag2:Lag1, data = Weekly, family = binomial, subset = train)

glm.probs = predict(glm.fit, Weekly.0910, type = "response")

glm.pred = rep("Down", length(glm.probs))

glm.pred[glm.probs > 0.5] = "Up"

Direction.0910 = Direction[!train]

table(glm.pred, Direction.0910)

mean(glm.pred == Direction.0910)

# LDA with Lag2 interaction with Lag1

lda.fit = lda(Direction ~ Lag2:Lag1, data = Weekly, subset = train)

lda.pred = predict(lda.fit, Weekly.0910)

mean(lda.pred$class == Direction.0910)

# QDA with sqrt(abs(Lag2))

qda.fit = qda(Direction ~ Lag2 + sqrt(abs(Lag2)), data = Weekly, subset = train)

qda.class = predict(qda.fit, Weekly.0910)$class

table(qda.class, Direction.0910)

mean(qda.class == Direction.0910)

# KNN k =10

knn.pred = knn(train.X, test.X, train.Direction, k = 10)

table(knn.pred, Direction.0910)

mean(knn.pred == Direction.0910)

# KNN k = 100

knn.pred = knn(train.X, test.X, train.Direction, k = 100)

table(knn.pred, Direction.0910)

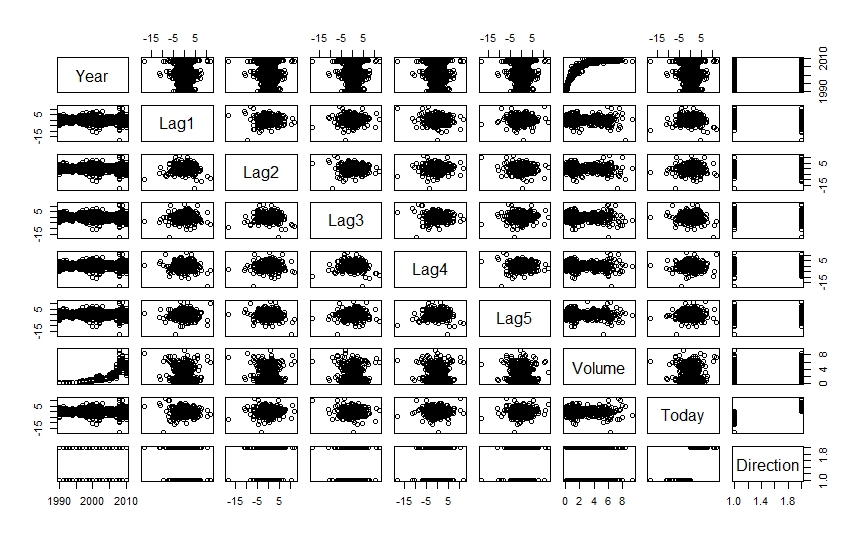
mean(knn.pred == Direction.0910)

1. Year and Volume appear to have a relationship. No other patterns are discernible.
2. Lag 2 appears to have some statistical significance with a Pr(>|z|) = 3%.
3. Percentage of correct predictions: (54+557)/(54+557+48+430) = 56.1%. Weeks the market goes up the logistic regression is right most of the time, 557/(557+48) = 92.1%. Weeks the market goes up the logistic regression is wrong most of the time 54/(430+54) = 11.2%.

(f) A correctness of 58.7% even though it picked Up the whole time.

(g) Logistic regression and LDA methods provide similar test error rates.

(h) Out of these permutations, the original LDA and logistic regression have better performance in terms of test error rate.



**Console:**

> library(ISLR)

> summary(Weekly)

Year Lag1 Lag2 Lag3 Lag4 Lag5

Min. :1990 Min. :-18.1950 Min. :-18.1950 Min. :-18.1950 Min. :-18.1950 Min. :-18.1950

1st Qu.:1995 1st Qu.: -1.1540 1st Qu.: -1.1540 1st Qu.: -1.1580 1st Qu.: -1.1580 1st Qu.: -1.1660

Median :2000 Median : 0.2410 Median : 0.2410 Median : 0.2410 Median : 0.2380 Median : 0.2340

Mean :2000 Mean : 0.1506 Mean : 0.1511 Mean : 0.1472 Mean : 0.1458 Mean : 0.1399

3rd Qu.:2005 3rd Qu.: 1.4050 3rd Qu.: 1.4090 3rd Qu.: 1.4090 3rd Qu.: 1.4090 3rd Qu.: 1.4050

Max. :2010 Max. : 12.0260 Max. : 12.0260 Max. : 12.0260 Max. : 12.0260 Max. : 12.0260

Volume Today Direction

Min. :0.08747 Min. :-18.1950 Down:484

1st Qu.:0.33202 1st Qu.: -1.1540 Up :605

Median :1.00268 Median : 0.2410

Mean :1.57462 Mean : 0.1499

3rd Qu.:2.05373 3rd Qu.: 1.4050

Max. :9.32821 Max. : 12.0260

> pairs(Weekly)

> cor(Weekly[, -9])

Year Lag1 Lag2 Lag3 Lag4 Lag5 Volume Today

Year 1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923 -0.030519101 0.84194162 -0.032459894

Lag1 -0.03228927 1.000000000 -0.07485305 0.05863568 -0.071273876 -0.008183096 -0.06495131 -0.075031842

Lag2 -0.03339001 -0.074853051 1.00000000 -0.07572091 0.058381535 -0.072499482 -0.08551314 0.059166717

Lag3 -0.03000649 0.058635682 -0.07572091 1.00000000 -0.075395865 0.060657175 -0.06928771 -0.071243639

Lag4 -0.03112792 -0.071273876 0.05838153 -0.07539587 1.000000000 -0.075675027 -0.06107462 -0.007825873

Lag5 -0.03051910 -0.008183096 -0.07249948 0.06065717 -0.075675027 1.000000000 -0.05851741 0.011012698

Volume 0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617 -0.058517414 1.00000000 -0.033077783

Today -0.03245989 -0.075031842 0.05916672 -0.07124364 -0.007825873 0.011012698 -0.03307778 1.000000000

> attach(Weekly)

> glm.fit = glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly, family = binomial)

> summary(glm.fit)

Call:

glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +

Volume, family = binomial, data = Weekly)

Deviance Residuals:

Min 1Q Median 3Q Max

-1.6949 -1.2565 0.9913 1.0849 1.4579

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 0.26686 0.08593 3.106 0.0019 \*\*

Lag1 -0.04127 0.02641 -1.563 0.1181

Lag2 0.05844 0.02686 2.175 0.0296 \*

Lag3 -0.01606 0.02666 -0.602 0.5469

Lag4 -0.02779 0.02646 -1.050 0.2937

Lag5 -0.01447 0.02638 -0.549 0.5833

Volume -0.02274 0.03690 -0.616 0.5377

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1496.2 on 1088 degrees of freedom

Residual deviance: 1486.4 on 1082 degrees of freedom

AIC: 1500.4

Number of Fisher Scoring iterations: 4

> glm.probs = predict(glm.fit, type = "response")

> glm.pred = rep("Down", length(glm.probs))

> glm.pred[glm.probs > 0.5] = "Up"

> table(glm.pred, Direction)

Direction

glm.pred Down Up

Down 54 48

Up 430 557

> train = (Year < 2009)

> Weekly.0910 = Weekly[!train, ]

> glm.fit = glm(Direction ~ Lag2, data = Weekly, family = binomial, subset = train)

> glm.probs = predict(glm.fit, Weekly.0910, type = "response")

> glm.pred = rep("Down", length(glm.probs))

> glm.pred[glm.probs > 0.5] = "Up"

> Direction.0910 = Direction[!train]

> table(glm.pred, Direction.0910)

Direction.0910

glm.pred Down Up

Down 9 5

Up 34 56

> library(MASS)

> lda.fit = lda(Direction ~ Lag2, data = Weekly, subset = train)

> lda.pred = predict(lda.fit, Weekly.0910)

> table(lda.pred$class, Direction.0910)

Direction.0910

Down Up

Down 9 5

Up 34 56

> mean(lda.pred$class == Direction.0910)

[1] 0.625

> qda.fit = qda(Direction ~ Lag2, data = Weekly, subset = train)

> qda.class = predict(qda.fit, Weekly.0910)$class

> table(qda.class, Direction.0910)

Direction.0910

qda.class Down Up

Down 0 0

Up 43 61

> mean(qda.class == Direction.0910)

[1] 0.5865385

> library(class)

> train.X = as.matrix(Lag2[train])

> test.X = as.matrix(Lag2[!train])

> train.Direction = Direction[train]

> set.seed(1)

> knn.pred = knn(train.X, test.X, train.Direction, k = 1)

> table(knn.pred, Direction.0910)

Direction.0910

knn.pred Down Up

Down 21 30

Up 22 31

> mean(knn.pred == Direction.0910)

[1] 0.5

> glm.fit = glm(Direction ~ Lag2:Lag1, data = Weekly, family = binomial, subset = train)

> glm.probs = predict(glm.fit, Weekly.0910, type = "response")

> glm.pred = rep("Down", length(glm.probs))

> glm.pred[glm.probs > 0.5] = "Up"

> Direction.0910 = Direction[!train]

> table(glm.pred, Direction.0910)

Direction.0910

glm.pred Down Up

Down 1 1

Up 42 60

> mean(glm.pred == Direction.0910)

[1] 0.5865385

> lda.fit = lda(Direction ~ Lag2:Lag1, data = Weekly, subset = train)

> lda.pred = predict(lda.fit, Weekly.0910)

> mean(lda.pred$class == Direction.0910)

[1] 0.5769231

> qda.fit = qda(Direction ~ Lag2 + sqrt(abs(Lag2)), data = Weekly, subset = train)

> qda.class = predict(qda.fit, Weekly.0910)$class

> table(qda.class, Direction.0910)

Direction.0910

qda.class Down Up

Down 12 13

Up 31 48

> mean(qda.class == Direction.0910)

[1] 0.5769231

> knn.pred = knn(train.X, test.X, train.Direction, k = 10)

> table(knn.pred, Direction.0910)

Direction.0910

knn.pred Down Up

Down 17 18

Up 26 43

> mean(knn.pred == Direction.0910)

[1] 0.5769231

> knn.pred = knn(train.X, test.X, train.Direction, k = 100)

> table(knn.pred, Direction.0910)

Direction.0910

knn.pred Down Up

Down 9 12

Up 34 49

> mean(knn.pred == Direction.0910)

[1] 0.5576923

**Question 12:**

**Solution:**

**R code:**

#Q12(a)

library(ISLR)

summary(Auto)

attach(Auto)

mpg01 = rep(0, length(mpg))

mpg01[mpg > median(mpg)] = 1

Auto = data.frame(Auto, mpg01)

#Q12(b)

cor(Auto[, -9])

pairs(Auto)

#Q12(c)

train = (year%%2 == 0)

test = !train

Auto.train = Auto[train, ]

Auto.test = Auto[test, ]

mpg01.test = mpg01[test]

#Q12(d)

library(MASS)

lda.fit = lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, subset = train)

lda.pred = predict(lda.fit, Auto.test)

mean(lda.pred$class != mpg01.test)

#Q12(e)

qda.fit = qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, subset = train)

qda.pred = predict(qda.fit, Auto.test)

mean(qda.pred$class != mpg01.test)

#Q12(f)

glm.fit = glm(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, family = binomial, subset = train)

glm.probs = predict(glm.fit, Auto.test, type = "response")

glm.pred = rep(0, length(glm.probs))

glm.pred[glm.probs > 0.5] = 1

mean(glm.pred != mpg01.test)

#Q12(g)

library(class)

train.X = cbind(cylinders, weight, displacement, horsepower)[train, ]

test.X = cbind(cylinders, weight, displacement, horsepower)[test, ]

train.mpg01 = mpg01[train]

set.seed(1)

knn.pred = knn(train.X, test.X, train.mpg01, k = 1) # KNN(k=1)

mean(knn.pred != mpg01.test)

knn.pred = knn(train.X, test.X, train.mpg01, k = 10) # KNN(k=10)

mean(knn.pred != mpg01.test)

knn.pred = knn(train.X, test.X, train.mpg01, k = 100) # KNN(k=100)

mean(knn.pred != mpg01.test)

**Console:**

library(ISLR)

> summary(Auto)

mpg cylinders displacement horsepower weight acceleration year

Min. : 9.00 Min. :3.000 Min. : 68.0 Min. : 46.0 Min. :1613 Min. : 8.00 Min. :70.00

1st Qu.:17.00 1st Qu.:4.000 1st Qu.:105.0 1st Qu.: 75.0 1st Qu.:2225 1st Qu.:13.78 1st Qu.:73.00

Median :22.75 Median :4.000 Median :151.0 Median : 93.5 Median :2804 Median :15.50 Median :76.00

Mean :23.45 Mean :5.472 Mean :194.4 Mean :104.5 Mean :2978 Mean :15.54 Mean :75.98

3rd Qu.:29.00 3rd Qu.:8.000 3rd Qu.:275.8 3rd Qu.:126.0 3rd Qu.:3615 3rd Qu.:17.02 3rd Qu.:79.00

Max. :46.60 Max. :8.000 Max. :455.0 Max. :230.0 Max. :5140 Max. :24.80 Max. :82.00

origin name

Min. :1.000 amc matador : 5

1st Qu.:1.000 ford pinto : 5

Median :1.000 toyota corolla : 5

Mean :1.577 amc gremlin : 4

3rd Qu.:2.000 amc hornet : 4

Max. :3.000 chevrolet chevette: 4

(Other) :365

> attach(Auto)

The following objects are masked from Auto (pos = 3):

acceleration, cylinders, displacement, horsepower, mpg, name, origin, weight, year

The following objects are masked from Auto (pos = 6):

acceleration, cylinders, displacement, horsepower, mpg, name, origin, weight, year

The following objects are masked from Auto (pos = 7):

acceleration, cylinders, displacement, horsepower, mpg, name, origin, weight, year

The following objects are masked from Auto (pos = 8):

acceleration, cylinders, displacement, horsepower, mpg, name, origin, weight, year

> mpg01 = rep(0, length(mpg))

> mpg01[mpg > median(mpg)] = 1

> Auto = data.frame(Auto, mpg01)

> cor(Auto[, -9])

mpg cylinders displacement horsepower weight acceleration year origin mpg01

mpg 1.0000000 -0.7776175 -0.8051269 -0.7784268 -0.8322442 0.4233285 0.5805410 0.5652088 0.8369392

cylinders -0.7776175 1.0000000 0.9508233 0.8429834 0.8975273 -0.5046834 -0.3456474 -0.5689316 -0.7591939

displacement -0.8051269 0.9508233 1.0000000 0.8972570 0.9329944 -0.5438005 -0.3698552 -0.6145351 -0.7534766

horsepower -0.7784268 0.8429834 0.8972570 1.0000000 0.8645377 -0.6891955 -0.4163615 -0.4551715 -0.6670526

weight -0.8322442 0.8975273 0.9329944 0.8645377 1.0000000 -0.4168392 -0.3091199 -0.5850054 -0.7577566

acceleration 0.4233285 -0.5046834 -0.5438005 -0.6891955 -0.4168392 1.0000000 0.2903161 0.2127458 0.3468215

year 0.5805410 -0.3456474 -0.3698552 -0.4163615 -0.3091199 0.2903161 1.0000000 0.1815277 0.4299042

origin 0.5652088 -0.5689316 -0.6145351 -0.4551715 -0.5850054 0.2127458 0.1815277 1.0000000 0.5136984

mpg01 0.8369392 -0.7591939 -0.7534766 -0.6670526 -0.7577566 0.3468215 0.4299042 0.5136984 1.0000000

> pairs(Auto)

> train = (year%%2 == 0)

> test = !train

> Auto.train = Auto[train, ]

> Auto.test = Auto[test, ]

> mpg01.test = mpg01[test]

> library(MASS)

> lda.fit = lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, subset = train)

> lda.pred = predict(lda.fit, Auto.test)

> mean(lda.pred$class != mpg01.test)

[1] 0.1263736

> qda.fit = qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, subset = train)

> qda.pred = predict(qda.fit, Auto.test)

> mean(qda.pred$class != mpg01.test)

[1] 0.1318681

> glm.fit = glm(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto, family = binomial, subset = train)

> glm.probs = predict(glm.fit, Auto.test, type = "response")

> glm.pred = rep(0, length(glm.probs))

> glm.pred[glm.probs > 0.5] = 1

> mean(glm.pred != mpg01.test)

[1] 0.1208791

> library(class)

> train.X = cbind(cylinders, weight, displacement, horsepower)[train, ]

> test.X = cbind(cylinders, weight, displacement, horsepower)[test, ]

> train.mpg01 = mpg01[train]

> set.seed(1)

> knn.pred = knn(train.X, test.X, train.mpg01, k = 1) # KNN(k=1)

> mean(knn.pred != mpg01.test)

[1] 0.1538462

> knn.pred = knn(train.X, test.X, train.mpg01, k = 10) # KNN(k=10)

> mean(knn.pred != mpg01.test)

[1] 0.1648352

> knn.pred = knn(train.X, test.X, train.mpg01, k = 100) # KNN(k=100)

> mean(knn.pred != mpg01.test)

[1] 0.1428571

(d) 12.6% test error rate.

(e) 13.2% test error rate

(f) 12.1% test error rate

(g) k=1, 15.4% test error rate. k=10, 16.5% test error rate. k=100, 14.3% test error rate.