Homework

Sai Jyothi

2023-03-02

```
# 9.
# install.packages("ISLR")
library(ISLR)
library(MASS)
data("Auto")
head(Auto)
```

```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
## 5  17         8          302        140   3449         10.5   70      1
## 6  15         8          429        198   4341         10.0   70      1
##                       name
## 1 chevrolet chevelle malibu
## 2         buick skylark 320
## 3        plymouth satellite
## 4             amc rebel sst
## 5               ford torino
## 6          ford galaxie 500
```

```
#
# # (a)
#  qualitative: name, origin
#  quantitative: mpg, displacement,  cylinders, horsepower,acceleration, year, weight


# # (b)
# # apply the range function to the first seven columns of Auto
sapply(Auto[, 1:7], range)
```

```
##       mpg cylinders displacement horsepower weight acceleration year
## [1,]  9.0         3           68         46   1613          8.0   70
## [2,] 46.6         8          455        230   5140         24.8   82
```

```
# #       mpg cylinders displacement horsepower weight acceleration year
# # [1,]  9.0         3           68         46   1613          8.0   70
# # [2,] 46.6         8          455        230   5140         24.8   82

# (c)
sapply(Auto[, 1:7], mean)
```

```
##          mpg    cylinders displacement   horsepower       weight acceleration
##    23.445918     5.471939   194.411990   104.469388  2977.584184    15.541327
##         year
##    75.979592
```

```
#         mpg    cylinders displacement   horsepower       weight acceleration
#   23.445918     5.471939   194.411990   104.469388  2977.584184    15.541327
#        year
#   75.979592
#
sapply(Auto[, 1:7], sd)
```

```
##          mpg    cylinders displacement   horsepower       weight acceleration
##     7.805007     1.705783   104.644004    38.491160   849.402560     2.758864
##         year
##     3.683737
```

```
# #         mpg    cylinders displacement   horsepower       weight acceleration
# #    7.805007     1.705783   104.644004    38.491160   849.402560     2.758864
# #        year
# #    3.683737
#
# # (d)
newAuto = Auto[-(10:85),]
dim(newAuto) == dim(Auto) - c(76,0)
```

```
## [1] TRUE TRUE
```

```
newAuto[9,] == Auto[9,]
```

```
##    mpg cylinders displacement horsepower weight acceleration year origin name
## 9 TRUE      TRUE         TRUE       TRUE   TRUE         TRUE TRUE   TRUE TRUE
```

```
newAuto[10,] == Auto[86,]
```

```
##     mpg cylinders displacement horsepower weight acceleration year origin name
## 87 TRUE      TRUE         TRUE       TRUE   TRUE         TRUE TRUE   TRUE TRUE
```

```
 sapply(newAuto[, 1:7], range)
```

```
##       mpg cylinders displacement horsepower weight acceleration year
## [1,] 11.0         3           68         46   1649          8.5   70
## [2,] 46.6         8          455        230   4997         24.8   82
```

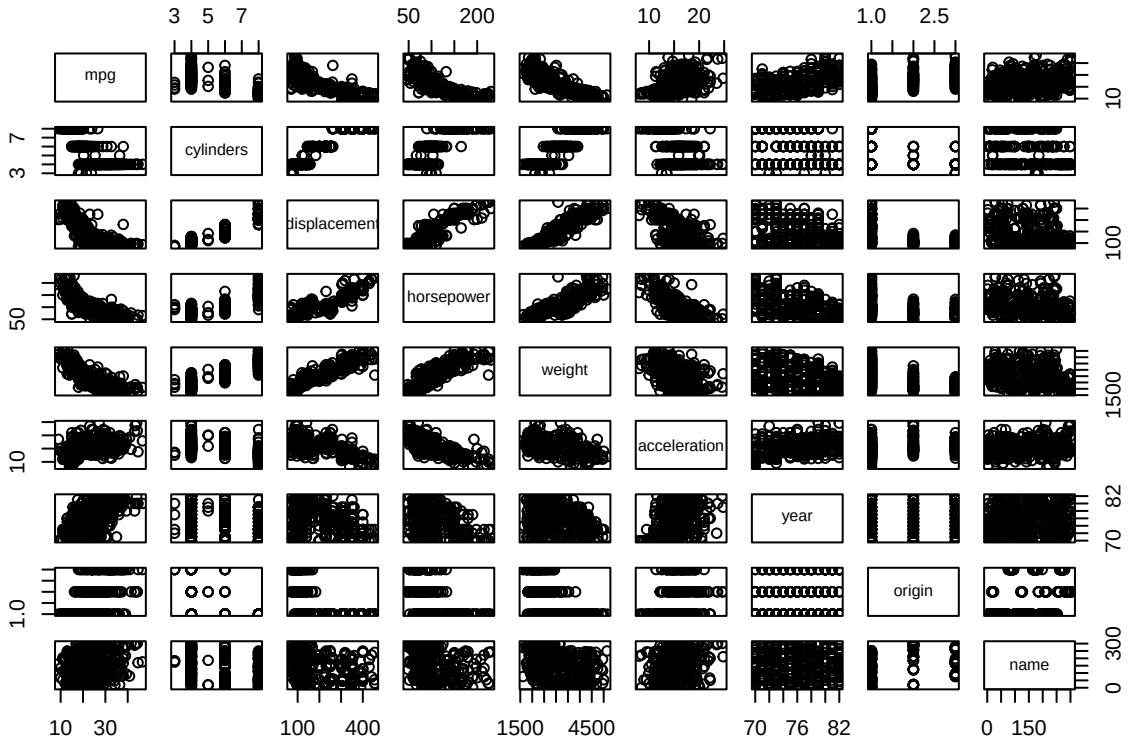
```
#       mpg cylinders displacement horsepower weight acceleration year
# [1,] 11.0         3           68         46   1649          8.5   70
# [2,] 46.6         8          455        230   4997         24.8   82
 sapply(newAuto[, 1:7], mean)
```

```
##          mpg    cylinders displacement   horsepower       weight acceleration
##    24.404430     5.373418   187.240506   100.721519  2935.971519    15.726899
##         year
##    77.145570
```

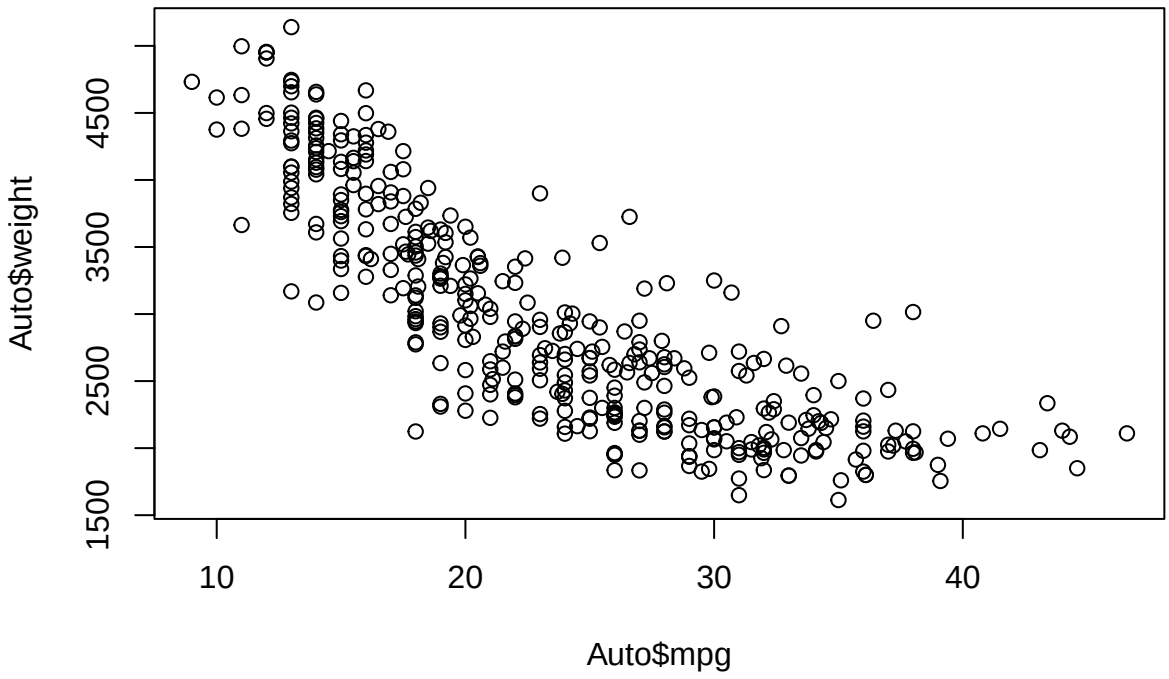
```
#          mpg    cylinders displacement   horsepower       weight acceleration
#   24.404430     5.373418   187.240506   100.721519  2935.971519    15.726899
#         year
#   77.145570
 sapply(newAuto[, 1:7], sd)
```

```
##          mpg    cylinders displacement   horsepower       weight acceleration
##     7.867283     1.654179    99.678367    35.708853   811.300208     2.693721
##         year
##     3.106217
```

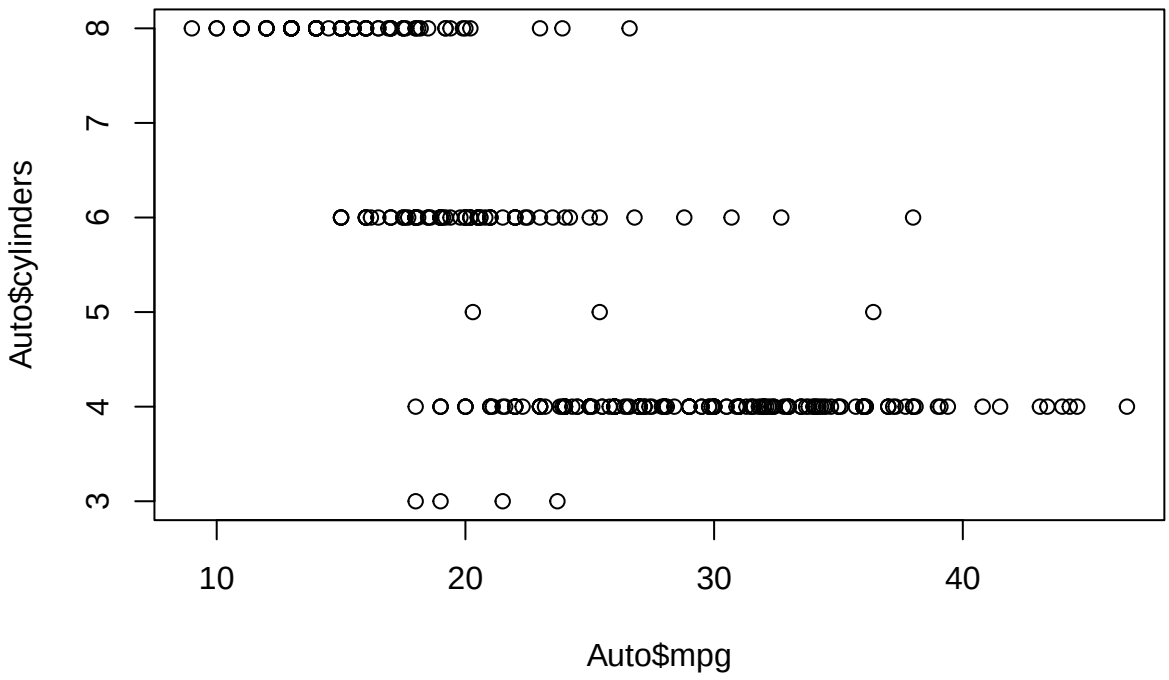
```
#          mpg    cylinders displacement   horsepower       weight acceleration
#    7.867283     1.654179    99.678367    35.708853   811.300208     2.693721
#         year
#    3.106217

# # (e)
 pairs(Auto)
```



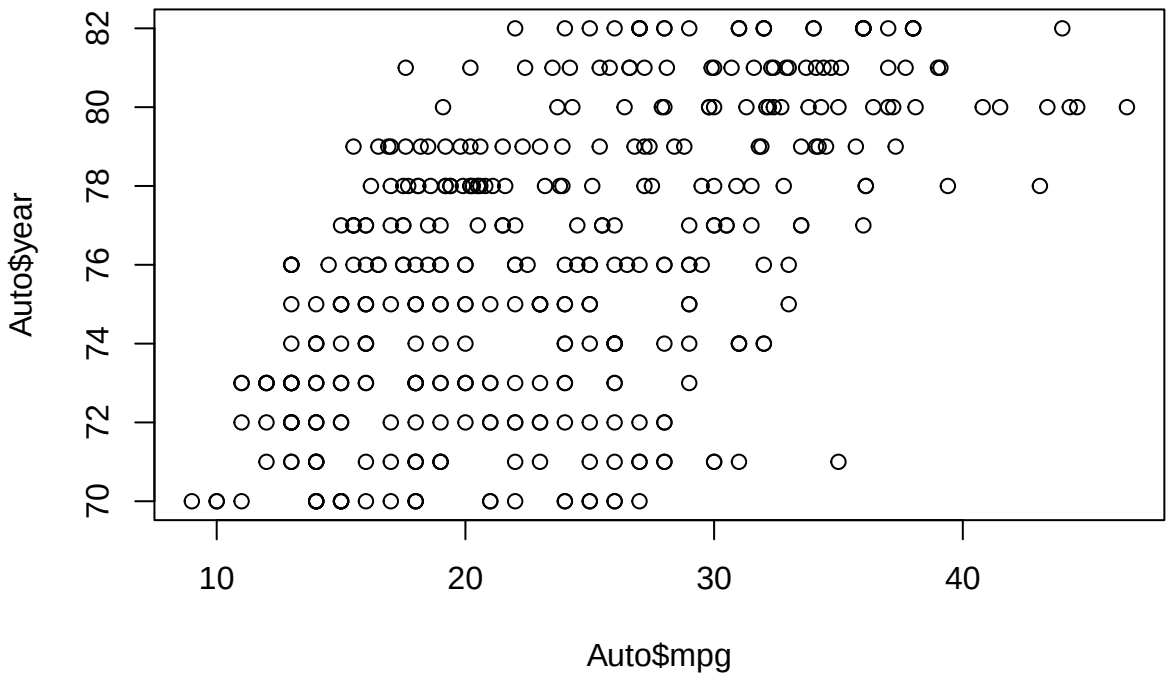
```
plot(Auto$mpg, Auto$weight)
```



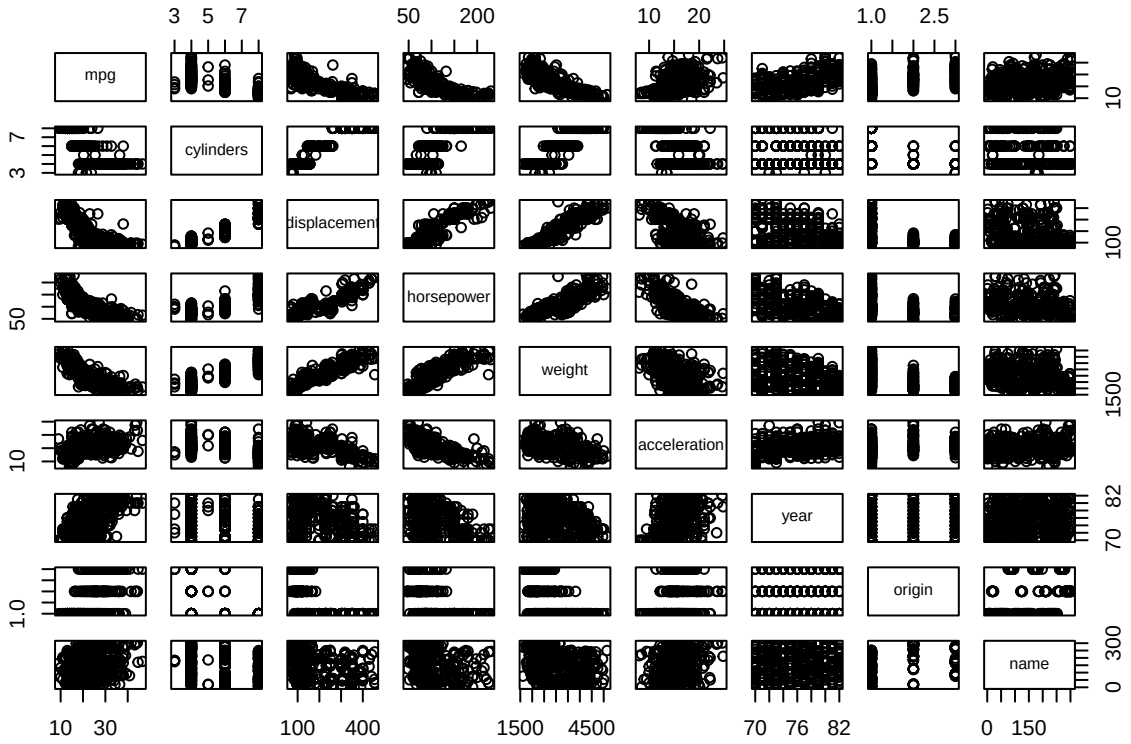

```
# Heavier weight correlates with lower mpg.
plot(Auto$mpg, Auto$cylinders)
```




```
# More cylinders, less mpg.
 plot(Auto$mpg, Auto$year)
```




```
# Cars become more efficient over time.
#
# # (f)
pairs(Auto)
```




```
# See descriptions of plots in (e).
# All of the predictors show some correlation with mpg. The name predictor has
# too little observations per name though, so using this as a predictor is
# likely to result in overfitting the data and will not generalize well.
```

```
#### CHAPTER 3 - Question 8
#### (A)
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
```

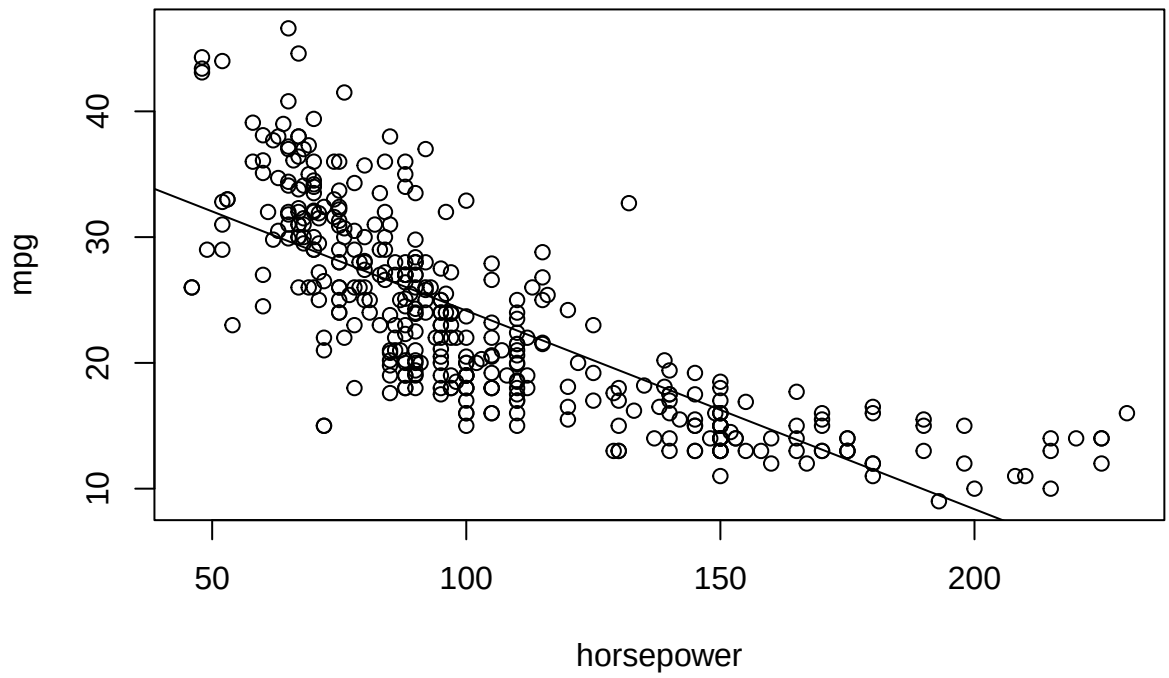
```
# (i).
# Yes, Assessing if there is a relationship between horsepower and mpg instead of the null hypothesis t

# (ii).
# To calculate the residual error in relation to the response, we use the response's mean and RSE. The
#
# (iii).
# There is a negative correlation between horsepower and mileage. According to linear regression, the m
#

# (iv).

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

```
##        fit      lwr      upr
## 1 24.46708 23.97308 24.96108
```

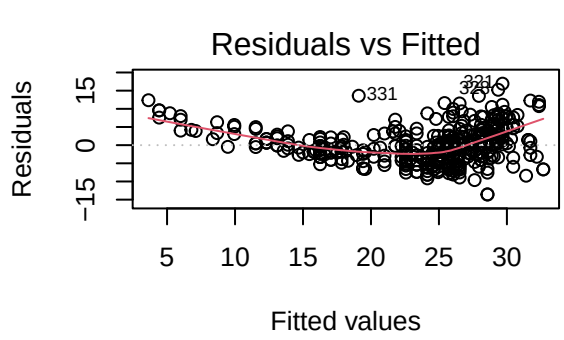
```
#
#
#
predict(lm.fit, data.frame(horsepower=c(98)), interval="prediction")
```

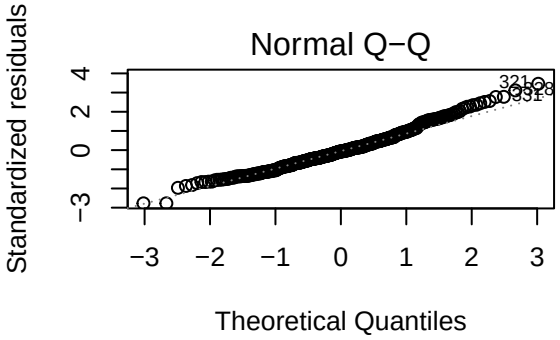
```
##        fit     lwr      upr
## 1 24.46708 14.8094 34.12476
```

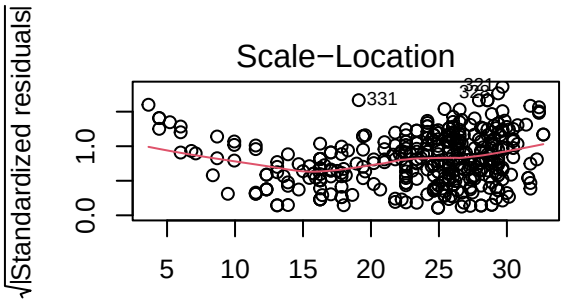
```
# 8b.
plot(horsepower, mpg)
abline(lm.fit)
```

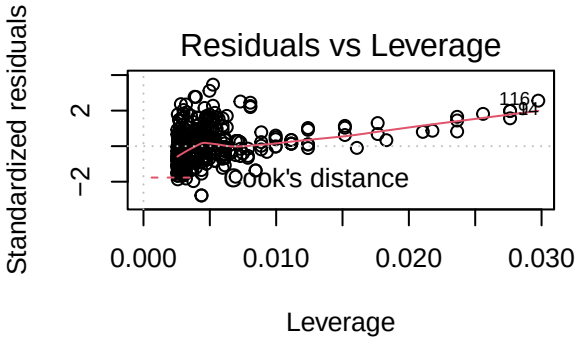



```
# 8c.
par(mfrow=c(2,2))
plot(lm.fit)
```







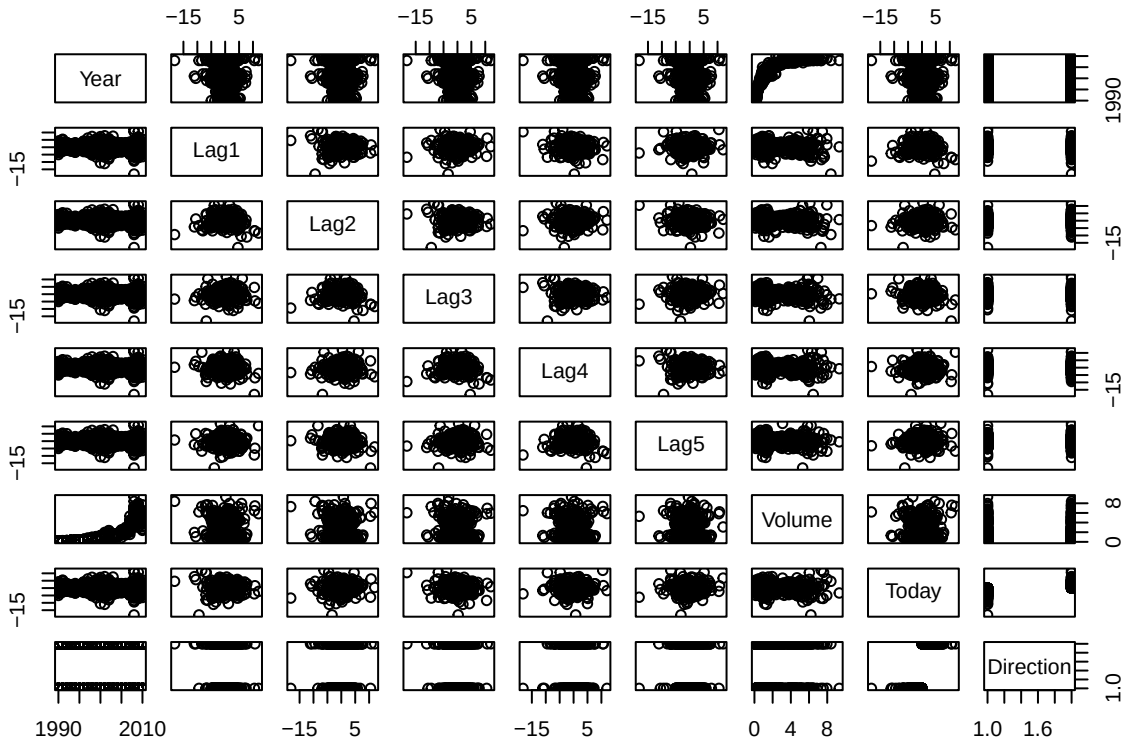



```
#### Chapter 4

#10 a
 library(ISLR)
 summary(Weekly)
```

```
##       Year           Lag1               Lag2               Lag3
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260
##       Lag4               Lag5              Volume            Today
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540
##  Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260
##  Direction
##  Down:484
##  Up  :605
##
##
##
```

##

```
pairs(Weekly)
```




```
cor(Weekly[, -9])
```

```
##              Year         Lag1        Lag2        Lag3         Lag4
## Year    1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923
## Lag1   -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876
## Lag2   -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535
## Lag3   -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865
## Lag4   -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000
## Lag5   -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027
## Volume  0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617
## Today  -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873
##                Lag5      Volume        Today
## Year   -0.030519101  0.84194162 -0.032459894
## Lag1   -0.008183096 -0.06495131 -0.075031842
## Lag2   -0.072499482 -0.08551314  0.059166717
## Lag3    0.060657175 -0.06928771 -0.071243639
## Lag4   -0.075675027 -0.06107462 -0.007825873
## Lag5    1.000000000 -0.05851741  0.011012698
## Volume -0.058517414  1.00000000 -0.033077783
## Today   0.011012698 -0.03307778  1.000000000
```

```r
# b
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
```

```r
#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)
```

```
##         Direction
## glm.pred Down  Up
##     Down   54  48
##     Up    430 557
```

```r
# 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)
```

```
##         Direction.0910
## glm.pred Down Up
##     Down    9  5
##     Up     34 56
```

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

```
## [1] 0.625
```

```
#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)
```

```
##       Direction.0910
##        Down Up
##   Down    9  5
##   Up     34 56
```

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

```
## [1] 0.625
```

```
# f
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
```

```
# 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)
```

```
##         Direction.0910
## knn.pred Down Up
##     Down   21 30
##     Up     22 31
```

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

```
## [1] 0.5
```

```
#h  Logistic regression and LDA methods provide similar test error rates.

# i
# 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)
```

```
##         Direction.0910
## glm.pred Down Up
##     Down    1  1
##     Up     42 60
```

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

```
## [1] 0.5865385
```

```
# 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)
```

```
## [1] 0.5769231
```

```
# 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)
```

```
##          Direction.0910
## qda.class Down Up
##      Down   12 13
##      Up     31 48
```

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

```
## [1] 0.5769231
```

```
# KNN k =10
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 k = 100
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
```