Module 3: Recommended Exercises

alexaoh

21.01.2021

Problem 1 (Extension from Book Ex. 9)

This question involves the use of multiple linear regression on the `Auto` data set from `ISLR` package (you may use `?Auto` to see a description of the data). First we exclude from our analysis the variable `name` and look at the data summary and structure of the dataset.

```r
library(ISLR)
Auto = subset(Auto, select = -name)
# Auto$origin = factor(Auto$origin)
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
#>  Min.   : 8.00   Min.   :70.00   Min.   :1.000
#>  1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000
#>  Median :15.50   Median :76.00   Median :1.000
#>  Mean   :15.54   Mean   :75.98   Mean   :1.577
#>  3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000
#>  Max.   :24.80   Max.   :82.00   Max.   :3.000
```

```r
str(Auto)
```

```
#> 'data.frame':    392 obs. of  8 variables:
#>  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
#>  $ cylinders   : num  8 8 8 8 8 8 8 8 8 8 ...
#>  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
#>  $ horsepower  : num  130 165 150 150 140 198 220 215 225 190 ...
#>  $ weight      : num  3504 3693 3436 3433 3449 ...
#>  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
#>  $ year        : num  70 70 70 70 70 70 70 70 70 70 ...
#>  $ origin      : num  1 1 1 1 1 1 1 1 1 1 ...
```

We obtain a summary and see that all variables are numerical (continuous). However, when we check the description of the data (again with `?Auto`) we immediately see that `origin` is actually encoding for either American (origin=1), European (origin=2) or Janapense (origin=3) origin of the car, thus the values 1, 2 and 3 do not have any actual numerical meaning. We therefore need to first change the data type of that variable to let R know that we are dealing with a qualitative (categorical) variable, instead of a continuous one (otherwise we will obtain wrong model fits). In R such variables are called *factor variables*, and before we continue to do any analyses we first need to convert `origin` into a factor variable (a synonymous for "qualitative predictor"):

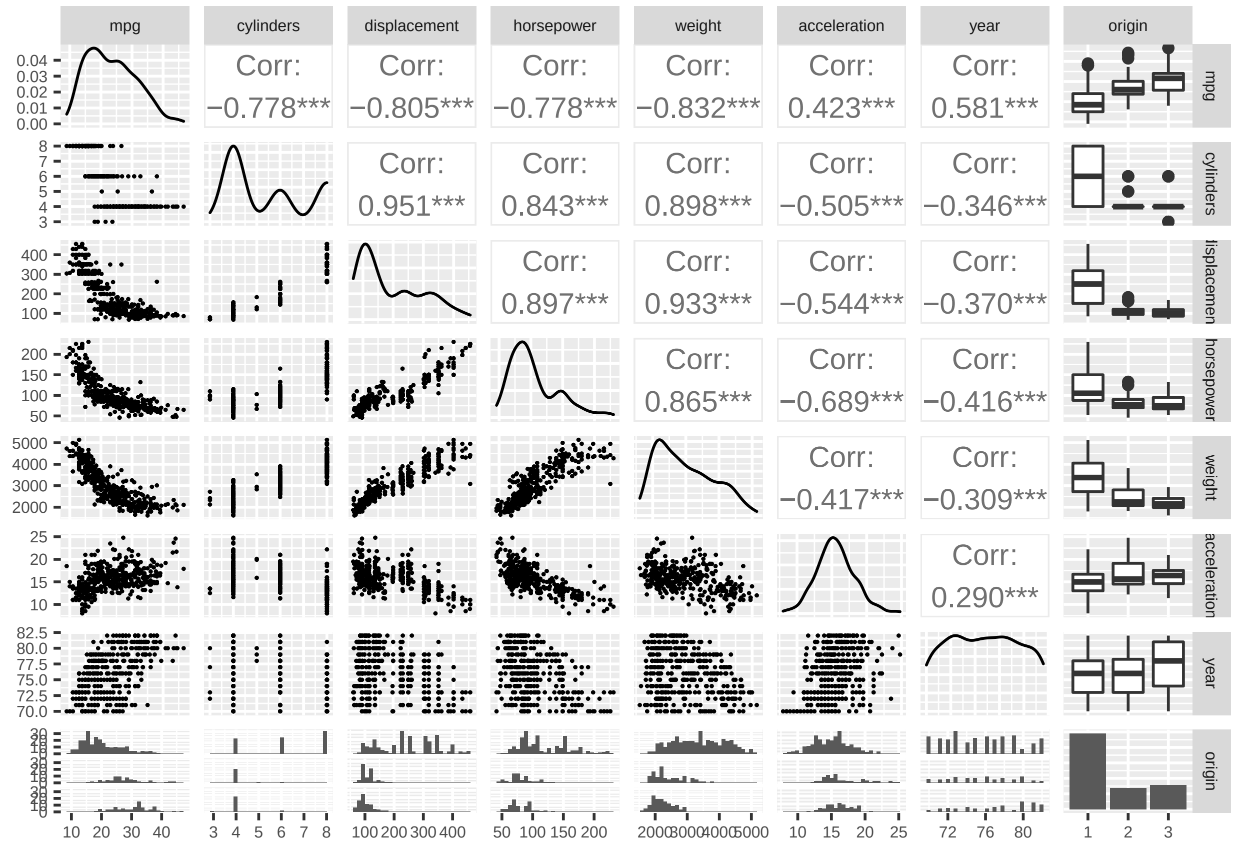
```
Auto$origin = factor(Auto$origin)
```

a)

Use the function `ggpairs()` from `GGally` package to produce a scatterplot matrix which includes all of the variables in the data set.

```
library(GGally)
#ggpairs(Auto) # This was my standard solution. Could have done the more fancy version from LF, below:

ggpairs(Auto, lower = list(continuous = wrap("points", size=0.1))) + # change points size
  theme(text = element_text(size = 7)) # change text size
```



b)

Compute the correlation matrix between the variables. You will need to remove the factor covariate `origin`, because this is no longer a continuous variable.

```
variables <- Auto[-c(8)]   # Remove 'origin' from the data set.
Sigma <- cor(variables)
Sigma
```

```
#>                     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
#>              acceleration       year
#> mpg             0.4233285  0.5805410
#> cylinders      -0.5046834 -0.3456474
#> displacement   -0.5438005 -0.3698552
#> horsepower     -0.6891955 -0.4163615
#> weight         -0.4168392 -0.3091199
#> acceleration    1.0000000  0.2903161
#> year            0.2903161  1.0000000
```

c)

Use the `lm()` function to perform a multiple linear regression with `mpg` (miles per gallon, a measure for fuel consumption) as the response and all other variables (except `name`) as the predictors. Use the `summary()` function to print the results. Comment on the output. In particular:

i. Is there a relationship between the predictors and the response?
ii. Is there evidence that the weight of a car influences `mpg`? Interpret the regression coefficient β_{weight} (what happens if a car weights 1000kg more, for example?).
iii. What does the coefficient for the year variable suggest?

```
mreg <- lm(mpg ~ ., data = Auto)  # name has already been removed from Auto.
summary(mreg)
```

```
#>
#> Call:
#> lm(formula = mpg ~ ., data = Auto)
#>
#> Residuals:
#>     Min      1Q  Median      3Q     Max
#> -9.0095 -2.0785 -0.0982  1.9856 13.3608
#>
#> Coefficients:
#>                Estimate Std. Error t value Pr(>|t|)
#> (Intercept)  -1.795e+01  4.677e+00  -3.839 0.000145 ***
#> cylinders    -4.897e-01  3.212e-01  -1.524 0.128215
#> displacement  2.398e-02  7.653e-03   3.133 0.001863 **
#> horsepower   -1.818e-02  1.371e-02  -1.326 0.185488
#> weight       -6.710e-03  6.551e-04 -10.243  < 2e-16 ***
#> acceleration  7.910e-02  9.822e-02   0.805 0.421101
#> year          7.770e-01  5.178e-02  15.005  < 2e-16 ***
#> origin2       2.630e+00  5.664e-01   4.643 4.72e-06 ***
#> origin3       2.853e+00  5.527e-01   5.162 3.93e-07 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 3.307 on 383 degrees of freedom
#> Multiple R-squared:  0.8242, Adjusted R-squared:  0.8205
#> F-statistic: 224.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

Comments on the output:

i) The F-statistic is 224.5, with a p-value of less than 2.2e−6, which signals that there is a relationship between the predictors and the response. Moreover, several of the p-values of the coefficients related to the predictors are significant. As noted on page 77 in ISLR, each of these p-values give the partial effect of adding that specific variable to the model, while keeping the others in. However, as also noted, there are some flaws with only noting these values, which is why the F-statistic is of main concern when concluding whether or not the predictors are useful in predicting the response.

ii) The p-value of the coefficient β_{weight} is 2e−16 which could be evidence that the weight of a car influences `mpg` (at least it is not evidence against whether or not the weight of a car influences `mpg`). The interpretation of the coefficient is that the `mpg` changes, on average, by $\beta_{\text{weight}} = 6.710\text{e}{-3}$ for every one-valued increase in the weight of the car, given that all the other predictors are fixed. This means that, e.g., if a car weighs 1000kg, the `mpg` is estimated to be reduced by 6.710.

iii) The coefficient for the year variable suggests that `mpg` is increased by 0.770 for each increase in model year of the car.

d)

Look again at the regression output from question c). Now we want to test whether the `origin` variable is important. How does this work for a factor variable with more than only two levels?

We construct dummy variables such that he coefficients can be estimated in the regression. If we have k levels in the factor variable, we construct $k-1$ dummy variables. In this way, a baseline level is made, and each coefficient says something about the difference in the response when each respective category is fulfilled with respect to the baseline level. This will become more clear after the example in this task.

`Origin2` and `Origin3` are the two factor-coefficients that R made. This means that origin category 1 will be regarded as the baseline, which is given by the interceipt. In this case we can see that both the estimations of European (2) and Japanese (3) are significant (from their p-values) and that they give positive values of the `mpg` when added to the interceipt (the baseline, which is American (1)). Have a look at the R-block below for the dummy variables that R made automatically for the categorical variable.

```r
contrasts(Auto$origin)
```

```
#>   2 3
#> 1 0 0
#> 2 1 0
#> 3 0 1
```

As one can see, the dummy variables 2 and 3 were made automatically by R, with 1 as a baseline, as predicted. This means that the dummy variable 'Origin2' takes on the value 1 if the the origin is 2 (European) and zero otherwise. Moreover, the dummy variable 'Origin3' takes on the value 1 if the origin is 3 (Japanese) and zero otherwise. The baseline, 1 (American), corresponds to when both dummy variables are zero-valued. Hence, this is given by the interceipt.

As explained in the LF, we can use the following to do an F-test on whether $\beta_{\text{origin2}} = \beta_{\text{origin3}} = 0$ at the same time

```r
anova(mreg)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
cylinders	1	1.440308e+04	1.440308e+04	1317.3787511	0.0000000
displacement	1	1.073344e+03	1.073344e+03	98.1734677	0.0000000
horsepower	1	4.034081e+02	4.034081e+02	36.8977402	0.0000000
weight	1	9.757250e+02	9.757250e+02	89.2447341	0.0000000
acceleration	1	9.660708e-01	9.660708e-01	0.0883617	0.7664315
year	1	2.419120e+03	2.419120e+03	221.2649607	0.0000000
origin	2	3.559553e+02	1.779777e+02	16.2787372	0.0000002

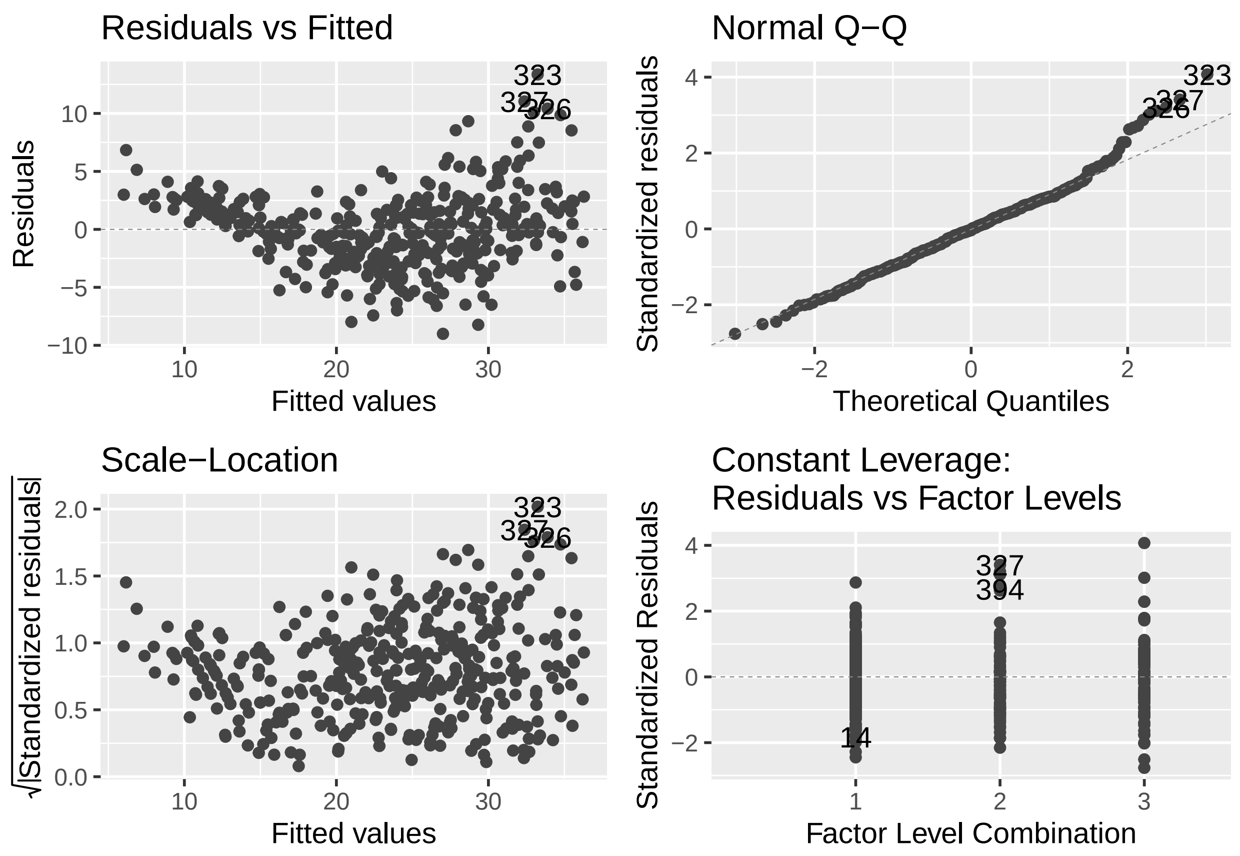
	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Residuals	383	4.187392e+03	1.093314e+01	NA	NA

As we can see, the p-value of `origin` is small, which signals that the origin of the car has an influence on the response.

e)

Use the `autoplot()` function from the `ggfortify` package to produce diagnostic plots of the linear regression fit by setting `smooth.colour = NA`, as sometimes the smoothed line can be misleading. Comment on any problems you see with the fit. Do the residual plots suggest any unusually large outliers? Does the leverage plot identify any observations with unusually high leverage?

```r
library(ggfortify)
autoplot(mreg, smooth.colour = NA)
```



Comments on the plots:
From the residual plot in the upper left it looks like the residuals form a pattern closer to a quadratic, which suggests that there might be some problems with the fit. More specifically, it looks like the variance is increasing with the fitted values, which suggests that the assumption about constant variance is not fulfilled. However, the assumption of 0 expectation of the residuals seems to be fulfilled. The Scale-Location plot suggests what has already been noted as well, that the variance is increasing with increasing fitted value.
I cannot, however, identify any unusually large outliers in this plot.

Moreover, it looks like there are a few points in each of the categories that are high leverage points, especially in factor level 3.

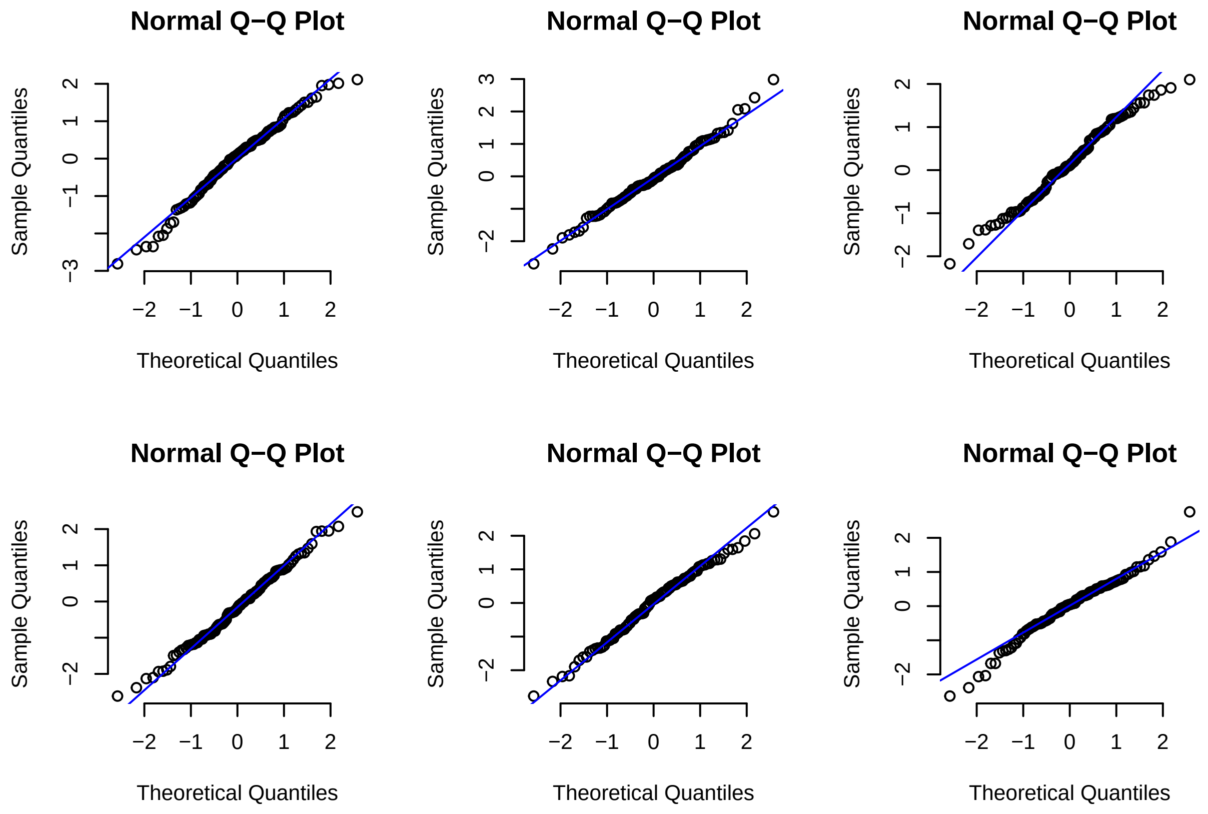
The points 323, 326 and 327 especially caught my eye, since they have large residuals and they deviate a lot in the upper right part of the QQ-plot. Also, some of these are marked in the leverage-plat as well. This might suggest that these points are outliers and they might need to be checked for errors in the experiment/data.

Based on the "QQ-plot" test below, I would say that the QQ-plot does not look too good in this case, since the sample quantiles show a highly deviating trend in the top right of the plot, which looks to be more extreme than in some of the plots below.

f)

For beginners, it can be difficult to decide whether a certain QQ plot looks "good" or "bad", because we only look at it and do not test anything. A way to get a feeling for how "bad" a QQ plot may look, even when the normality assumption is perfectly ok, we can use simulations: We can simply draw from the normal distribution and plot the QQ plot. Use the following code to repeat this six times:

```r
set.seed(2332)
n = 100
par(mfrow = c(2, 3))
for (i in 1:6) {
    sim = rnorm(n)
    qqnorm(sim, pch = 1, frame = FALSE)
    qqline(sim, col = "blue", lwd = 1)
}
```



g)

Let us look at interactions. These can be included via the `*` or `:` symbols in the linear predictor of the regression function (see Section 3.6.4 in the course book).

Fit another model for `mpg`, including only `displacement`, `weight`, `year` and `origin` as predictors, plus an interaction between `year` and `origin` (interactions can be included as `year*origin`; this adds the main effects and the interaction at once). Is there evidence that the interactions term is relevant? Give an interpretation of the result.

```r
interactions.fit <- lm(mpg ~ displacement + weight + year * origin, data = Auto)
summary(interactions.fit)
```

```
#>
#> Call:
#> lm(formula = mpg ~ displacement + weight + year * origin, data = Auto)
#>
#> Residuals:
#>     Min      1Q  Median      3Q     Max
#> -8.7710 -2.0204 -0.0207  1.7045 13.0017
#>
#> Coefficients:
#>                Estimate Std. Error t value Pr(>|t|)
#> (Intercept)  -5.117e+00  5.259e+00  -0.973 0.331220
#> displacement  4.803e-03  5.032e-03   0.955 0.340420
#> weight       -6.685e-03  5.543e-04 -12.060  < 2e-16 ***
#> year          6.152e-01  6.614e-02   9.302  < 2e-16 ***
#> origin2      -3.735e+01  1.026e+01  -3.642 0.000307 ***
#> origin3      -2.532e+01  9.441e+00  -2.682 0.007631 **
#> year:origin2  5.187e-01  1.342e-01   3.865 0.000130 ***
#> year:origin3  3.564e-01  1.213e-01   2.937 0.003514 **
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 3.257 on 384 degrees of freedom
#> Multiple R-squared:  0.829,  Adjusted R-squared:  0.8259
#> F-statistic: 265.9 on 7 and 384 DF,  p-value: < 2.2e-16
```

```r
fit.without.interactions <- lm(mpg ~ displacement + weight + year + origin,
    data = Auto)
summary(fit.without.interactions)
```

```
#>
#> Call:
#> lm(formula = mpg ~ displacement + weight + year + origin, data = Auto)
#>
#> Residuals:
#>     Min      1Q  Median      3Q     Max
#> -9.2633 -2.1572  0.0205  1.8226 13.5061
#>
#> Coefficients:
#>                Estimate Std. Error t value Pr(>|t|)
#> (Intercept)  -1.959e+01  4.064e+00  -4.822 2.05e-06 ***
#> displacement  9.321e-03  5.001e-03   1.864   0.0631 .
#> weight       -6.820e-03  5.637e-04 -12.099  < 2e-16 ***
#> year          7.980e-01  5.081e-02  15.705  < 2e-16 ***
#> origin2       2.383e+00  5.606e-01   4.251 2.67e-05 ***
#> origin3       2.438e+00  5.309e-01   4.592 5.94e-06 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 3.327 on 386 degrees of freedom
#> Multiple R-squared:  0.8206, Adjusted R-squared:  0.8183
#> F-statistic: 353.2 on 5 and 386 DF,  p-value: < 2.2e-16
```

```r
anova(interactions.fit, fit.without.interactions)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
384	4073.055	NA	NA	NA	NA
386	4271.965	-2	-198.91	9.376433	0.0001057

The R-squared is slightly bigger than the value obtained when only fitting the main effects, while the F-statistic is slightly smaller. Moreover, the interaction terms could be described as significant in the fit with interactions (based on relatively small p-values), but they cannot be used to state the significance of the interaction terms definitely. However, based on the small differences in R-squared (and F-statistic) between the two fits, I would say that there is no clear evidence that the interaction terms are relevant. On the other hand, the anova results in a relatively small Pr(>F) in my opinion, which might suggest that the fit with the interactions could have some merit. This last note is correct: It is the F-test that is most important when testing whether the coefficients of the interaction terms are zero at the same time (which is the NULL-hypothesis). The p-value of the F-test value is small, which gives evidence that the interaction terms have merit.

As noted in the LF, I could just have calculated the anova of the fit with the interactions, like so

```r
anova(interactions.fit)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
displacement	1	15440.1719	15440.17185	1455.670595	0.0000000
weight	1	1208.5135	1208.51348	113.936396	0.0000000
year	1	2601.3981	2601.39814	245.254963	0.0000000
origin	2	296.9452	148.47262	13.997722	0.0000014
year:origin	2	198.9100	99.45502	9.376433	0.0001057
Residuals	384	4073.0547	10.60691	NA	NA

h)

Try a few different transformations of the variables, such as $\log(X)$, $\sqrt{X}$, X^2. See Section 3.6.5 in the course book for how to do this. Perhaps you manage to improve the residual plots that you got in e)? Comment on your findings.

```r
# log-transformation.  log <- lm(mpg ~ . + I(log(weight)) +
# I(log(year)), data = Auto) summary(log) autoplot(log)

# sqrt-transformation. sqrt <- lm(mpg ~ . + I(sqrt(weight)) +
# I(sqrt(year)), data = Auto) summary(sqrt) autoplot(sqrt)

# x^2-transformation. squared <- lm(mpg ~ . + I(year^2) +
# I(weight^2), data = Auto) summary(squared) autoplot(squared)
```

I think that the residual plots seem to gain less patterns when using these transformations on some of the variables, but I think the QQ-plot only gets worse. Also, the same outliers as noted earlier as still present, which further substantiates my suspicion that these points should be checked.

Problem 2

a)

A core finding for the least-squares estimator $\hat{\boldsymbol{\beta}}$ of linear regression models is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \ ,$$

with $\hat{\boldsymbol{\beta}} \sim N_p(\boldsymbol{\beta}, \sigma^2(\mathbf{X}^T\mathbf{X})^{-1})$.

- Show that $\hat{\boldsymbol{\beta}}$ has this distribution with the given mean and covariance matrix.
- What do you need to assume to get to this result?
- What does this imply for the distribution of the jth element of $\hat{\boldsymbol{\beta}}$?
- In particular, how can we calculate the variance of $\hat{\beta}_j$?

PROOF:

- First, we show that $\mathsf{Cov}(\hat{\boldsymbol{\beta}}) = \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$. We begin the proof by defining the quantity $\mathbf{C} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ such that $\hat{\boldsymbol{\beta}} = \mathbf{CY}$. Now, calculating the covariance matrix gives

$$\begin{aligned}\mathsf{Cov}(\hat{\boldsymbol{\beta}}) = \mathsf{Cov}(\mathbf{CY}) &= \mathbf{C}\cdot\mathsf{Cov}(\mathbf{Y})\cdot\mathbf{C}^{\mathbf{T}}\\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\cdot\sigma^2\mathbf{I}\cdot((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T)^T\\
&= \sigma^2\{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\cdot\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-T}\}\\
&= \sigma^2\{(\mathbf{X}^T\mathbf{X})^{-1}\cdot\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\}\\
&= \sigma^2(\mathbf{X}^T\mathbf{X})^{-1},\end{aligned}$$

where we assume that $\mathsf{Cov}(\mathbf{Y}) = \sigma^2\mathbf{I}$.

Next we show that $\mathsf{E}(\hat{\boldsymbol{\beta}}) = \boldsymbol{\beta}$. This is shown by regular calculation of multivariate expectation, as done in the following

$$\begin{aligned}\mathsf{E}(\hat{\boldsymbol{\beta}}) = \mathsf{E}(\mathbf{CY}) = \mathbf{C}\cdot\mathsf{E}(\mathbf{Y}) &= \mathbf{C}\cdot\mathbf{X}\beta\\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\beta = \boldsymbol{\beta},\end{aligned}$$

where we have assumed that $\mathsf{E}(\mathbf{Y}) = \mathbf{X}\beta$.

Finally, $\hat{\boldsymbol{\beta}}$ has a multivariate normal distribution since it is a linear combination of multivariate normal distributions, assuming that $\mathbf{Y}$ is normally distributed. Hence, the distribution of $\hat{\boldsymbol{\beta}}$ is given by

$$\hat{\boldsymbol{\beta}} \sim N_p(\boldsymbol{\beta}, \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}).$$

The distribution is of dimension p (here) because of the dimensions of the design matrix.

- Assumptions: The necessary assumptions are, as shortly noted earlier, that

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \ , \quad \boldsymbol{\varepsilon} \sim N_n(\mathbf{0}, \sigma^2\mathbf{I})$$

which results in

$$\mathbf{Y} \sim N_n(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}).$$

- For the j^{th} element of $\hat{\boldsymbol{\beta}}$ this implies that it is (univariate) normally distributed with itself as expected value and the j^{th} element of $\sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$ as variance.
- The variance of $\hat{\beta}_j$ can be found by looking up the j^{th} (diagonal, Σ_{jj}) element of $\sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$.

b)

What is the interpretation of a 95% confidence interval? Hint: repeat experiment (on Y), on average how many CIs cover the true β_j? The following code shows an interpretation of a 95% confidence interval.

- Model: $Y = 1 + 3X + \varepsilon$, with $\varepsilon \sim \mathsf{N}(0, 1)$.

```r
beta0 = 1
beta1 = 3
true_beta = c(beta0, beta1)  # vector of model coefficients
true_sd = 1  # choosing true sd
X = runif(100, 0, 1)  # simulate the predictor variable X
Xmat = model.matrix(~X, data = data.frame(X))  # create design matrix
ci_int = ci_x = 0  # Counts how many times the true value is within the confidence interval
nsim = 1000
for (i in 1:nsim) {
    y = rnorm(n = 100, mean = Xmat %*% true_beta, sd = rep(true_sd, 100))
    mod = lm(y ~ x, data = data.frame(y = y, x = X))
    ci = confint(mod)
    ci_int[i] = ifelse(true_beta[1] > ci[1, 1] & true_beta[1] < ci[1,
        2], 1, 0)  # if true value of beta0 is within the CI then 1 else 0
    ci_x[i] = ifelse(true_beta[2] > ci[2, 1] & true_beta[2] < ci[2, 2],
        1, 0)  # if true value of beta_1 is within the CI then 1 else 0
}
c(mean(ci_int), mean(ci_x))
```

```
#> [1] 0.955 0.947
```

The interpretation of the 95% confidence interval is: If values are drawn, parameters are estimated and confidence intervals are computed many times (for each of these draws), then 95% of these confidence intervals will contain the true value. As one can see from the simulations, this makes sense.

c)

What is the interpretation of a 95% prediction interval? Hint: repeat experiment (on Y) for a given $\boldsymbol{x}_0$. Write R code that shows the interpretation of a 95% PI. Hint: In order to produce the PIs use the data point $x_0 = 0.4$. Furthermore you may use a similar code structure as in b).

```r
beta0 = 1
beta1 = 3
true_beta = c(beta0, beta1)  # vector of model coefficients
true_sd = 1  # choosing true sd
X = runif(100, 0, 1)  # simulate the predictor variable X
Xmat = model.matrix(~X, data = data.frame(X))  # create design matrix
pi_count = 0  # Counts how many times the true value is within the prediction interval
nsim = 1000
next.value <- 0.4
for (i in 1:nsim) {
    y = rnorm(n = 100, mean = Xmat %*% true_beta, sd = rep(true_sd, 100))
    mod = lm(y ~ x, data = data.frame(y = y, x = X))
    pi = predict(mod, newdata = data.frame(x = next.value), interval = "prediction",
        type = "response")
    pred.value = 1 + 3 * next.value + rnorm(n = 1)  # predicted value of unobserved value response
    pi_count[i] = ifelse(pred.value > pi[1, 2] & pred.value < pi[1, 3],
        1, 0)  # if true value of beta_1 is within the CI then 1 else 0
}
mean(pi_count)
```

```
#> [1] 0.945
```

The interpretation of a prediction interval is: If values are drawn, parameters are estimated and a prediction of the next observation is made (for each of these draws) many times, then the true next (un-observed) value will be in 95% of these prediction intervals. As is seen from the code the interpretation of the prediction interval is verified.

A difference between the confidence intervals and prediction intervals in this case is that the confidence intervals are calculated based one each parameter $\hat{\boldsymbol{\beta}}$ while the prediction interval is only calculated with respect to one predicted response from the linear model.

d)

Construct a 95% CI for $\boldsymbol{x}_0^T\beta$. Explain what is the connections between a CI for β_j, a CI for $\boldsymbol{x}_0^T\beta$ and a PI for Y at $\boldsymbol{x}_0$.

A 95% CI for $\boldsymbol{x}_0^T\beta$ can be constructed in the same way as for β_j. Check LF for more details about the connection between the three mentioned constructs. I cannot be bothered to elaborate further at this moment in time.

e)

Explain the difference between *error* and *residual*. What are the properties of the raw residuals? Why don't we want to use the raw residuals for model check? What is our solution to this?

The *error* is the squared difference between the predicted value and the true value in the population (summed over all points). This is therefore unknown, something we can never calculate exactly, since the true value in the population is never known.

The *residual* is the difference between the predicted value and the observed value (summed over all points). More precisely, the residual is given by

$$\hat{\varepsilon} = Y - \hat{Y} = (I - \underbrace{X(X^TX)^{-1}X^\top}_{=H})Y,$$

where H is called the "hat matrix". This quantity can be used to estimate the true unknown error in the model.

The properties of the raw residuals are that they are normally distributed with expectation zero and $\mathsf{Cov}(\hat{\varepsilon}) = \sigma^2(I - H)$. Because of these properties, we do not want to use the raw residuals for model check, since they may have different covariances depending on the value of the predictors and they may be correlated. One possible solution to this is therefore to use standardized or studentized residuals, since then we can estimate the errors more closely from the residuals, because of the assumptions on the errors being independent, homoscedastic and independent of the covariates.