Importing college.txt and setup

```r
college = read.table("college.txt")
college$Elite=as.factor(college$Elite)
college$Private=as.factor(college$Private)
attach(college)
summary(college)
```

```
##  Private        Apps           Accept          Enroll       Top10perc    
##  No :212   Min.   :   81   Min.   :   72   Min.   :  35   Min.   : 1.00  
##  Yes:565   1st Qu.:  776   1st Qu.:  604   1st Qu.: 242   1st Qu.:15.00  
##            Median : 1558   Median : 1110   Median : 434   Median :23.00  
##            Mean   : 3002   Mean   : 2019   Mean   : 780   Mean   :27.56  
##            3rd Qu.: 3624   3rd Qu.: 2424   3rd Qu.: 902   3rd Qu.:35.00  
##            Max.   :48094   Max.   :26330   Max.   :6392   Max.   :96.00  
##    Top25perc      F.Undergrad     P.Undergrad         Outstate    
##  Min.   :  9.0   Min.   :  139   Min.   :    1.0   Min.   : 2340  
##  1st Qu.: 41.0   1st Qu.:  992   1st Qu.:   95.0   1st Qu.: 7320  
##  Median : 54.0   Median : 1707   Median :  353.0   Median : 9990  
##  Mean   : 55.8   Mean   : 3700   Mean   :  855.3   Mean   :10441  
##  3rd Qu.: 69.0   3rd Qu.: 4005   3rd Qu.:  967.0   3rd Qu.:12925  
##  Max.   :100.0   Max.   :31643   Max.   :21836.0   Max.   :21700  
##    Room.Board       Books           Personal         PhD        
##  Min.   :1780   Min.   :  96.0   Min.   : 250   Min.   :  8.00  
##  1st Qu.:3597   1st Qu.: 470.0   1st Qu.: 850   1st Qu.: 62.00  
##  Median :4200   Median : 500.0   Median :1200   Median : 75.00  
##  Mean   :4358   Mean   : 549.4   Mean   :1341   Mean   : 72.66  
##  3rd Qu.:5050   3rd Qu.: 600.0   3rd Qu.:1700   3rd Qu.: 85.00  
##  Max.   :8124   Max.   :2340.0   Max.   :6800   Max.   :103.00  
##     Terminal       S.F.Ratio      perc.alumni        Expend     
##  Min.   : 24.0   Min.   : 2.50   Min.   : 0.00   Min.   : 3186  
##  1st Qu.: 71.0   1st Qu.:11.50   1st Qu.:13.00   1st Qu.: 6751  
##  Median : 82.0   Median :13.60   Median :21.00   Median : 8377  
##  Mean   : 79.7   Mean   :14.09   Mean   :22.74   Mean   : 9660  
##  3rd Qu.: 92.0   3rd Qu.:16.50   3rd Qu.:31.00   3rd Qu.:10830  
##  Max.   :100.0   Max.   :39.80   Max.   :64.00   Max.   :56233  
##    Grad.Rate       Elite    
##  Min.   : 10.00   No :699  
##  1st Qu.: 53.00   Yes: 78  
##  Median : 65.00            
##  Mean   : 65.46            
##  3rd Qu.: 78.00            
##  Max.   :118.00            
```

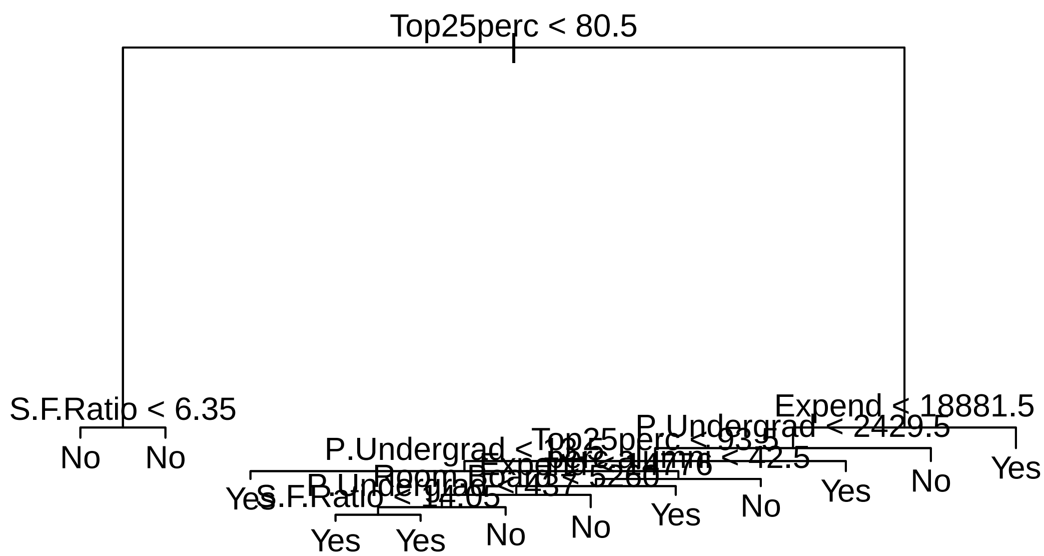
Q1 a)

First, a decision tree is created, excluding Top10perc.

```r
library(tree)
tree_elite = tree(Elite ~ . - Top10perc, data = college)
summary(tree_elite)
```

```
##
## Classification tree:
## tree(formula = Elite ~ . - Top10perc, data = college)
## Variables actually used in tree construction:
## [1] "Top25perc"   "S.F.Ratio"   "Expend"      "P.Undergrad" "perc.alumni"
## [6] "Room.Board"
## Number of terminal nodes:  12
## Residual mean deviance:  0.04263 = 32.61 / 765
## Misclassification error rate: 0.009009 = 7 / 777
```

```r
plot(tree_elite, main = "Decision Tree for Elites")
text(tree_elite, pretty = 0)
```



Q1 b)

Using a set seed for reproducible results, the new tree is trained on 500 random observations from the data set and tested against the remaining observations.

```r
set.seed(1)
training_set = sample(1:nrow(college), 500)
test_set = college[-training_set,]
new_tree_elite = tree(Elite ~ . - Top10perc, college, subset = training_set)
summary(new_tree_elite)
```

```
##
## Classification tree:
## tree(formula = Elite ~ . - Top10perc, data = college, subset = training_set)
## Variables actually used in tree construction:
## [1] "Top25perc"   "S.F.Ratio"   "Expend"      "perc.alumni" "Apps"
## [6] "P.Undergrad" "Outstate"
## Number of terminal nodes:  11
## Residual mean deviance:  0.05905 = 28.88 / 489
## Misclassification error rate: 0.014 = 7 / 500
```

```r
test_predictions = predict(new_tree_elite, test_set, type = "class")
table = table(test_predictions, test_set$Elite)
print(table)
```

```
##
## test_predictions  No Yes
##              No  242   8
##              Yes   7  20
```

```r
print ((table[1, 2] + table[2, 1])/sum(table))
```

```
## [1] 0.05415162
```

15 observations are misclassified, giving an error rate of ~5.4%, higher than when the tree was tested on all the data used for fitting - which caused overfitting. This is a more realistic error rate.

Q1 c)

```r
logistic_elite = glm(Elite ~  Top25perc + S.F.Ratio + Expend + P.Undergrad + perc.alumni + Room.Board
 + Outstate + Apps, data = college, family = binomial, subset = training_set)
```

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

```r
summary(logistic_elite)
```

```
##
## Call:
## glm(formula = Elite ~ Top25perc + S.F.Ratio + Expend + P.Undergrad +
##     perc.alumni + Room.Board + Outstate + Apps, family = binomial,
##     data = college, subset = training_set)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.701e+01  6.492e+00  -4.161 3.17e-05 ***
## Top25perc    3.630e-01  7.356e-02   4.935 8.02e-07 ***
## S.F.Ratio   -9.896e-02  1.690e-01  -0.586  0.55813
## Expend       3.584e-04  1.202e-04   2.982  0.00286 **
## P.Undergrad -1.231e-03  7.181e-04  -1.715  0.08639 .
## perc.alumni -1.034e-01  3.806e-02  -2.717  0.00659 **
```

```
## Room.Board  -1.044e-03  5.300e-04  -1.970  0.04886 *
## Outstate     8.712e-05  1.423e-04   0.612  0.54027
## Apps         3.793e-05  1.478e-04   0.257  0.79744
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 325.083  on 499  degrees of freedom
## Residual deviance:  61.914  on 491  degrees of freedom
## AIC: 79.914
##
## Number of Fisher Scoring iterations: 10
```

Remove non-significant variables: in this case, 'Apps' has the highest P-value.

```
logistic_elite = glm(Elite ~  Top25perc + S.F.Ratio + Expend + P.Undergrad + perc.alumni + Room.Board
 + Outstate, data = college, family = binomial, subset = training_set)
```

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

```
summary(logistic_elite)
```

```
##
## Call:
## glm(formula = Elite ~ Top25perc + S.F.Ratio + Expend + P.Undergrad +
##     perc.alumni + Room.Board + Outstate, family = binomial, data = college,
##     subset = training_set)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.773e+01  5.973e+00  -4.643 3.44e-06 ***
## Top25perc    3.686e-01  7.150e-02   5.154 2.54e-07 ***
## S.F.Ratio   -8.484e-02  1.582e-01  -0.536  0.59167
## Expend       3.677e-04  1.142e-04   3.219  0.00129 **
## P.Undergrad -1.111e-03  5.254e-04  -2.115  0.03440 *
## perc.alumni -1.051e-01  3.774e-02  -2.786  0.00533 **
## Room.Board  -1.036e-03  5.298e-04  -1.956  0.05043 .
## Outstate     9.260e-05  1.407e-04   0.658  0.51044
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 325.08  on 499  degrees of freedom
## Residual deviance:  61.98  on 492  degrees of freedom
## AIC: 77.98
##
## Number of Fisher Scoring iterations: 10
```

Remove non-significant variables: in this case, 'S.F.Ratio' has the highest p-value.

```r
logistic_elite = glm(Elite ~  Top25perc + Expend + P.Undergrad + perc.alumni + Room.Board
 + Outstate, data = college, family = binomial, subset = training_set)
```

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

```r
summary(logistic_elite)
```

```
##
## Call:
## glm(formula = Elite ~ Top25perc + Expend + P.Undergrad + perc.alumni +
##     Room.Board + Outstate, family = binomial, data = college,
##     subset = training_set)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.919e+01  5.474e+00  -5.333 9.66e-08 ***
## Top25perc    3.662e-01  7.077e-02   5.175 2.28e-07 ***
## Expend       3.810e-04  1.069e-04   3.564 0.000365 ***
## P.Undergrad -1.096e-03  5.198e-04  -2.109 0.034982 *
## perc.alumni -1.033e-01  3.725e-02  -2.772 0.005570 **
## Room.Board  -9.671e-04  5.077e-04  -1.905 0.056815 .
## Outstate     9.697e-05  1.410e-04   0.688 0.491701
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 325.083  on 499  degrees of freedom
## Residual deviance:  62.272  on 493  degrees of freedom
## AIC: 76.272
##
## Number of Fisher Scoring iterations: 10
```

Remove non-significant variables: in this case, 'Outstate' has the highest p-value.

```r
logistic_elite = glm(Elite ~  Top25perc + Expend + P.Undergrad + perc.alumni + Room.Board,
data = college, family = binomial, subset = training_set)
```

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

```r
summary(logistic_elite)
```

```
##
## Call:
## glm(formula = Elite ~ Top25perc + Expend + P.Undergrad + perc.alumni +
##     Room.Board, family = binomial, data = college, subset = training_set)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.900e+01  5.480e+00  -5.293 1.21e-07 ***
```

```
## Top25perc    3.619e-01  7.063e-02   5.123 3.00e-07 ***
## Expend       3.993e-04  1.018e-04   3.921 8.83e-05 ***
## P.Undergrad -1.221e-03  4.988e-04  -2.449  0.01434 *
## perc.alumni -9.338e-02  3.435e-02  -2.718  0.00656 **
## Room.Board  -7.478e-04  3.770e-04  -1.984  0.04730 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 325.083  on 499  degrees of freedom
## Residual deviance:  62.755  on 494  degrees of freedom
## AIC: 74.755
##
## Number of Fisher Scoring iterations: 10
```

```r
predicted_logistic = predict(logistic_elite, newdata = test_set, type = "response")
summary(predicted_logistic)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000000 0.0000026 0.0875313 0.0009995 0.9999966
```

When using the testing data in the logistic regression, on average, a college has an estimated ~8.75% probability of being elite.

```r
predicted_elite = rep("No", 277)
predicted_elite[predicted_logistic > 0.5] = "Yes"
table = table(predicted_elite, test_set$Elite)
print(table)
```

```
##
## predicted_elite  No Yes
##             No  247   5
##             Yes   2  23
```

```r
print((table[1, 2] + table[2, 1])/(sum(table)))
```

```
## [1] 0.02527076
```

Using the logistic regression, the error rate is ~2.53%, which is lower than the decision tree and is more accurate in this case.

Q2

Importing the package and viewing data summary

```r
library(ISLR)
View(Auto)
?Auto
```

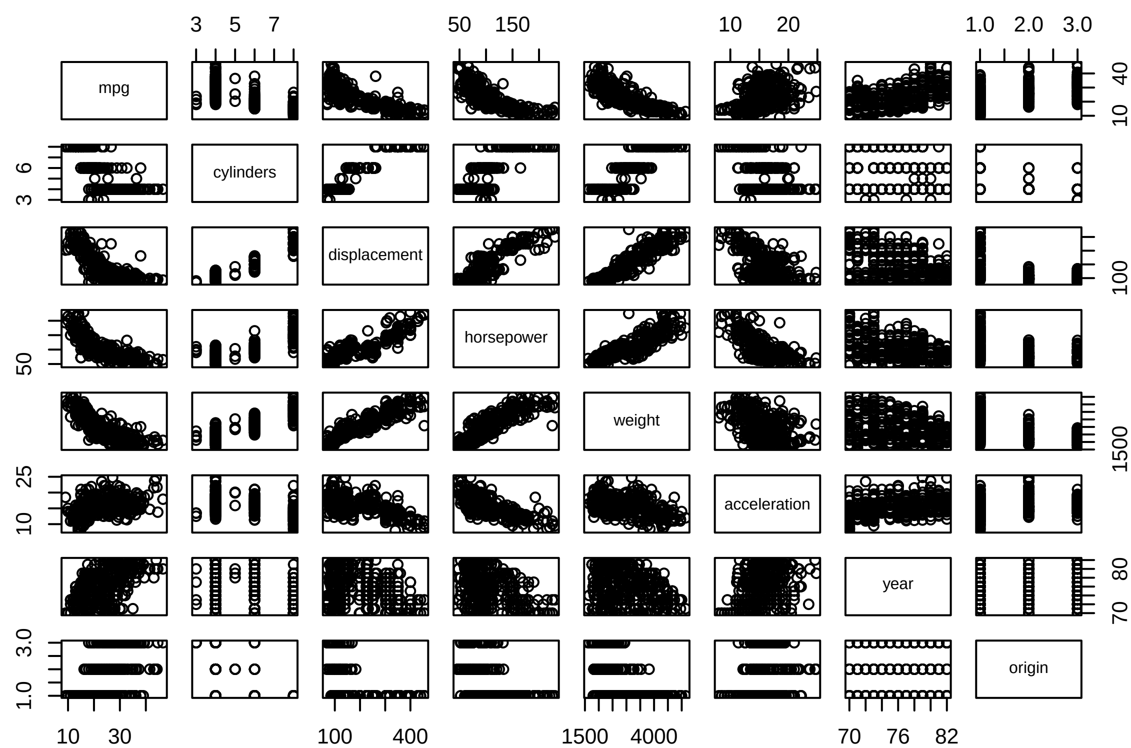
```
## starting httpd help server ... done
```

```r
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  
```

Q2 a)

```r
pairs(Auto[,1:8])
```



Q2 b)

```
cor(Auto[,1:8])
```

```
##                     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
##              acceleration       year     origin
## mpg             0.4233285  0.5805410  0.5652088
## cylinders      -0.5046834 -0.3456474 -0.5689316
## displacement   -0.5438005 -0.3698552 -0.6145351
## horsepower     -0.6891955 -0.4163615 -0.4551715
## weight         -0.4168392 -0.3091199 -0.5850054
## acceleration    1.0000000  0.2903161  0.2127458
## year            0.2903161  1.0000000  0.1815277
## origin          0.2127458  0.1815277  1.0000000
```

Q2 c)

```
mpg_regression = lm(mpg ~ . -name, data = Auto)
summary(mpg_regression)
```

```
##
## Call:
## lm(formula = mpg ~ . - name, data = Auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5903 -2.1565 -0.1169  1.8690 13.0604
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -17.218435   4.644294  -3.707  0.00024 ***
## cylinders     -0.493376   0.323282  -1.526  0.12780
## displacement   0.019896   0.007515   2.647  0.00844 **
## horsepower    -0.016951   0.013787  -1.230  0.21963
## weight        -0.006474   0.000652  -9.929  < 2e-16 ***
## acceleration   0.080576   0.098845   0.815  0.41548
## year           0.750773   0.050973  14.729  < 2e-16 ***
## origin         1.426141   0.278136   5.127 4.67e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.328 on 384 degrees of freedom
```

```
## Multiple R-squared:  0.8215, Adjusted R-squared:  0.8182
## F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

i) Yes, there is a relationship between the predidctors and the response. When testing the null hypothesis that all regression coefficients are 0, the F-statistic is returned - with a value of 252.4, suggesting that the overall regression is statistically significant. This is supported by the low p-value.

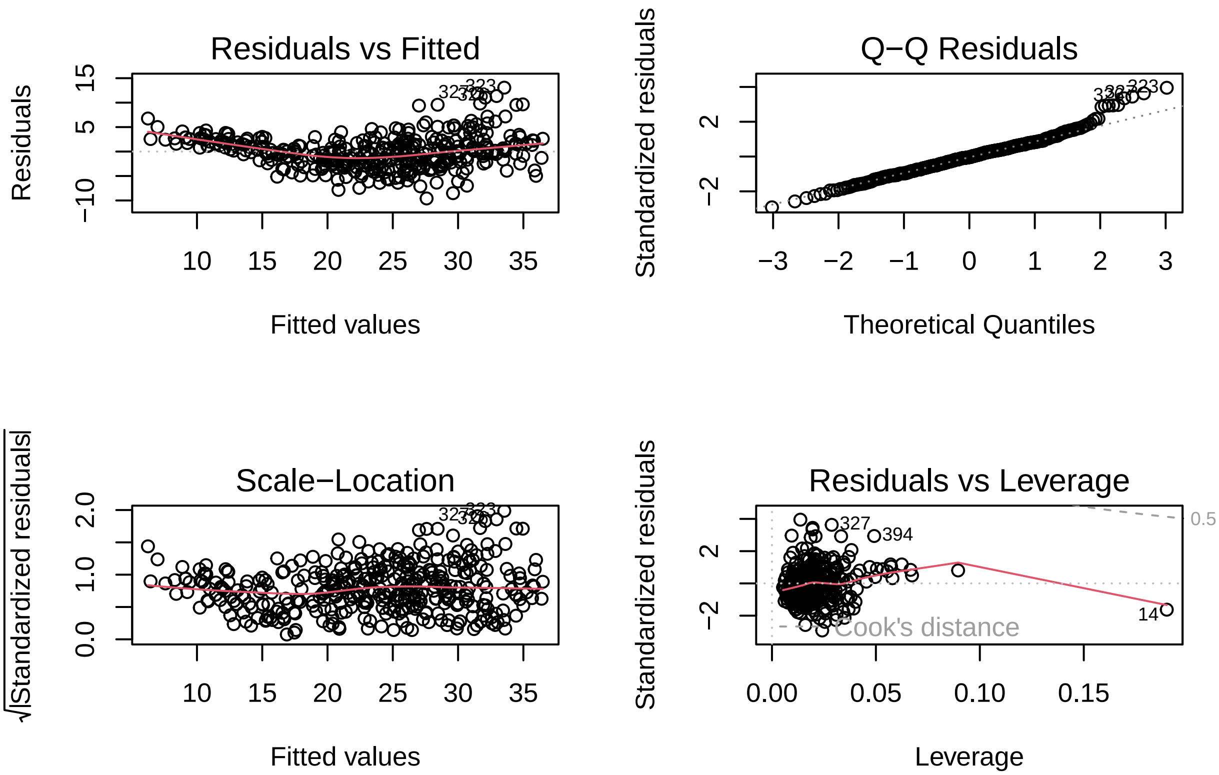
ii) The predictors with a statistically significant relationship are the predictors with low p-values, with < 0.05 being statistically significant. These predictors are: displacement, weight, year and origin. It appears there is a high probability that these regressors affect mpg. These results are supported by the pairs() plot produced earlier, with the exception of horsepower which appears to correlate with mpg in the plot. The lack of statistical significance of acceleration, however, is also supported by the plot.

iii) The coefficients for year and origin are both positive, suggesting a positive relationship between these regressors and mpg. In the case of the year coefficient, an increase in the year results in a ~0.75 increase in mpg. Intuitively, this is plausible; later years indicate newer cars and more time for better technology to develop, resulting in efficient cars with higher miles per gallon. The coefficient of origin is ~1.43, almost double the coefficient on year. This suggests the origin of the car has a higher impact on mpg, with American cars being the least efficient, followed by European and Japanese cars.

iv) The insignificant predictors are cylinders, horsepower and acceleration. The first possbility would be to remove the insignificant predictors. This can reduce overfitting and improve generalisability when applying the model to unseen data. This must be done one predictor at a time, in order to observe if previously-insignificant variables become significant. For example, from the plot it can be seen that horsepower and acceleration are highly correlated, so removing one of these predictors can reduce multicollinearity and increase significance, as well as reduce standard error. Another possibility would be to create an interaction term between these two correlated regressors in order to capture the joint effect on mpg, which would avoid omitted variable bias.

Q2 d)

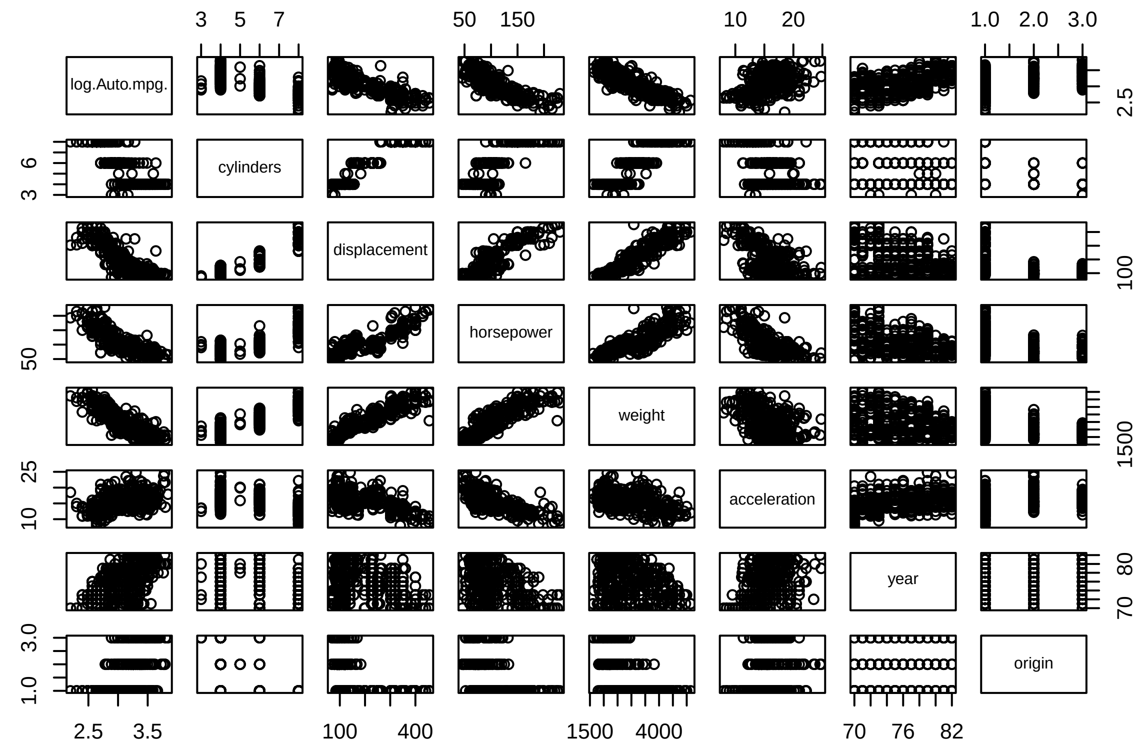
```r
par(mfrow=c(2,2))
plot(mpg_regression)
```



From the diagnosis plots, we see the following: - The Residuals vs Fitted plot curves slightly, suggesting that the relationship between the preditors and the response contains some non-linearity. Residuals are higher for smaller fitted values, decreasing as fitted values approach 20, then increasing again. - The Q Q residuals plot shows most residuals have a normal distribution, with the exception of a few outliers towards the end of the graph. - The Scale-Location plot shows some heteroskedasticity, with the same outliers pulling the line upwards due to much higher variance. - The Residuals vs Leverage plot shows clear leverage from point 14, suggesting significant influence on the regression and coefficients.

Q2 e)

```
pairs(data.frame(log(Auto$mpg),Auto[,-c(1,9)]))
```




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

```
## [1] -0.8322442
```

```
cor((log(Auto$mpg)),Auto$weight)
```

```
## [1] -0.8756582
```

Overall, correlations appear to be stronger between log mpg and most variables as seen in the plot. For example, the correlation between log mpg and weight is higher than when mpg is not logged. Log transformations can be more appropriate and result in higher correlation, especially when the relationship was not completely linear to begin with.

Q3 a) i)

```
set.seed(3)
x1=runif(150) # 150 U(0,1) random numbers
x2=0.5*runif(150)+rnorm(150)/5 # rnorm(150) returns 150 N(0,1) random numbers
y=2+2*x1+x2+rnorm(150)
```

B0 = 2, B1 = 2, B2 = 1, E~N(0,1)

Q3 a) ii)

```
ytrain = y[1:100]; ytest=y[101:150] # splits y into training and test sets, with 100 and 50 observation
x = data.frame(x1, x2); x.train=x[1:100,]; x.test=x[101:150,]
m1 = lm(ytrain~x1+x2, data=x.train)
summary(m1)
```

```
##
## Call:
## lm(formula = ytrain ~ x1 + x2, data = x.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3324 -0.6809 -0.0203  0.4915  3.5541
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.9065     0.2257   8.449 2.96e-13 ***
## x1            2.0503     0.3761   5.452 3.80e-07 ***
## x2            1.1604     0.4104   2.827  0.00571 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.058 on 97 degrees of freedom
## Multiple R-squared:  0.3108, Adjusted R-squared:  0.2966
## F-statistic: 21.88 on 2 and 97 DF,  p-value: 1.44e-08
```

```
confint.lm(m1)
```

```
##                 2.5 %   97.5 %
## (Intercept) 1.4586262 2.354350
## x1          1.3038742 2.796755
## x2          0.3457617 1.975008
```

The coefficient on x1 is 2.0503, while the coefficient on x2 is 1.1604. Both coefficients are significant, with very low p-values. B0 confidence intervals = [1.459, 2.354]. B1 confidence intervals = [1.304, 2.797]. B2 confidence intervals = [0.346, 1.975].

Q3 b) iii)

```
predict_y = predict(m1, x.test)
summary(predict_y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.587   2.751   3.177   3.199   3.535   4.397
```

```
squared_difference = (predict_y - ytest)^2
print (sqrt(mean(squared_difference)))
```

```
## [1] 1.07953
```

The rMSPE is ~1.080.

Q3 b) i)

```r
set.seed(3)
x1=runif(150)
x2=0.5*x1+rnorm(150)/5
y=2+2*x1+x2+rnorm(150)
print (cor(x1, x2))
```

```
## [1] 0.5844431
```

The correlation between x1 and x2 is 0.584.

Q3 b) ii)

```r
ytrain = y[1:100]; ytest=y[101:150] # splits y into training and test sets, with 100 and 50 observation
x = data.frame(x1, x2); x.train=x[1:100,]; x.test=x[101:150,]
m1 = lm(ytrain~x1+x2, data=x.train)
summary(m1)
```

```
##
## Call:
## lm(formula = ytrain ~ x1 + x2, data = x.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3096 -0.6276 -0.0263  0.5668  3.5841
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.3265     0.2030  11.459  < 2e-16 ***
## x1            1.8250     0.4672   3.906 0.000174 ***
## x2            0.4190     0.5197   0.806 0.422167
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.029 on 97 degrees of freedom
## Multiple R-squared:  0.2558, Adjusted R-squared:  0.2405
## F-statistic: 16.67 on 2 and 97 DF,  p-value: 5.981e-07
```

```r
confint.lm(m1)
```

```
##                  2.5 %   97.5 %
## (Intercept)  1.9235661 2.729523
## x1           0.8976977 2.752343
## x2          -0.6125908 1.450504
```

The coefficient on x1 is 1.8250, while the coefficient on x2 is 0.4190. THe coefficient on x2 is not significant, while the x1 coefficient is. B0 confidence intervals = [1.924, 2.730]. B1 confidence intervals = [0.898, 2.752]. B2 confidence intervals = [-0.613, 1.451].

```r
predict_y = predict(m1, x.test)
summary(predict_y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.336   2.897   3.332   3.310   3.726   4.201
```

```r
squared_difference = (predict_y - ytest)^2
print(sqrt(mean(squared_difference)))
```

```
## [1] 1.090116
```

The rMSPE is 1.090.

Q3 b) iii)

In the second model run, the coefficient on x2 is not significant while it was in the first model run. x1 and x2 are highly correlated, as corr = 0.584. The model is unable to accurately differentiate between the effects of x1 and x2 on y, and as such the coefficient on x2 becomes statistically insignificant; this is also supported by the larger confidence intervals of the second regression. The new rMSPE, 1.09, is fairly similar to the first regression (rMSPE = 1.07953), suggesting that the difference between actual and predicted values is roughly the same for both regressions.

Q3 b) iv)

```r
m1 = lm(ytrain~x1, data=x.train)
summary(m1)
```

```
##
## Call:
## lm(formula = ytrain ~ x1, data = x.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2686 -0.6139 -0.0734  0.6011  3.5881
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.3182     0.2024  11.453  < 2e-16 ***
## x1            2.0641     0.3604   5.728 1.12e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.027 on 98 degrees of freedom
## Multiple R-squared:  0.2508, Adjusted R-squared:  0.2432
## F-statistic: 32.81 on 1 and 98 DF,  p-value: 1.117e-07
```

The coefficient on x1 is 2.0641 and is statistically significant.

```r
predict_y = predict(m1, x.test)
summary(predict_y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.403   2.893   3.331   3.319   3.757   4.204
```

```r
squared_difference = (predict_y - ytest)^2
print (sqrt(mean(squared_difference)))
```

```
## [1] 1.115781
```

The rMSPE is now 1.116, which is actually slightly higher than the previous regressions. The coefficient on x1, 2.0641, is higher than the coefficient on x1 (1.8250) when the regression with x2 was also run. This is evidence of multicollinearity, given that the effect of x1 is higher when x2 is not included in the regression. The coefficient on x1 is able to accurately capture the effect of x1 on y, without the redundancy of a correlated regressor. Finally, the R^2 is reduced, from 0.2558 in the previous regression to 0.2508. This is expected, as removing a regressor will remove some predictive power of the regression, with a lower proportion of variance being accounted for by the predictors; however, as the reduction is so low (only 0.005) it shows that x2 is very insignificant in predicting y.

Q3 c) i)

```r
set.seed(3)
x1=runif(150)
epsilon=rnorm(150)
x2=0.5*runif(150)+epsilon/5
y=2+2*x1+x2+epsilon
cor(x2, epsilon)
```

```
## [1] 0.8217773
```

The correlation between x2 and the error is 0.822.

Q3 c) ii)

```r
ytrain = y[1:100]; ytest=y[101:150] # splits y into training and test sets, with 100 and 50 observation
x = data.frame(x1, x2); x.train=x[1:100,]; x.test=x[101:150,]
m1 = lm(ytrain~x1+x2, data=x.train)
summary(m1)
```

```
##
## Call:
## lm(formula = ytrain ~ x1 + x2, data = x.train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.35583 -0.43974  0.06475  0.42365  1.19596
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.1972     0.1276   9.383 2.89e-15 ***
## x1            1.8863     0.2103   8.969 2.25e-14 ***
## x2            4.4114     0.2520  17.504  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5911 on 97 degrees of freedom
## Multiple R-squared:  0.8228, Adjusted R-squared:  0.8192
## F-statistic: 225.3 on 2 and 97 DF,  p-value: < 2.2e-16
```

```r
confint.lm(m1)
```

```
##                 2.5 %   97.5 %
## (Intercept) 0.9439413 1.450410
## x1          1.4689389 2.303755
## x2          3.9111854 4.911557
```

The coefficient on x1 is 1.886, while the coefficient on x2 is 4.411. The coefficients on x1 and x2 are both statistically significant. B0 confidence interval = [0.944, 1.450]. B1 confidence interval = [1.469, 2.304]. B2 confidence interval = [3.911, 4.912].

```r
predict_y = predict(m1, x.test)
summary(predict_y)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.099   2.455   3.491   3.433   4.476   6.459
```

```r
squared_difference = (predict_y - ytest)^2
print (sqrt(mean(squared_difference)))
```

```
## [1] 0.6414308
```

The rMPSE is 0.641.

Q3) c) iii)

B1 and B2 are both statistically significant. The R^2 is high, at 0.8228, suggesting the predictors capture most of the variance in y. rMSPE is also lower than previously, suggesting the model's estimates are accurate to the true values. This occurs due to endogeneity, causing biased estimates. Additionally, the confidence interval of B2 does not include its true value within the range due to the correlation between B2 and the error term.