assign5.R

harry

2020-11-01

```r
#Harry Pines Assignment 5
```

1)

```r
library("tidyverse")
```

```
## -- Attaching packages --------------------------------------------------------------------------- tid

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------ tidyverse
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(dplyr)

#read in data

autos<-na.omit(read.csv("Auto.csv", header=TRUE))
suppressWarnings(autos$horsepower <- as.numeric(as.character(autos$horsepower)))
autos <- autos[complete.cases(autos),]
```

a. (5%) Perform a multiple linear regression with mpg as the response and all other variables except name as the predictors.

```r
input = subset(autos, select=-c(name))
print(head(input))
```

```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
## 5  17         8          302        140   3449         10.5   70      1
## 6  15         8          429        198   4341         10.0   70      1
```

```r
# Build the model
model <- lm(mpg~cylinders+displacement+horsepower+weight+acceleration+year+origin,
            data = input)

# Show the model.
summary(model)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + year + origin, data = input)
##
## 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
```

```r
#the summary indicates that we have several strong predictors
#in the 99.9% confidence interval.
```

i) Which predictors appear to have a statistically significant relationship to the response, and how do you determine this?

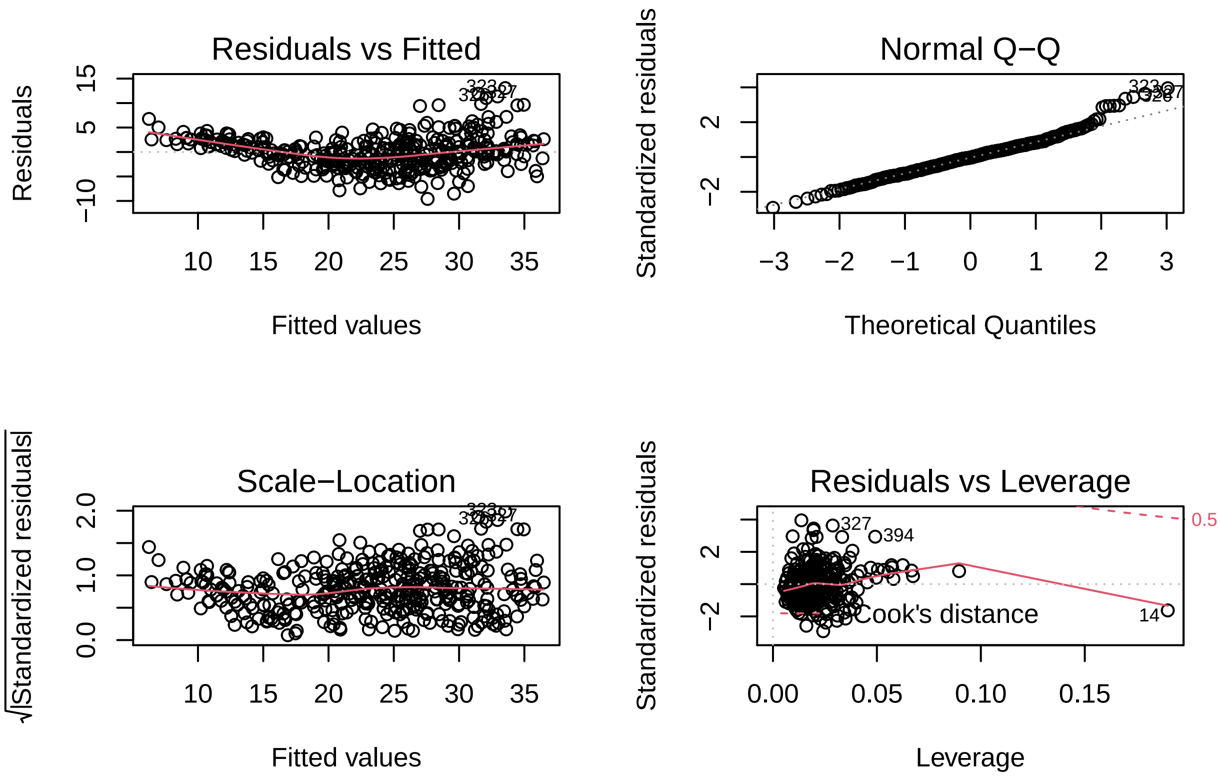
```r
#Based on the p values, the strongest predictors
#of miles to the gallon are the weight, year and origin
```

ii) What does the coefficient for the displacement variable suggest, in simple terms?

```r
# The coefficient of the variable displacement is estimated to be approx 0.01,
# this is the multiplier applied to the variable in the prediction.
# The p value indicates a value < 0.01 which implies
# statistical significance at a 99% confidence level.
```

b. (5%) Produce diagnostic plots of the linear regression fit.

```r
par(mfrow=c(2,2)) # Change the panel layout to 2 x 2
plot(model)
```




```r
par(mfrow=c(1,1)) # Change back to 1 x 1

#the residual plot appears to be good. the data is
#distributed around a relatively horizontal line without many outliers.
# the normal Q-Q also appears to be linear and the scale location is permissible.
# there are seemingly no cases directly beyond the cooks distance line,
```

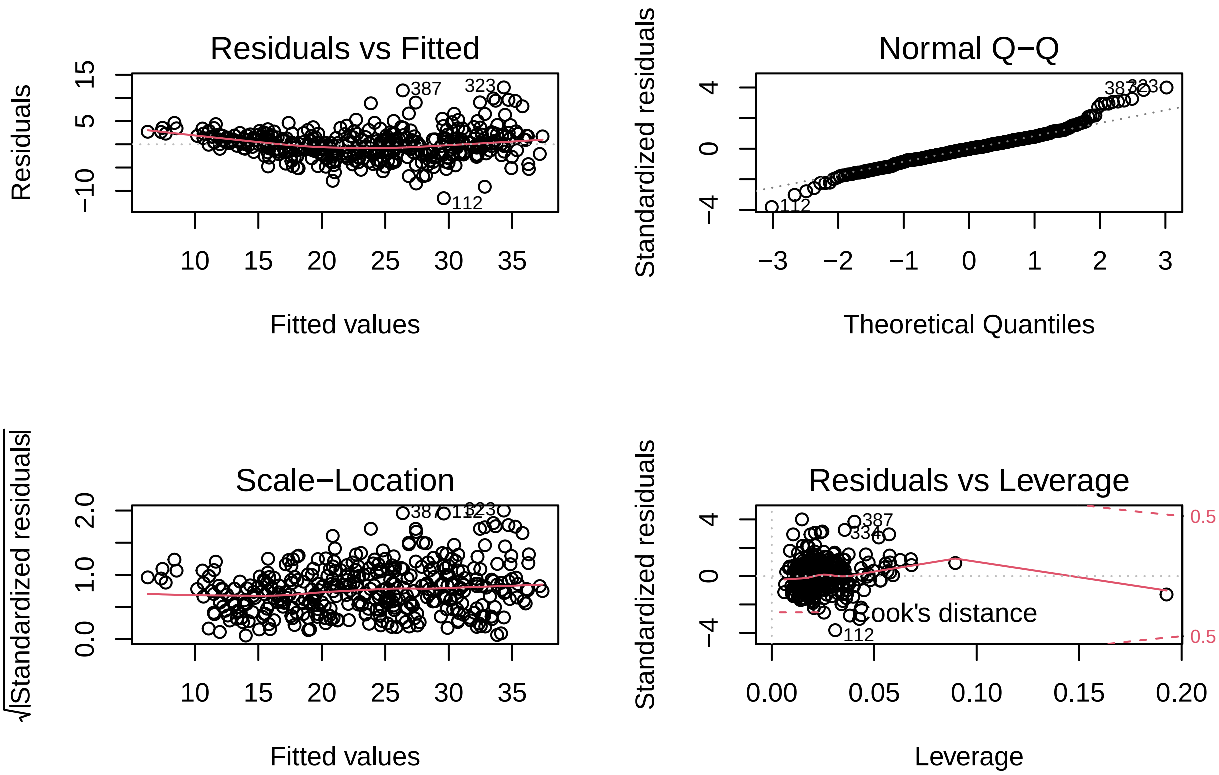
c. (5%) Fit linear regression models with interaction effects.

```r
#interactions between cylinders and displacement
# Build the model
model2 <- lm(mpg~cylinders*displacement+horsepower+weight+acceleration+year+origin,
             data = input)

# Show the model
summary(model2)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders * displacement + horsepower + weight +
##     acceleration + year + origin, data = input)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.6081  -1.7833  -0.0465   1.6821  12.2617
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -2.7096590  4.6858582  -0.578 0.563426
## cylinders              -2.6962123  0.4094916  -6.584 1.51e-10 ***
## displacement           -0.0774797  0.0141535  -5.474 7.96e-08 ***
## horsepower             -0.0476026  0.0133736  -3.559 0.000418 ***
## weight                 -0.0052339  0.0006253  -8.370 1.10e-15 ***
## acceleration            0.0597997  0.0918038   0.651 0.515188
## year                    0.7594500  0.0473354  16.044  < 2e-16 ***
## origin                  0.7087399  0.2736917   2.590 0.009976 **
## cylinders:displacement  0.0136081  0.0017209   7.907 2.84e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.089 on 383 degrees of freedom
## Multiple R-squared:  0.8465, Adjusted R-squared:  0.8433
## F-statistic: 264.1 on 8 and 383 DF,  p-value: < 2.2e-16
```

```r
#with a lower standard of error and higher statistical significance
#across the board, there is likely an interaction relationship
#between cylinders and displacement.
par(mfrow=c(2,2)) # Change the panel layout to 2 x 2
plot(model2)
```

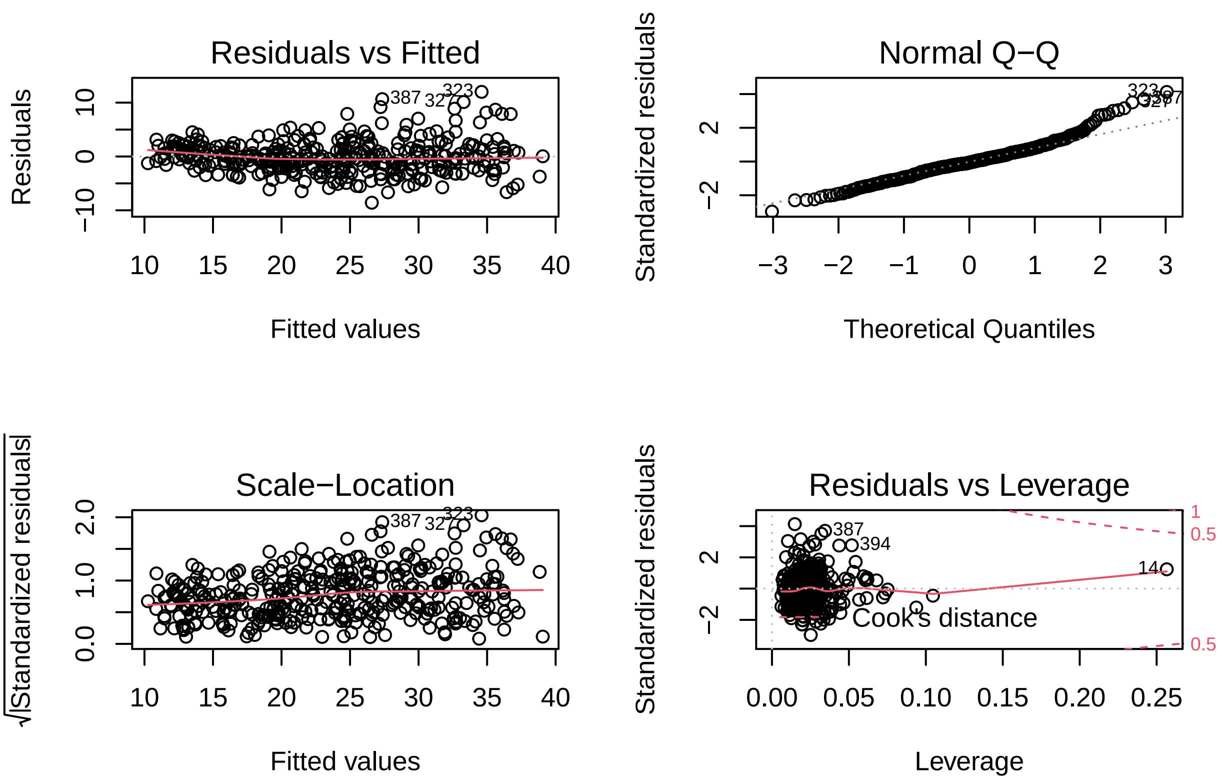



```r
par(mfrow=c(1,1)) # Change back to 1 x 1


#interactions between weight and horsepower
# Build the model
model3 <- lm(mpg~cylinders+displacement+horsepower*weight+acceleration+year+origin,
             data = input)

# Show the model
summary(model3)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower * weight +
##     acceleration + year + origin, data = input)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.589 -1.617 -0.184  1.541 12.001
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        2.876e+00  4.511e+00   0.638 0.524147
## cylinders         -2.955e-02  2.881e-01  -0.103 0.918363
## displacement       5.950e-03  6.750e-03   0.881 0.378610
```

```
## horsepower        -2.313e-01  2.363e-02  -9.791  < 2e-16 ***
## weight            -1.121e-02  7.285e-04 -15.393  < 2e-16 ***
## acceleration      -9.019e-02  8.855e-02  -1.019 0.309081
## year               7.695e-01  4.494e-02  17.124  < 2e-16 ***
## origin             8.344e-01  2.513e-01   3.320 0.000986 ***
## horsepower:weight  5.529e-05  5.227e-06  10.577  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.931 on 383 degrees of freedom
## Multiple R-squared:  0.8618, Adjusted R-squared:  0.859
## F-statistic: 298.6 on 8 and 383 DF,  p-value: < 2.2e-16
```

```r
par(mfrow=c(2,2)) # Change the panel layout to 2 x 2
plot(model3)
```

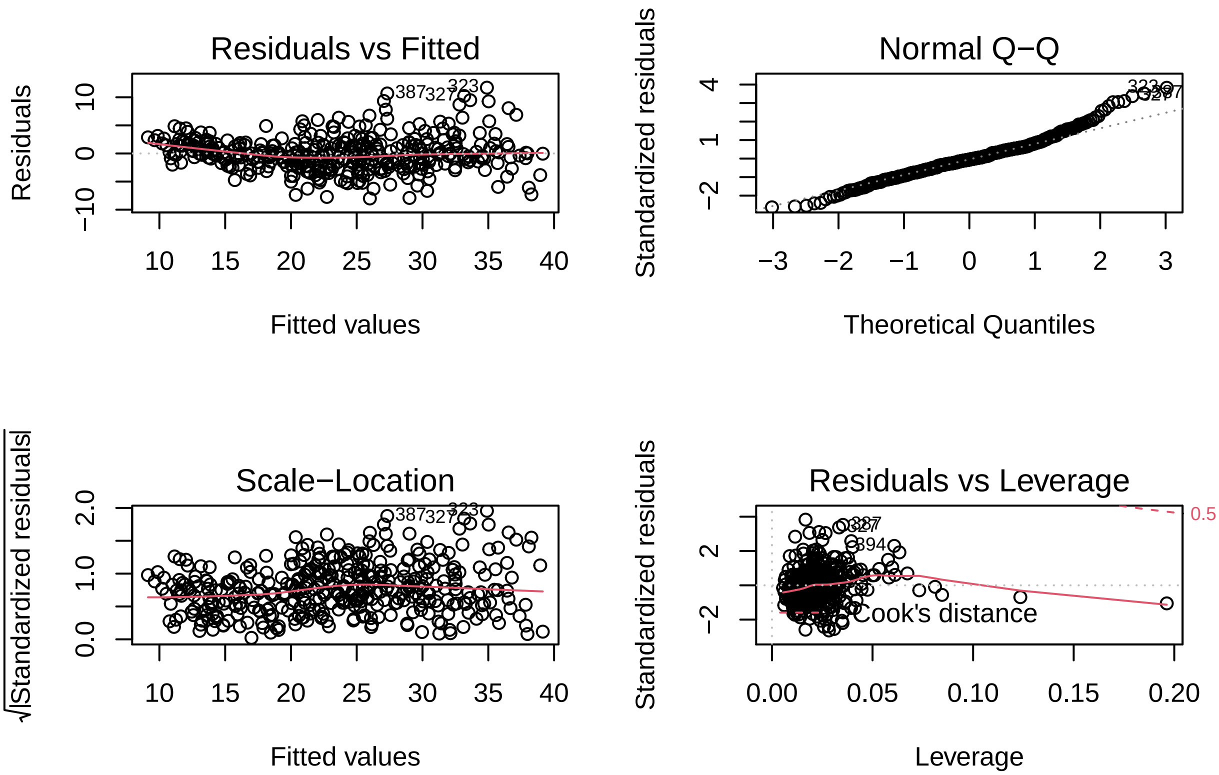



```r
par(mfrow=c(1,1)) # Change back to 1 x 1

#the r2 value indicates this model is also better than one without any
#interactions



#interactions between weight and year
# Build the model
```

```r
model4 <- lm(mpg~cylinders+displacement+horsepower+weight*year+acceleration+origin,
             data = input)

# Show the model
summary(model4)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight *
##     year + acceleration + origin, data = input)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9995 -1.8495 -0.1559  1.6061 11.7042
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.186e+02  1.338e+01  -8.864  < 2e-16 ***
## cylinders    -1.218e-01  3.032e-01  -0.402   0.6881
## displacement  1.293e-02  7.019e-03   1.842   0.0663 .
## horsepower   -2.877e-02  1.286e-02  -2.236   0.0259 *
## weight        3.044e-02  4.652e-03   6.543 1.94e-10 ***
## year          2.084e+00  1.732e-01  12.033  < 2e-16 ***
## acceleration  1.447e-01  9.196e-02   1.574   0.1164
## origin        1.174e+00  2.597e-01   4.519 8.30e-06 ***
## weight:year  -4.879e-04  6.097e-05  -8.002 1.47e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.084 on 383 degrees of freedom
## Multiple R-squared:  0.847,  Adjusted R-squared:  0.8439
## F-statistic: 265.1 on 8 and 383 DF,  p-value: < 2.2e-16
```

```r
par(mfrow=c(2,2)) # Change the panel layout to 2 x 2
plot(model4)
```




```r
par(mfrow=c(1,1)) # Change back to 1 x 1

#the r2 value indicates this model isn't vastly different from the initial model.
#there likely isn't a strong relationship
```

2) This problem involves the Boston data set, which we saw in class. We will now try to predict per capita crime rate using the other variables in this data set. In other words, per capita crime rate is the response, and the other variables are the predictors.

```r
boston<-na.omit(read.csv("HousingData.csv", header=TRUE))
boston <- boston[complete.cases(boston),]
summary(boston)
```

```
##       CRIM                ZN              INDUS            CHAS        
##  Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000  
##  1st Qu.: 0.08196   1st Qu.:  0.00   1st Qu.: 5.13   1st Qu.:0.00000  
##  Median : 0.26888   Median :  0.00   Median : 8.56   Median :0.00000  
##  Mean   : 3.69014   Mean   : 11.46   Mean   :11.00   Mean   :0.06853  
##  3rd Qu.: 3.43597   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000  
##  Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000  
##       NOX               RM             AGE              DIS        
##  Min.   :0.3890   Min.   :3.561   Min.   :  2.90   Min.   : 1.130  
##  1st Qu.:0.4530   1st Qu.:5.879   1st Qu.: 45.48   1st Qu.: 2.110  
##  Median :0.5380   Median :6.202   Median : 77.70   Median : 3.199  
```

```
##  Mean   :0.5532   Mean   :6.280   Mean   : 68.93   Mean   : 3.805
##  3rd Qu.:0.6240   3rd Qu.:6.606   3rd Qu.: 94.25   3rd Qu.: 5.117
##  Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127
##       RAD              TAX           PTRATIO            B
##  Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  2.6
##  1st Qu.: 4.000   1st Qu.:280.2   1st Qu.:17.40   1st Qu.:376.7
##  Median : 5.000   Median :330.0   Median :19.10   Median :392.2
##  Mean   : 9.404   Mean   :406.4   Mean   :18.54   Mean   :358.5
##  3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.9
##  Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.9
##      LSTAT             MEDV
##  Min.   : 1.730   Min.   : 5.00
##  1st Qu.: 7.125   1st Qu.:16.80
##  Median :11.300   Median :21.05
##  Mean   :12.769   Mean   :22.36
##  3rd Qu.:17.117   3rd Qu.:25.00
##  Max.   :37.970   Max.   :50.00
```

a. (6%) For each predictor, fit a simple linear regression model to predict the response. Include the code, but not the output for all models in your solution.

```r
boston.crime.lm.ZN <- lm(CRIM ~ ZN,data = boston)
summary(boston.crime.lm.ZN)
```

```
##
## Call:
## lm(formula = CRIM ~ ZN, data = boston)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.493 -4.279 -2.769  1.186 84.458
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.51820    0.50553   8.937  < 2e-16 ***
## ZN          -0.07225    0.01906  -3.791 0.000173 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.05 on 392 degrees of freedom
## Multiple R-squared:  0.03537,    Adjusted R-squared:  0.03291
## F-statistic: 14.37 on 1 and 392 DF,  p-value: 0.0001734
```

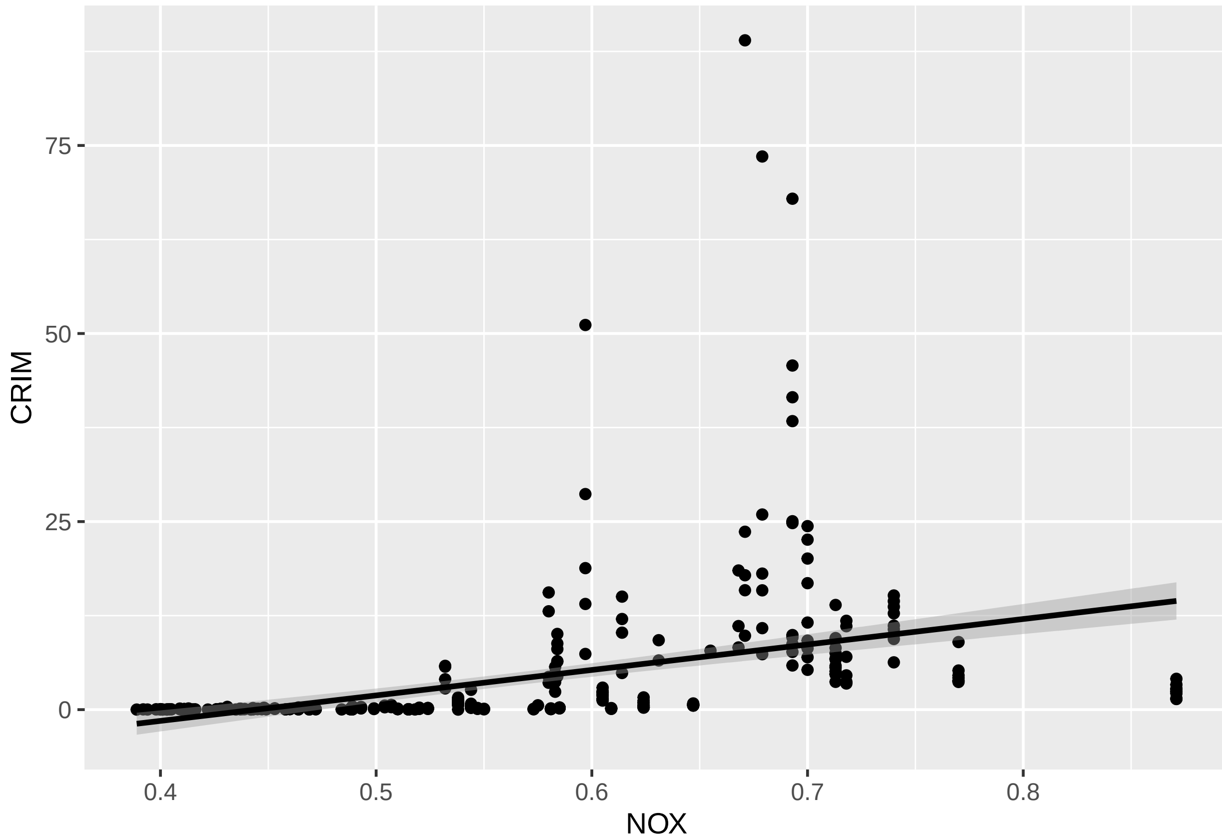
```r
boston.crime.lm.INDUS <- lm(CRIM ~ INDUS,data = boston)
#summary(boston.crime.lm.INDUS)

boston.crime.lm.CHAS <- lm(CRIM ~ CHAS,data = boston)
#summary(boston.crime.lm.CHAS)

boston.crime.lm.NOX <- lm(CRIM ~ NOX,data = boston)
#summary(boston.crime.lm.NOX)

boston.crime.lm.RM <- lm(CRIM ~ RM,data = boston)
```

```r
#summary(boston.crime.lm.RM)

boston.crime.lm.AGE <- lm(CRIM ~ AGE,data = boston)
#summary(boston.crime.lm.AGE)

boston.crime.lm.DIS <- lm(CRIM ~ DIS,data = boston)
#summary(boston.crime.lm.DIS)

boston.crime.lm.RAD <- lm(CRIM ~ RAD,data = boston)
#summary(boston.crime.lm.RAD)

boston.crime.lm.TAX <- lm(CRIM ~ TAX,data = boston)
#summary(boston.crime.lm.TAX)

boston.crime.lm.PTRATIO <- lm(CRIM ~ PTRATIO,data = boston)
#summary(boston.crime.lm.PTRATIO)

boston.crime.lm.B <- lm(CRIM ~ B,data = boston)
#summary(boston.crime.lm.B)

boston.crime.lm.LSTAT <- lm(CRIM ~ LSTAT,data = boston)
#summary(boston.crime.lm.LSTAT)


boston.crime.lm.MEDV <- lm(CRIM ~ MEDV,data = boston)
#summary(boston.crime.lm.MEDV)
```

b. (6%)

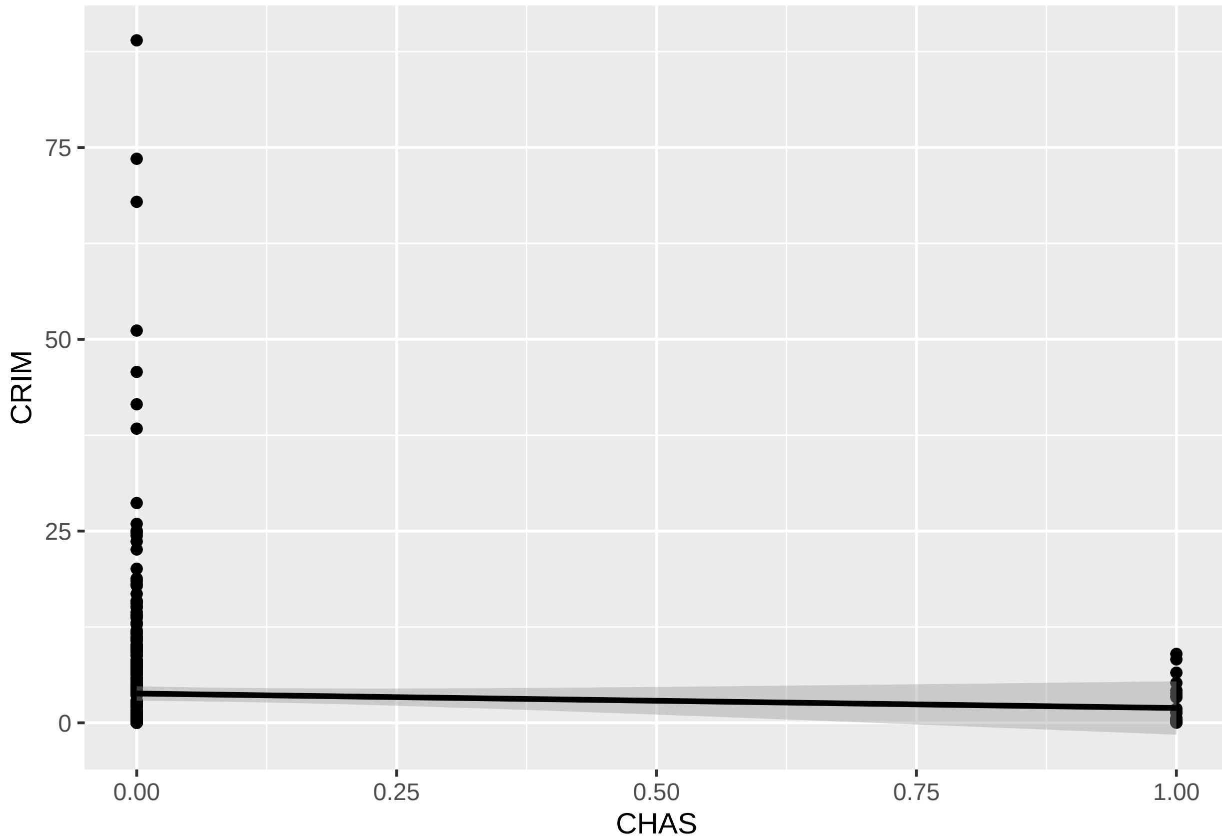
```r
#for each model, all were significant with the exception of CHAS.

#The variable meanings are defined as follows:
#CRIM - per capita crime rate by town

#NOX - nitric oxides concentration (parts per 10 million)
boston.graph.NOX<-ggplot(boston, aes(x=NOX, y=CRIM))+
  geom_point()
boston.graph.NOX <- boston.graph.NOX + geom_smooth(method="lm", col="black")

boston.graph.NOX
```

```
## `geom_smooth()` using formula 'y ~ x'
```



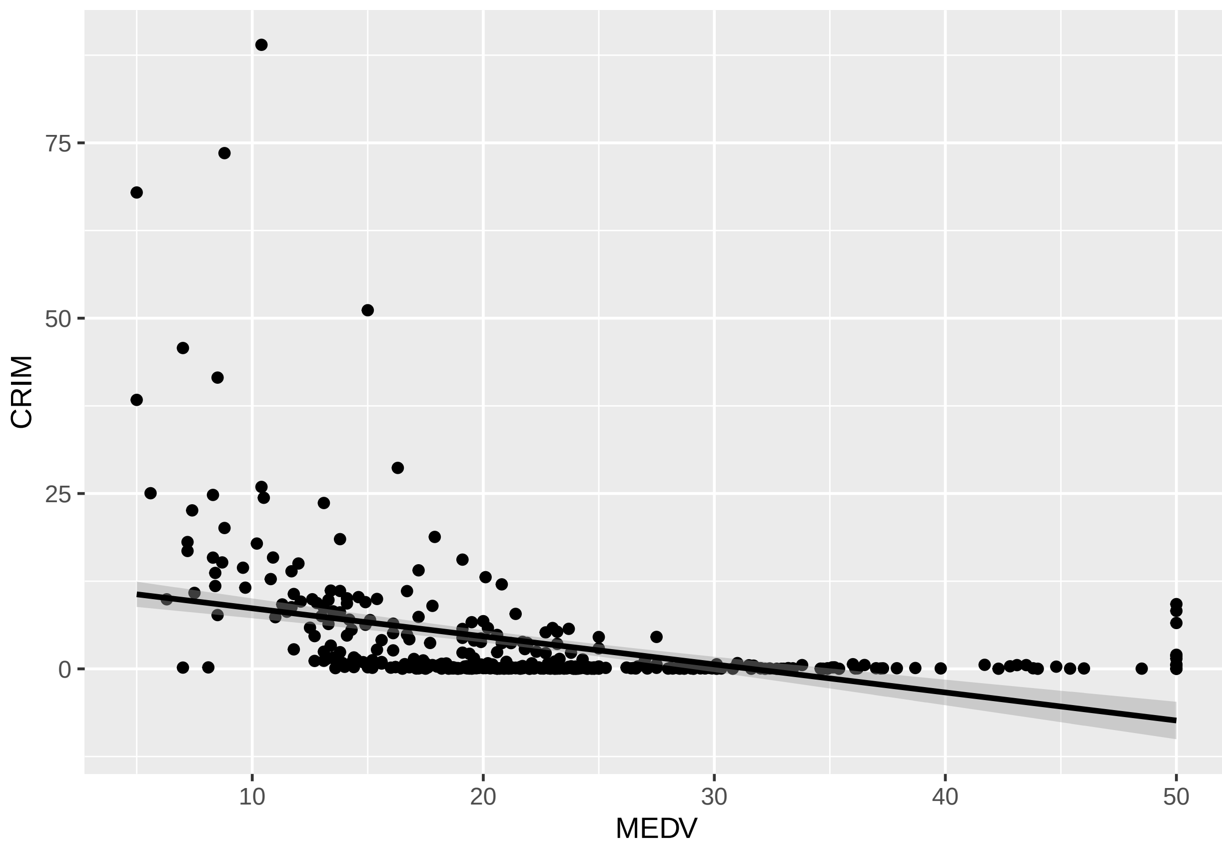

```
#Here, the relationship appears to be linearly increasing.


#CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
boston.graph.CHAS<-ggplot(boston, aes(x=CHAS, y=CRIM))+
  geom_point()
boston.graph.CHAS <- boston.graph.CHAS + geom_smooth(method="lm", col="black")
boston.graph.CHAS
```

```
## `geom_smooth()` using formula 'y ~ x'
```



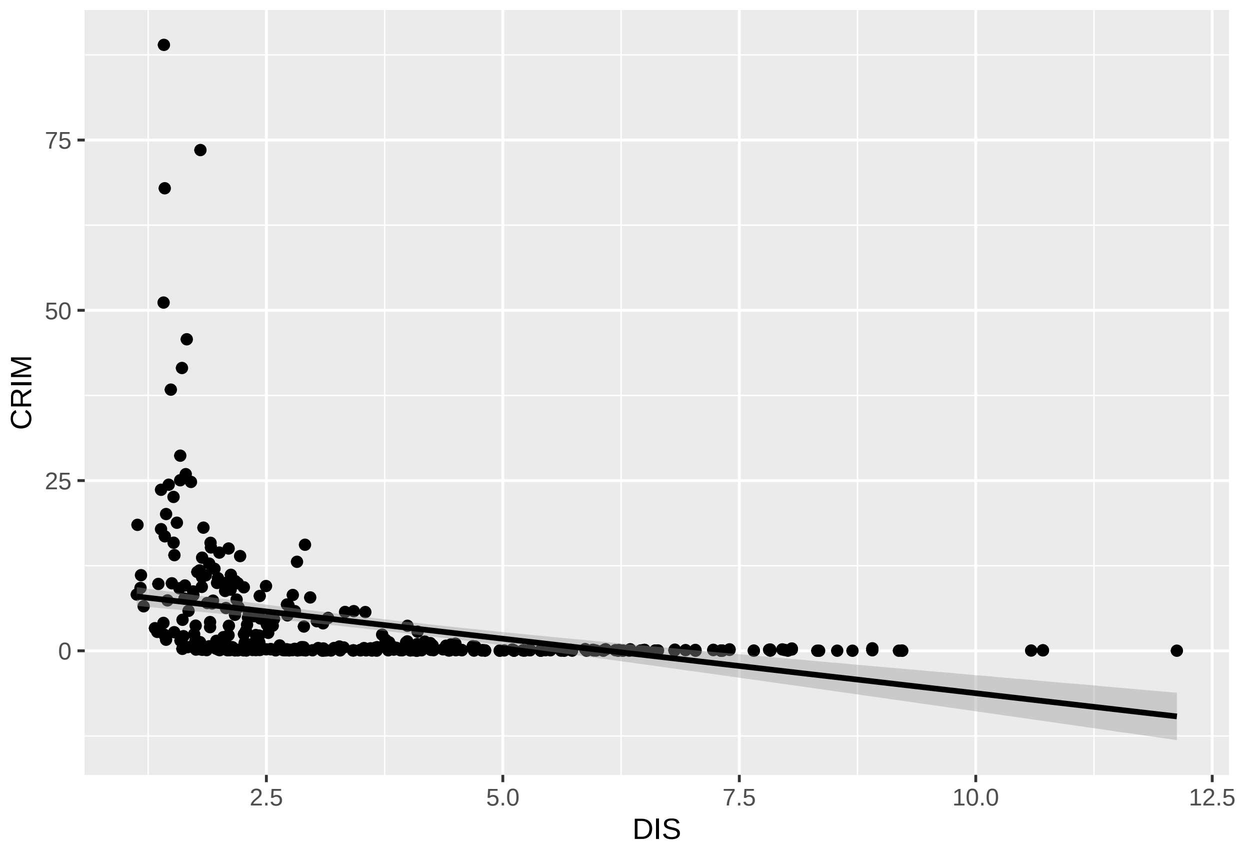

```r
#here the relationship indicates crime is more prevalent away from the river

#MEDV - Median value of owner-occupied homes in $1000's
boston.graph.MEDV<-ggplot(boston, aes(x=MEDV, y=CRIM))+
  geom_point()

boston.graph.MEDV <- boston.graph.MEDV + geom_smooth(method="lm", col="black")
boston.graph.MEDV
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
#Here, the relationship appears to be linearly decreasing.


#DIS - weighted distances to five Boston employment centres.
boston.graph.DIS<-ggplot(boston, aes(x=DIS, y=CRIM))+
  geom_point()

boston.graph.DIS <- boston.graph.DIS + geom_smooth(method="lm", col="black")
boston.graph.DIS
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
#Here, the relationship appears to be linearly decreasing.
```

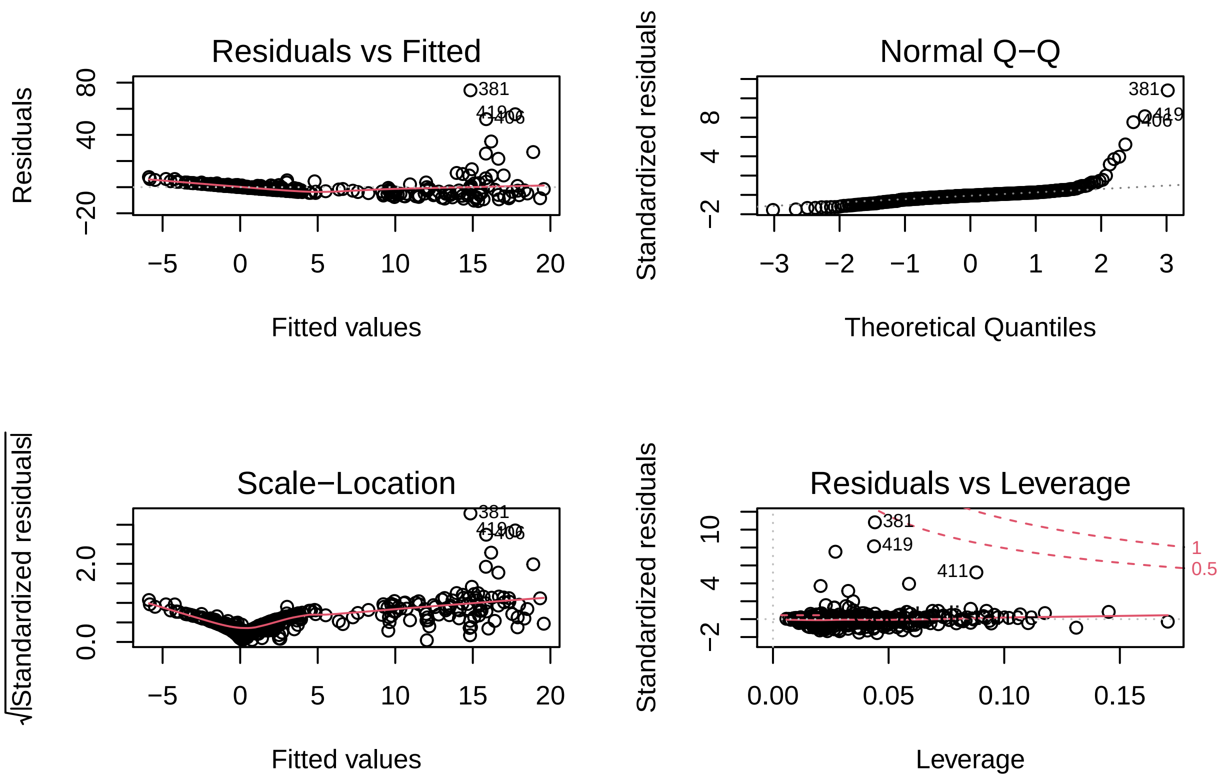
c. (6%)

```
model5 <- lm(CRIM~ZN+INDUS+CHAS+NOX+RM+AGE+DIS+RAD+TAX+PTRATIO+B+LSTAT+MEDV,
             data = boston)
summary(model5)
```

```
##
## Call:
## lm(formula = CRIM ~ ZN + INDUS + CHAS + NOX + RM + AGE + DIS +
##     RAD + TAX + PTRATIO + B + LSTAT + MEDV, data = boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.674  -2.224  -0.458   1.017  74.130
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.992e+01  9.196e+00   2.166  0.03094 *
## ZN           4.680e-02  2.270e-02   2.061  0.03995 *
## INDUS       -4.086e-02  1.030e-01  -0.397  0.69178
## CHAS        -1.144e+00  1.461e+00  -0.783  0.43412
## NOX         -1.318e+01  6.748e+00  -1.954  0.05148 .
## RM           7.206e-01  8.106e-01   0.889  0.37455
```

```
## AGE          5.548e-05  2.261e-02   0.002  0.99804
## DIS         -1.079e+00  3.453e-01  -3.125  0.00192 **
## RAD          6.434e-01  1.056e-01   6.093 2.72e-09 ***
## TAX         -6.205e-03  6.260e-03  -0.991  0.32221
## PTRATIO     -3.447e-01  2.308e-01  -1.493  0.13622
## B           -8.550e-03  4.682e-03  -1.826  0.06864 .
## LSTAT        1.475e-01  9.210e-02   1.601  0.11011
## MEDV        -2.381e-01  7.919e-02  -3.007  0.00282 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.009 on 380 degrees of freedom
## Multiple R-squared:  0.4391, Adjusted R-squared:  0.4199
## F-statistic: 22.88 on 13 and 380 DF,  p-value: < 2.2e-16
```

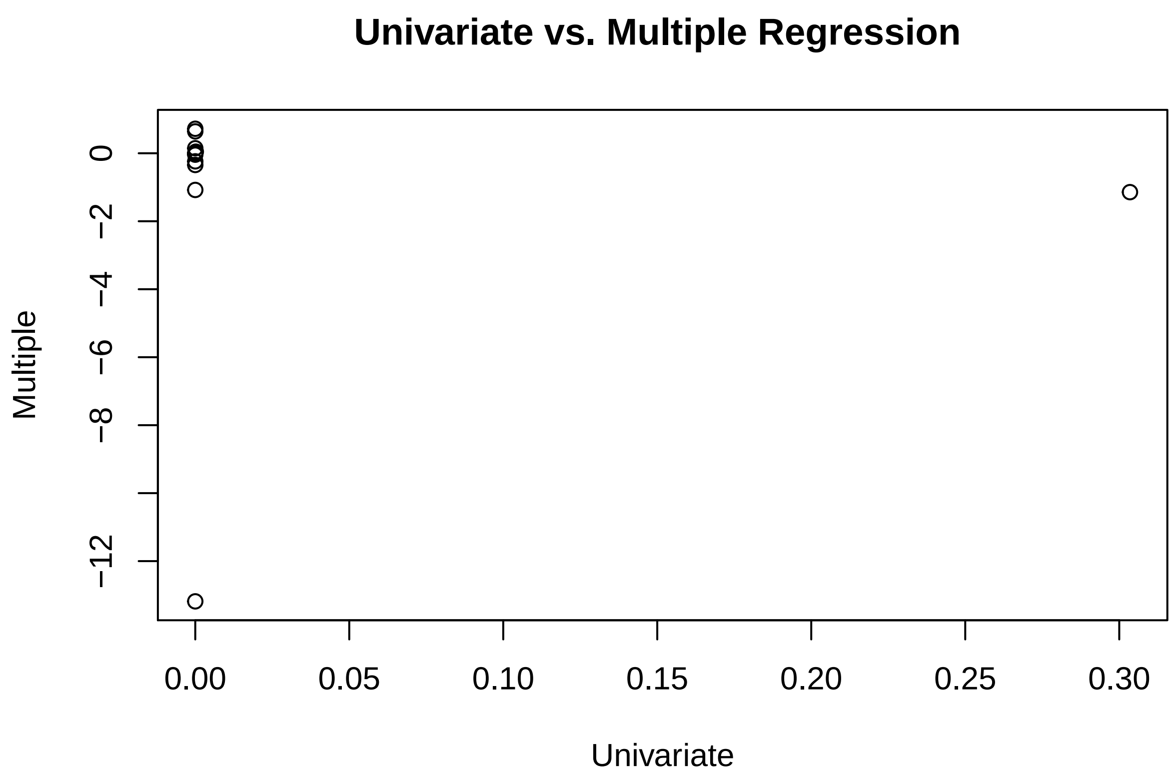
```r
#examining all of them together suggests the most significant predictor is RAD,
#followed by DIS and MEDV.

#we can reject the null hypothesis for RAD,DIS,MEDV and ZN.
#NOX and B both have >90% certainty but that's outside our 95% field.

par(mfrow=c(2,2)) # Change the panel layout to 2 x 2
plot(model5)
```



```r
par(mfrow=c(1,1)) # Change back to 1 x 1

#with the highest R2 value, this model is the better predictor when compared
#to the individual models.
```

d. (6%) What does this plot tell you about the various predictors?

```r
#build the coefficient table,
name <- colnames(boston)[2:14]
univ <- c(coef(summary(boston.crime.lm.ZN))["ZN","Pr(>|t|)"],
coef(summary(boston.crime.lm.INDUS))["INDUS","Pr(>|t|)"],
coef(summary(boston.crime.lm.CHAS))["CHAS","Pr(>|t|)"],
coef(summary(boston.crime.lm.NOX))["NOX","Pr(>|t|)"],
coef(summary(boston.crime.lm.RM))["RM","Pr(>|t|)"],
coef(summary(boston.crime.lm.AGE))["AGE","Pr(>|t|)"],
coef(summary(boston.crime.lm.DIS))["DIS","Pr(>|t|)"],
coef(summary(boston.crime.lm.RAD))["RAD","Pr(>|t|)"],
coef(summary(boston.crime.lm.TAX))["TAX","Pr(>|t|)"],
coef(summary(boston.crime.lm.PTRATIO))["PTRATIO","Pr(>|t|)"],
coef(summary(boston.crime.lm.B))["B","Pr(>|t|)"],
coef(summary(boston.crime.lm.LSTAT))["LSTAT","Pr(>|t|)"],
coef(summary(boston.crime.lm.MEDV))["MEDV","Pr(>|t|)"])
coef_table <- data.frame(name,univ)

coef_table$multi <- model5$coefficients[2:14]


plot(coef_table$univ,coef_table$multi,
     main = "Univariate vs. Multiple Regression ",
     xlab = "Univariate", ylab = "Multiple")
```




```
#the table suggests two of the predictors differ extensively compared
#to the others across comparisons
```

e. (6%) Is there evidence of non-linear association between any of the predictors and the response? Hint: use the poly() function in R. Again, include the code, but not the output for each model in your solution, and instead describe any non-linear trends you uncover.

```
deterfit <- function(column) {
  fit_1 = lm(CRIM~column, data = boston)
  fit_2 = lm(CRIM~poly(column,2), data = boston)
  fit_3 = lm(CRIM~poly(column,3), data = boston)
  fit_4 = lm(CRIM~poly(column,4), data = boston)
  fit_5 = lm(CRIM~poly(column,5), data = boston)
  print(anova(fit_1,fit_2,fit_3,fit_4,fit_5))
}

deterfit(boston$ZN)
```

```
## Analysis of Variance Table
##
## Model 1: CRIM ~ column
## Model 2: CRIM ~ poly(column, 2)
## Model 3: CRIM ~ poly(column, 3)
## Model 4: CRIM ~ poly(column, 4)
## Model 5: CRIM ~ poly(column, 5)
```

```
##   Res.Df   RSS Df Sum of Sq      F  Pr(>F)
## 1    392 32104
## 2    391 31633  1    471.27 5.8008 0.01648 *
## 3    390 31548  1     84.75 1.0431 0.30774
## 4    389 31528  1     19.50 0.2401 0.62443
## 5    388 31523  1      5.79 0.0712 0.78968
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#the poly determinate of ZN indicates that a nonlinear association
#would be more suitable

deterfit(boston$INDUS)
```

```
## Analysis of Variance Table
##
## Model 1: CRIM ~ column
## Model 2: CRIM ~ poly(column, 2)
## Model 3: CRIM ~ poly(column, 3)
## Model 4: CRIM ~ poly(column, 4)
## Model 5: CRIM ~ poly(column, 5)
##   Res.Df   RSS Df Sum of Sq       F    Pr(>F)
## 1    392 28179
## 2    391 27535  1    643.35 10.6844  0.001177 **
## 3    390 25025  1   2510.31 41.6895 3.201e-10 ***
## 4    389 24950  1     74.67  1.2401  0.266147
## 5    388 23363  1   1587.09 26.3572 4.498e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#the poly determinate of INDUS suggests a quartic polynomial would be best

#Chas is a dummy variable and thus has no linear fit.

#deterfit(boston$NOX)
#the poly determinate of NOX suggests a quartic polynomial would be best

#deterfit(boston$RM)
#the poly determinate of RM suggests a cubic polynomial would be best

#deterfit(boston$AGE)
#the poly determinate of age suggests a quartic polynomial would be best

#deterfit(boston$DIS)
#the poly determinate of DIS suggests a quintic polynomial would be best but
#the true fit likley is beyond 6

#deterfit(boston$RAD)
#the poly determinate of RAD suggests a cubic polynomial would be best

#deterfit(boston$TAX)
#the poly determinate of TAX suggests a square polynomial would be best
```

```r
#deterfit(boston$PTRATIO)
#the poly determinate of PRATIO suggests linear is appropriate

#deterfit(boston$B)
#the poly determinate of B suggests linear is appropriate

#deterfit(boston$LSTAT)
#the poly determinate of lstat suggests a square polynomial would be best

#deterfit(boston$MEDV)
#the poly determinate of MEDV suggests quintic polynomial would be best
```

3) Suppose we collect data for a group of students in a statistics class with variables:

a. (5%) Estimate the probability that a student who studies for 32 h, has a PSQI score of 12 and has an undergrad GPA of 3.0 gets an A in the class. Show your work.

```r
#PREDICTION -> y = -7 + 0.1(hours) + 1(undergradgpa) + -0.04(pSQI)
a_predict <- -7 + (0.1*32) + 1*3 + (-0.04*12)
a_predict
```

```
## [1] -1.28
```

```r
#since its less than 0, the prediction would indicate the probability is 0.
```

b. (5%) How many hours would the student in part

(a) need to study to have a 50 % chance of getting an A in the class?

```r
#if y = 0.5,  0.5 = -7 + (0.1*x) + 3 + (-0.04*12)
# 7.5 = 0.1x + 3 -0.48
# 4.5 = 0.1x -0.48
# 4.98 = 0.1x
# 49.8 = x

#thus the student would need to study for 49.8 hours.
```

c. (5%)

```r
#if y = 0.5,  0.5 = -7 + (0.1*x) + 3 + (-0.04*3)
# 7.5 = 0.1x + 3
# 4.5 = 0.1x -0.12
# 4.62 = 0.1x
# 46.2 = x

#thus the student would need to study for 46.2 hours.
```

4)

```r
library("tm")
```

```
## Loading required package: NLP

##
## Attaching package: 'NLP'

## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```r
library("SnowballC")
originaldataset = read.csv("GuardianArticles.csv",stringsAsFactors = FALSE)

# Load the data
docs <- VCorpus(VectorSource(originaldataset$body))
#docs <- Corpus(VectorSource(text))

# Convert the text to lower case, removing whitespace, stopwords, punctuation and
#setting it all to lowercase
docs <- tm_map(docs, content_transformer(tolower))
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removeWords, stopwords("english"))
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, stripWhitespace)
docs <- tm_map(docs, stemDocument)



dtm <- TermDocumentMatrix(docs)
dtm = removeSparseTerms(dtm, 0.99)
m <- as.matrix(dtm)
```

a. Tokenization (20%)

```r
print(originaldataset$body[10])
```

```
## [1] "it is rare to come from an exhibition so buoyed up so ravished and so covetous as i did after s
```

```r
chosenrow <-(m[,10])
chosenrow[which(chosenrow==0)] = NA_character_
demo <- data.frame(chosenrow)
demo <- demo[complete.cases(demo),]
print(demo)
```

```
##  [1] "4"  "1"  "1"  "1"  "1"  "2"  "1"  "1"  "1"  "1"  "1"  "1"  "11" "1"  "1"
## [16] "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "2"  "1"  "1"  "1"  "1"  "1"  "2"
## [31] "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "2"  "1"  "1"  "1"  "1"
## [46] "2"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "4"  "1"  "1"  "1"  "4"  "1"
## [61] "2"  "1"  "1"  "1"  "1"  "1"  "1"  "2"  "1"  "1"  "1"  "1"  "5"  "1"  "1"
```

```
##  [76] "2"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "2"  "1"  "1"
##  [91] "1"  "1"  "2"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "2"  "2"  "1"  "1"
## [106] "3"  "2"  "1"  "1"  "1"  "1"  "1"  "4"  "1"  "2"  "2"  "1"  "1"  "1"  "1"
## [121] "1"  "2"  "3"  "1"  "1"  "1"  "1"  "3"  "1"  "2"  "1"  "1"  "2"  "1"  "1"
## [136] "1"  "2"  "1"  "1"  "1"  "3"  "1"  "2"  "2"  "2"  "3"  "1"  "2"  "1"  "1"
## [151] "1"  "1"  "8"  "1"  "1"  "2"  "1"  "3"  "1"  "1"  "1"  "1"  "1"  "1"  "1"
## [166] "1"  "1"  "1"  "1"  "1"  "1"  "6"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"
## [181] "1"  "1"  "2"  "1"  "1"  "1"  "1"  "1"  "3"  "1"  "1"  "1"  "1"  "2"  "1"
## [196] "1"  "1"  "2"  "1"  "1"  "3"  "1"  "1"  "2"  "1"  "2"  "1"  "1"  "1"  "1"
## [211] "2"  "2"  "1"  "1"  "1"  "1"  "1"  "3"  "1"  "1"  "1"  "1"  "2"  "2"  "1"
## [226] "1"  "3"  "1"  "1"  "1"
```

b. Classification (20%)

```r
library("tidyverse")
library(dplyr)
# Loading package
library(e1071)
library(caTools)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
article_type <- recode(originaldataset$section, 'culture'=0,
                       'sport'=1, 'technology'=2, 'world'=3,
                       'artanddesign'=4,'business'=5,.default = 7)

max(article_type)
```

```
## [1] 5
```

```r
features <- t(m)

correlationMatrix <- cor(features)

highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.75)

features <- t(m)
features <- features[,-highlyCorrelated]
feature_set <- cbind(features,article_type)

df = as.data.frame(feature_set)
df <- as.data.frame(feature_set)
```

```r
fit = glm(article_type ~.,family ="gaussian",data = df)
  temp <- varImp(fit)
library(dplyr)
n <- as.data.frame(temp)

#keep the top 80.0198 by importance%
n_sorted <- head(arrange(n,desc(Overall)),2000)
names <- colnames(t(n_sorted))

#somehow we lost some
names <- names[-622]
print(names[1625])
```

```
## [1] "`repeat`"
```

```r
names <- names[-1625]
print(names[1645])
```

```
## [1] "`next`"
```

```r
names <- names[-1645]
output <- df[, names[1:1997]]

#re-attach the feature sets
#make it binary
output[output > 0] <- 1
output <- cbind(output,article_type)
output$article_type <- as.factor(output$article_type)
set.seed(1234)

samp <- createDataPartition(output[,1], p = 0.8, list = FALSE)
training <- output[samp,]
testing <- output[-samp,]

classifier <- naiveBayes(article_type ~ ., data = training)
y_pred <- predict(classifier,testing[1:1997])
cm <- table(testing$article_type, y_pred)

confusionMatrix(cm)
```

```
## Confusion Matrix and Statistics
##
##    y_pred
##       0   1   2   3   4   5
##   0 236  14  19  13  81   5
##   1  20 335   8   7   2   5
##   2  21   7 285  18   4  62
##   3  16  11  16 271  24  51
##   4  36  10  11   9 317  10
##   5   3   4  29  17   2 312
##
## Overall Statistics
```

```
##
##                Accuracy : 0.7665
##                  95% CI : (0.7486, 0.7837)
##     No Information Rate : 0.1942
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7198
##
##  Mcnemar's Test P-Value : 4.15e-09
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity            0.7108   0.8793   0.7745   0.8090   0.7372   0.7011
## Specificity            0.9326   0.9780   0.9418   0.9397   0.9592   0.9702
## Pos Pred Value         0.6413   0.8886   0.7179   0.6967   0.8066   0.8501
## Neg Pred Value         0.9501   0.9760   0.9562   0.9664   0.9405   0.9309
## Prevalence             0.1449   0.1663   0.1606   0.1462   0.1877   0.1942
## Detection Rate         0.1030   0.1462   0.1244   0.1183   0.1384   0.1362
## Detection Prevalence   0.1606   0.1646   0.1733   0.1698   0.1715   0.1602
## Balanced Accuracy      0.8217   0.9286   0.8581   0.8743   0.8482   0.8357
```