Exercise 6

Konstantinos Vakalopoulos

2022-12-15

Premilinary work

First, the Auto from the ISLR package were loaded,

```
library(ISLR)

data("Auto")
data <- Auto
```

the missing values were omitted, the variable "name" was deleted and the variables "cylinders" and "origin" were transformed to factors.

```
data <- na.omit(data)
data <- data[,-9]
data$cylinders <- as.factor(data$cylinders)
data$origin <- as.factor(data$origin)
```

More information about the data are presented to the plots using the library Hmisc.

```
library(Hmisc)
```

```
hist.data.frame(data)
```

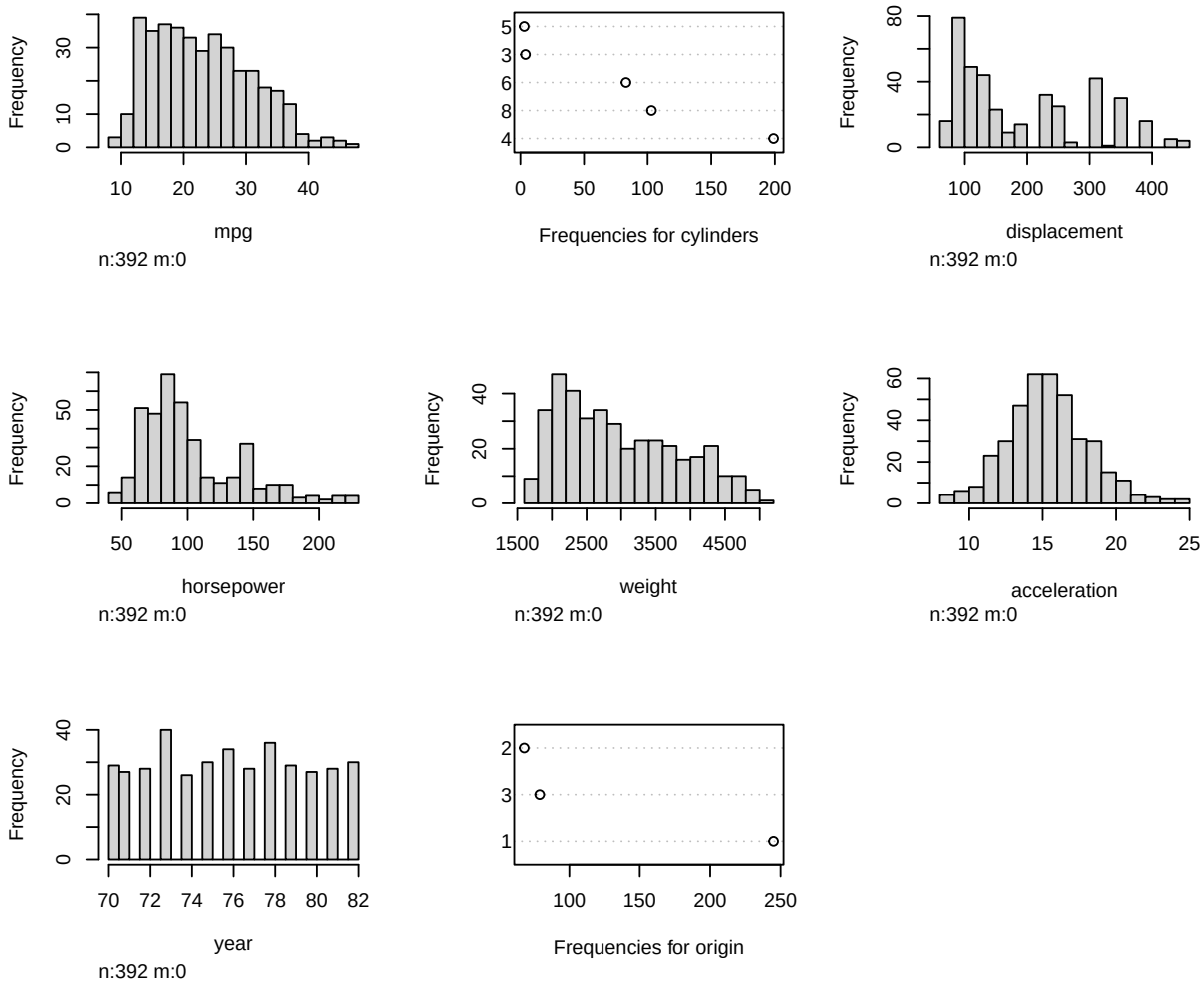

Figure 1: Histograms of the variables

Furthermore, the numeric variables "mpg", "weight" and "horsepower" were transformed using the log function. Below are presented the plots of the transformed data.

```
data$mpg <- log(data$mpg)
data$weight <- log(data$weight)
data$horsepower <- log(data$horsepower)
```

```
hist.data.frame(data)
```

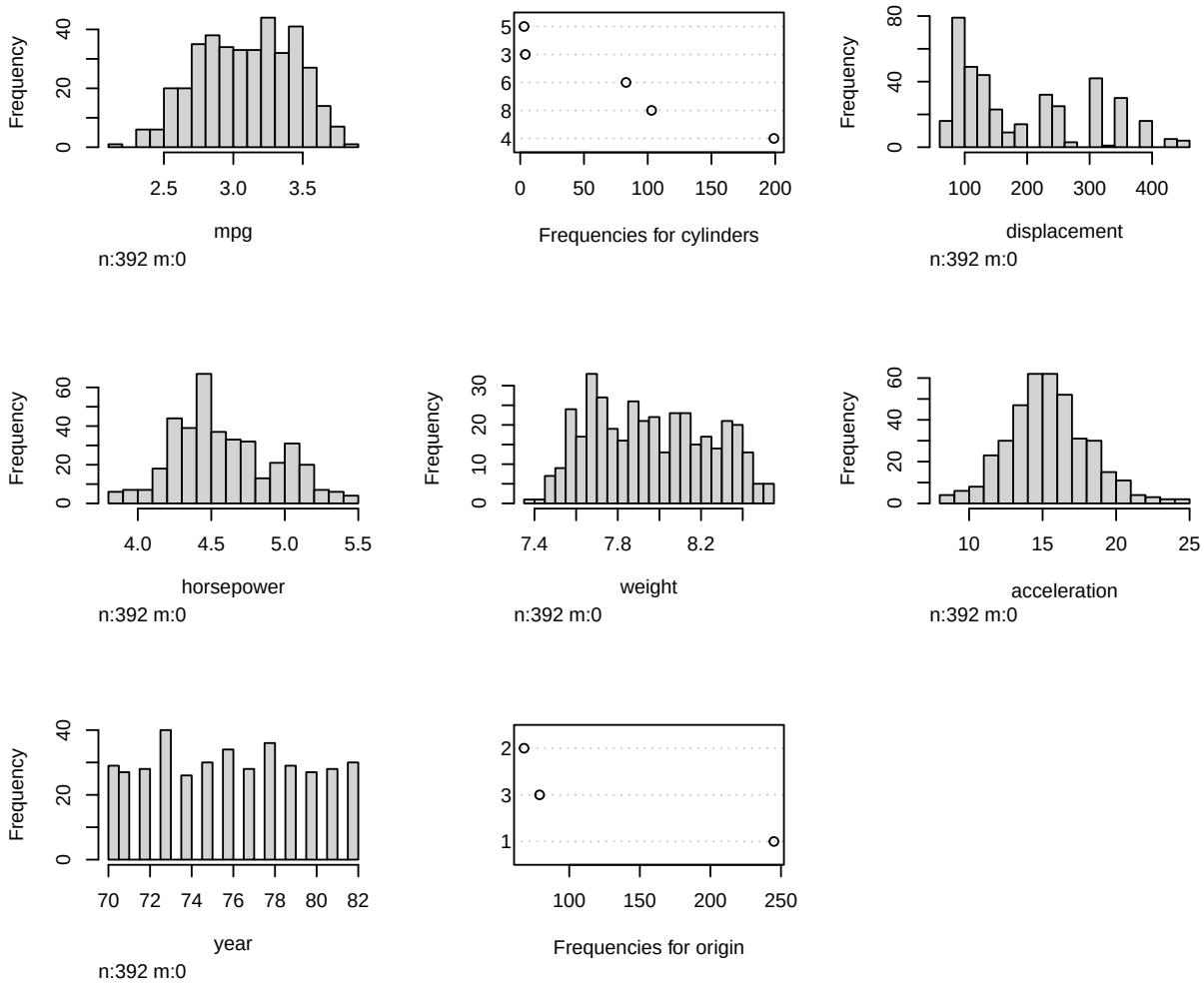

Figure 2: Histograms of the variables

Question 1(a)

For this part of the exercise, the variable "mpg" was considered the response and "acceleration" the explanatory variable. Also, three different data sets were created based on the variable "origin".

```
newdata <- subset(data, select=c("mpg", "acceleration"))
American <- newdata[which(data$origin==1),]
European <- newdata[which(data$origin==2),]
Japanese <- newdata[which(data$origin==3),]
```

For the desired number of degrees of freedom, smoothing splines were used in each of the data set using cross validation. In R, the library splines was used.

```
library(splines)

American.df <-  smooth.spline(American$acceleration, American$mpg, cv=TRUE)$df
```

```
European.df <-  smooth.spline(European$acceleration, European$mpg, cv=TRUE)$df
Japanese.df <-  smooth.spline(Japanese$acceleration, Japanese$mpg, cv=TRUE)$df
```

First the data were sorted according to the variable "acceleration" and based on the degrees of freedom, the B-splines were calculated using the bs() function for each data set and the lm() to fit the model.

```
American <- American[order(American$acceleration),]
European <- European[order(European$acceleration),]
Japanese <- Japanese[order(Japanese$acceleration),]

lm.American <- lm(mpg ~ bs(acceleration, df=American.df), data = American)
lm.European <- lm(mpg ~ bs(acceleration, df=European.df), data = European)
lm.Japanese <- lm(mpg ~ bs(acceleration, df=Japanese.df), data = Japanese)
```

The plots "mpg" vs "acceleration" and the fitted lines are presented below.

```
par(mfrow = c(2, 2))
plot(American$acceleration, American$mpg, xlab="Acceleration", ylab="MPG",
     main = "American")
lines(American$acceleration, fitted(lm.American), col="blue")

plot(European$acceleration, European$mpg, xlab="Acceleration", ylab="MPG",
     main = "European")
lines(European$acceleration, fitted(lm.European), col="blue")

plot(Japanese$acceleration, Japanese$mpg, xlab="Acceleration", ylab="MPG",
     main = "Japanese")
lines(Japanese$acceleration, fitted(lm.Japanese), col="blue")
```

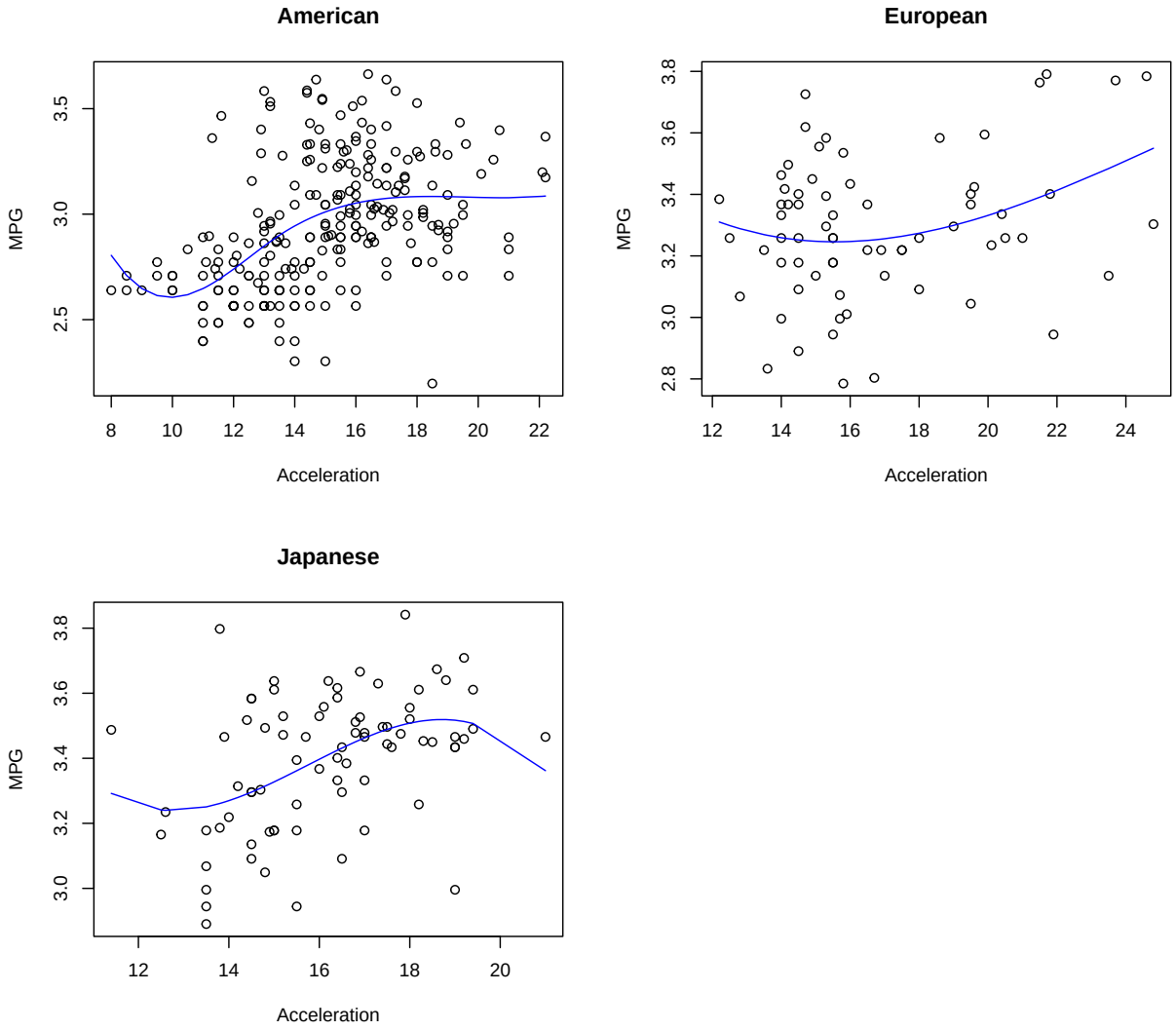

Figure 3: mpg versus accelerations

Question 1(b)

The same procedure was done for the natural cubic splines, but this time the function ns() was used. Furthermore, the degrees of freedom were rounded due to an error provided by the function ns()

```
American.df <- round(American.df, digits = 0)
European.df <- round(European.df, digits = 0)
Japanese.df <- round(Japanese.df, digits = 0)

lm.American <- lm(mpg ~ ns(acceleration, df=American.df), data = American)
lm.European <- lm(mpg ~ ns(acceleration, df=European.df), data = European)
lm.Japanese <- lm(mpg ~ ns(acceleration, df=Japanese.df), data = Japanese)
```

The plots “mpg” vs “acceleration” and the fitted lines are presented below.

```
par(mfrow = c(2, 2))
plot(American$acceleration, American$mpg, xlab="Acceleration", ylab="MPG",
     main = "American")
lines(American$acceleration, fitted(lm.American), col="blue")

plot(European$acceleration, European$mpg, xlab="Acceleration", ylab="MPG",
     main = "European")
lines(European$acceleration, fitted(lm.European), col="blue")

plot(Japanese$acceleration, Japanese$mpg, xlab="Acceleration", ylab="MPG",
     main = "Japanese")
lines(Japanese$acceleration, fitted(lm.Japanese), col="blue")
```

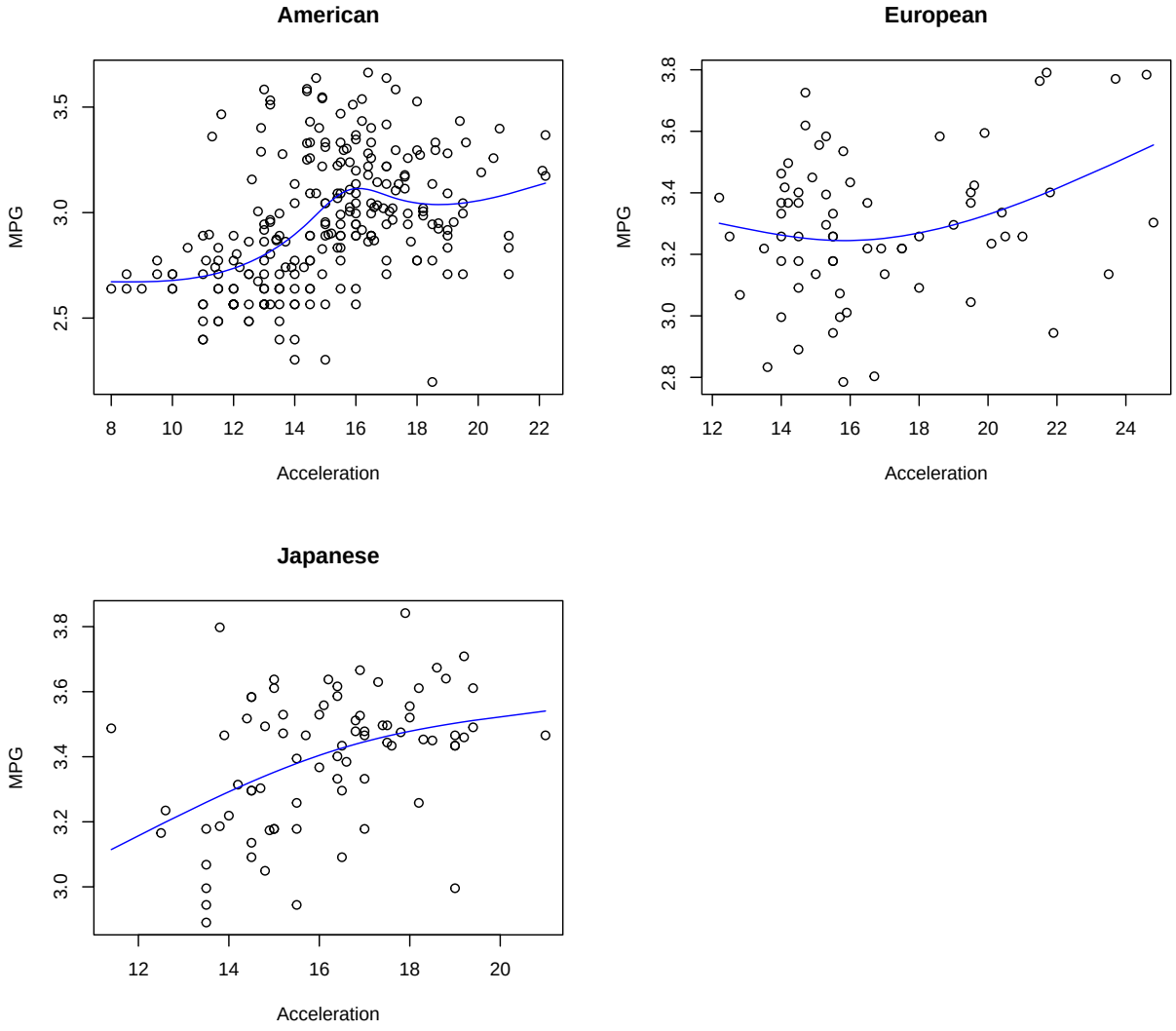

Figure 4: mpg versus accelerations

Question 1(c)

For the smoothing splines, the sorting is not required. However, the data were kept sorted. As in the question 1(a) the function smooth.spline() was used with cross validation for the optimal degrees of freedom.

```
American.smooth <-  smooth.spline(American$acceleration, American$mpg, cv=TRUE)
European.smooth <-  smooth.spline(European$acceleration, European$mpg, cv=TRUE)
Japanese.smooth <-  smooth.spline(Japanese$acceleration, Japanese$mpg, cv=TRUE)
```

The plots "mpg" vs "acceleration" and the fitted lines are presented below.

```
par(mfrow = c(2, 2))
plot(American$acceleration, American$mpg, xlab="Acceleration", ylab="MPG",
     main = "American")
lines(American.smooth, col="blue")

plot(European$acceleration, European$mpg, xlab="Acceleration", ylab="MPG",
     main = "European")
lines(European.smooth, col="blue")

plot(Japanese$acceleration, Japanese$mpg, xlab="Acceleration", ylab="MPG",
     main = "Japanese")
lines(Japanese.smooth, col="blue")
```

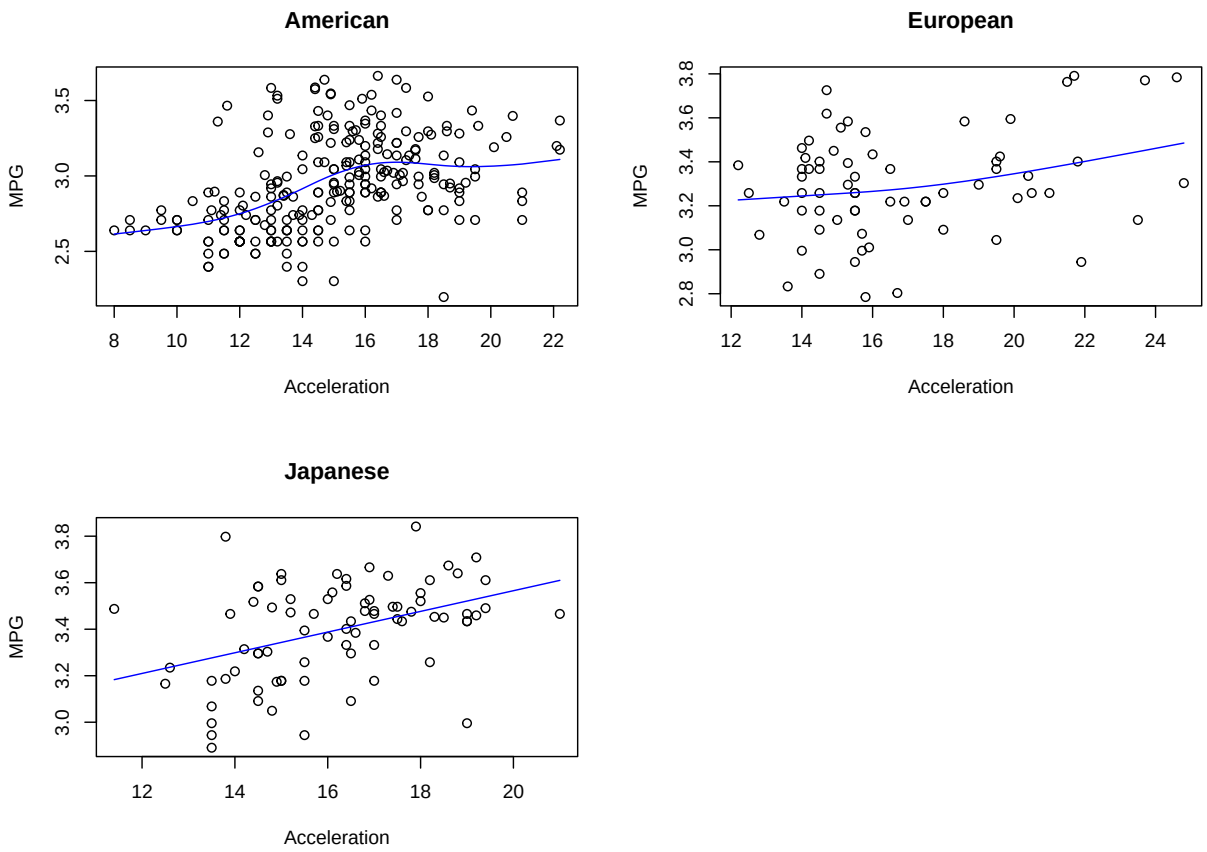

Figure 5: mpg versus accelerations

Question 2(a)

The data were split randomly into train and test set (2/3 train and 1/3 test).

```
set.seed(12223236)
n <- nrow(data)
train <- sample(1:n,round(n*2/3))
test <- (1:n)[-train]
```

For the linear model with natural cubic splines the function ns() from the library splines and the lm() function was used. Also, in the model the variables cylinders and origin were entered in the usual way without splines because they are categorical variables.

```
library(splines)
model <- lm(mpg~cylinders+ns(displacement,4)+ns(horsepower,4)
            +ns(weight,4)+ns(acceleration,4)+ns(year,4)+origin, data=data,subset=train)
```

The summary of the model is presented below.

```
summary(model)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + ns(displacement, 4) + ns(horsepower,
##     4) + ns(weight, 4) + ns(acceleration, 4) + ns(year, 4) +
##     origin, data = data, subset = train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.278172 -0.057671  0.005659  0.052539  0.253807
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            3.21674    0.13041  24.667  < 2e-16 ***
## cylinders4             0.49025    0.08728   5.617 5.49e-08 ***
## cylinders5             0.65546    0.11906   5.505 9.66e-08 ***
## cylinders6             0.48493    0.09447   5.133 5.99e-07 ***
## cylinders8             0.52994    0.11553   4.587 7.32e-06 ***
## ns(displacement, 4)1 -0.20330    0.10151  -2.003 0.046363 *
## ns(displacement, 4)2 -0.46531    0.12299  -3.783 0.000197 ***
## ns(displacement, 4)3 -0.50773    0.17513  -2.899 0.004097 **
## ns(displacement, 4)4 -0.03966    0.12555  -0.316 0.752349
## ns(horsepower, 4)1   -0.21850    0.08124  -2.689 0.007671 **
## ns(horsepower, 4)2   -0.24992    0.09143  -2.734 0.006745 **
## ns(horsepower, 4)3   -0.77702    0.16804  -4.624 6.22e-06 ***
## ns(horsepower, 4)4   -0.67215    0.10060  -6.681 1.71e-10 ***
## ns(weight, 4)1       -0.13539    0.08800  -1.539 0.125250
## ns(weight, 4)2       -0.17960    0.09640  -1.863 0.063706 .
## ns(weight, 4)3       -0.24332    0.17189  -1.416 0.158239
## ns(weight, 4)4       -0.37923    0.09868  -3.843 0.000157 ***
## ns(acceleration, 4)1 -0.08190    0.07933  -1.032 0.302952
## ns(acceleration, 4)2 -0.10367    0.05854  -1.771 0.077894 .
## ns(acceleration, 4)3 -0.07778    0.16307  -0.477 0.633801
```

```
## ns(acceleration, 4)4 -0.10471    0.08689  -1.205 0.229394
## ns(year, 4)1          0.02633    0.03161   0.833 0.405807
## ns(year, 4)2          0.21547    0.03092   6.968 3.25e-11 ***
## ns(year, 4)3          0.16566    0.05596   2.960 0.003389 **
## ns(year, 4)4          0.31817    0.02894  10.992  < 2e-16 ***
## origin2               0.01721    0.02570   0.669 0.503877
## origin3               0.02180    0.02471   0.882 0.378440
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09525 on 234 degrees of freedom
## Multiple R-squared:  0.9329, Adjusted R-squared:  0.9255
## F-statistic: 125.2 on 26 and 234 DF,  p-value: < 2.2e-16
```

Regarding the categorical variables, only the "cylinders" variable is significant based on the p value. Regarding the rest of the variables, can be seen that we have a block of 4 variables (artificial variables) for each input variable. Thus, based on the summary table and the p values we can determine if the block is significant or not. For the variable "displacement" 3 of the 4 basis functions are significant and thus we have to keep all the block. We cannot remove the fourth basis function and the keep the rest. Either we keep or remove all the block. Same thing applies for the variable "year", "weight" and "horsepower". Finally, for the "acceleration" variable non of the 4 basis functions are significant and this variable could be removed from the model.

The RMSE for the test set is:

```
pred <- predict(model,newdata=data[test,])
sqrt(mean((data$mpg[test]-pred)^2))
```

```
## [1] 0.1223106
```

Question 2(b)

In this part of the exercise, the stepwise variable selection was applied using the function step().

```
model.step <-  step(model,direction="both")
```

The summary of the model is presented below.

```
summary(model.step)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + ns(displacement, 4) + ns(horsepower,
##     4) + ns(weight, 4) + ns(acceleration, 4) + ns(year, 4), data = data,
##     subset = train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.276583 -0.058108  0.004435  0.055330  0.262365
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)           3.23427    0.12859  25.153  < 2e-16 ***
## cylinders4            0.49545    0.08630   5.741 2.87e-08 ***
## cylinders5            0.67273    0.11533   5.833 1.78e-08 ***
## cylinders6            0.49953    0.09243   5.405 1.59e-07 ***
## cylinders8            0.54249    0.11429   4.747 3.58e-06 ***
## ns(displacement, 4)1 -0.24601    0.08799  -2.796 0.005604 **
## ns(displacement, 4)2 -0.51150    0.11078  -4.617 6.39e-06 ***
## ns(displacement, 4)3 -0.56164    0.16379  -3.429 0.000715 ***
## ns(displacement, 4)4 -0.07927    0.11652  -0.680 0.496986
## ns(horsepower, 4)1   -0.21022    0.08050  -2.611 0.009596 **
## ns(horsepower, 4)2   -0.23575    0.08975  -2.627 0.009189 **
## ns(horsepower, 4)3   -0.76174    0.16673  -4.569 7.91e-06 ***
## ns(horsepower, 4)4   -0.66308    0.09972  -6.650 2.02e-10 ***
## ns(weight, 4)1       -0.13342    0.08748  -1.525 0.128566
## ns(weight, 4)2       -0.17319    0.09470  -1.829 0.068689 .
## ns(weight, 4)3       -0.24923    0.17121  -1.456 0.146789
## ns(weight, 4)4       -0.37245    0.09742  -3.823 0.000169 ***
## ns(acceleration, 4)1 -0.07995    0.07910  -1.011 0.313138
## ns(acceleration, 4)2 -0.10240    0.05743  -1.783 0.075872 .
## ns(acceleration, 4)3 -0.06990    0.16235  -0.431 0.667175
## ns(acceleration, 4)4 -0.09773    0.08509  -1.148 0.251943
## ns(year, 4)1          0.02485    0.03149   0.789 0.430737
## ns(year, 4)2          0.21523    0.03062   7.029 2.22e-11 ***
## ns(year, 4)3          0.16394    0.05553   2.952 0.003472 **
## ns(year, 4)4          0.31759    0.02861  11.100  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09501 on 236 degrees of freedom
## Multiple R-squared:  0.9327, Adjusted R-squared:  0.9258
## F-statistic: 136.2 on 24 and 236 DF,  p-value: < 2.2e-16
```

Based on the summary table it is observed that the variable "origin" has been removed from the model. However, the rest of the variables, regarding the significance, are similar to the full model from the question 2(a).

The RMSE for the test set is:

```
pred <- predict(model.step,newdata=data[test,])
sqrt(mean((data$mpg[test]-pred)^2))
```

```
## [1] 0.123132
```

Question 2(c)

The variables from the reduced model, despite the factor variable cylinders, are: "displacement", "horsepower", "weight", "acceleration" and "year".

```
X <- model.matrix(model.step)
```

In figure 6, the variable "displacement" against their estimated values.

```
plot(data[train,]$displacement,X[,6:9]%*%model.step$coefficients[6:9],
     xlab="Displacement", ylab="Estimated Displacement")
```

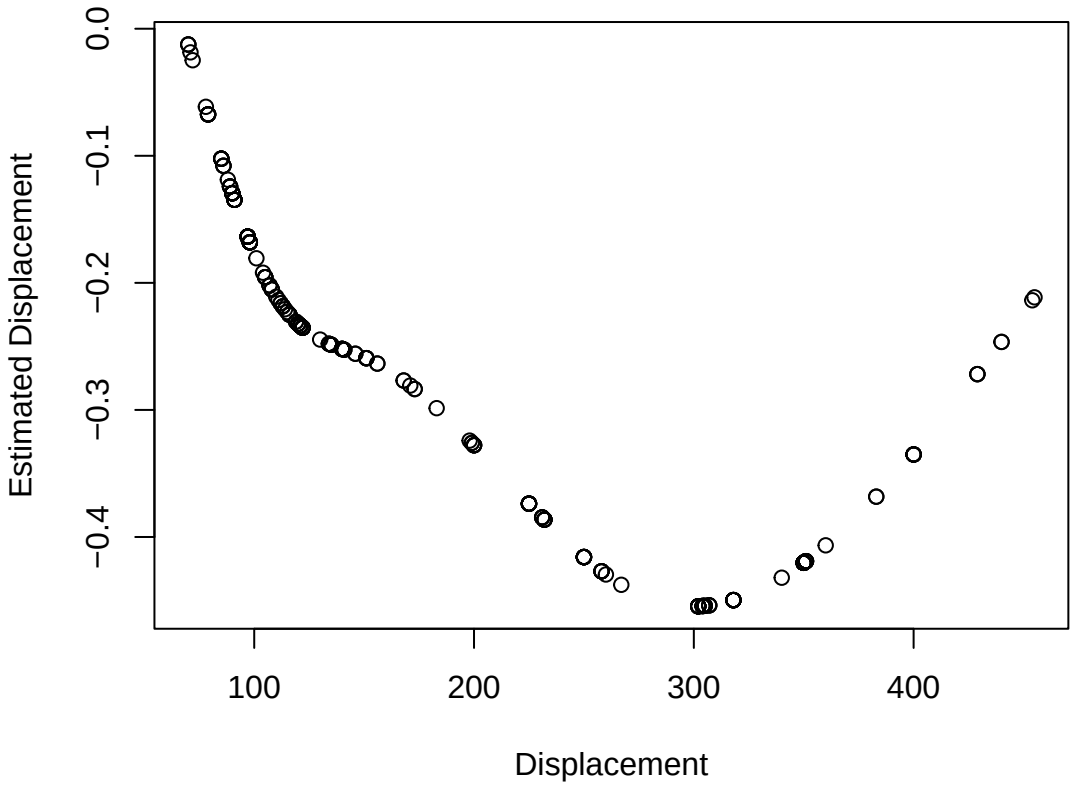

Figure 6: Variable Displacement

Based on the plot, it is observed that from 80 to 300 both the displacement and the estimated values increase. Also, between 100 and 200 a small non linearity can be seen. The biggest difference in the plot is observed when the variable displacement exceeds the value 300. From that point onwards the increase in displacement is equivalent to an increase in estimated values

In figure 7, the variable "horsepower" against their estimated values.

```
plot(data[train,]$horsepower,X[,10:13]%*%model.step$coefficients[10:13],
     xlab="Horsepower", ylab="Estimated Horsepower")
```

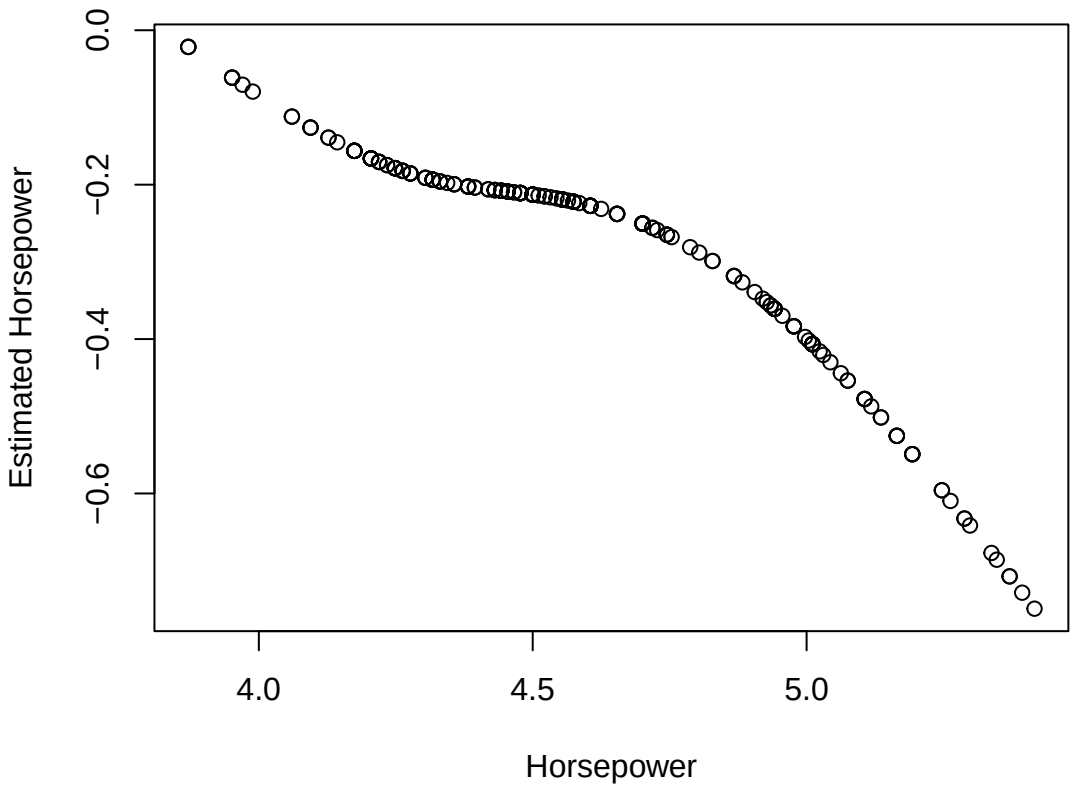

Figure 7: Variable Horsepower

In the figure above, similar to "displacement" variable, while the horsepower increases, the estimates values decrease. Also, non linearity is observed from 4.2 to 5.0.

In figure 8, the variable "weight" against their estimated values.

```
plot(data[train,]$weight,X[,14:17]%*%model.step$coefficients[14:17],
     xlab="Weight", ylab="Estimated Weight")
```

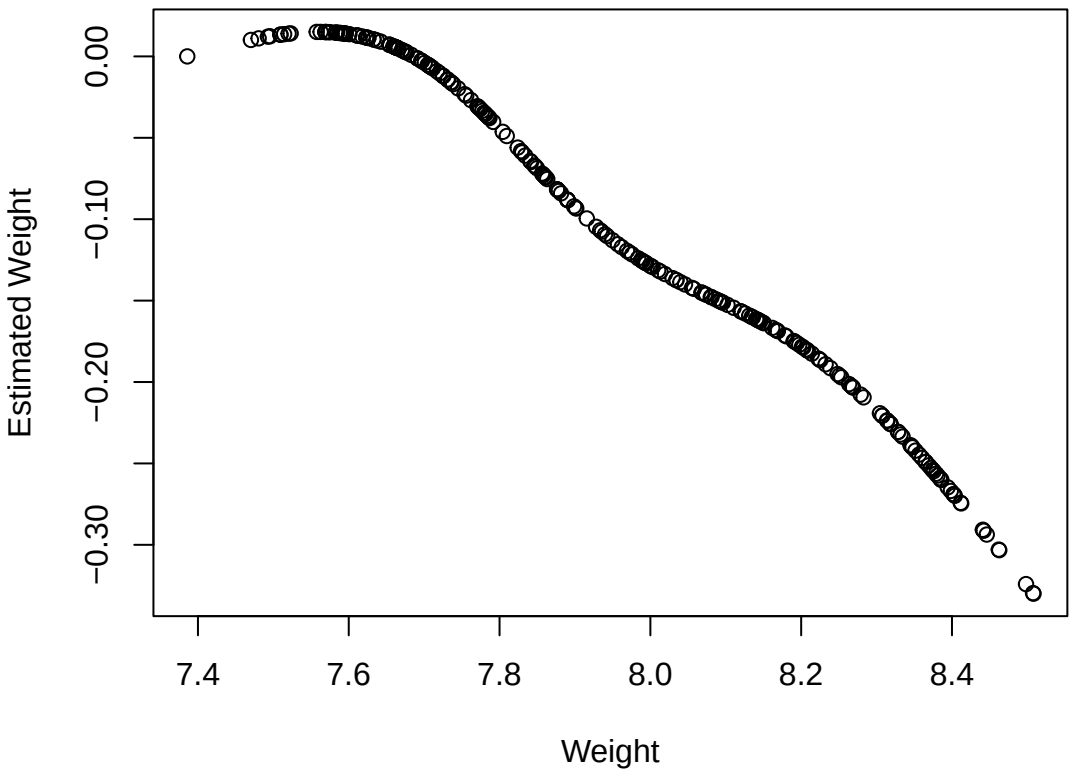

Figure 8: Variable Weight

In figure 8, from 7.4 to 7.7 the weight increases but the estimated values are towards zero. Consequently, negative linear relationship is observed between the two variables in the whole range of weight values. However, there are still some points in the plot where non linearity takes place.

In figure 9, the variable "acceleration" against their estimated values.

```
plot(data[train,]$acceleration,X[,18:21]%*%model.step$coefficients[18:21],
     xlab="Acceleration", ylab="Estimated Acceleration")
```

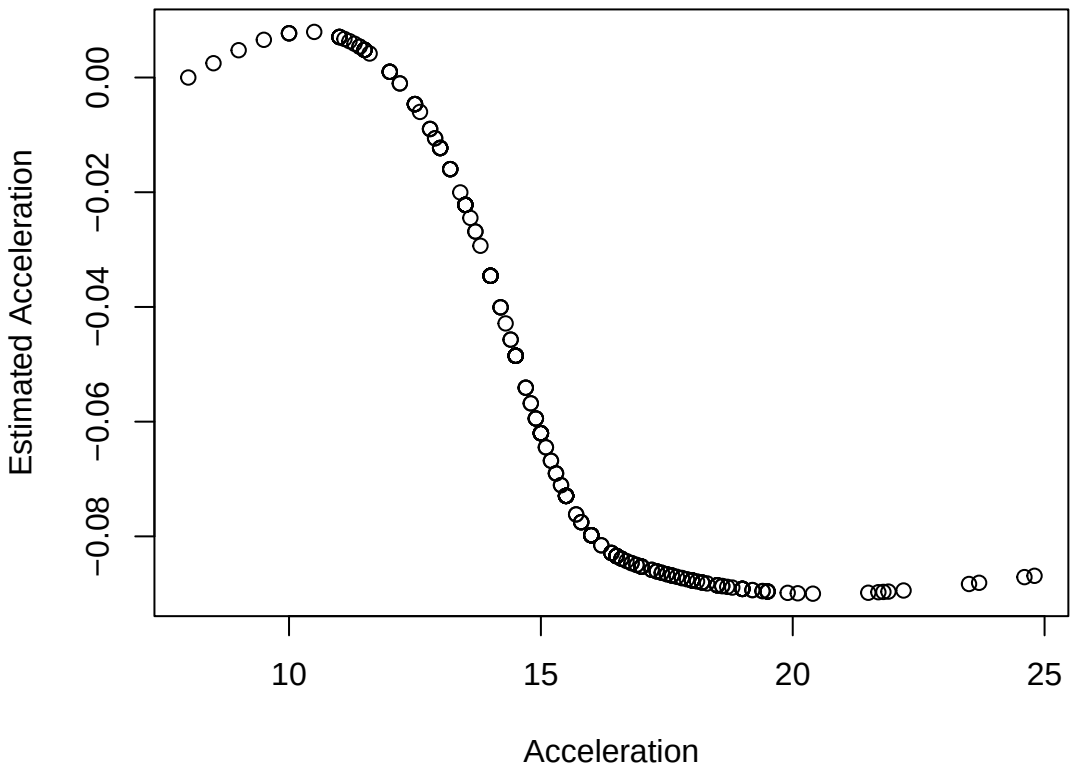

Figure 9: Variable Acceleration

In figure 9, the estimated values for the variable "acceleration" fluctuate over the whole range. More specifically, firstly there is an increase in values. Then, the estimated values start to decrease and for acceleration from 12.5 to 16 this decrease is linear. After that, the estimated are stable towards -0.75 but while the acceleration increases, the estimated values tend to increase.

In figure 10, the variable "year" against their estimated values.

```
plot(data[train,]$year,X[,22:25]%*%model.step$coefficients[22:25],
     xlab="Year", ylab="Estimated Year")
```

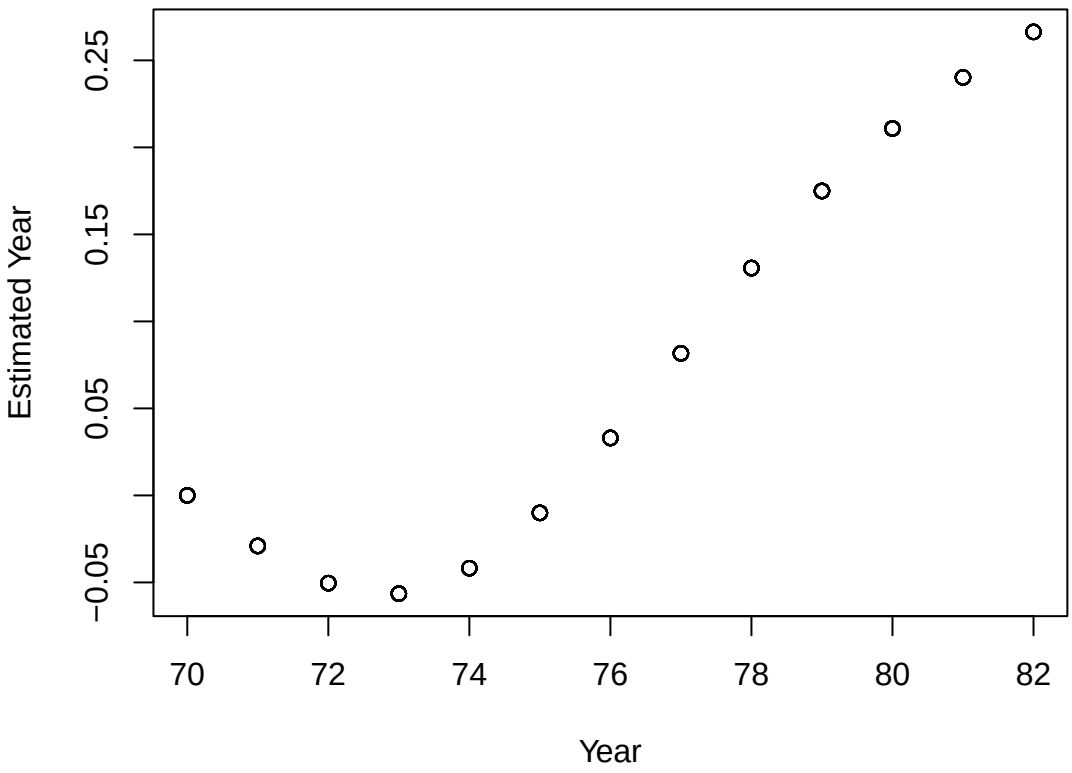

Figure 10: Variable Year

Based on the figure 10, for the years 70-73 the estimated values are decreasing. Although, from the year 74 to 82 the estimated values are increasing linearly.