Exercise #4

Linear regression with R

Brian Schweigler; 16-102-071

06/04/2022

Preliminaries

Loading the education dataset:

```r
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

education_df = read.csv("EducationBis.txt", header = TRUE, sep = "\t", comment.char = "#")
```

Check for outliers / erronous data input:

```r
range(education_df$Wage)
```

```
## [1] 2047.2 8453.5
```

```r
range(education_df$Education)
```

```
## [1]  5 22
```

```r
unique(education_df$Gender)
```

```
## [1] "male"   "female"
```

No outliers directly visible, Gender is encoded as binary, years of education and wage range seem reasonable without additional information.

1. Build two linear models, one for men and one for women, and use the Education variable to explain the Wage. Describe the results of the output provided by the function lm() for both models.

Splitting by gender, then creating the linear model:

```
female = education_df[which(education_df$Gender == "female"),]
male = education_df[which(education_df$Gender == "male"),]
lm_male = lm(male$Wage ~ male$Education)
lm_female = lm(female$Wage ~ female$Education)

summary(lm_female)
```

```
##
## Call:
## lm(formula = female$Wage ~ female$Education)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -220.461  -78.869   -4.028   73.437  271.939
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -563.61      37.50  -15.03   <2e-16 ***
## female$Education   397.54       2.58  154.10   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 104.3 on 196 degrees of freedom
## Multiple R-squared:  0.9918, Adjusted R-squared:  0.9918
## F-statistic: 2.375e+04 on 1 and 196 DF,  p-value: < 2.2e-16
```

```
summary(lm_male)
```

```
##
## Call:
## lm(formula = male$Wage ~ male$Education)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -243.654  -74.152    6.096   70.923  279.797
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      24.199     27.937   0.866    0.387
## male$Education  398.250      1.919 207.559   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 97.41 on 297 degrees of freedom
## Multiple R-squared:  0.9932, Adjusted R-squared:  0.9931
## F-statistic: 4.308e+04 on 1 and 297 DF,  p-value: < 2.2e-16
```

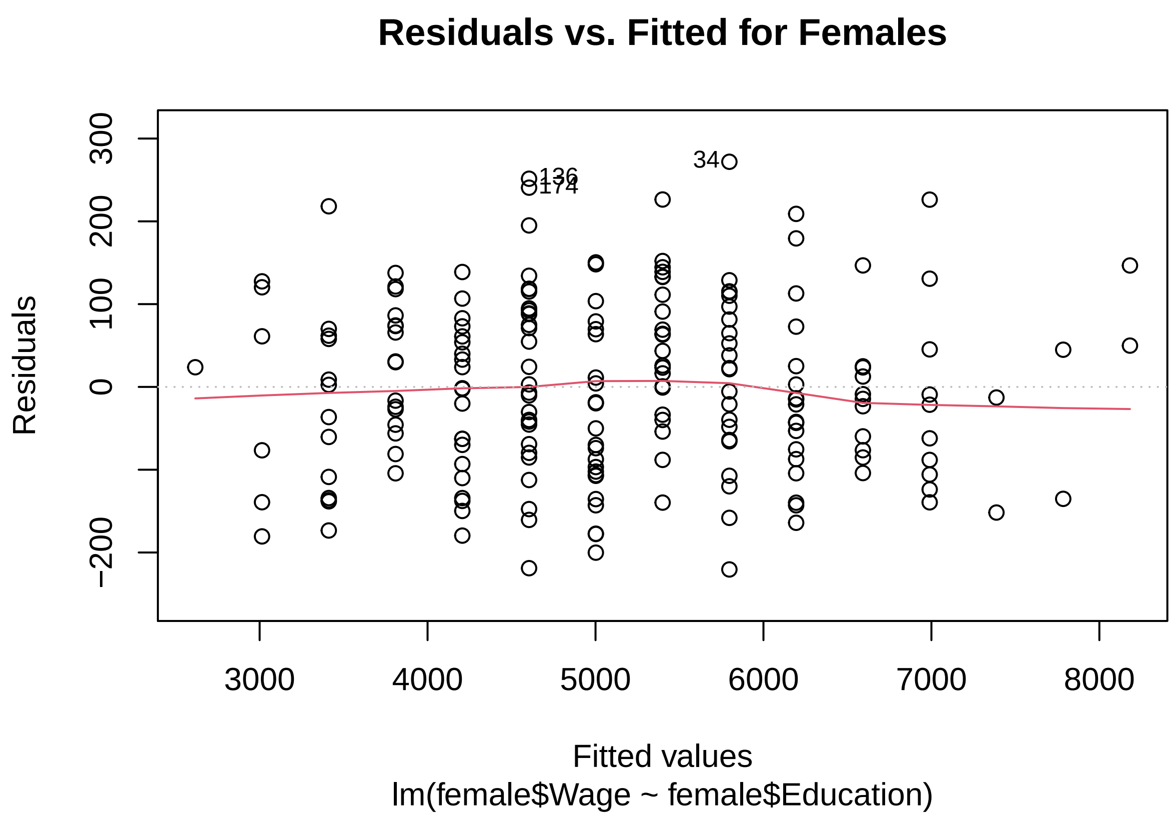
The `female$Education` has a slope of 397.54, while `male$Education` of 398.25, so both are slightly below 400. This also directly answers question 2, yes the slopes are significantly different from 0.

Furthermore, the y-intersect is at -563.61 for females and 24.20 for males

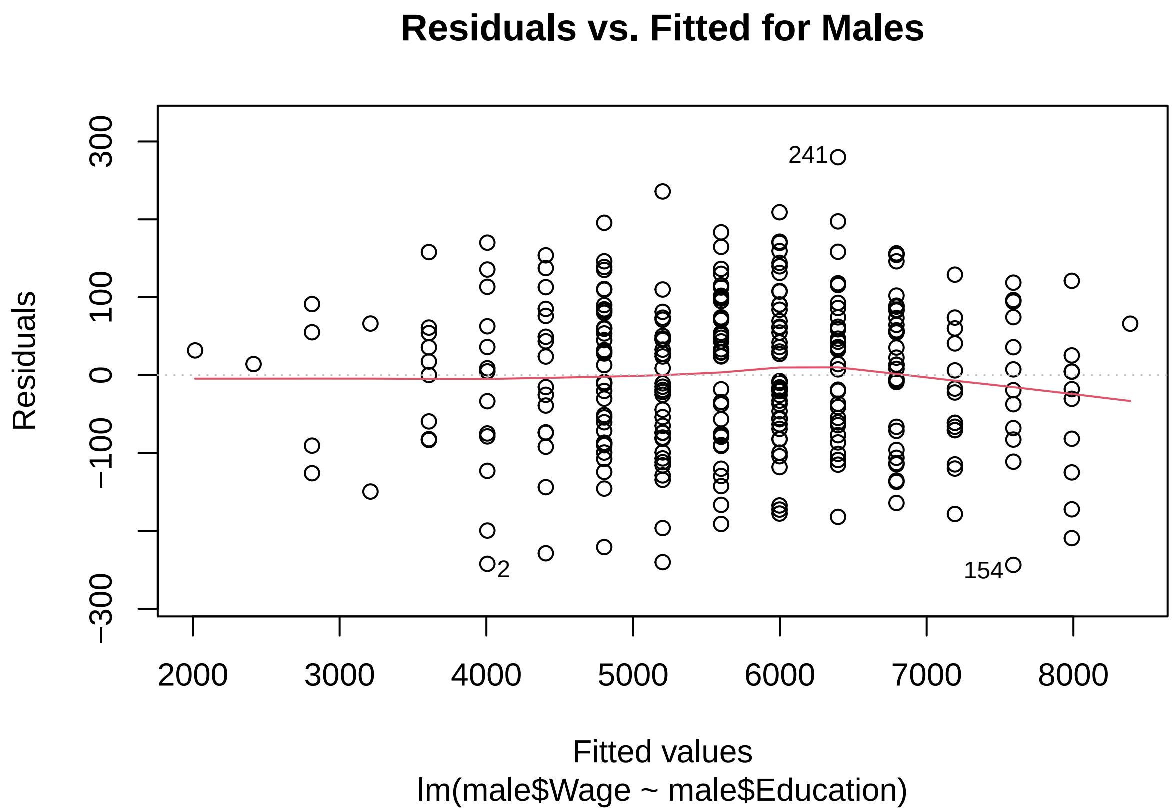
2. Are the two slopes significantly different from 0?

As seen in question 1, yes they are siginicantly different from 400. But, as a treat, have the Residuals vs. fitted and Normal Q-Q plot for both linear models below:

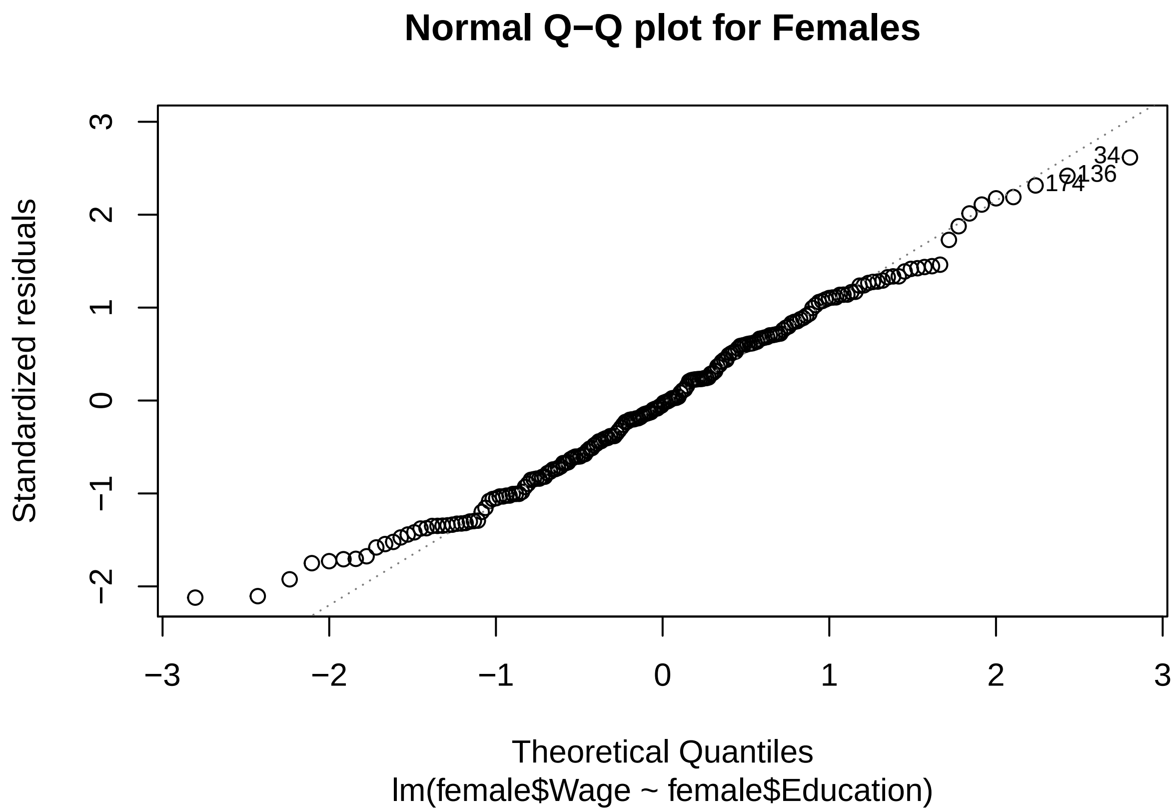
```
plot(lm_female, which = c(1), main = "Residuals vs. Fitted for Females", caption = "")
```



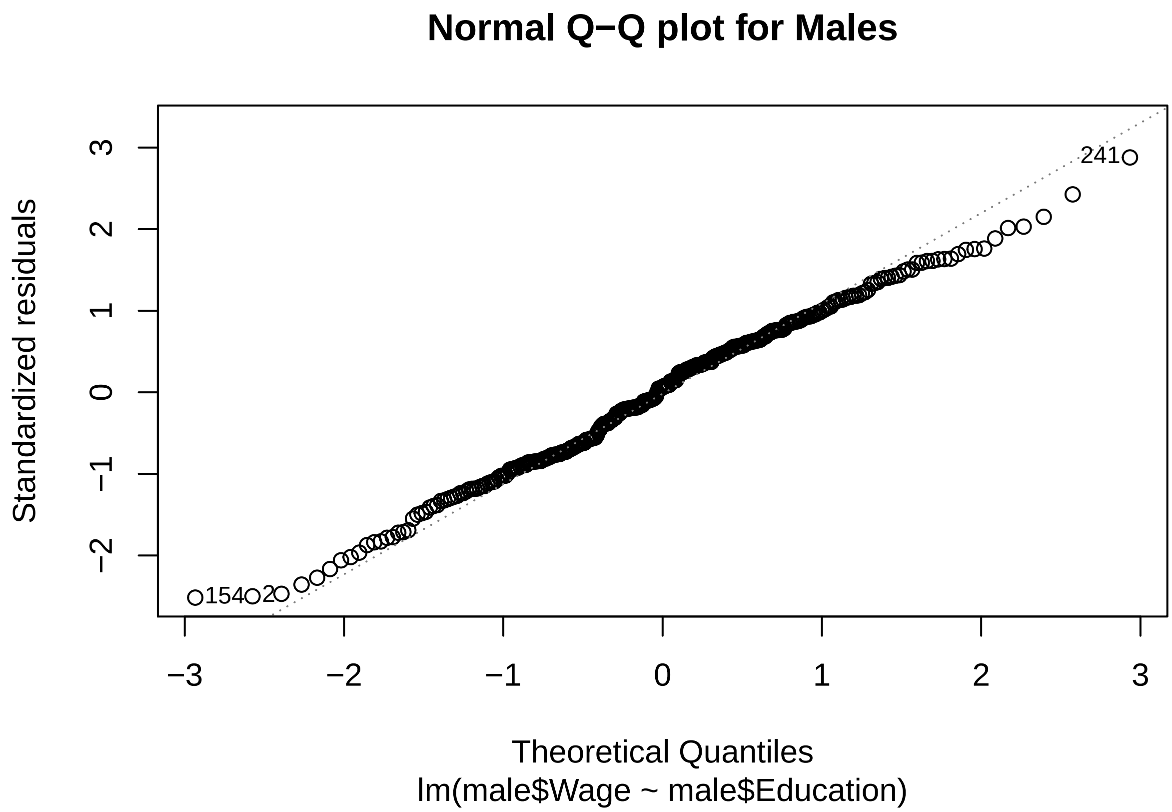

```
plot(lm_male, which = c(1), main = "Residuals vs. Fitted for Males", caption = "")
```




```
plot(lm_female, which = c(2), main = "Normal Q-Q plot for Females", caption = "")
```




```
plot(lm_male, which = c(2), main = "Normal Q-Q plot for Males", caption = "")
```


Normal Q−Q plot for Males
Standardized residuals
Theoretical Quantiles
lm(male$Wage ~ male$Education)
241
154
2
−3
−2
−1
0
1
2
3

3. Can you build a simple lm() model using all the predictors? Describe this unified model.

```
lm_education = lm(education_df$Wage ~ education_df$ID + education_df$Education + education_df$Gender)
summary(lm_education)
```

```
##
## Call:
## lm(formula = education_df$Wage ~ education_df$ID + education_df$Education +
##     education_df$Gender)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -243.334  -75.119    2.669   70.007  276.238
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -576.22317   24.08079 -23.929   <2e-16 ***
## education_df$ID            0.02952    0.03114   0.948    0.344
## education_df$Education   397.90661    1.54440 257.645   <2e-16 ***
## education_df$Gendermale  597.94707    9.17422  65.177   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 100.1 on 493 degrees of freedom
## Multiple R-squared:  0.9931, Adjusted R-squared:  0.9931
## F-statistic: 2.362e+04 on 3 and 493 DF,  p-value: < 2.2e-16
```

As we would expect, ID does not explain the wage at all, as its estimate is close to 0. The years of Education and the Gender are better indicators.

Preliminaries Computers Dataset

Loading the computers dataset:

```
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

computers_df = read.csv("Computers.txt", header = TRUE, sep = "\t"  , comment.char = "#")
```

Check for outliers / erronous data input:

```
unique(computers_df$vendor)
```

```
##  [1] "adviser"      "amdahl"       "apollo"       "basf"         "bti"
##  [6] "burroughs"    "c.r.d"        "cdc"          "cambex"       "dec"
## [11] "dg"           "formation"    "four-phase"   "gould"        "hp"
## [16] "harris"       "honeywell"    "ibm"          "ipl"          "magnuson"
## [21] "microdata"    "nas"          "ncr"          "nixdorf"      "perkin-elmer"
## [26] "prime"        "siemens"      "sperry"       "sratus"       "wang"
```

30 vendors as specified in the accompanying PDF.

We will not be looking at the models, are there are too many different unique ones.

```
range(computers_df$MYCT)
```

```
## [1]   17 1500
```

```
range(computers_df$MYCT[(min(computers_df$MYCT) < computers_df$MYCT)
                                                & (computers_df$MYCT < max(computers_df$MYCT))
                                                ])
```

```
## [1]   23 1100
```

```
range(computers_df$MMIN)
```

```
## [1]    64 32000
```

```
range(computers_df$MMIN[(min(computers_df$MMIN) < computers_df$MMIN)
                                                & (computers_df$MMIN < max(computers_df$MMIN))
                                                ])
```

```
## [1]    96 16000
```

```
range(computers_df$MMAX)
```

```
## [1]    64 64000
```

```
range(computers_df$MMAX[(min(computers_df$MMAX) < computers_df$MMAX)
                                                & (computers_df$MMAX < max(computers_df$MMAX))
                                                ])
```

```
## [1]   512 32000
```

```
range(computers_df$CACH)
```

```
## [1]   0 256
```

```
range(computers_df$CACH[(min(computers_df$CACH) < computers_df$CACH)
                                                & (computers_df$CACH < max(computers_df$CACH))
                                                ])
```

```
## [1]   1 160
```

```r
range(computers_df$CGMIN)
```

```
## [1]  0 52
```

```r
range(computers_df$CGMIN[(min(computers_df$CGMIN) < computers_df$CGMIN)
                                                & (computers_df$CGMIN < max(computers_df$CGMIN))
                                                ])
```

```
## [1]  1 32
```

```r
range(computers_df$CHMAX)
```

```
## [1]   0 176
```

```r
range(computers_df$CHMAX[(min(computers_df$CHMAX) < computers_df$CHMAX)
                                                & (computers_df$CHMAX < max(computers_df$CHMAX))
                                                ])
```

```
## [1]   1 128
```

```r
range(computers_df$PRP)
```

```
## [1]    6 1150
```

```r
range(computers_df$PRP[(min(computers_df$PRP) < computers_df$PRP)
                                                & (computers_df$PRP < max(computers_df$PRP))
                                                ])
```

```
## [1]    7 1144
```

```r
range(computers_df$ERP)
```

```
## [1]   15 1238
```

```r
range(computers_df$ERP[(min(computers_df$ERP) < computers_df$ERP)
                                                & (computers_df$ERP < max(computers_df$ERP))
                                                ])
```

```
## [1]  17 978
```

The second largest and second smallest values seem to often be "near" (as far as can be guessed) to the max and min value. More testing would have to be done, but it seems that no "major outliers" sneaked into the data. This was to be expected as the data has already been used in other papers.

4. Check the different variables (predictor) you have to predict PRP. In your opinion, which are the variables that cannot be used to explain the system performance?

Variables such as the model or vendor are not applicable to predict performance in a linear model. ERP should not be used, as it is the result of a linear prediction of PRP (based on the available predictor values).

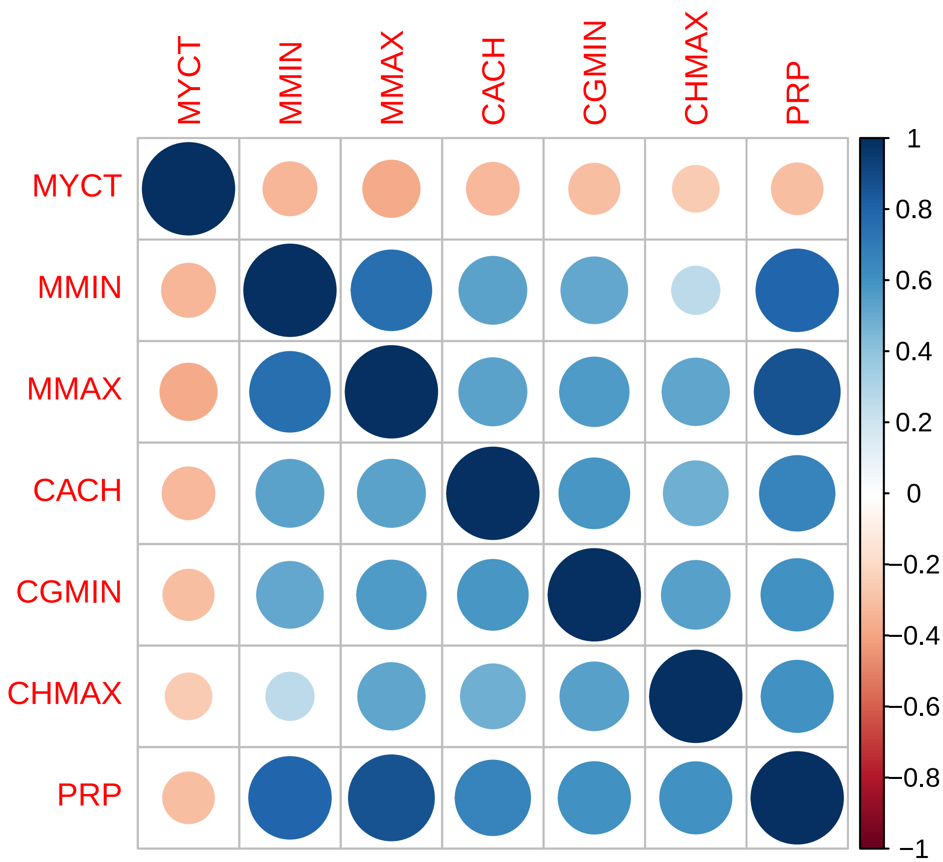
5. You're allowed to use only a single variable (predictor) to predict the value of PRP. Which one would you use? Does your model explain something? What is the confidence interval around the slope?

We will be using corrplot library for plotting the correlations.

```
library(corrplot)
```

```
## corrplot 0.88 loaded
```

```
computers_df_cor = cor(
  subset(
    # to remove the non-numeric fields (model & vendor)
    computers_df[sapply(computers_df, is.numeric)],
    # Removing the ERP field that should not be used
    select = -c(ERP),
  )
)
corrplot(computers_df_cor)
```



MMAX is likely to be a good linear predictor for the PRP value, with MMIN as the alternative. This is to be expected, that if The Maximum Main Memory in Kilobytes is of interest, then so is the minimum main memory in kilobytes.

```
computers_df_lm_MMAX = lm(PRP ~ MMAX, data = computers_df)
summary(computers_df_lm_MMAX)
```

```
##
## Call:
```

```
## lm(formula = PRP ~ MMAX, data = computers_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -230.76  -36.07    3.31   29.33  426.48
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.400e+01  8.001e+00   -4.25 3.24e-05 ***
## MMAX         1.184e-02  4.816e-04   24.58  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 81.45 on 207 degrees of freedom
## Multiple R-squared:  0.7448, Adjusted R-squared:  0.7435
## F-statistic: 604.1 on 1 and 207 DF,  p-value: < 2.2e-16
```

```
confint(computers_df_lm_MMAX, 'MMAX', level = 0.95)
```

```
##           2.5 %     97.5 %
## MMAX 0.01088673 0.01278561
```

```
computers_df_lm_MMIN = lm(PRP ~ MMIN, data = computers_df)
summary(computers_df_lm_MMIN)
```

```
##
## Call:
## lm(formula = PRP ~ MMIN, data = computers_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -174.78  -32.01  -12.01    7.47  875.22
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.088903   8.421540   1.317    0.189
## MMIN         0.032962   0.001749  18.851   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 97.81 on 207 degrees of freedom
## Multiple R-squared:  0.6319, Adjusted R-squared:  0.6301
## F-statistic: 355.4 on 1 and 207 DF,  p-value: < 2.2e-16
```

```
confint(computers_df_lm_MMIN, 'MMIN', level = 0.95)
```

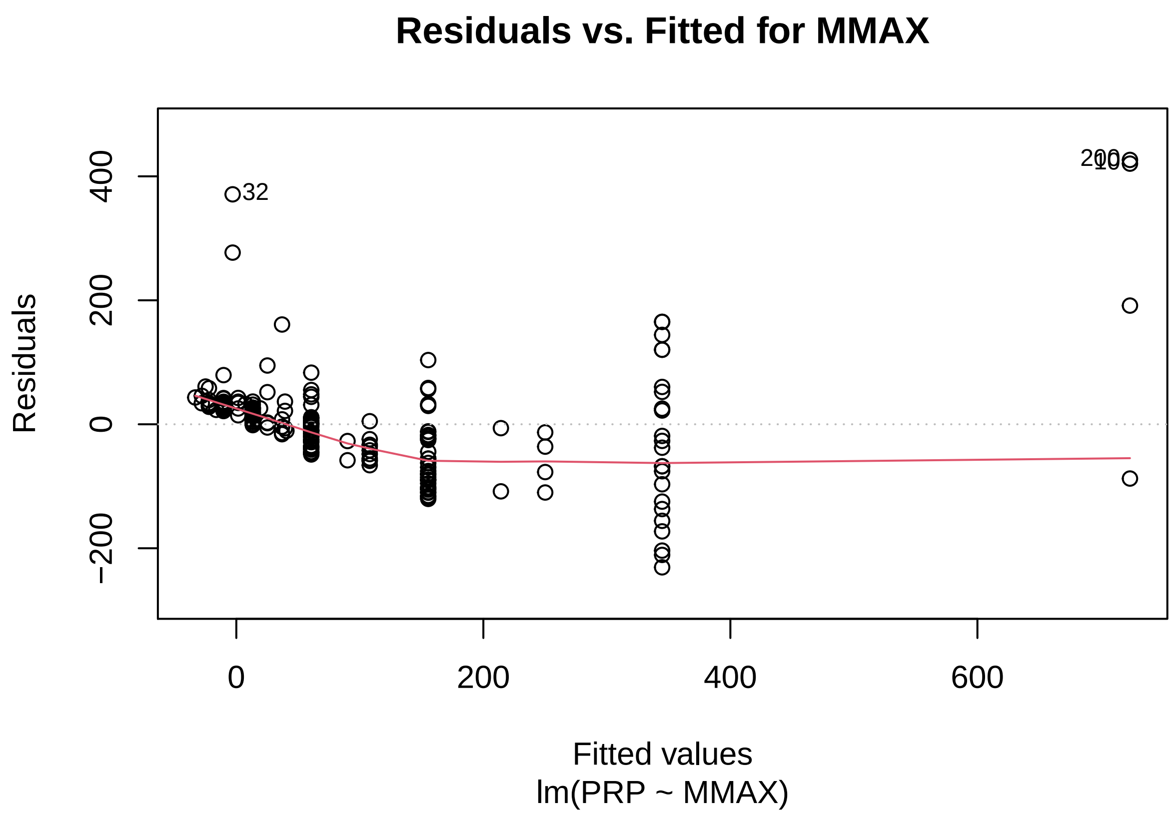
```
##          2.5 %     97.5 %
## MMIN 0.0295144 0.03640871
```

MMAX seems better suited than MMIN, as can be seen with the Coefficients. It might be that MMIN might change slightly slower than MMAX for compatibility reasons. MMAX will likely always be "cutting-edge".

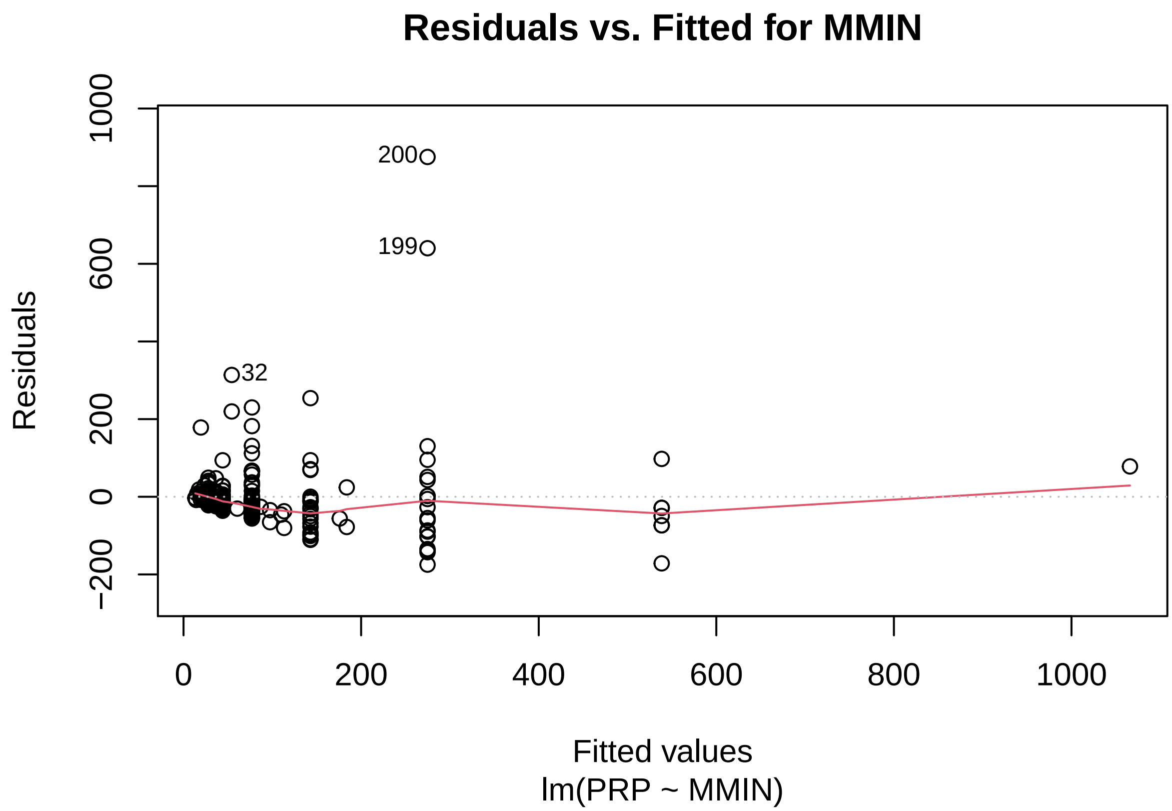
The confidence-interval around the slope is 0.011-0.013 for MMAX and 0.029-0.36 for MMIN.

6. Visualize graphically the (linear) relationship that you found.

```
plot(computers_df_lm_MMAX, which = c(1), main = "Residuals vs. Fitted for MMAX", caption = "")
```




```
plot(computers_df_lm_MMIN, which = c(1), main = "Residuals vs. Fitted for MMIN", caption = "")
```



The derivation of the points from the residual = 0 line is smallest for small values, of which there are more for MMIN and MMAX. Of note is that the mean residuals for MMAX are consistently below the residual = 0 line after around 150, while for MMIN we have larger outliers and the mean even goes above the line for higher MMIN values.

All this tells us, is that we have a lot more data for smaller MMIN and MMAX values and that MMAX correlates more strongly with PRP than MMIN. It is likely that after a certain threshold (of technological innovation), the MMIN did not need to increase to be in line with MMAX anymore, which is why MMAX is the better predictor than MMIN.

Preliminaries

Loading the cars dataset:

```
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))

cars_df = read.csv("Cars.txt", header = TRUE, sep = "\t", comment.char = "#")
```

Check for outliers:

```
range(cars_df$mpg)
```

```
## [1]  9.0 46.6
```

```
range(cars_df$mpg[(min(cars_df$mpg) < cars_df$mpg)
                                                & (cars_df$mpg < max(cars_df$mpg))
                                                ])
```

```
## [1] 10.0 44.6
```

```
unique(cars_df$cylinders)
```

```
## [1] 8 4 6 3 5
```

```
range(cars_df$displacement)
```

```
## [1]  68 455
```

```
range(cars_df$displacement[(min(cars_df$displacement) < cars_df$displacement)
                                                & (cars_df$displacement < max(cars_df$displacement))
                                                ])
```

```
## [1]  70 454
```

```
unique(cars_df$horsepower)
```

```
##  [1] 130 165 150 140 198 220 215 225 190 170 160  95  97  85  88  46  87  90 113
## [20] 200 210 193  NA 100 105 175 153 180 110  72  86  70  76  65  69  60  80  54
## [39] 208 155 112  92 145 137 158 167  94 107 230  49  75  91 122  67  83  78  52
## [58]  61  93 148 129  96  71  98 115  53  81  79 120 152 102 108  68  58 149  89
## [77]  63  48  66 139 103 125 133 138 135 142  77  62 132  84  64  74 116  82
```

There are 6 NAs for horsepower that need to be removed -> delete the whole corresponding row. These are also mentioned in Cars.pdf.

```
cars_df_cleaned <- cars_df[!is.na(cars_df$horsepower),]
```

```
range(cars_df_cleaned$weight)
```

```
## [1] 1613 5140
```

```
range(cars_df_cleaned$weight[(min(cars_df_cleaned$weight) < cars_df_cleaned$weight)
                                                & (cars_df_cleaned$weight < max(cars_df_cleaned$weight)
                                                ])
```

```
## [1] 1649 4997
```

```
range(cars_df_cleaned$acceleration)
```

```
## [1]  8.0 24.8
```

```
range(cars_df_cleaned$acceleration[(min(cars_df_cleaned$acceleration) < cars_df_cleaned$acceleration)
                                              & (cars_df_cleaned$acceleration < max(cars_df_cleaned$ac
                                              ])
```

```
## [1]  8.5 24.6
```

```
range(cars_df_cleaned$year)
```

```
## [1] 70 82
```

```
range(cars_df_cleaned$origin)
```

```
## [1] 1 3
```

Origin is Ternary.

Names are a list of names, thus no need to check it.

Besides the NAs found for horspower, the data seems to be good (without additional information.)

7. Check the different variables (predictor) you have to predict mpg. In your opinion, which are the variables that cannot be used to explain the system performance?

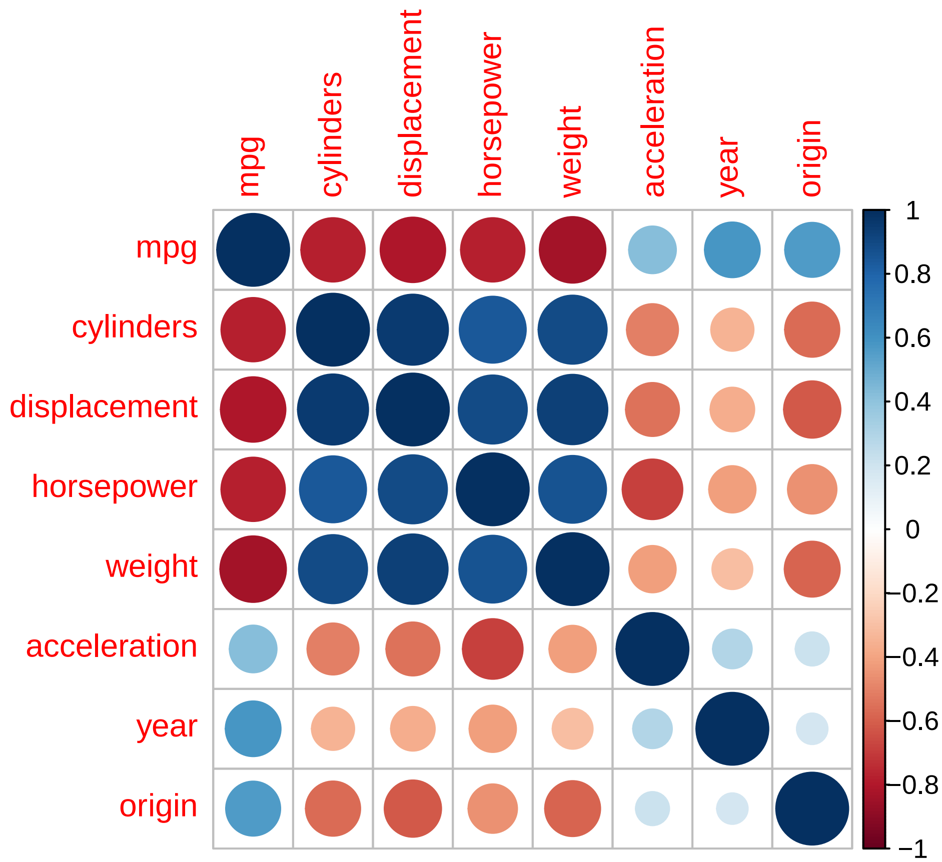
Obviously name can't be used to explain the mpg, origin is unlikely to be of impact too. Furthermore, the year might corrolate with mpg if the efficiency increased over time, but is not a direct predictor for mpg.

8. You're allowed to use only a single variable (predictor) to predict the value of mpg. Which one would you use? Does your model explain something? What is the confidence interval around the slope?

We will be using corrplot library for plotting the correlations.

```
library(corrplot)
```

```
cars_df_cor = cor(
  subset(
    # to remove the non-numeric fields
    cars_df_cleaned[sapply(cars_df_cleaned, is.numeric)],
  )
)
corrplot(cars_df_cor)
```



As expected origin is not useful. Best seems to be weight, which is to be expected as the more weight needs to be propelled forward, the higher the energy required. Thus we will be using weight.

On a side-note, it can be said that weight, horsepower, displacements, and cylinders all relate to each other. If the weight increases, so thus the horsepower of the engine, the number of cylinders and the displacement.

```
cars_df_cleaned_lm_weight = lm(mpg ~ weight, data = cars_df_cleaned)
summary(cars_df_cleaned_lm_weight)
```

```
##
## Call:
## lm(formula = mpg ~ weight, data = cars_df_cleaned)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.9736  -2.7556  -0.3358   2.1379  16.5194
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 46.216524   0.798673   57.87   <2e-16 ***
## weight      -0.007647   0.000258  -29.64   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.333 on 390 degrees of freedom
## Multiple R-squared:  0.6926, Adjusted R-squared:  0.6918
## F-statistic: 878.8 on 1 and 390 DF,  p-value: < 2.2e-16
```

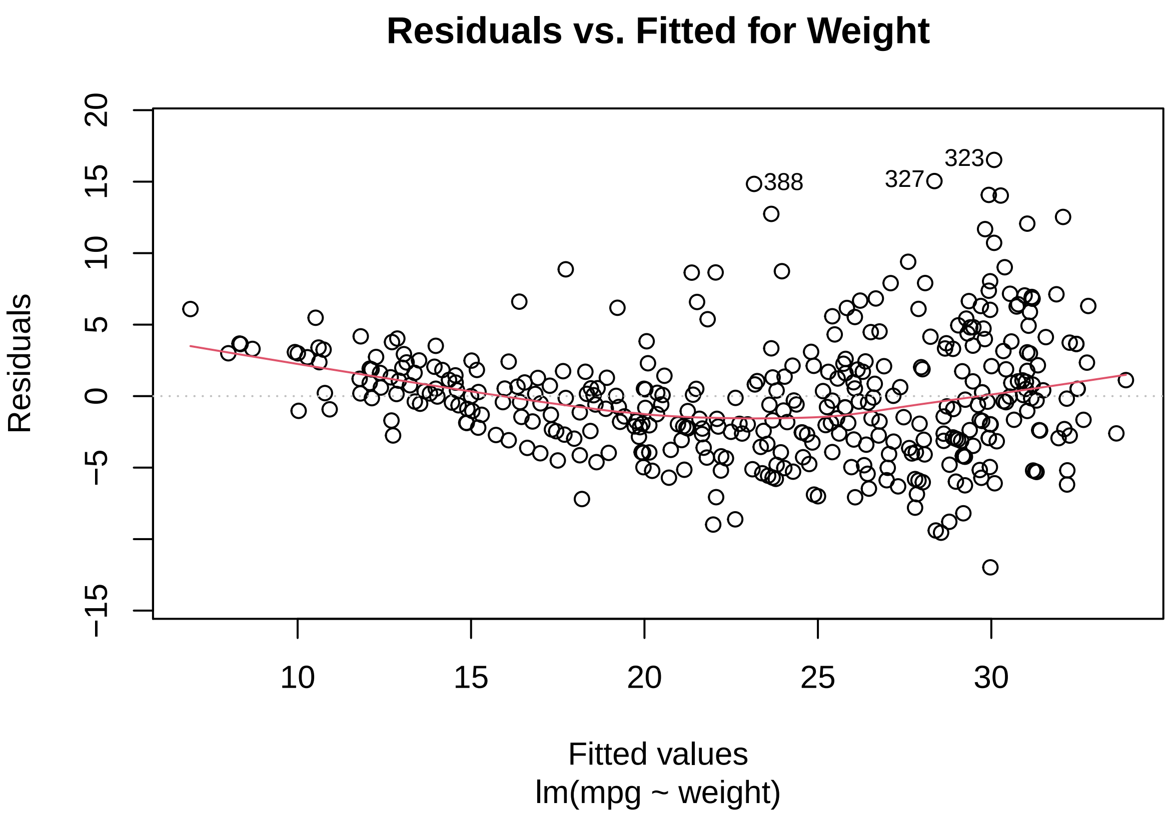
```r
confint(cars_df_cleaned_lm_weight, 'weight', level = 0.95)
```

```
##               2.5 %      97.5 %
## weight -0.008154515 -0.00714017
```

As we have a negative correlation (if weight increases, mpg decreases) the confidence-interval aroudn the slope is the negative values -0.008 - -0.007.

9. Visualize graphically the (linear) relationship that you found

```
plot(cars_df_cleaned_lm_weight, which = c(1), main = "Residuals vs. Fitted for Weight", caption = "")
```



Of interest is that higher values tend to have more outliers in terms of residuals. Do note that overall the residual = 0 line, seems to be quite fitting.

I would be wary to use the model as a predictor for smaller values, as we have a lack of data points for this range.