# STT 481 HW 2

Derien Weatherspoon

2023-02-13

## Question 1:

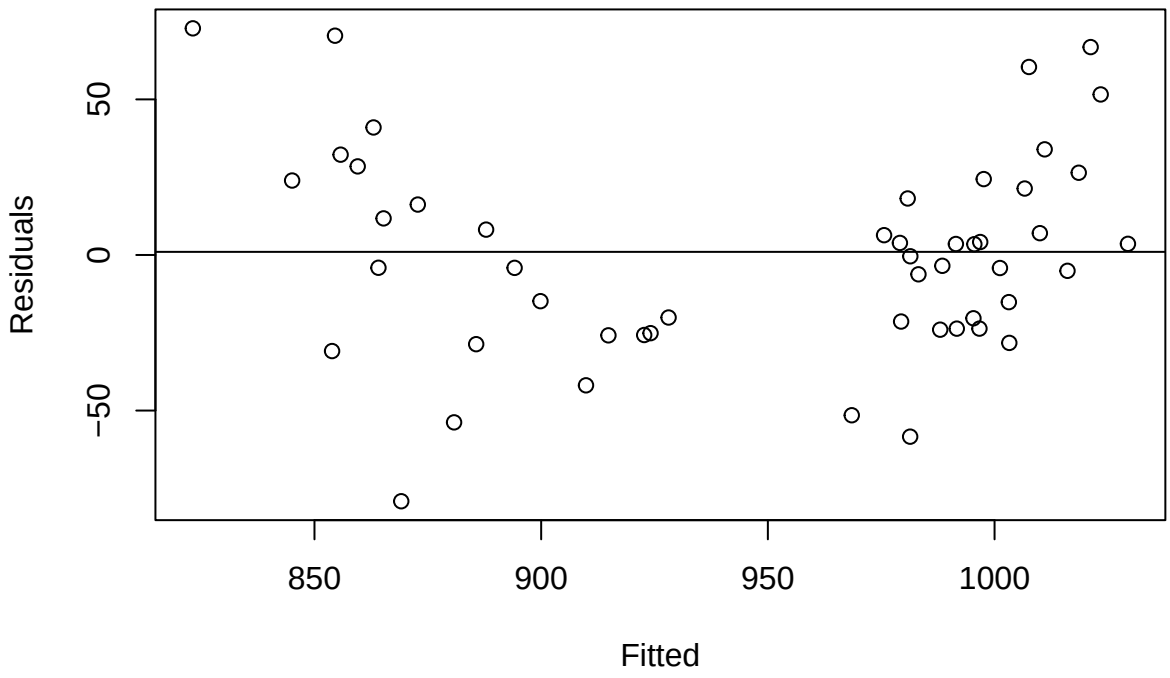
In 1982, average SAT scores were published with breakdowns of state-by-state performance in the United States. The average SAT scores varied considerably by state, with mean scores falling between 790 (South Carolina) to 1088 (Iowa). Two researchers examined compositional and demographic variables to examine to what extent these characteristics were tied to SAT scores. Fit a model with "sat" as the response, and "expend", "income", "public", and "takers" as predictors. Perform regression diagnostics on this model to answer the following questions. Display any plots that are relevant.

a) Check the constant variance assumption for the errors.

```
sat.fit <- lm(sat ~ expend + income + public + takers, data = SAT) #Fitting the model
summary(sat.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
```

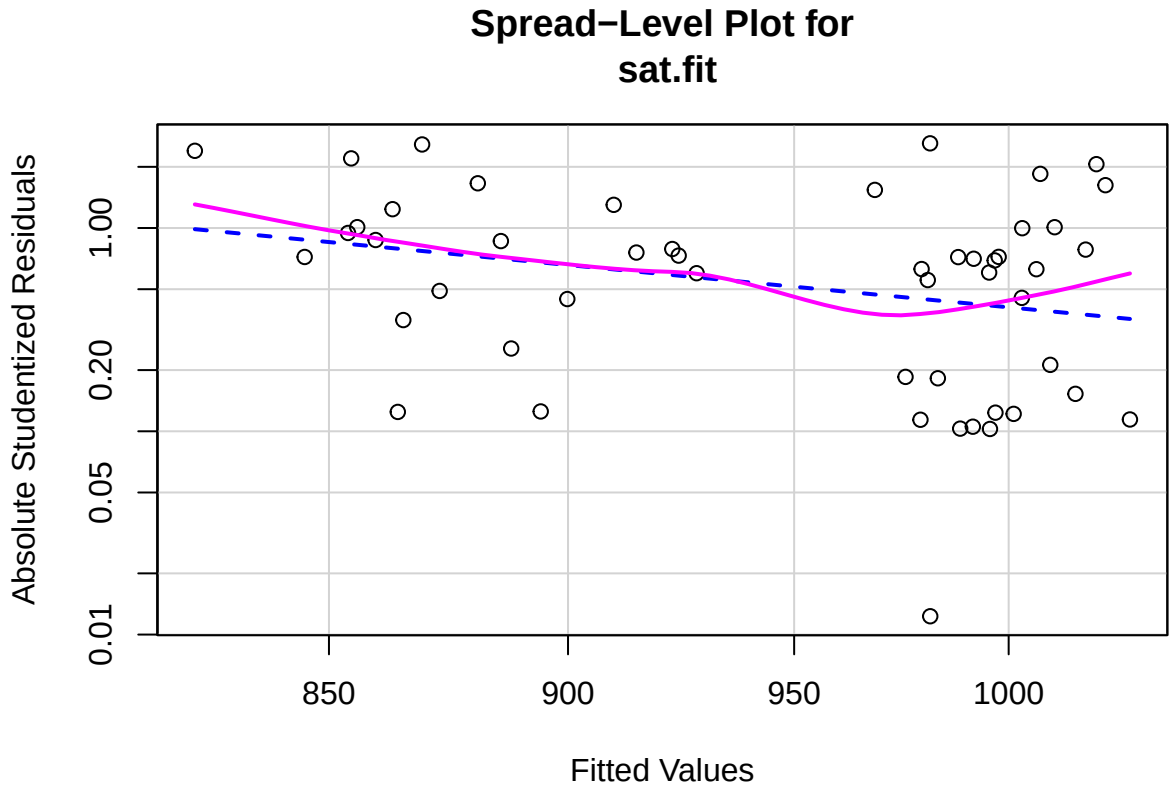
```
plot(fitted(sat.fit), residuals(sat.fit), xlab = "Fitted", ylab = "Residuals") #checking constant varia
abline(h=1)
```




```
ncvTest(sat.fit) #non-constant variance test for homoscedasticity
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 3.309361, Df = 1, p = 0.068886
```

```
spreadLevelPlot(sat.fit) #plotting studentized residuals vs fitted values
```

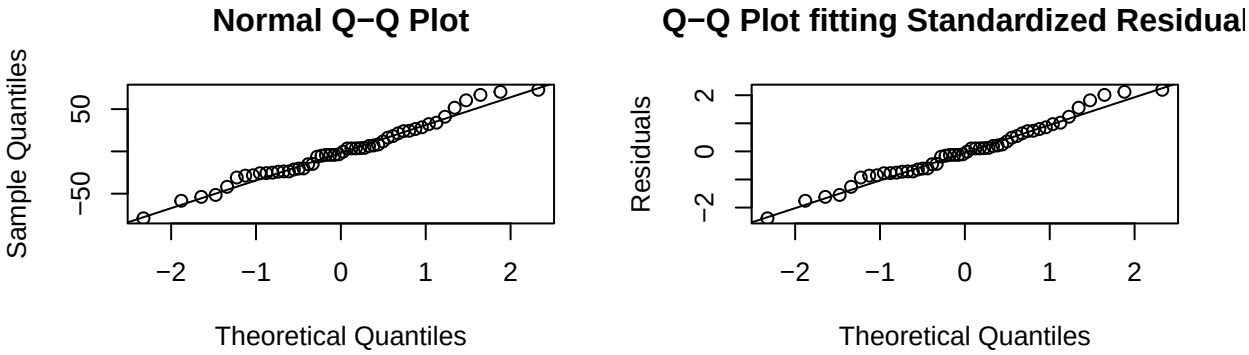



```
##
## Suggested power transformation:  5.54796
```

Looking at our plot, there doesn't appear to be anything wrong with it. The variances is pretty constant across the board for the fitted values. Also, I ran a non-constant variance score test, and the behavior of the points on our plot seem to be in line with the values from the test. There are no constant variance or homoscedasticity and residuals follow a pattern in the residual plot

b) Check the normality assumption.

```
par(mfrow=c(2,2))
qqnorm(residuals(sat.fit, ylab = "Residuals", main = "Q-Q Plot fitting Residuals"))
qqline(residuals(sat.fit))

qqnorm(scale(residuals(sat.fit)),ylab="Residuals",main="Q-Q Plot fitting Standardized Residuals")
qqline(scale(residuals(sat.fit)))
```



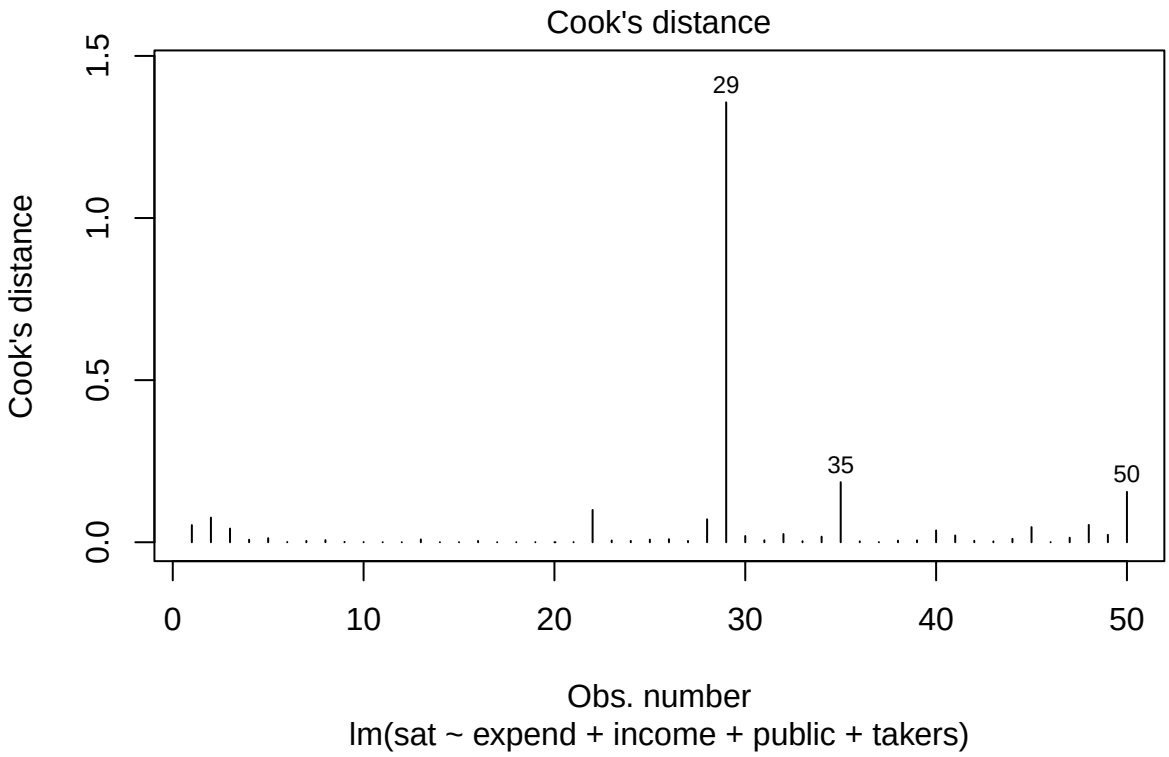

The residuals do not follow a straight line, they do not follow a normal distribution.

c) Check for large leverage points.

```
hat <- hatvalues(sat.fit)
leverage_cut <- 5 * 2 * 1 / nrow(SAT)
high_lev <- SAT[hat > leverage_cut,]
high_lev #printing the table for desired
```

```
##        state  sat takers income years public expend rank
## 6    Montana 1033      8    263 15.91   93.7  29.48 86.4
## 22 Louisiana  975      5    394 16.85   44.8  19.72 82.9
## 29    Alaska  923     31    401 15.32   96.5  50.10 79.6
```

```
cut_off<- 4/((nrow(SAT)-length(sat.fit$coefficients)-2)) #changing the cutoff for our CD plot
plot(sat.fit, which=4, cook.levels=cut_off) #CD plot
```




```
#Now, seeing our plots, select points that stand out from cook's distance
CD <-data.frame(cooks.distance(sat.fit))
names(CD) <- "cook.distance"
mean.cooks.distance <- mean(CD$cook.distance)

influential.points <- CD[CD$cook.distance > 3*mean.cooks.distance,,drop=F]
influential.points #points that stand out from cooks distance
```

```
##    cook.distance
## 29     1.3568528
## 35     0.1852304
## 50     0.1553311
```

Points with a higher leverage point than 2p/n should individually looked at, it is an indicator of high leverage

d) Check for the outliers.

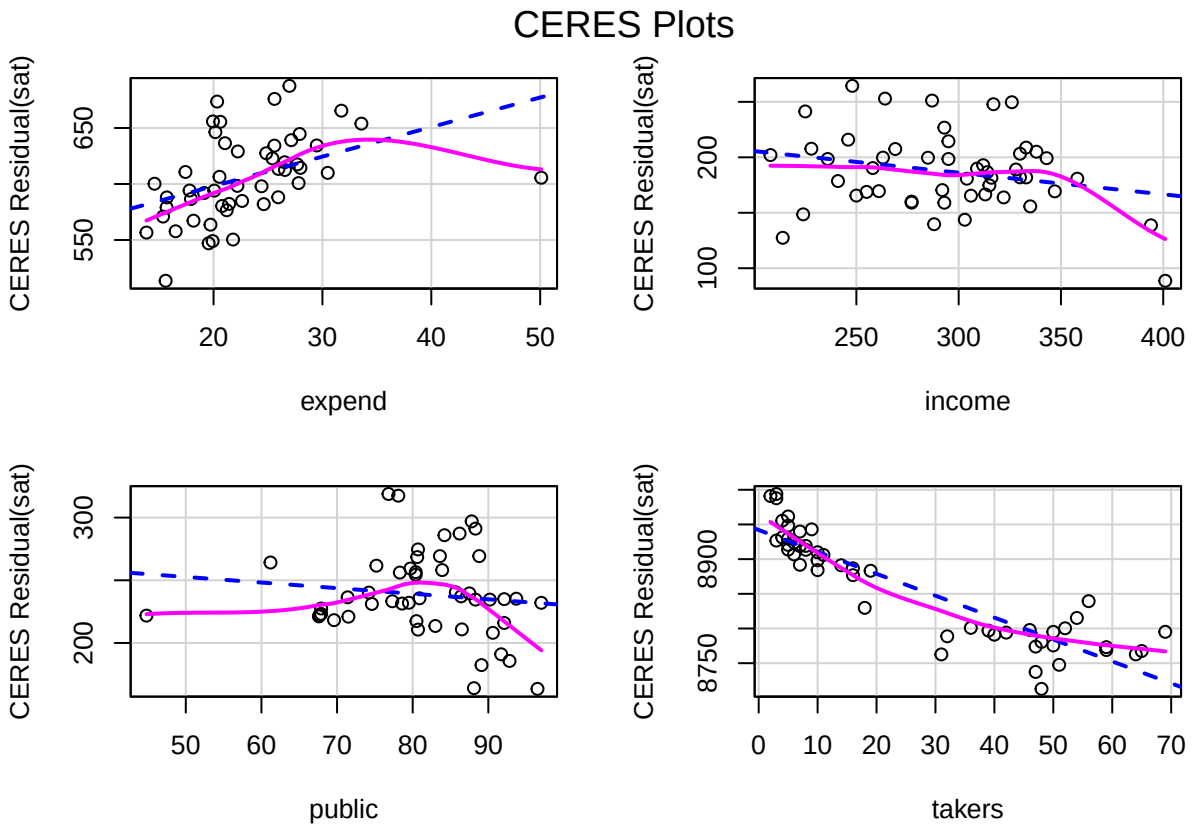
```
outlierTest(sat.fit)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##     rstudent unadjusted p-value Bonferroni p
## 29 -2.609248           0.012353      0.61764
```

No Studentized residuals with Bonferroni $p < 0.05$, so there are no outliers in this set.

e) Check the structure of the relationship between the predictors and the response.

```
ceresPlots(sat.fit) #crPlot with all predictors involved
```



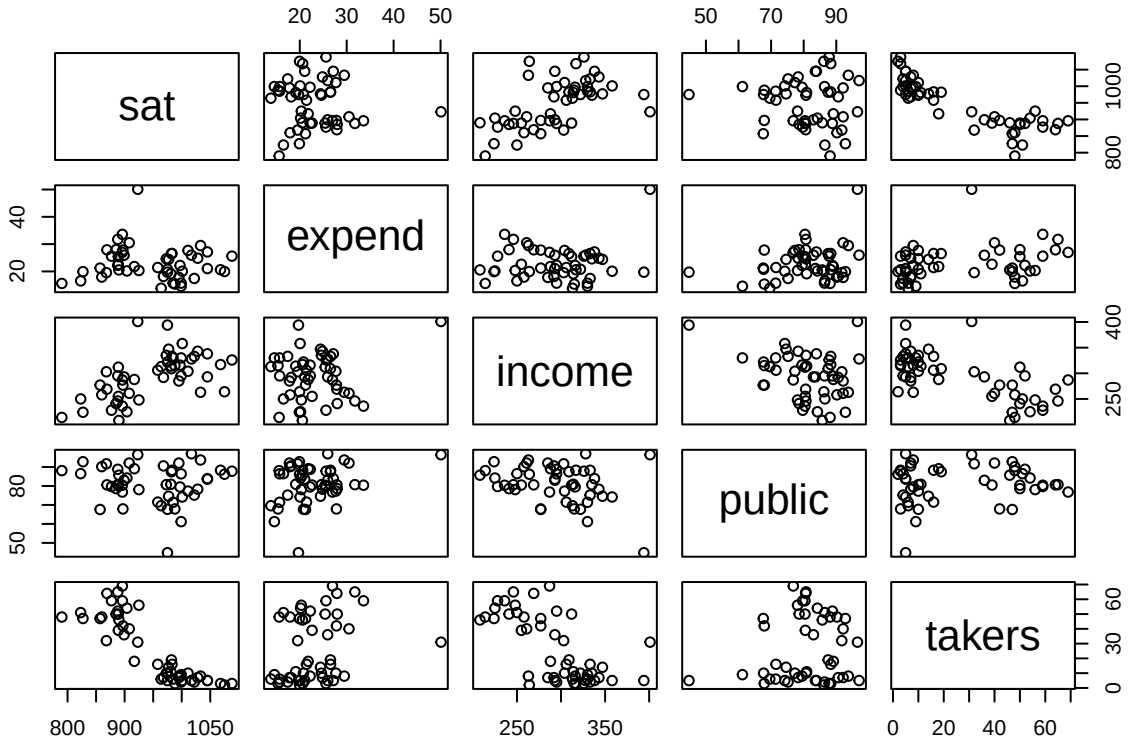

```
sat.pred <- sat.fit$fitted.values

cor(SAT$sat,sat.pred)
```

```
## [1] 0.8833184
```

88% correlation between actual and predicted values from the model.

f) Use pairs function in R to see the relationship between sat and other predictors. You can see takers appears to have a quadratic relationship with sat. Include this quadratic effect in your current model and perform the regression diagnostics. Check the structure of the relationship between the predictors and the response again.

```
pairs(SAT[, c("sat", "expend", "income", "public", "takers")])
```




```
quad_sat.fit <- lm(sat ~ expend + income + public + takers + I(takers^2), data = SAT) #include quadrati
summary(quad_sat.fit)
```

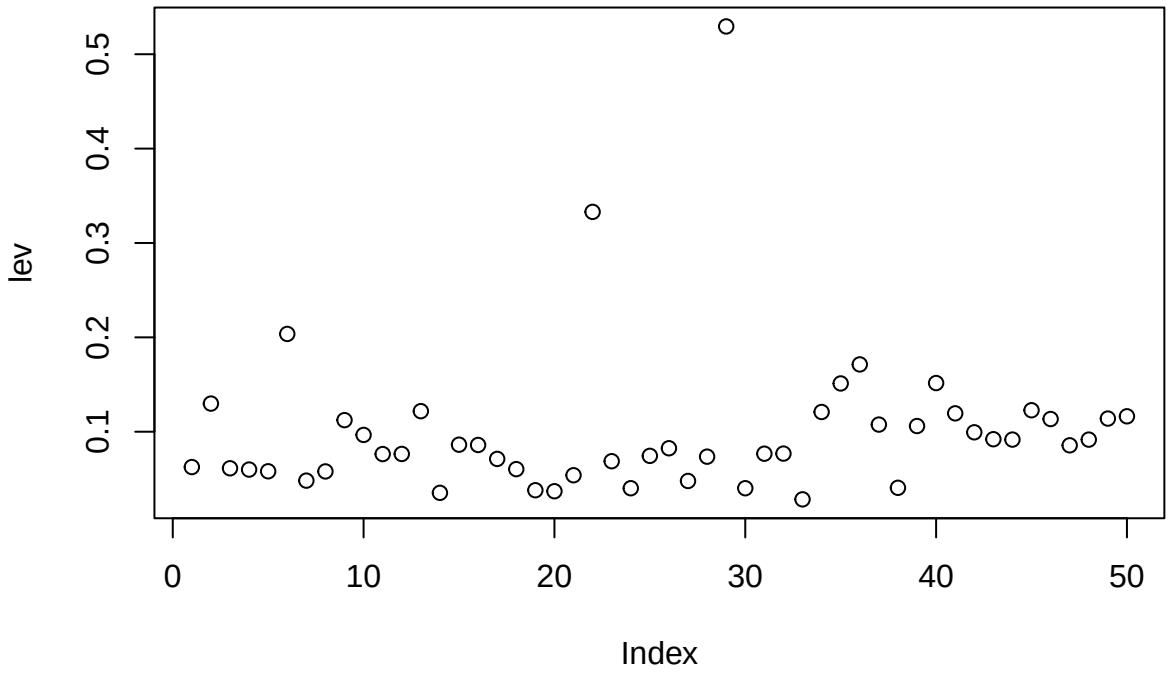
```
##
## Call:
## lm(formula = sat ~ expend + income + public + takers + I(takers^2),
##     data = SAT)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.342 -15.703  -0.402  14.122  63.631
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1063.99010   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
```

```
takers2 <- SAT$takers*SAT$takers
```

The predictors "takers" and "expend" are still very significant for this model, and "expend" appear to be more significant in this quadratic model than the default linear regression model. Notably, we see that Adjusted R^2 has gone from 76% to 87%, which is desirable.

g) Using the model in (a)-(e) (no quadratic effect), remove the large leverage points you found and perform the regression diagnostics. Check for large leverage points again.

```
lev <- hat(model.matrix(sat.fit))
plot(lev)
```



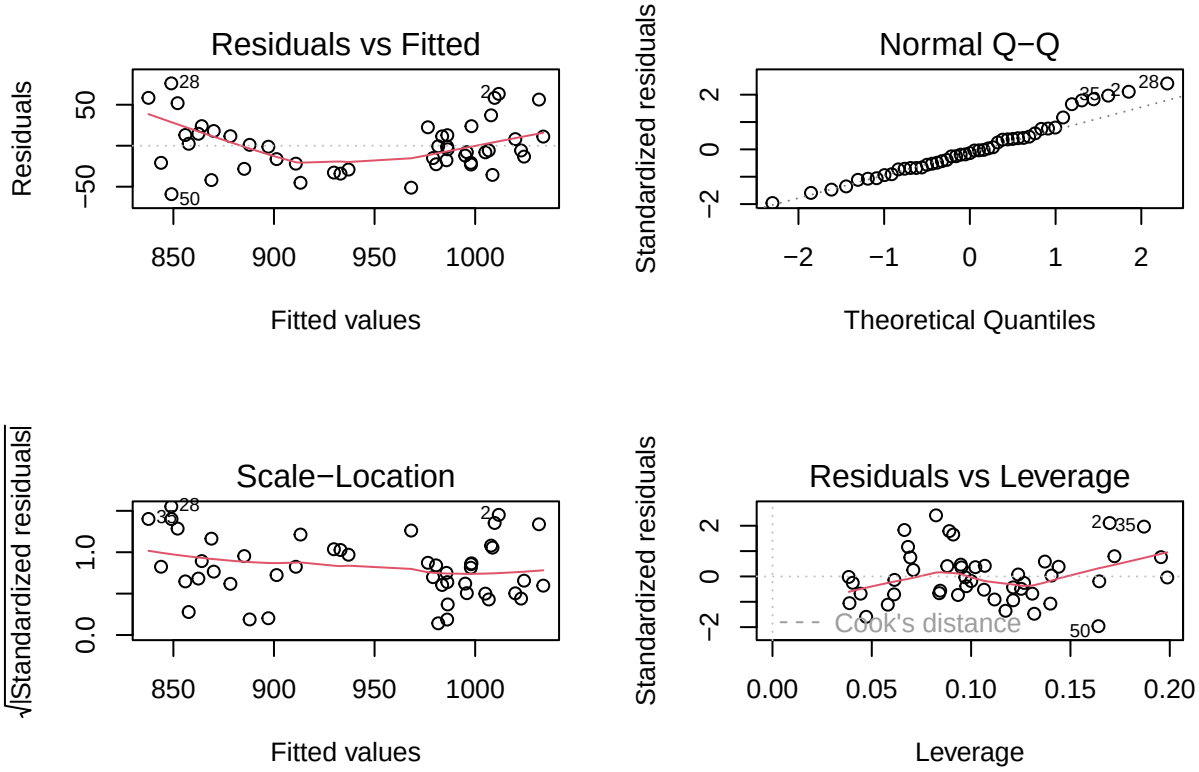

```
SAT[lev>0.2,] # obtain leverage values greater than 0.2
```

```
##        state  sat takers income years public expend rank
## 6    Montana 1033      8    263 15.91   93.7  29.48 86.4
## 22 Louisiana  975      5    394 16.85   44.8  19.72 82.9
## 29    Alaska  923     31    401 15.32   96.5  50.10 79.6
```

```
#fit new model without the 3 leverage points
SAT2 <- SAT[!(lev>0.2),]

sat.fit_no_lev <- lm(sat~expend+income+public+takers, data = SAT2)
summary(sat.fit_no_lev)
```

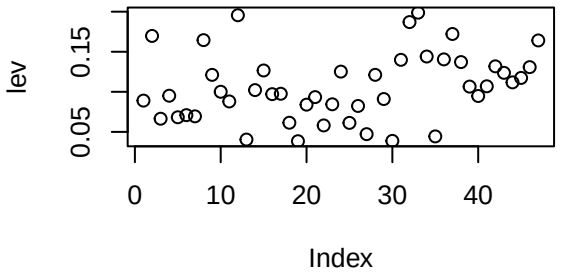
```
##
## Call:
## lm(formula = sat ~ expend + income + public + takers, data = SAT2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -59.074 -21.375  -4.355  13.817  76.033
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 974.05471   95.52283  10.197 6.25e-13 ***
## expend        4.41097    1.18906   3.710 0.000603 ***
## income       -0.01749    0.21197  -0.083 0.934631
## public       -0.45775    0.65383  -0.700 0.487725
## takers       -3.11920    0.38458  -8.111 3.92e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32.95 on 42 degrees of freedom
## Multiple R-squared:  0.8078, Adjusted R-squared:  0.7895
## F-statistic: 44.14 on 4 and 42 DF,  p-value: 1.626e-14
```

```
par(mfrow=c(2,2))
plot(sat.fit_no_lev)
```



```
lev <-  hat(model.matrix(sat.fit_no_lev))
plot(lev)
# obtain leverage values greater than 0.2
SAT2[lev>0.2,]
```

```
## [1] state  sat    takers income years  public expend rank
## <0 rows> (or 0-length row.names)
```




There are no leverage points in the new model ran, this caused a slight increased in adjusted R^2 from 76% to 78%.

h) Comment on which model we should use. The model in (f) or the model in (a)-(e) with the removal of large leverage points that you did in (g)?

Solution: I think it would be best if the model with the quadratic effect applied in (f) was used. The addition of the quadratic effect caused a much greater increase in adjusted R^2 than the removal of 3 large leverage points did. The model in (f) improved model performance much greater than the model in (g).

## Question 2:

It was stated in the text that classifying an observation to the class for which (4.17) is largest is equivalent to classifying an observation to the class for which (4.18) is largest. Prove that this is the case. In other words, under the assumption that the observations in the kth class are drawn from a N(mu_k,sigma^2) distribution, the Bayes classifier assigns an observation to the class for which the discriminant function is maximized.

Solution: ON SEPARATE PAPER (Done)

## Question 3:

We now examine the differences between LDA and QDA.

a) If the Bayes decision boundary is linear, do we expect LDA or QDA to perform better on the training set? On the test set?

Solution: The QDA on the training set. This is because it is more flexible than LDA on training. On the test set, we use the LDA. This is due to the bias-variance trade off, and QDA has a higher chance to overfit a training set.

b) If the Bayes decision boundary is non-linear, do we expect LDA or QDA to perform better on the training set? On the test set?

Solution: For this, QDA on both works best. LDA isn't great working with assuming things are non-linear.

c) In general, as the sample size n increases, do we expect the test prediction accuracy of QDA relative to LDA to improve, decline, or be unchanged? Why?

Solution: We expect it to either increase or not change at all. This is due to earlier fact that QDA is more likely to overfit training data.

d) True or False: Even if the Bayes decision boundary for a given problem is linear, we will probably achieve a superior test error rate using QDA rather than LDA because QDA is flexible enough to model a linear decision boundary. Justify your answer.

Solution: False. Since QDA doesn't make assumptions on the decision boundary because it is a flexible analysis. It only influences bias-variance trade off slightly.

## Question 4:

Suppose we collect data for a group of students in a statistics class with variables x_1 = hours studied, x_2 = undergrad GPA, and y = receive an A. We fit a logistic regression and produce estimated coefficient, Beta^0 = -6, Beta^1 = 0.05, Beta^2 = 1.

a) Estimate the probability that a student who studies for 40 h and has an undergrad GPA of 3.5 gets an A in the class.

```
probability_4 <- function(a,b){
  c <- exp(-6 + 0.05*a + 1*b);
  return( round(c/(1+c),2))
}
#create a logit function that evaluates what we desire
probability_4(40,3.5) #input the values given into our function
```

```
## [1] 0.38
```

b) How many hours would the student in part (a) need to study to have a 50 % chance of getting an A in the class?

```
hours <- seq(40,60,1) #40 hours to 60 hours, 1 sequence
probability_4_2 <- mapply(hours, 3.5, FUN=probability_4) #GPA of 3.5 (A)
names(probability_4_2) <- paste(hours)
probability_4_2
```

```
##   40   41   42   43   44   45   46   47   48   49   50   51   52   53   54   55
## 0.38 0.39 0.40 0.41 0.43 0.44 0.45 0.46 0.48 0.49 0.50 0.51 0.52 0.54 0.55 0.56
##   56   57   58   59   60
## 0.57 0.59 0.60 0.61 0.62
```

In order to get a 50%, you need to study at least 50 hours a week.

## Question 5:

This problem has to do with odds.

a) On average, what fraction of people with an odds of 0.37 of defaulting on their credit card payment will in fact default?

```
#can be seen with the function p(x)/1-p(x)
0.37/(1+0.37) #odds of card being defaulted = true
```

```
## [1] 0.270073
```

b) Suppose that an individual has a 16% chance of defaulting on her credit card payment. What are the odds that she will default?

```
#can be seen with the function p(x)/1-p(x)
.16/(1-.16) #odds of defaulting
```

```
## [1] 0.1904762
```

## Question 6:

Download the csv file arthritis.csv. These data from Koch & Edwards (1988) from a double-blind clinical trial investigating a new treatment for rheumatoid arthritis.

a) Fit a logistic regression with the Improved as the response and Treatment, Sex and Age as predictors

```
#with logistic regression, use glm.
treatment.factor <- as.factor(arthritis$Treatment) # == "Placebo"
sex.factor <- as.factor(arthritis$Sex) # == "Female"
improved.factor <- as.factor(arthritis$Improved) # == "No"
arthritis.fit <- glm(improved.factor ~ Treatment + Sex + Age, data = arthritis, family = binomial)
summary(arthritis.fit)
```

```
##
## Call:
## glm(formula = improved.factor ~ 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
```

b) Use log odds to interpret the coefficients Beta^1, Beta^2, and Beta^3.

```
log(exp(coef(arthritis.fit)))
```

```
##      (Intercept) TreatmentTreated          SexMale              Age
##      -3.01546086       1.75980413      -1.48783119       0.04874654
```

The interpretation of log odds ratio: The intercept here, with all predictors being involved, that the chances of being improved = yes with these factors is -3.01546086. Increasing "Improved" by 1 unit lowers the chances of improving from arthritis by a factor of -3.01546086.

c) Use odds to interpret the coefficients Beta^1, Beta^2, and Beta^3.

```
exp(coef(arthritis.fit))
```

```
##      (Intercept) TreatmentTreated          SexMale              Age
##       0.04902324       5.81129902       0.22586198       1.04995419
```

The interpretation of odds ratio: The intercept here, with all predictors being involved, that the chances of being improved = yes with these factors is 0.04902324. Increasing "Improved" by 1 unit raises the chances of improving from arthritis by a factor of 0.0490232.

d) Construct 95% confidence intervals for the coefficients.

```
confint(arthritis.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
```

e) Add the interaction Sex:Age in your model. For this model, for a one year increase in age, how much does the log odds of some improvement (versus none) increase? Explain it separately with respect to Sex.

```
arthritis3.fit <- glm(improved.factor ~ Treatment + Sex + Age + Sex:Age, data = arthritis, family = bin
#summary(arthritis3.fit)
log(exp(coef(arthritis3.fit)))
```

```
##      (Intercept) TreatmentTreated          SexMale              Age
##      -4.55476656       1.79704618       2.75065541       0.07734209
##      SexMale:Age
##      -0.07944668
```

For males, a one year increase in age increases the log odds of some improvement (versus none) by the amount given by -0.07944668

For females, a one year increase in age increases the log odds of some improvement (versus none) by the amount given by 0.7734209. ## Question 7: This question should be answered using the "Weekly" data set, which is part of the ISLR2 package. This data is similar in nature to the Smarket data from this chapter's lab, except that it contains 1, 089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

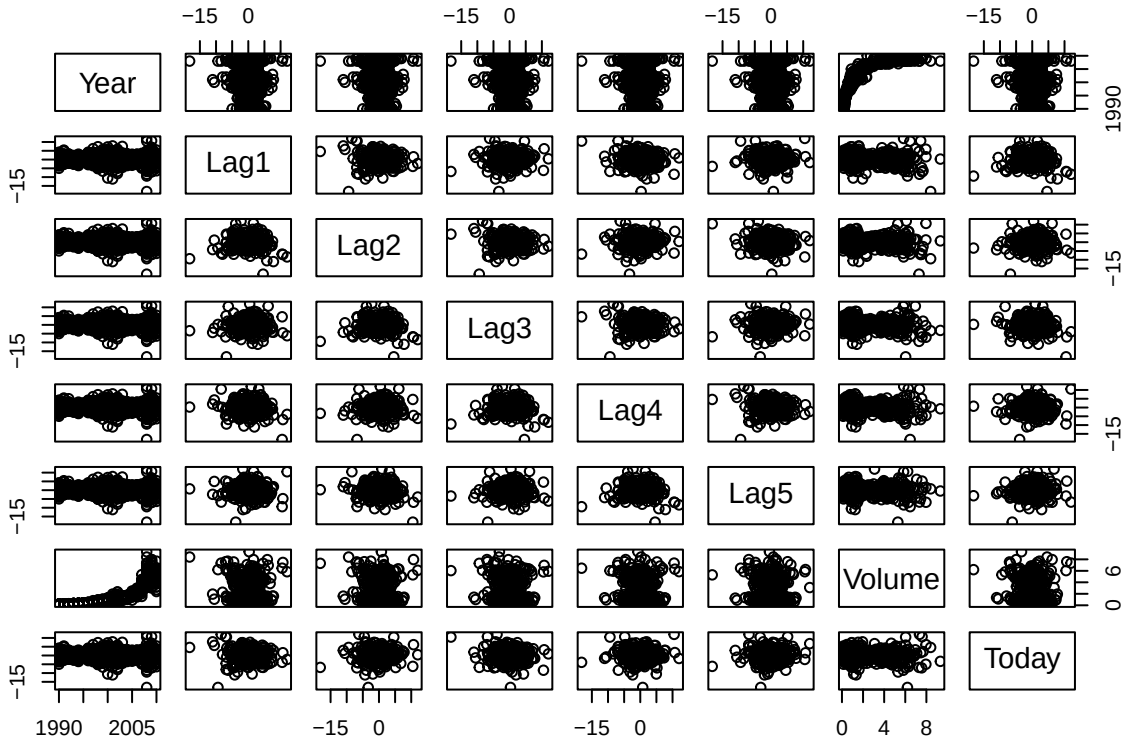
a) Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?

```
summary(Weekly)
```

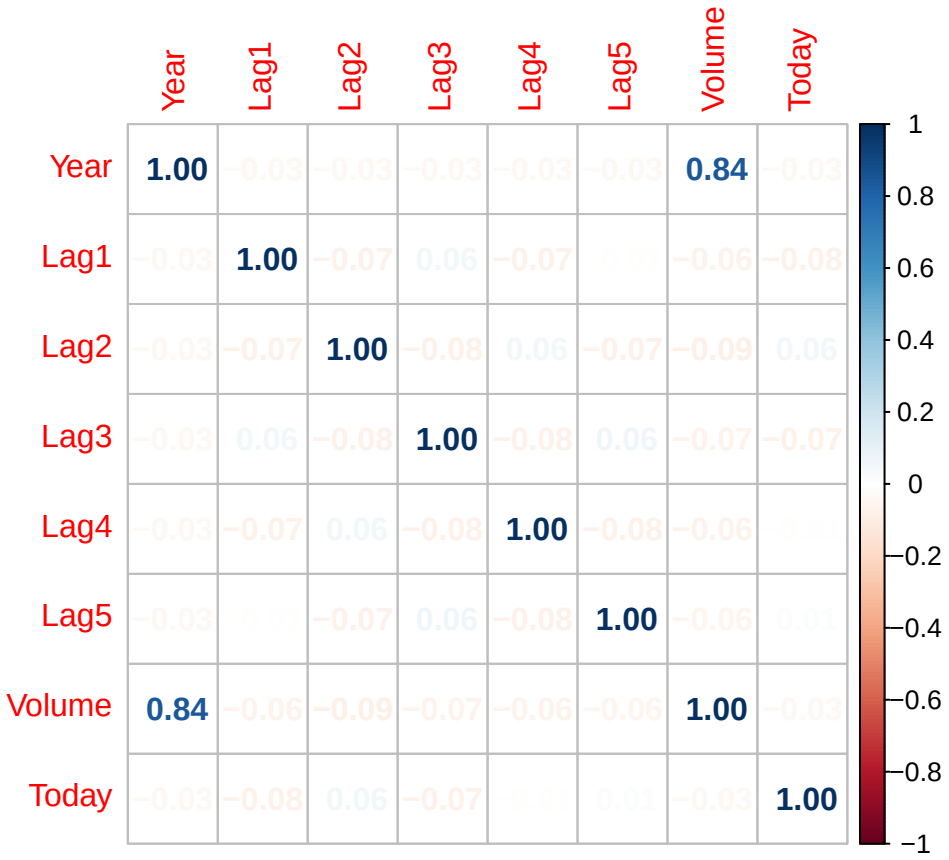
```
##       Year           Lag1               Lag2               Lag3
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410
##  Mean   :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[,-9])
```




```
corrplot(cor(Weekly[,-9]),method = "number")
```




```
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
```

From what we can observe from our correlation plot and scatter plot matrix, it appears that this data set has a high correlation between the variables 'year' and a volume of 0.84. Lags look like they don't have much of a correlation with anything. Also, it seems like volume has been increasing during the years of 1990 through around the 2010.

b) Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

```
weekly.fit <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, family = "binomial", data = We
summary(weekly.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
```

All Lags are of negative coefficient, except Lag2. Lag2 seems to be the only statistically significant of all of predictors when it comes to predicting Direction. We fail to reject the null hypothesis, where Beta = 0 for all other variables besides Lag2. With a coefficient of 0.05844, Lag2 has an effect that means an increase in 1 in Lag2 would represent an increase of exp(0.058), which equals 1.06 in odds of Direction increasing.

c) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

```
weekly.probs = predict(weekly.fit, type = "response")
weekly.preds = rep("Down", length(weekly.probs))
weekly.preds[weekly.probs > 0.5] = "Up"
table(weekly.preds, Weekly$Direction)
```

```
##
## weekly.preds Down  Up
##         Down   54  48
##         Up    430 557
```

We use the math: $(54 + 557)/(54 + 48 + 430 + 557)$ to find percentage of current predictions. This equates to 0.5611. So, this means that the model predicted the weekly market trend accurately about 56% of the time. Furthermore, if separate the matrix and predict the "Up" and "Down" trends separately, we get: $557/48 + 557 = 0.92$ for "Up" and $54/430 + 54 = 0.1115$ for "Down".

d) Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag2 as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 and 2010).

```
weekly.train = Weekly[Weekly$Year < 2009,] # make training set
weekly.test = Weekly[Weekly$Year > 2008,] # make test set
weekly.fit.trained <- glm(Direction ~ Lag2, data = weekly.train, family = "binomial")
summary(weekly.fit.trained)
```

```
##
## Call:
## glm(formula = Direction ~ Lag2, family = "binomial", data = weekly.train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.536  -1.264   1.021   1.091   1.368
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.20326    0.06428   3.162  0.00157 **
## Lag2         0.05810    0.02870   2.024  0.04298 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1354.7  on 984  degrees of freedom
## Residual deviance: 1350.5  on 983  degrees of freedom
## AIC: 1354.5
##
## Number of Fisher Scoring iterations: 4
```

```
## Confusion matrix below, follow similar structure from previous part
test.probs = predict(weekly.fit.trained, type = "response", newdata = weekly.test)
test.dir = Weekly$Direction[Weekly$Year > 2008]
test.preds = rep("Down", length(test.probs))
test.preds[test.probs>0.5] = "Up"
table(test.preds,test.dir)
```

```
##           test.dir
## test.preds Down Up
##       Down    9  5
##       Up     34 56
```

e) Repeat (d) using LDA.

```r
#Should be relatively the same. Should be the same table outcome too.
weekly.lda <- lda(Direction ~ Lag2, data = weekly.train)
weekly.lda
```

```
## Call:
## lda(Direction ~ Lag2, data = weekly.train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
##
## Coefficients of linear discriminants:
##            LD1
## Lag2 0.4414162
```

```r
weekly.lda.pred <- predict(weekly.lda, newdata = weekly.test, type = "response")
lda.class <- weekly.lda.pred$class
table(lda.class, weekly.test$Direction)
```

```
##
## lda.class Down Up
##      Down    9  5
##      Up     34 56
```

f) Repeat (d) using QDA.

```r
#copy paste and make it QDA
weekly.qda <- qda(Direction ~ Lag2, data = weekly.train)
weekly.qda
```

```
## Call:
## qda(Direction ~ Lag2, data = weekly.train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
```

```r
weekly.qda.pred <- predict(weekly.qda, newdata = weekly.test, type = "response")
qda.class <- weekly.qda.pred$class
table(qda.class, weekly.test$Direction)
```

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

g) Repeat (d) using KNN with K = 1.

```
#need to set seed when using KNN
set.seed(0)
#refer to KNN example code from homework 1
train.x = cbind(weekly.train$Lag2)
test.x = cbind(weekly.test$Lag2)
train.x = cbind(weekly.train$Direction)
knn.pred = knn(train.x, test.x, train.x, k=1) #set K to 1
table(knn.pred, weekly.test$Direction)
```

```
##
## knn.pred Down Up
##        1   28 39
##        2   15 22
```

The model is good at predicting T Positives, not very good at predicting T Negatives. (T = True)

i) Which of these methods appears to provide the best results on this data?

As far as which methods provide the best results, it appears logistic regression and LDA. Both of their rates are about ~60%.

j) Experiment with different combinations of predictors, including possible transformations and interactions, for each of the methods. Report the variables, method, and associated confusion matrix that appears to provide the best results on the held out data. Note that you should also experiment with values for K in the KNN classifier.

```
#try a new qda model maybe?
weekly.qda2 <- qda(Direction~Lag1 + Lag2 + Lag4, data= weekly.train)
weekly.qda2
```

```
## Call:
## qda(Direction ~ Lag1 + Lag2 + Lag4, data = weekly.train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##              Lag1        Lag2       Lag4
## Down  0.289444444 -0.03568254 0.15925624
## Up   -0.009213235  0.26036581 0.09220956
```

```r
#use former code syntax with new qda
weekly.qda.pred2 <- predict(weekly.qda2, newdata = weekly.test, type = "response")
qda.class2 <- weekly.qda.pred2$class
table(qda.class2, weekly.test$Direction)
```

```
##
## qda.class2 Down Up
##       Down    9 20
##       Up     34 41
```

```r
#try the same with a new lda
weekly.lda2 <- lda(Direction~Lag1 + Lag2 + Lag4, data= weekly.train)
weekly.lda2
```

```
## Call:
## lda(Direction ~ Lag1 + Lag2 + Lag4, data = weekly.train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##               Lag1        Lag2       Lag4
## Down  0.289444444 -0.03568254 0.15925624
## Up   -0.009213235  0.26036581 0.09220956
##
## Coefficients of linear discriminants:
##             LD1
## Lag1 -0.2984478
## Lag2  0.2960224
## Lag4 -0.1113485
```

```r
weekly.lda.pred2 <- predict(weekly.lda2, newdata = weekly.test, type = "response")
lda.class2 <- weekly.lda.pred2$class
table(lda.class2, weekly.test$Direction)
```

```
##
## lda.class2 Down Up
##       Down    9  7
##       Up     34 54
```

Neither models are significantly better than the models with Lag2 as its only predictor.

## Question 8:

In this problem, you will develop a model to predict whether a given car gets high or low gas mileage based on the Auto data set.

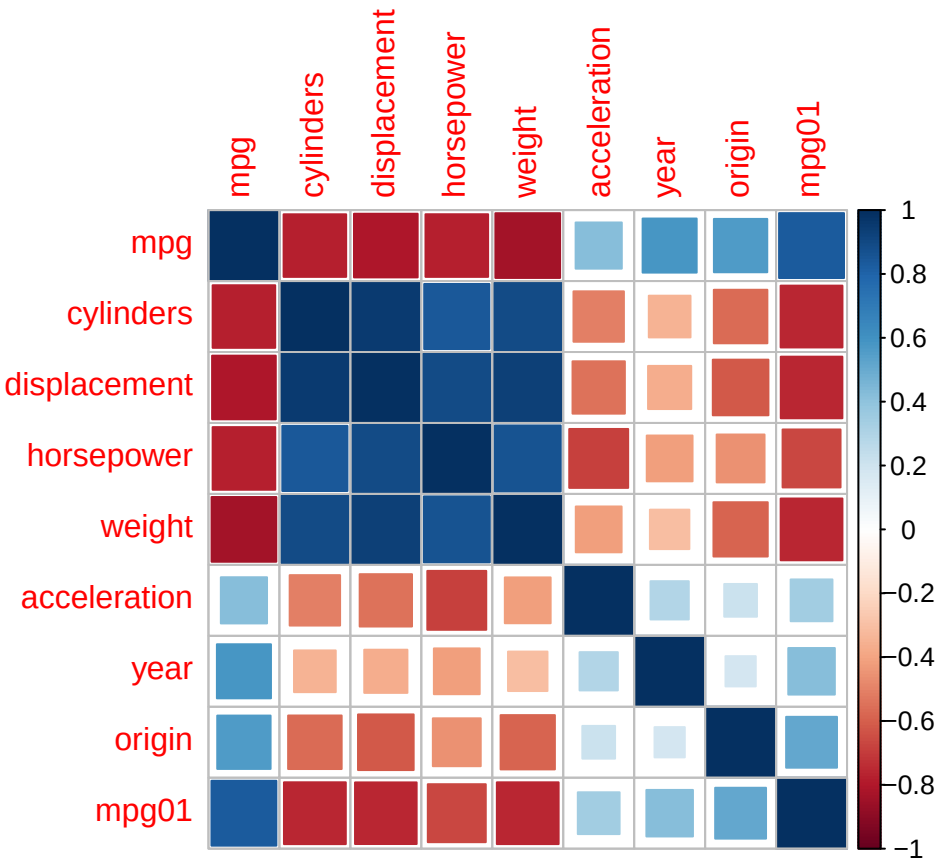
a) Create a binary variable, mpg01, that contains a 1 if mpg contains a value above its median, and a 0 if mpg contains a value below its median. You can compute the median using the median() function. Note you may find it helpful to use the data.frame() function to create a single data set containing both mpg01 and the other Auto variables.

```
data("Auto")
median_mpg <- median(Auto$mpg)
mpg01 <- rep(0, length(Auto$mpg))
mpg01[Auto$mpg > median_mpg] = 1
Auto1 <- data.frame(Auto, mpg01)
summary(Auto1)
```

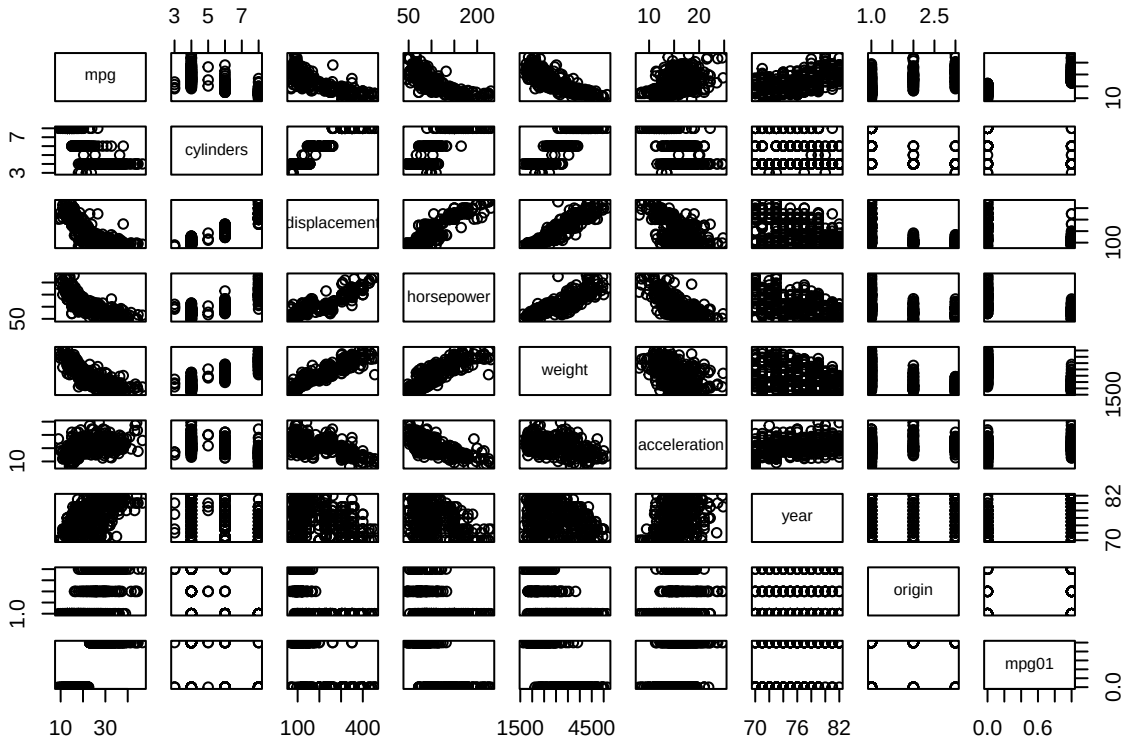
```
##       mpg          cylinders      displacement     horsepower        weight
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225
##  Median :22.75   Median :4.000   Median :151.0   Median : 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
##      mpg01
##  Min.   :0.0
##  1st Qu.:0.0
##  Median :0.5
##  Mean   :0.5
##  3rd Qu.:1.0
##  Max.   :1.0
##
```

b) Explore the data graphically in order to investigate the association between mpg01 and the other features. Which of the other features seem most likely to be useful in predicting mpg01? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.

```
corrplot(cor(Auto1[,-9]), method = "square")
```




```
pairs(Auto1[,-9])
```




It appears that Cylinders, Displacement, and Weight correlate strongly with mpg01; a strong negative correlation. Furthermore, horsepower and origin have a slight correlation with mpg01. So we will include these 5 in our model.

c) Split the data into a training set and a test set.

```
attach(Auto1)
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     mpg01
```

```
## The following object is masked from package:ggplot2:
##
##     mpg
```

```
t <- (year %% 2 == 0)
auto.train <- Auto1[t,]
auto.test <- Auto1[-t,]
```

d) Perform LDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

```
auto.lda <- lda(mpg01 ~ cylinders + displacement + weight + origin + horsepower, data = auto.train)
auto.lda
```

```
## Call:
## lda(mpg01 ~ cylinders + displacement + weight + origin + horsepower,
##     data = auto.train)
##
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##   cylinders displacement   weight   origin horsepower
## 0  6.812500     271.7396 3604.823 1.166667  133.14583
## 1  4.070175     111.6623 2314.763 2.035088   77.92105
##
## Coefficients of linear discriminants:
##                       LD1
## cylinders    -0.696710813
## displacement  0.001567657
## weight       -0.001087183
## origin        0.130758405
## horsepower    0.004275955
```

```
auto.lda.pred <- predict(auto.lda, auto.test)
table(auto.lda.pred$class, auto.test$mpg01) #confusion matrix
```

```
##
##       0   1
##   0 169  15
##   1  26 181
```

```
mean(auto.lda.pred$class != auto.test$mpg01) #test error
```

```
## [1] 0.1048593
```

From the LDA model, the test error rate equates to be 10.48%.

e) Perform QDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

```
auto.qda <- qda(mpg01 ~ cylinders + displacement + weight + origin + horsepower, data = auto.train)
auto.qda
```

```
## Call:
## qda(mpg01 ~ cylinders + displacement + weight + origin + horsepower,
##     data = auto.train)
##
## Prior probabilities of groups:
##         0         1
```

```
## 0.4571429 0.5428571
##
## Group means:
##   cylinders displacement   weight   origin horsepower
## 0  6.812500     271.7396 3604.823 1.166667  133.14583
## 1  4.070175     111.6623 2314.763 2.035088   77.92105
```

```
auto.qda.pred <- predict(auto.qda, auto.test)
table(auto.qda.pred$class, auto.test$mpg01) #confusion matrix
```

```
##
##       0   1
##   0 174  19
##   1  21 177
```

```
mean(auto.qda.pred$class != auto.test$mpg01) #test error
```

```
## [1] 0.1023018
```

Using the QDA method, the model resulted in a test error rate of 10.23%. Not too far off from the LDA method.

f) Perform logistic regression on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

```
auto.fit <- glm(mpg01 ~ horsepower + cylinders + displacement + origin, data = auto.train, family = "bi
summary(auto.fit)
```

```
##
## Call:
## glm(formula = mpg01 ~ horsepower + cylinders + displacement +
##     origin, family = "binomial", data = auto.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5331  -0.0251   0.1314   0.3719   2.6232
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  13.10623    2.58726   5.066 4.07e-07 ***
## horsepower   -0.07067    0.02460  -2.873  0.00406 **
## cylinders    -0.92497    0.65347  -1.415  0.15693
## displacement -0.01354    0.01468  -0.922  0.35646
## origin        0.21268    0.43511   0.489  0.62499
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 289.577  on 209  degrees of freedom
## Residual deviance:  89.635  on 205  degrees of freedom
```

```
## AIC: 99.635
##
## Number of Fisher Scoring iterations: 7
```

```
auto.probs = predict(auto.fit, auto.test, type = "response")
auto.pred = rep(0, length(auto.probs))
auto.pred[auto.probs > 0.5] = 1
table(auto.pred, auto.test$mpg01) #confusion matrix
```

```
##
## auto.pred   0   1
##         0 172  13
##         1  23 183
```

```
mean(auto.pred != auto.test$mpg01)
```

```
## [1] 0.09207161
```

Using a logistic regression model resulted in a test error rate of 9.2%.

h) Perform KNN on the training data, with several values of K, in order to predict mpg01. Use only the variables that seemed most associated with mpg01 in (b). What test errors do you obtain? Which value of K seems to perform the best on this data set?

```
train.X <-  cbind(auto.train$cylinders, auto.train$weight, auto.train$displacement, auto.train$horsepow
test.X <-  cbind(auto.test$cylinders, auto.test$weight, auto.test$displacement, auto.test$horsepower, a
```

Below, we will try 4 different KNN values and see what we get.

```
set.seed(1)
auto.knn.pred <-  knn(train.X, test.X, auto.train$mpg01, k = 1)
mean(auto.knn.pred != auto.test$mpg01)
```

```
## [1] 0.07161125
```

```
auto.knn.pred2 <-  knn(train.X, test.X, auto.train$mpg01, k = 7)
mean(auto.knn.pred2 != auto.test$mpg01)
```

```
## [1] 0.1202046
```

```
auto.knn.pred3 <-  knn(train.X, test.X, auto.train$mpg01, k = 12)
mean(auto.knn.pred3 != auto.test$mpg01)
```

```
## [1] 0.1278772
```

```
auto.knn.pred4 <-  knn(train.X, test.X, auto.train$mpg01, k = 100)
mean(auto.knn.pred4 != auto.test$mpg01)
```

```
## [1] 0.1176471
```

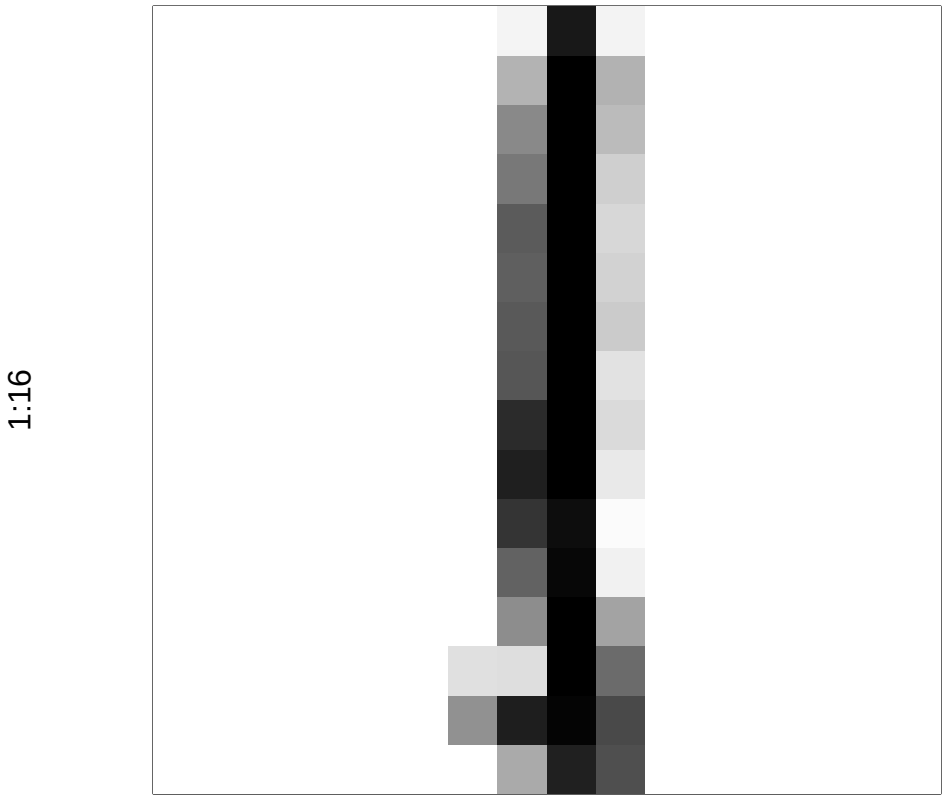
K = 1 seems to be likely that it'll have the lowest error rate of 7.16%. Typically, as K increases, the error rate also increases for the model.

## Question 9:

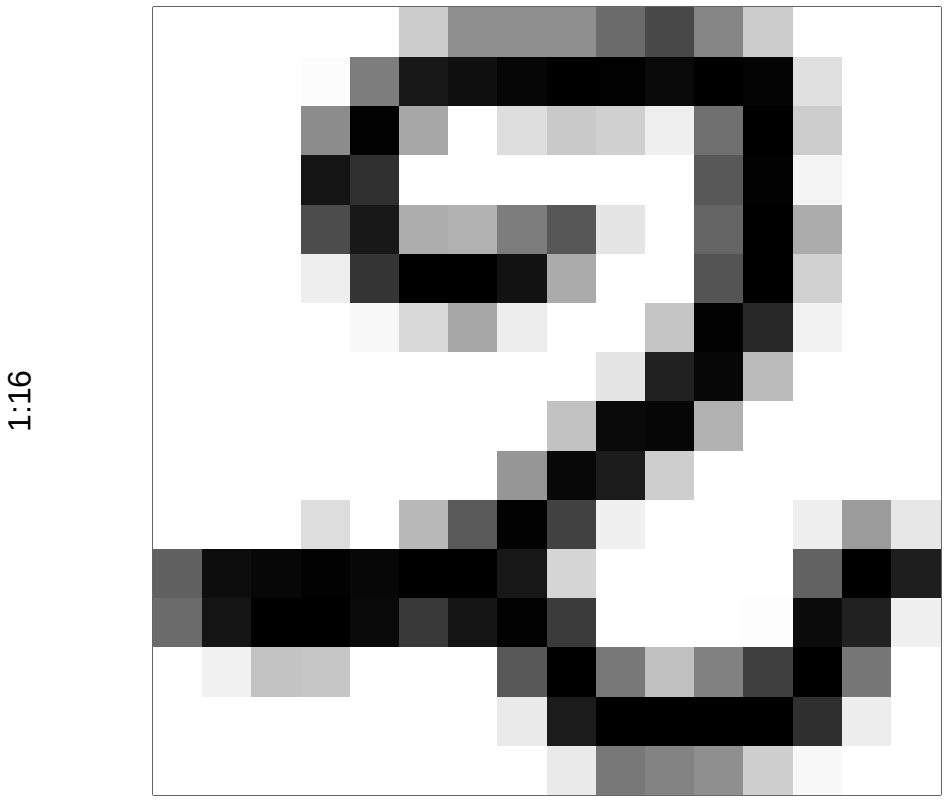
Download the csv files zipcode_train.csv and zipcode_test.csv. Load and visualize the files using the following command lines. In the dataset, you have 1736 training images and 462 test images, where each image is a handwritten digit and it can be either 1 or 2. The description of columns is below.

a) Perform logistic regression, LDA, and KNN models.

```
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)
COLORS <- c("white", "black")
CUSTOM_COLORS <- colorRampPalette(colors = COLORS)
vis <- function(i){
  par(pty = "s", mar = c(1, 1, 1, 1), xaxt = "n", yaxt = "n")
  z <- matrix(as.numeric(train.dat[i,1:256]), 16, 16)
  image(1:16,1:16,z[,16:1], col = CUSTOM_COLORS(256))
}
vis(2) # hand written 1 (from 1 to 1005)
```




```
vis(1500) # hand written 2 (from 1006 to 1736)
```




```
#zip.fit <- glm(data = train.dat, Y ~ ., family = binomial)
#glm.pred <- ifelse(predict(zip.fit, newdata = test.dat) > 0.5, 1)
#zip.matrix <- table(test.dat$Y, glm.pred)
#test.acc <- sum(diag(zip.matrix)) / sum(zip.matrix)
#print(1-test.acc)

#what am I doing wrong with this code?
```

```
zip.lda <- lda(Y ~ . - p16 - p32, data = train.dat)
lda.pred <- predict(zip.lda, newdata = test.dat)
lda.c <- lda.pred$class
lda.matrix <- table(test.dat$Y, lda.c)
lda.acc <- sum(diag(lda.matrix)) / sum(lda.matrix)
1 - lda.acc
```

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

```
#zip.knn <- knn(train = train.dat, test = test.dat, cl = train.dat$Y, k = 1)
#knn.matrix <- table(zip.knn, test.dat$Y)
#knn.acc <- sum(diag(knn.matrix)) / sum(knn.matrix)
#1-knn.acc

#what am I doing wrong with this code?
```

b) Using the test.dat, which of these methods appears to provide the best results on the test data?

Unable to conjure up actual evidence from the code, but I can guess that knn will have best results with this classification data set. I assume it will have the lowest error

## References

https://www.statology.org/test-for-normality-in-r/ https://statisticsglobe.com/calculate-leverage-statistics-r https://statsandr.com/blog/outliers-detection-in-r/ https://rdrr.io/cran/car/man/ncvTest.html https://cran.r-project.org/web/packages/olsrr/vignettes/influence_measures.html https://www.statology.org/quadratic-regression-r/ https://stats.oarc.ucla.edu/r/dae/logit-regression/ https://stackoverflow.com/questions/47546658/logistic-regression-on-factor-error-in-evalfamilyinitialize-y-values-must https://stackoverflow.com/questions/41384075/r-calculate-and-interpret-odds-ratio-in-logistic-regression https://cran.r-project.org/web/packages/corrplot/vignettes/corrplot-intro.html#:~:text=R%20package%20corrplot%20provides%20
https://www.statology.org/confusion-matrix-in-r/ https://datascienceplus.com/how-to-perform-logistic-regression-lda-qda-in-r/