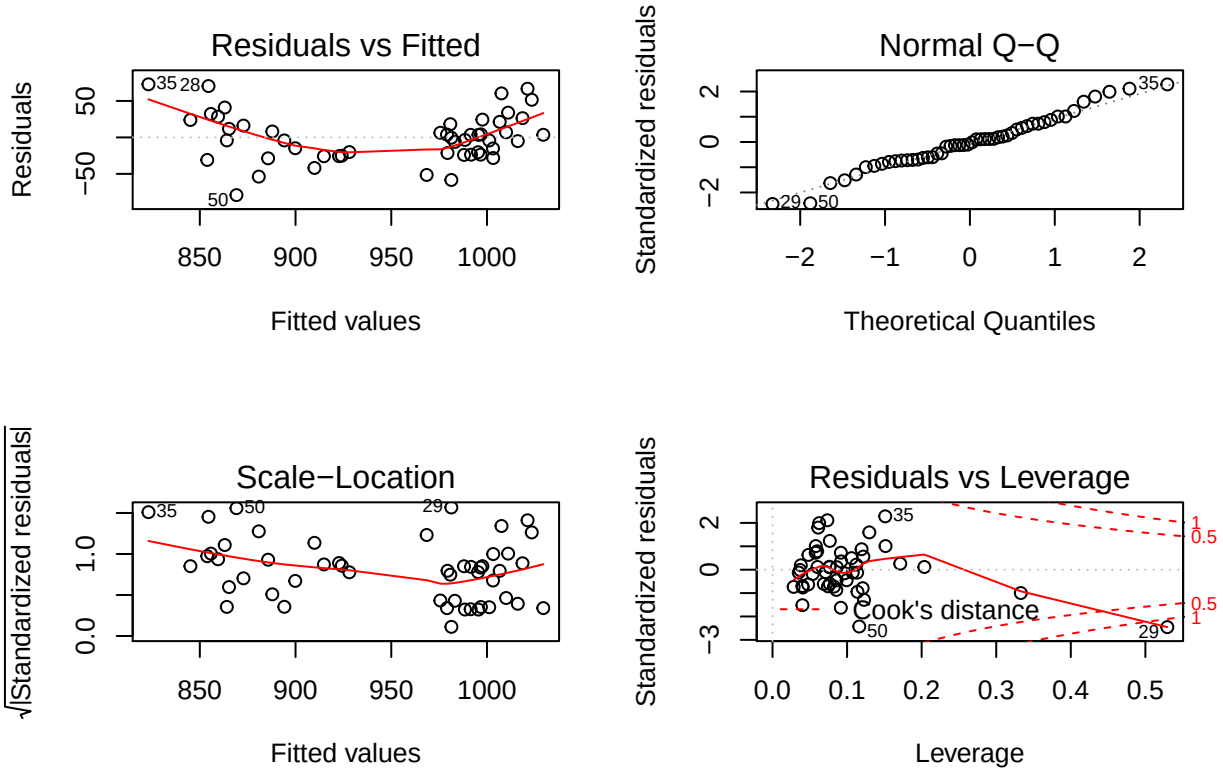
STT481 HW2 solution

100 points total

1.

```
SAT <- read.csv("sat.csv")
lm.fit <- lm(sat ~ expend + income + public + takers, data = SAT)
par(mfrow=c(2,2))
plot(lm.fit)
```



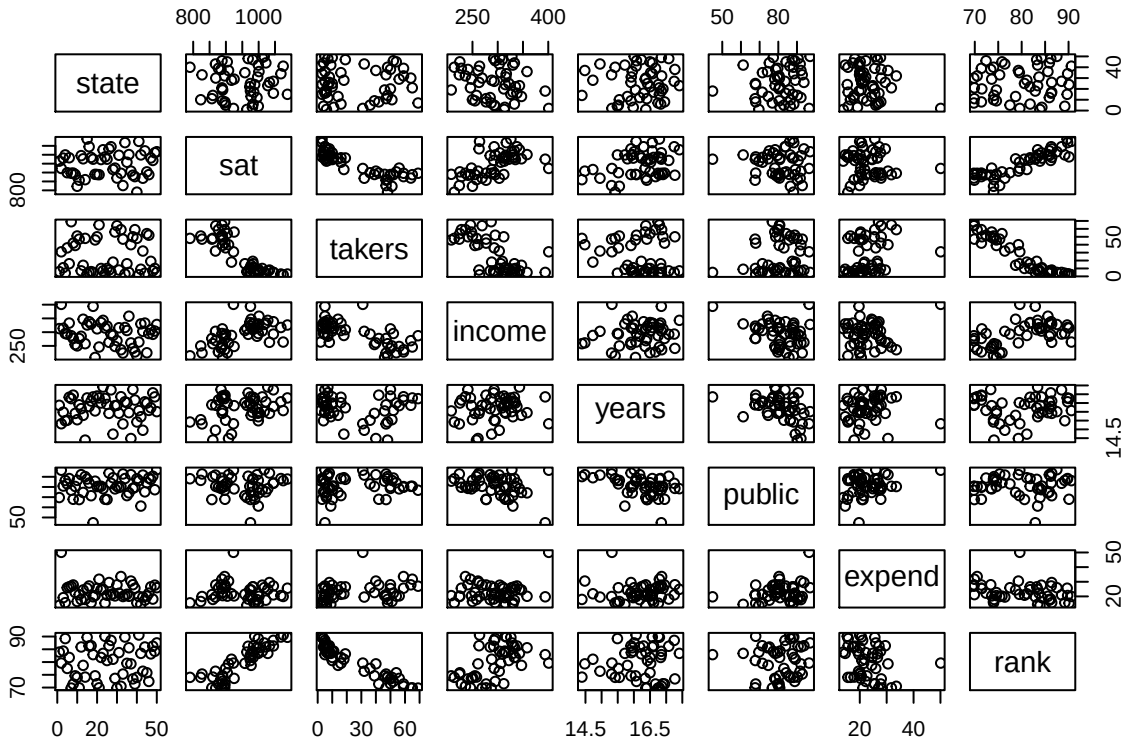

```
summary(lm.fit)
```

```
##
## Call:
## lm(formula = sat ~ expend + income + public + takers, data = SAT)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -79.139 -23.673  -1.953  20.554  72.827
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1083.1366    84.9776  12.746  < 2e-16 ***
```

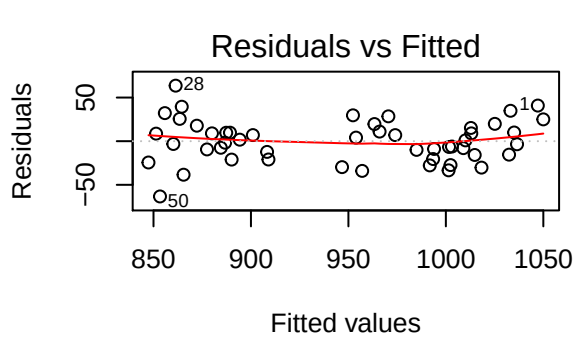
```
## expend         3.1332     1.0589   2.959  0.00491 **
## income        -0.2533     0.1929  -1.313  0.19582
## public        -0.5656     0.6012  -0.941  0.35177
## takers        -3.3093     0.3692  -8.965 1.42e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34.66 on 45 degrees of freedom
## Multiple R-squared:  0.7803, Adjusted R-squared:  0.7607
## F-statistic: 39.94 on 4 and 45 DF,  p-value: 2.896e-14
```

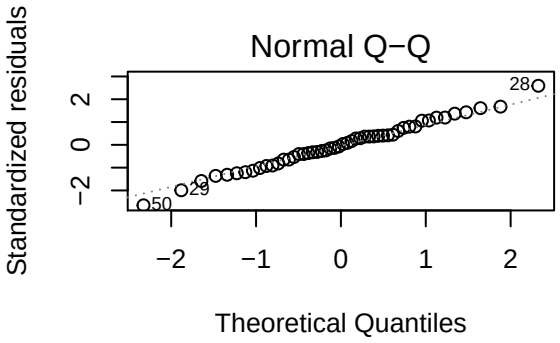
(a) From the Residuals vs Fitted plot (top left), the constant variance assumption looks fine.
(b) From the Normal Q-Q plot, most of points are located on the straight line. Even though some points are identified as questionable points, the distribution seems not to be far away from normal distribution.
(c) From the Residual vs Leverage plot, observation 29 has large Cook's distance (larger than 1), so I will identify this observation as large leverage point.
(d) From the Scale-Location plot, the standardized residuals are smaller than 3, or equivalently $\sqrt{\text{standardized residuals}} < \sqrt{3}$, so it seems that no outlier is in this dataset.
(e) From the Residuals vs Fitted plot, it appears that there is some structure of the relationship between the predictors and the response.
(f) It appears that `takers` appears to have a quadratic relationship with `sat`.

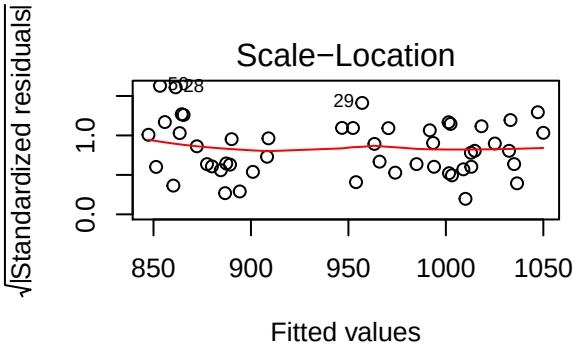
```
pairs(SAT)
```

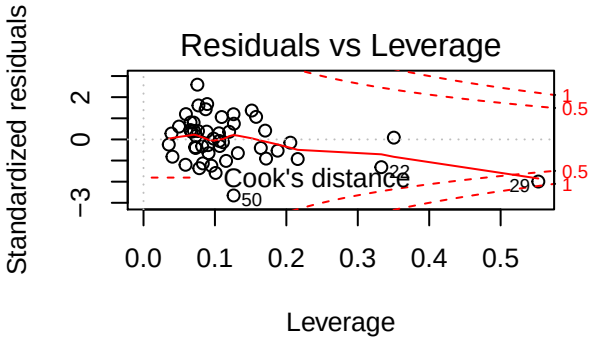



```
lm.fit <- lm(sat ~ expend + income + public + takers + I(takers^2), data = SAT)
par(mfrow=c(2,2))
plot(lm.fit)
```









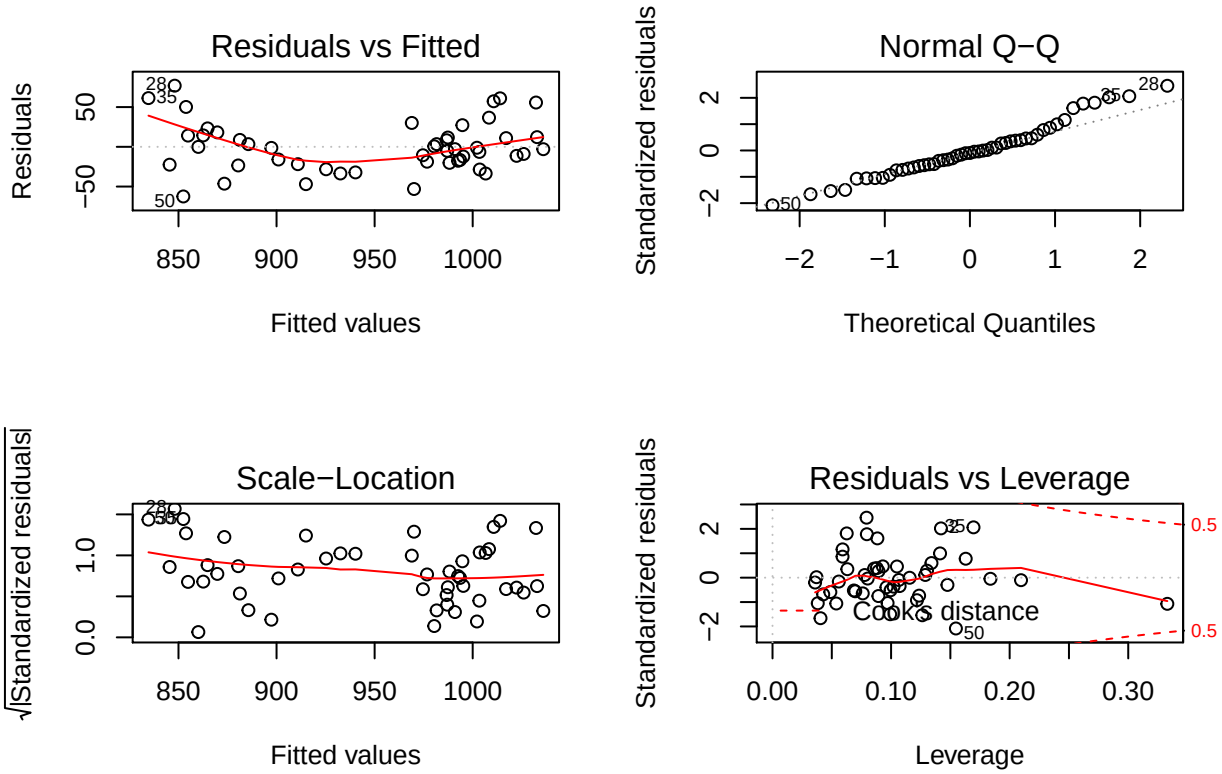

```
summary(lm.fit)
```

```
##
## Call:
## lm(formula = sat ~ expend + income + public + takers + I(takers^2),
##     data = SAT)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.342 -15.703  -0.402  14.122  63.631
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1063.99010   62.69582  16.971  < 2e-16 ***
## expend         3.18845    0.78036   4.086 0.000183 ***
## income        -0.21083    0.14229  -1.482 0.145554
## public        -0.07225    0.45000  -0.161 0.873179
## takers        -8.01746    0.80269  -9.988 6.93e-13 ***
## I(takers^2)    0.07636    0.01225   6.234 1.53e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25.54 on 44 degrees of freedom
## Multiple R-squared:  0.8833, Adjusted R-squared:  0.8701
## F-statistic: 66.62 on 5 and 44 DF,  p-value: < 2.2e-16
```

It appears that there is no structure of the relationship between the predictors and the response. Also, the observation 29 now is not a high leverage point.

(g)

```r
SAT <- SAT[-29,]
lm.fit <- lm(sat ~ expend + income + public + takers, data = SAT)
par(mfrow=c(2,2))
plot(lm.fit)
```




```r
summary(lm.fit)
```

```
##
## Call:
## lm(formula = sat ~ expend + income + public + takers, data = SAT)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -62.47 -20.19  -2.94  14.12  76.92
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 960.09205   92.84121  10.341 2.35e-13 ***
## expend        4.41310    1.11072   3.973  0.00026 ***
## income       -0.05193    0.19723  -0.263  0.79353
## public       -0.16418    0.58630  -0.280  0.78077
## takers       -3.14341    0.35319  -8.900 2.14e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32.62 on 44 degrees of freedom
## Multiple R-squared:  0.8092, Adjusted R-squared:  0.7919
## F-statistic: 46.65 on 4 and 44 DF,  p-value: 2.797e-15
```

It appears that there is still some structure of the relationship between the predictors and the response.

(h) I would choose (f) because not only the residual diagnostics show that the assumptions for linear models all hold, but also removing a high leverage point or outlier without checking with data collector can be dangerous. It can be due to a deficiency with the model, such as the model missing a quadratic effect.

2. Question 2 in 4.8

Assuming that $f_k(x)$ is normal, the probability that an observation x is in class k is given by

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu_k)^2)}{\sum \pi_l \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu_l)^2)}$$

while the discriminant function is given by

$$\delta_k(x) = x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

Claim: Maximizing $p_k(x)$ is equivalent to maximizing $\delta_k(x)$.

Proof. Let x remain fixed and observe that we are maximizing over the parameter k. Suppose that $\delta_k(x) \geq \delta_i(x)$. We will show that $f_k(x) \geq f_i(x)$. From our assumption we have

$$x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k) \geq x\frac{\mu_i}{\sigma^2} - \frac{\mu_i^2}{2\sigma^2} + \log(\pi_i).$$

Exponentiation is a monotonically increasing function, so the following inequality holds

$$\pi_k \exp(x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2}) \geq \pi_i \exp(x\frac{\mu_i}{\sigma^2} - \frac{\mu_i^2}{2\sigma^2})$$

Multipy this inequality by the positive constant

$$c = \frac{\frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}x^2)}{\sum \pi_l \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu_l)^2)}$$

and we have that

$$\frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu_k)^2)}{\sum \pi_l \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu_l)^2)} \geq \frac{\pi_i \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu_i)^2)}{\sum \pi_l \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}(x-\mu_l)^2)}$$

or equivalently, $f_k(x) \geq f_i(x)$. Reversing these steps also holds, so we have that maximizing δ_k is equivalent to maximizing p_k.

3. Question 5 in 4.8

(a) If the Bayes decision boundary is linear, we expect QDA to perform better on the training set because it's higher flexiblity will yield a closer fit. On the test set, we expect LDA to perform better than QDA because QDA could overfit the linearity of the Bayes decision boundary.

(b) If the Bayes decision bounary is non-linear, we expect QDA to perform better both on the training and test sets.

(c) We expect the test prediction accuracy of QDA relative to LDA to improve, in general, as the the sample size n increases because a more flexibile method will yield a better fit as more samples can be fit and variance is offset by the larger sample sizes.

(d) False. With fewer sample points, the variance from using a more flexible method, such as QDA, would lead to overfit, yielding a higher test rate than LDA.

4. Question 6 in 4.8

$$p(X) = \frac{\exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2)}{1 + \exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2)}$$
$$X_1 = hoursstudied, X_2 = undergradGPA$$
$$\beta_0 = -6, \beta_1 = 0.05, \beta_2 = 1$$

a.

$$X = [40hours, 3.5GPA]$$
$$p(X) = \frac{\exp(-6 + 0.05X_1 + X_2)}{1 + \exp(-6 + 0.05X_1 + X_2)}$$
$$= \frac{\exp(-6 + 0.0540 + 3.5)}{1 + \exp(-6 + 0.0540 + 3.5)}$$
$$= \frac{\exp(-0.5)}{1 + \exp(-0.5)}$$
$$= 37.75\%$$

b.

$$X = [X_1 hours, 3.5GPA]$$
$$p(X) = \frac{\exp(-6 + 0.05X_1 + X_2)}{1 + \exp(-6 + 0.05X_1 + X_2)}$$
$$0.50 = \frac{\exp(-6 + 0.05X_1 + 3.5)}{1 + \exp(-6 + 0.05X_1 + 3.5)}$$
$$0.50(1 + \exp(-2.5 + 0.05X_1)) = \exp(-2.5 + 0.05X_1)$$
$$0.50 + 0.50\exp(-2.5 + 0.05X_1)) = \exp(-2.5 + 0.05X_1)$$
$$0.50 = 0.50\exp(-2.5 + 0.05X_1)$$
$$\log(1) = -2.5 + 0.05X_1$$
$$X_1 = 2.5/0.05 = 50hours$$

5.Question 9 in 4.8

a.

$$\frac{p(X)}{1-p(X)} = 0.37$$
$$p(X) = 0.37(1-p(X))$$
$$1.37p(X) = 0.37$$
$$p(X) = \frac{0.37}{1.37} = 27\%$$

b.

$$odds = \frac{p(X)}{1-p(X)} = .16/.84 = 0.19$$

6.(a)

```
arthritis <- read.csv("arthritis.csv")
glm.fit <- glm(Improved ~ Treatment + Sex + Age, family = binomial, data = arthritis)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Improved ~ Treatment + Sex + Age, family = binomial,
##     data = arthritis)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.10833  -0.91158   0.05362   0.91681   1.84659
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -3.01546    1.16777  -2.582  0.00982 **
## TreatmentTreated  1.75980    0.53650   3.280  0.00104 **
## SexMale          -1.48783    0.59477  -2.502  0.01237 *
## Age               0.04875    0.02066   2.359  0.01832 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 116.449  on 83  degrees of freedom
## Residual deviance:  92.063  on 80  degrees of freedom
## AIC: 100.06
##
```

```
## Number of Fisher Scoring iterations: 4
```

6.(b)

- With treatment, the log odds of some improvement (versus none) increases by 1.8.
- For males, the log odds of some improvement (versus none) decreases by 1.5 (versus females).
- For a one year increase in age, the log odds of some improvement (versus none) increases by 0.05.

6.(c)

- With treatment, the odd ratios of some improvement (versus none) increases by `round(exp(glm.fit$coefficients)[2],` =5.81 (versus no treatment).
- For males, the odd ratios of some improvement (versus none) decreases by `round(exp(-glm.fit$coefficients)[3],2)` = 4.43 (versus females) or increases by `round(exp(glm.fit$coefficients)[3],2)` = 0.23 (versus females).
- For a one year increase in age, the odd ratios of some improvement (versus none) increases by `round(exp(glm.fit$coefficients)[4],2)`=1.05.

6.(d)

```
confint(glm.fit)
```

```
## Waiting for profiling to be done...
```

```
##                          2.5 %      97.5 %
## (Intercept)      -5.477304188 -0.84351926
## TreatmentTreated  0.750803551  2.87506538
## SexMale          -2.729719456 -0.37239953
## Age               0.009951561  0.09194283
```

6.(e)

```
glm.fit <- glm(Improved ~ Treatment + Sex + Age + Sex*Age, family = binomial, data = arthritis)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Improved ~ Treatment + Sex + Age + Sex * Age, family = binomial,
##     data = arthritis)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.30304  -0.88850   0.01037   0.91087   2.12913
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -4.55477    1.56009  -2.920  0.00351 **
## TreatmentTreated  1.79705    0.55996   3.209  0.00133 **
## SexMale           2.75066    2.34871   1.171  0.24155
## Age               0.07734    0.02775   2.787  0.00532 **
## SexMale:Age      -0.07945    0.04313  -1.842  0.06545 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 116.45  on 83  degrees of freedom
## Residual deviance:  88.64  on 79  degrees of freedom
## AIC: 98.64
##
## Number of Fisher Scoring iterations: 4
```

For a one year increase in age, the log odds of some improvement (versus none) increases by `0.07734 - 0.07945` if the patient is male and by `0.07734` if the patient is female.

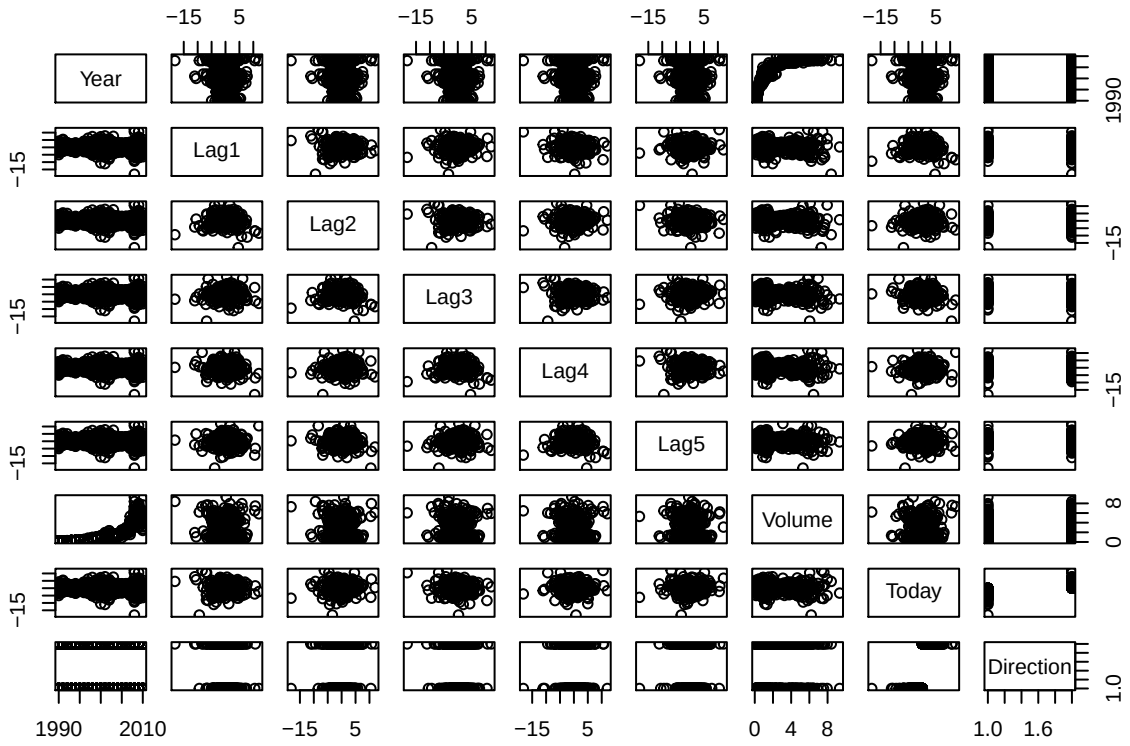
7. Question 13 in Section 4.8 except (h)

7.(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
```

Year and Volume appear to have a relationship. No other patterns are discernible.

7.(b)

```
attach(Weekly)
glm.fit2 = glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly, family = binomial)
summary(glm.fit2)
```

```
##
## 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
```

Lag 2 appears to have some statistical significance with type-I error = 0.05.

7.(c)

```
glm.probs = predict(glm.fit2, 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
```

Percentage of currect 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 down the logistic regression is wrong most of the time 54/(430+54) = 11.2%.

7.(d)

```
train.index <- (Year < 2009)
Weekly.train <- Weekly[train.index, ]
Weekly.test <- Weekly[!train.index, ]
glm.fit <- glm(Direction ~ Lag2, data = Weekly.train, family = binomial)
glm.probs <- predict(glm.fit, Weekly.test, type = "response")
glm.pred = rep("Down", length(glm.probs))
glm.pred[glm.probs > 0.5] = "Up"
```

```
Direction.test = Weekly.test[,"Direction"]
table(glm.pred, Direction.test)
```

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

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

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

7.(e)

```
library(MASS)
lda.fit3 = lda(Direction ~ Lag2, data = Weekly.train)
lda.pred = predict(lda.fit3, Weekly.test)
table(lda.pred$class, Direction.test)
```

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

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

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

7.(f)

```
qda.fit2 = qda(Direction ~ Lag2, data = Weekly.train)
qda.class = predict(qda.fit2, Weekly.test)$class
table(qda.class, Direction.test)
```

```
##          Direction.test
## qda.class Down Up
##      Down    0  0
##      Up     43 61
```

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

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

A correctness of 58.65% even though it picked Up the whole time!

7.(g)

```
library(class)
train.X = as.matrix(Weekly.train[,"Lag2"])
test.X = as.matrix(Weekly.test[,"Lag2"])
train.Direction = Weekly.train[,"Direction"]
set.seed(1)
knn.pred = knn(train.X, test.X, train.Direction, k = 1)
table(knn.pred, Direction.test)
```

```
##         Direction.test
## knn.pred Down Up
```

```
##     Down   21 30
##     Up     22 31
```

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

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

7.(i)

Logistic regression and LDA methods provide similar test error rates.

7.(j)

Here I add an interaction and quadratic effects for logistic regression, LDA, and QDA methods. For KNN method, I try K = 10 and 100 using three predictors: Lag1, Lag 2, and Volume.

```
# Logistic regression
glm.fit3 = glm(Direction ~ Lag1*Lag2 + Lag1^2 + Lag2^2, data = Weekly.train, family = binomial)
glm.probs = predict(glm.fit3, Weekly.test, type = "response")
glm.pred = rep("Down", length(glm.probs))
glm.pred[glm.probs > 0.5] = "Up"
Direction.test = Weekly.test[,"Direction"]
table(glm.pred, Direction.test)
```

```
##         Direction.test
## glm.pred Down Up
##     Down    7  8
##     Up     36 53
```

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

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

```
# LDA
lda.fit4 = lda(Direction ~ Lag1*Lag2 + Lag1^2 + Lag2^2, data = Weekly.train)
lda.pred4 = predict(lda.fit4, Weekly.test)
mean(lda.pred4$class == Direction.test)
```

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

```
# QDA with sqrt(abs(Lag2))
qda.fit3 = qda(Direction ~ Lag1*Lag2 + Lag1^2 + Lag2^2, data = Weekly.train)
qda.class = predict(qda.fit3, Weekly.test)$class
table(qda.class, Direction.test)
```

```
##          Direction.test
## qda.class Down Up
##      Down   23 36
##      Up     20 25
```

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

```
## [1] 0.4615385
```

```
# KNN k =10
train.X <- Weekly.train[,c("Lag1", "Lag2", "Volume")]
train.Direction <- Weekly.train[,"Direction"]
test.X <- Weekly.test[,c("Lag1", "Lag2", "Volume")]
Direction.test <- Weekly.test[,"Direction"]
```

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

```
##         Direction.test
## knn.pred Down Up
##     Down   26 32
##     Up     17 29
```

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

```
## [1] 0.5288462
```

```
# KNN k = 100
knn.pred = knn(train.X, test.X, train.Direction, k = 100)
table(knn.pred, Direction.test)
```

```
##         Direction.test
## knn.pred Down Up
##     Down   14 14
##     Up     29 47
```

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

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

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

8. Question 14 in Section 4.8 except (g)

8.(a)

```
library(ISLR)
summary(Auto)
```

```
##       mpg          cylinders      displacement     horsepower        weight
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140
##
##   acceleration        year           origin                      name
##  Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador       :  5
##  1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto        :  5
##  Median :15.50   Median :76.00   Median :1.000   toyota corolla    :  5
##  Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin       :  4
##  3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet        :  4
##  Max.   :24.80   Max.   :82.00   Max.   :3.000   chevrolet chevette:  4
##                                                  (Other)           :365
```

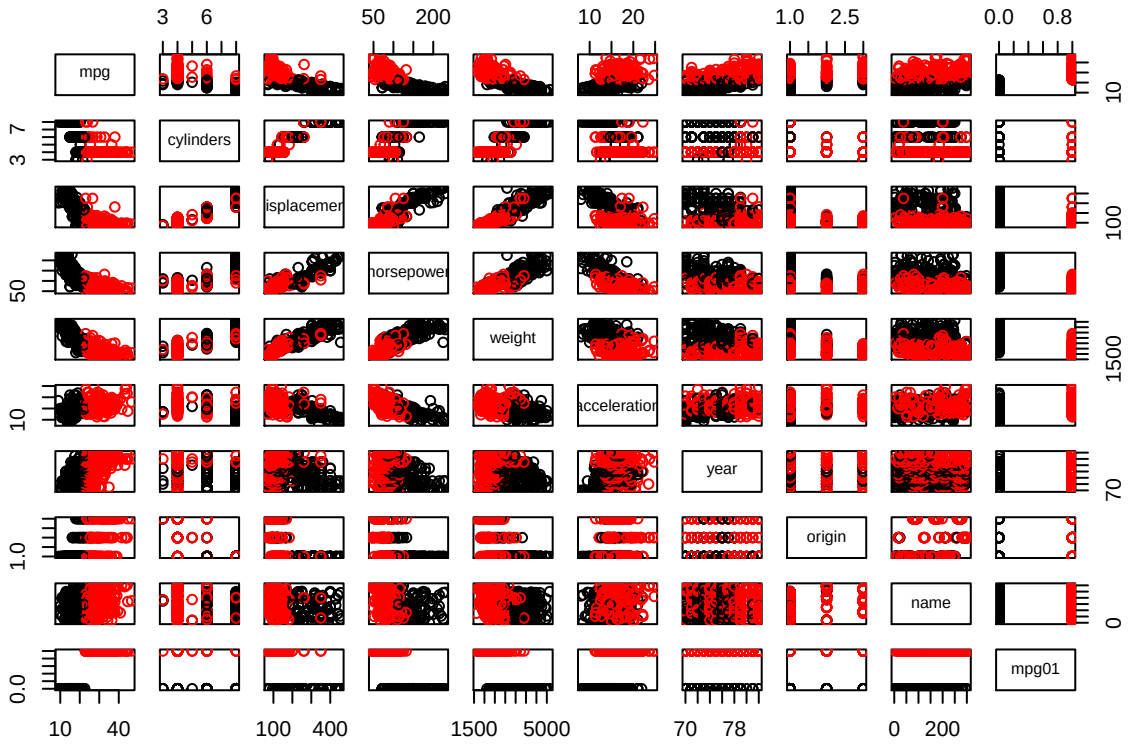
```
attach(Auto)
mpg01 = rep(0, length(mpg))
mpg01[mpg > median(mpg)] = 1
Auto = data.frame(Auto, mpg01)
```

8.(b)

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

```
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
## horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
## weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
## acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
## mpg01         0.8369392 -0.7591939   -0.7534766 -0.6670526 -0.7577566
##              acceleration       year     origin      mpg01
## mpg             0.4233285  0.5805410  0.5652088  0.8369392
## cylinders      -0.5046834 -0.3456474 -0.5689316 -0.7591939
## displacement   -0.5438005 -0.3698552 -0.6145351 -0.7534766
## horsepower     -0.6891955 -0.4163615 -0.4551715 -0.6670526
## weight         -0.4168392 -0.3091199 -0.5850054 -0.7577566
## acceleration    1.0000000  0.2903161  0.2127458  0.3468215
## year            0.2903161  1.0000000  0.1815277  0.4299042
## origin          0.2127458  0.1815277  1.0000000  0.5136984
## mpg01           0.3468215  0.4299042  0.5136984  1.0000000
```

```
pairs(Auto, col = mpg01 + 1)
```



Correlated with cylinders, weight, displacement, horsepower. (mpg, of course)

8.(c)

```
train = (year%%2 == 0)  # if the year is even
test = !train
Auto.train = Auto[train, ]
Auto.test = Auto[test, ]
mpg01.test = mpg01[test]
```

8.(d)

```
# LDA
library(MASS)
lda.fit = lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto.train)
lda.pred = predict(lda.fit, Auto.test)
mean(lda.pred$class != mpg01.test)
```

```
## [1] 0.1263736
```

Thus, 12.64% test error rate.

8.(e)

```
# QDA
qda.fit = qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto.train)
qda.pred = predict(qda.fit, Auto.test)
mean(qda.pred$class != mpg01.test)
```

```
## [1] 0.1318681
```

Thus, 13.19% test error rate.

8.(f)

```
# Logistic regression
glm.fit = glm(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto.train, family = binom
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
```

Thus, 12.09% test error rate.

8.(h)

```
library(class)
train.X = Auto.train[,c("cylinders", "weight", "displacement", "horsepower")]
test.X = Auto.test[,c("cylinders", "weight", "displacement", "horsepower")]
train.mpg01 = Auto.train[,"mpg01"]
set.seed(1)
# KNN(k=1)
knn.pred = knn(train.X, test.X, train.mpg01, k = 1)
mean(knn.pred != mpg01.test)
```

```
## [1] 0.1538462
```

```r
# KNN(k=10)
knn.pred = knn(train.X, test.X, train.mpg01, k = 10)
mean(knn.pred != mpg01.test)
```

```
## [1] 0.1648352
```

```r
# KNN(k=100)
knn.pred = knn(train.X, test.X, train.mpg01, k = 100)
mean(knn.pred != mpg01.test)
```

```
## [1] 0.1428571
```

k=1, 15.38% test error rate. k=10, 16.48% test error rate. k=100, 14.29% test error rate. K of 100 seems to perform the best. 100 nearest neighbors.

9.(a)

```r
library(MASS)
train.dat <- read.csv("zipcode_train.csv")
train.dat$Y <- as.factor(train.dat$Y)
test.dat <- read.csv("zipcode_test.csv")
test.dat$Y <- as.factor(test.dat$Y)
glm.fit <- glm(Y ~ ., family = binomial, data = train.dat)
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
y <- predict(glm.fit, newdata = test.dat, type="response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type
## == : prediction from a rank-deficient fit may be misleading
```

```r
yhat <- rep(1,nrow(test.dat))
yhat[y>0.5] <- 2
print(table(test.dat$Y, yhat))
```

```
##    yhat
##       1   2
##   1 260   4
##   2   4 194
```

```r
print(mean(test.dat$Y!=yhat))
```

```
## [1] 0.01731602
```

```r
lda.fit <- lda(Y ~ .-p16-p32, data = train.dat)
y <- predict(lda.fit, newdata = test.dat)

print(table(test.dat$Y, y$class))
```

```
##
##       1   2
##   1 259   5
##   2   2 196
```

```r
print(mean(y$class!=test.dat$Y))
```

```
## [1] 0.01515152
```

```
library(class)
set.seed(1)
y <- knn(train.dat[,1:256],test.dat[,1:256], train.dat[,"Y"],k = 3)
print(table(test.dat$Y, y))
```

```
##    y
##       1   2
##   1 261   3
##   2   2 196
```

```
print(mean(y!=test.dat$Y))
```

```
## [1] 0.01082251
```

```
y <- knn(train.dat[,1:256],test.dat[,1:256], train.dat[,"Y"],k = 10)
print(table(test.dat$Y, y))
```

```
##    y
##       1   2
##   1 261   3
##   2   3 195
```

```
print(mean(y!=test.dat$Y))
```

```
## [1] 0.01298701
```

9.(b)

It appears that the KNNs have the lowest test error.