Übung 4

Aufgaben 19 bis 24

1.11.2022

Aufgabe 19:

$$y = X\beta + \epsilon$$

$$E(\epsilon) = 0 cov(\epsilon) = \sigma^2 I_n$$

a. Man zeige, dass $X^T r = 0$***:***

Man weiß, dass Residuenvektor als Differenz zwischen y und $\hat{y}$ dargestellt werden kann. Dem zufolge gilt: $r = y - \hat{y}$. Es ist noch bekannt, dass Beziehungen $\hat{y} = X\hat{\beta}$ und $\hat{\beta} = (X^T X)^{-1} X^T y$ gegeben sind.

Dies kann man dan leicht in die Gleichung $X^T r = 0$ einsetzen und überprüfen, was rauskommt:

$X^T r = X^T(y - \hat{y}) = X^T(y - X\hat{\beta}) = X^T y - X^T X\hat{\beta} =$

$= X^T y - X^T X(X^T X)^{-1} X^T y =$

$= X^T y - \frac{X^T X}{X^T X} X^T y = X^T y - X^T y = 0$

Dies heißt, dass Korrelation zwischen Kovarianten x_i und Residuen r_i immer 0 ist.

b. Man zeige, dass die Beziehung $\sum_{i=1}^{n} r_i = 0$ ***gilt.***

In der Gleichung $y = X\beta + \epsilon$ ist y Antwortsvektor, X Design-Matrix und ϵ Fehlervektor. Es ist hier wichtig anzumerken, dass die erste Spalte der Design-Matrix als $1 = Xr$ geschrieben werden kann, da unser Modell ein Intercept = 1 beinhaltet (ganze erste Spalte besteht aus Einsen).

In Gleichung $1 = Xr$ ist r Spaltenvektor mit allen Nullen bis auf den ersten Wert. In Matrixschreibweise ist die Summe der Residuen nichts anderes als $1^T(y - \hat{y})$. Dabei ist $\hat{y} = Hy$ und H ist die Projektionsmatrix. $H = X(X^T X)^{-1} X^T$.

Somit gilt dann:

$1^T(y - \hat{y}) = 1^T(I - H)y =$

$= r^T X^T(I - X(X^T X)^{-1} X^T)y =$

$= r^T(X^T - X^T X(X^T X)^{-1} X^T)y =$

$= r^T(X^T - X^T)y =$

$= 0$

c.

```
data("mtcars", package = "datasets")
y <- mtcars$mpg
X1 <- mtcars$disp
X2 <- mtcars$hp
m <- lm(y ~ X1 + X2)
r <- residuals(m)
head(r)
```

```
##         1         2         3         4         5         6
## -2.148091 -2.148091 -2.348379  1.225844  3.235770 -3.199783
```

```
standardized_resids <- function(r,X){
  n <- length(r)
  p <- ncol(X) - 1
  sigma_hat <- sqrt(sum(r^2) / (n - p - 1))
  H <- X %*% solve(t(X) %*% X) %*%t (X)
  return (r / sqrt(1-diag(H)) /sigma_hat)
}

design_mat = cbind(rep(1,length(X2)), X1, X2)
#head(design_mat)
#head(model.matrix(m)) #returns design matrix
standardized_resids(r,design_mat)
```

```
##          1          2          3          4          5          6          7
## -0.7020670 -0.7020670 -0.7758853  0.4079066  1.0795485 -1.0552598  0.1935267
##          8          9         10         11         12         13         14
## -0.1152271 -0.4279150 -1.1080264 -1.5650067 -0.4877968 -0.1941765 -0.8792905
##         15         16         17         18         19         20         21
## -0.3279304 -0.3616238  1.0355939  1.9015231  1.0936326  2.3221987 -1.0479571
##         22         23         24         25         26         27         28
## -0.6166510 -0.8515789 -0.2457508  1.6970296  0.2007351  0.3872396  1.7881741
##         29         30         31         32
##  0.7833102 -0.7830296  0.7752712 -0.9745009
```

d.

```
using_built_in <- rstandard(m)
from_my_function <- standardized_resids(r, design_mat)
#using_built_in - from_my_function
all.equal(using_built_in, from_my_function)
```

```
## [1] TRUE
```

Werte, die Builtin-Funktion und händisch geschriebene Funktion ausrechnen unterscheiden sich minimal. Das sieht man am besten, wenn man sich Differenz der zwei Residuenvetoren ausrechnet. Einige Werte sind ganz gleich und einige unterscheiden sich in ganz kleinen Nachkommastellen, die nicht mehr relevant sind. Dazu kommt es üblicherweise durch numerische und Rundungsfehler. Man kann auch mit Funktion *all.equal* überprüfen, ob alle ausgerechnete Werte in Wirklichkeit gleich sind. Das ist hier der Fall.

e.

```
studentized_resids <- function(r, x){
  ri <- standardized_resids(r, x)
  n <- nrow(x)
  p <- ncol(x) - 1
  return (ri *sqrt((n - p -2) / (n - p- 1 - ri^2) ))

}
studentized_resids(r, design_mat)
```

```
##          1          2          3          4          5          6          7
## -0.6957946 -0.6957946 -0.7704291  0.4019668  1.0827517 -1.0574065  0.1902837
##          8          9         10         11         12         13         14
## -0.1132489 -0.4218062 -1.1125598 -1.6071513 -0.4812913 -0.1909234 -0.8757501
##         15         16         17         18         19         20         21
## -0.3228260 -0.3561381  1.0369364  1.9970953  1.0974811  2.5290323 -1.0497999
##         22         23         24         25         26         27         28
## -0.6099379 -0.8474304 -0.2417284  1.7570378  0.1973810  0.3814920  1.8627667
##         29         30         31         32
##  0.7779603 -0.7776756  0.7698064 -0.9736260
```

f.

```
student_built_in <- rstudent(m)
student_my_fun <- studentized_resids(r, design_mat)
all.equal(student_built_in, student_my_fun)
```

```
## [1] TRUE
```

Wie in der Unteraufgebe d. kann man hier Funktion *all.equal* anwenden, um numeriche fehler zu beseitigen. Es liefern beide Funktionen gleiche Vektoren.

Aufgabe 20:

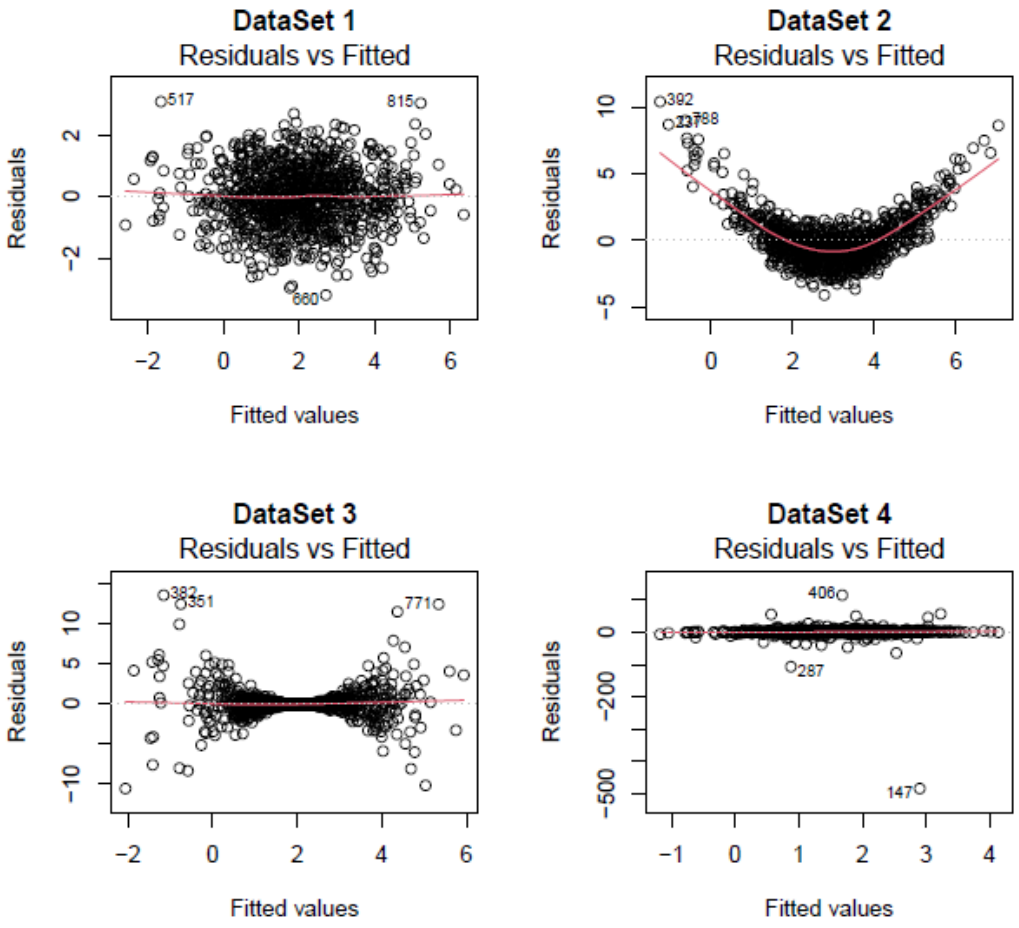


Figure 1: Residuals vs. fitted plot

DataSet1: Residuen sind in Umgebung von 0 zentriert und haben approximativ gleiche Varianz. Sie deuten kein Muster an und sind unabhägig von den Fitted Values.

DataSet2: Residuen sind nicht in Umgebung von 0 zentriert, aber es kann eine quadratische Verteilung angenommen werden.

DataSet3: Die residuen sind gegen 0 zentriert, aber je weiter sie vom Fitted Value 2 entfernt sind, desto größer wird ihre Varianz. Dieses Verhalten weist darauf hin, dass DataSet3 unhomoskedastisch ist.

DataSet4: Residuen sind wiederum gegen 0 zentriert und haben eine konstante Varianz. Davon sind aber nur ganz große Residuen ausgenommen.

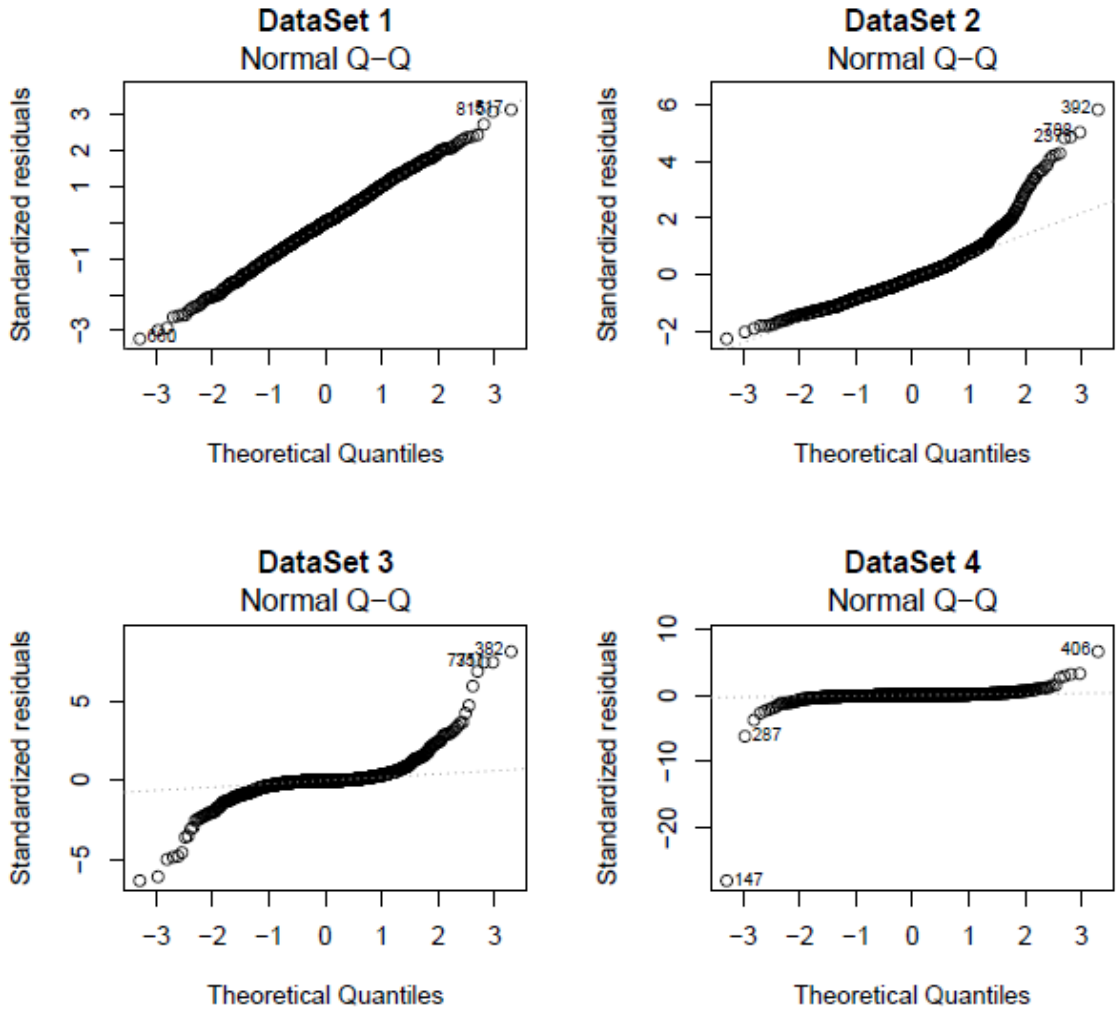


Figure 2: QQplots for the residuals

DataSet1: Standardisierte Residuen liegen entlang der QQ-Linie, daher kann Normalverteilung angenommen werden. Annahmen der Linearität, Homoskedastizität und Normalvertelung sind nicht verletzt.

DataSet2: Residuen schwanken im oberen Teil der Verteilung stark nach oben, was die Annahme der Normalverteilung direkt verletzt. Dies bedeutet, dass die Verteilung rechtsschief ist. Anhand dessen ist es klar, dass Annahme der Linearität und Normalverteilung verletzt ist.

DataSet3: An beiden oberen und untereren Teilen der Verteilung schwanken Residuen stark nach oben bzw. unten, weswegen die Ananahme der Normalität nicht beibehalten sein darf. Somit kommt zu Annahmeverletzung der Homoskedastizität und Normalverteilung.

DataSet4: An beiden oberen und untereren Teilen der Verteilung schwanken Residuen nach oben bzw. unten, jedoch nicht so stark wie in DataSet3. Ausreißer dürfen jedoch nicht vernachlässigt werden und die Annahme der Normalität muss verworfen werden.

Aufgabe 21:

$$\text{Model 1:} \quad \widehat{sales} = 7 + 0.03 \cdot TV + 0.2 \cdot youtube + 0.5 \cdot social$$

$$\text{Model 2:} \quad \widehat{sales} = 6 + 0.2 \cdot youtube + 0.5 \cdot social + 0.25 \cdot social \cdot youtube$$

$$\text{Model 3:} \quad \widehat{sales} = 5.5 + 0.2 \cdot youtube + 0.3 \cdot youtube^2 + 0.1 \cdot social$$

$$\text{Model 4:} \quad \widehat{sales} = 200 + 0.05 \cdot (TV - \bar{TV}) + 0.2 \cdot (youtube - \bar{youtube}) + 0.5 \cdot social$$

a. Man interpretiere im Modell 1 TV-Koeffizient

Falls man TV-Budget um x Prozent erhöhen und keine anderen Variablen ändern würde, wären $0.03 * x * 100 = 3x$ mehr Einheiten erwartet.

b. Man interpretiere im Modell 1 Intercept

Intercept ist der Umsatz (in hundert Einheiten), den man ohe zusätzliche Werbung erwartet. Wenn keine Webung miteinbezogen wird, erwartet man 700 Einheiten zu verkaufen.

c.

Wenn man Youtube-Budget um x erhöhen und andere Variablen nicht ändern würde, erwarteter Verkauf würde um $0.2 * x + 0.25 * x * social$ steigen. Wenn Budget von soziallen Medien um x erhöht wäre (ohne andere Variablen zu ändern), erwartet man eine Verkaufssteigerung von $0.5 * x + 0.25 * youtube * x$. Wenn ein großes Social Media- Budget gegeben ist, je mehr man Youtube-Budget erhöht, desto hat diese Erhöhung einen größeren Effekt auf erwarteten Verkauf.

d.

Es besteht eine quadratische Beziehung zwischen *sales* und *youtube*. Wenn ein bereits großes Youtube-Budget gegeben ist, mit jeder Erhöhung, rechten man mit einem erhöhten Verkauf, bezogen auf positive quadratische Komponente. D.h. um möglichst viel Umsatz zu machen, sollten Firmen so viel wie es geht, in Youtube-Werbung investieren.

Wäre Koeffizient von $youtube^2$ negativ, würde erwarteter Verkauf zuerst erhöht, wenn man Youtube-Budget erhöhen würde. Dann würde der Verkauf senen, da mann eine negative quadratische Komponente hat. Dies kann wie folgt ausgedrückt werden: je größer Youtube-Budget, desto weniger positive wirkung auf weiteren verkauf. D.h. Firmen sollen eine gerechtfertigte Ziffer in Youtube-Werbung investieren (nicht zu viel, aber auch nicht zu wenig).

e. Man interpretiere intercept im Modell 4

Intercept ist erwartete Anzahl an Verkäufen (in hundert Einheiten), wenn TV-Budget = $\bar{TV}$. Youtube-Budget ist $\bar{youtube}$ und Social Media Budget ist 0. Somit erwartet man 20000 Einheiten zu verkaufen.

f. Koeffizient von $(TV - \bar{TV})$

Wenn TV-Budget um x erhöht wird, steigt erwarteter Verkauf um $x * 0.005 * 100 = 5$ Einheiten.

Aufgabe 22:

```
data("ToothGrowth", package ="datasets")
```

a.

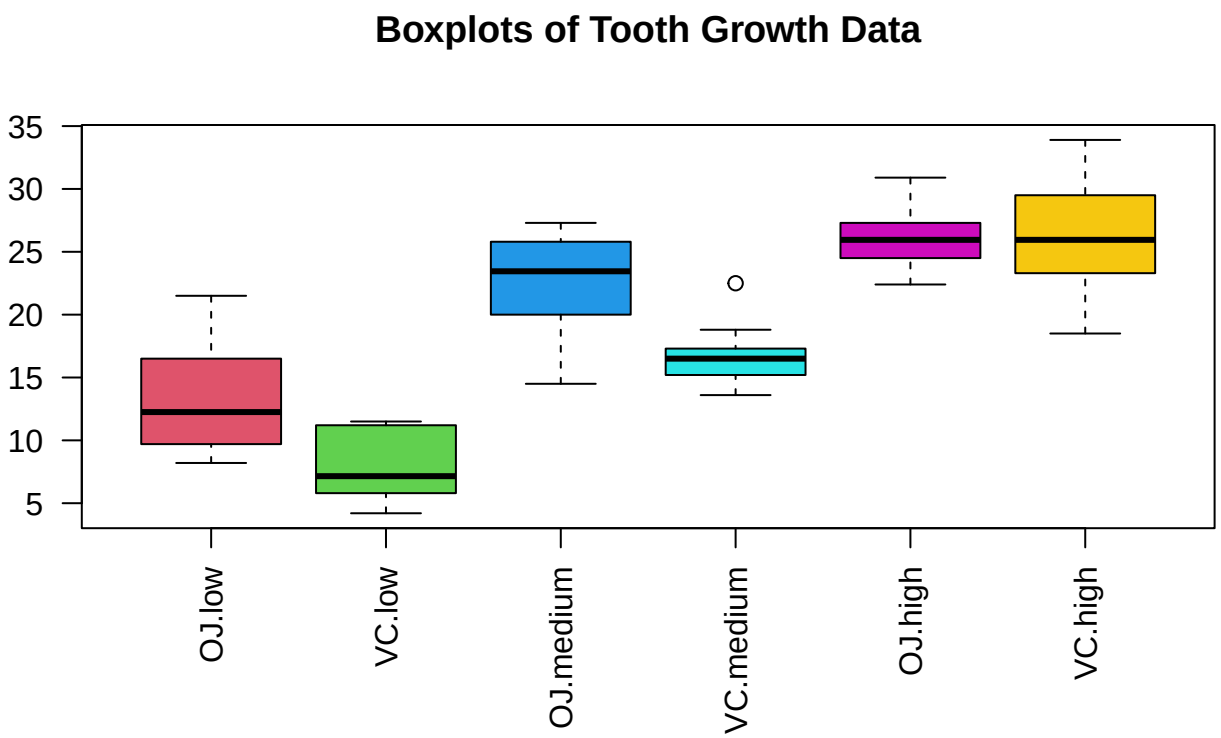
```
ToothGrowth$dose <- factor(ToothGrowth$dose,
                           labels = c("low", "medium", "high"))
```

```
summary(ToothGrowth)
```

```
##       len        supp        dose
##  Min.   : 4.20   OJ:30   low   :20
##  1st Qu.:13.07   VC:30   medium:20
##  Median :19.25           high  :20
##  Mean   :18.81
##  3rd Qu.:25.27
##  Max.   :33.90
```

Jedes Suplement war 30 mal während der Beobachtung verwendet und jede Dose war 20 mal genutzt.

```
par(mar = c(7, 2, 5 , 1), mgp = c(6, 1, 0))
boxplot(len ~ supp * dose, data = ToothGrowth,
        xlab = "Supplement and dose",
        ylab = "Tooth Length",
        main = "Boxplots of Tooth Growth Data",
        col = c(2:7), las = 2)
```




```
tab1 <- ToothGrowth |>
  with(aggregate(data.frame(len), data.frame(supp),
                 function(x) c(m = mean(x), s = sd(x), l = length(x))))
```

```
tab2 <- ToothGrowth |>
  with(aggregate(data.frame(len), data.frame(dose),
                 function(x) c(m = mean(x), s = sd(x), l = length(x))))
```

```
tab3 <- ToothGrowth |>
  with(aggregate(data.frame(len), data.frame(supp, dose),
                 function(x) c(m = mean(x), s = sd(x), l = length(x))))
```

```
tab4 <- ToothGrowth |>
  with(aggregate(data.frame(len), data.frame(supp, dose),
                 mean))
```

b.

```
model1 <- lm(len ~ supp + dose, data = ToothGrowth)
summary(model1)
```

```
##
## Call:
## lm(formula = len ~ supp + dose, data = ToothGrowth)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.085 -2.751 -0.800  2.446  9.650
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12.4550     0.9883  12.603  < 2e-16 ***
## suppVC       -3.7000     0.9883  -3.744 0.000429 ***
## dosemedium    9.1300     1.2104   7.543 4.38e-10 ***
## dosehigh     15.4950     1.2104  12.802  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.828 on 56 degrees of freedom
## Multiple R-squared:  0.7623, Adjusted R-squared:  0.7496
## F-statistic: 59.88 on 3 and 56 DF,  p-value: < 2.2e-16
```

***Interpretation*:** Intercept ist erwartete Länge von Odontoblasten, wenn Meerschwein eine niedige Dosisvom Vitamin C durch Orangensaft erhalten hat.

suppVC ist eine negative Zahl, weswegen sie eigentlich als erwartete Verringerung der Länge von Ortoblasten Interpretiert werden kann, falls Meerschwein Vitamin C durch *Asorbic Acid* erhalten hat statt durch Orangensaft.

Dosemedium ist die erwartete Erhöhung, wenn Wasserschwein eine mittlere Dosis vom Vitamin C erhalten hat.

c.

```
model2 <- lm(len ~ supp + dose,
             data = ToothGrowth,
             contrasts = list(supp = "contr.sum",
                              dose = "contr.sum"))
summary(model2)
```

```
##
## Call:
## lm(formula = len ~ supp + dose, data = ToothGrowth, contrasts = list(supp = "contr.sum",
##     dose = "contr.sum"))
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.085 -2.751 -0.800  2.446  9.650
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  18.8133     0.4941  38.073  < 2e-16 ***
## supp1         1.8500     0.4941   3.744 0.000429 ***
## dose1        -8.2083     0.6988 -11.746  < 2e-16 ***
## dose2         0.9217     0.6988   1.319 0.192573
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.828 on 56 degrees of freedom
## Multiple R-squared:  0.7623, Adjusted R-squared:  0.7496
## F-statistic: 59.88 on 3 and 56 DF,  p-value: < 2.2e-16
```

Interpretation:*dose1* ist Differenz der erwarteten Auswirkung für eine niedrigere Dosis in Bezug auf gesamten Mittelwert.

dose2 ist Differenz der erwarteten Auswirkung für eine mittlere Dosis, wobei der Gesamtmittelwert miteinbezogen wird.

supp1 ist die Differenz der erwarteten Auswirkung, wenn Vitamin C durch Orangejuice verabreicht wird. Gesamtmittelwert muss miteinbezogen werden un darf sich nicht ändern.

d.

```
model3 <- lm(len ~ dose*supp, data = ToothGrowth)
summary(model3)
```

```
##
## Call:
## lm(formula = len ~ dose * supp, data = ToothGrowth)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -8.20  -2.72  -0.27   2.65   8.27
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         13.230      1.148  11.521 3.60e-16 ***
## dosemedium           9.470      1.624   5.831 3.18e-07 ***
## dosehigh            12.830      1.624   7.900 1.43e-10 ***
## suppVC              -5.250      1.624  -3.233  0.00209 **
## dosemedium:suppVC   -0.680      2.297  -0.296  0.76831
## dosehigh:suppVC      5.330      2.297   2.321  0.02411 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.631 on 54 degrees of freedom
## Multiple R-squared:  0.7937, Adjusted R-squared:  0.7746
## F-statistic: 41.56 on 5 and 54 DF,  p-value: < 2.2e-16
```

```
#treatment contrasts are there by default
```

```
values <- unique(cbind(model.matrix(model3), ToothGrowth[, -1]))

contrast_matrix <- rbind(c(1, 0, 0, 0, 0, 0),
```

```
                          c(1, 1, 0, 0, 0, 0),
                          c(1, 0, 1, 0, 0, 0),
                          c(1, 0, 0, 1, 0, 0),
                          c(1, 1, 0, 1, 1, 0),
                          c(1, 0, 1, 1, 0, 1))

contrast_matrix%*%coefficients(model3)
```

```
##       [,1]
## [1,] 13.23
## [2,] 22.70
## [3,] 26.06
## [4,]  7.98
## [5,] 16.77
## [6,] 26.14
```

Werte der Kontrastmatrix können aus Tabelle abgelesen werden, die durch den Befehl *unique(cbind(model.matrix(model3), ToothGrowth[, -1]))* erzeugt wurden. So ist der H**aupteffekt von OJ.low** Intercept (nur Intercept ist auf 1 gesetzt).

Haupteffekt von OJ.medium: intercept + dosemedium

Haupteffekt von OJ.high: intercept + dosehigh

Haupteffekt von VC.low: intercept + suppVC

Haupteffekt von VC.medium: intercept + dosemedium + dosemedium:suppVC

Haupteffekt von VC.high: intercept + dosehigh + suppVC + dosehigh:suppVC

contrast_matrix%%coefficients(model3)* ergibt den geschätzten Haupteffekt für jede Faktorkombination, da Kontrastmatrix mit $\hat{\beta}$ multipliziert wird.

Man sollte dieses Modell noch mit Mitteln in vergleichen:

```
with(ToothGrowth, aggregate(len, list(dose, supp), mean))
```

```
##   Group.1 Group.2     x
## 1     low      OJ 13.23
## 2  medium      OJ 22.70
## 3    high      OJ 26.06
## 4     low      VC  7.98
## 5  medium      VC 16.77
## 6    high      VC 26.14
```

Wie man leicht sehen kann, sind der Haupteffekt von jeder Faktorkombination und Mittel von jeder Faktorkombination gleich.

Aufgabe 23:

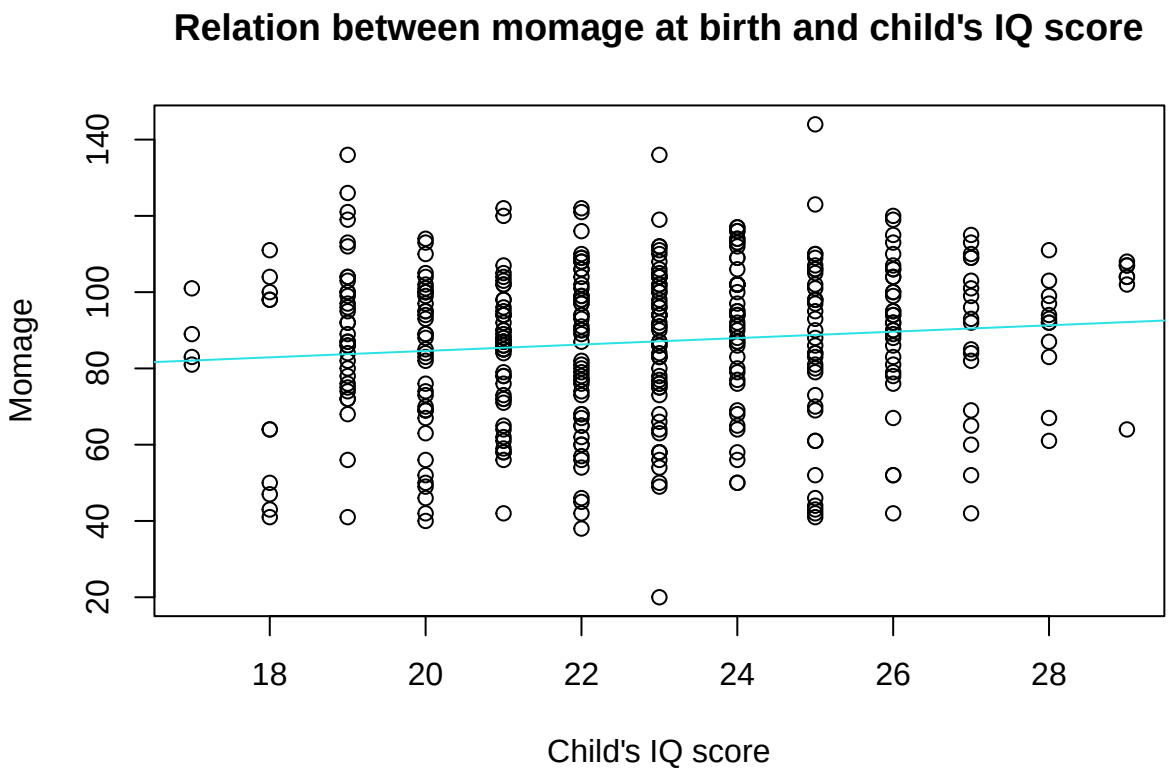
```
#install.packages('haven')
library(haven)
child_iq <- read_dta("child.iq.dta")
```

a.

```
model <- lm(ppvt ~ momage, data = child_iq)
summary(model)
```

```
##
## Call:
## lm(formula = ppvt ~ momage, data = child_iq)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -67.109 -11.798   2.971  14.860  55.210
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  67.7827     8.6880   7.802 5.42e-14 ***
## momage        0.8403     0.3786   2.219    0.027 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.34 on 398 degrees of freedom
## Multiple R-squared:  0.01223,    Adjusted R-squared:  0.009743
## F-statistic: 4.926 on 1 and 398 DF,  p-value: 0.02702
```

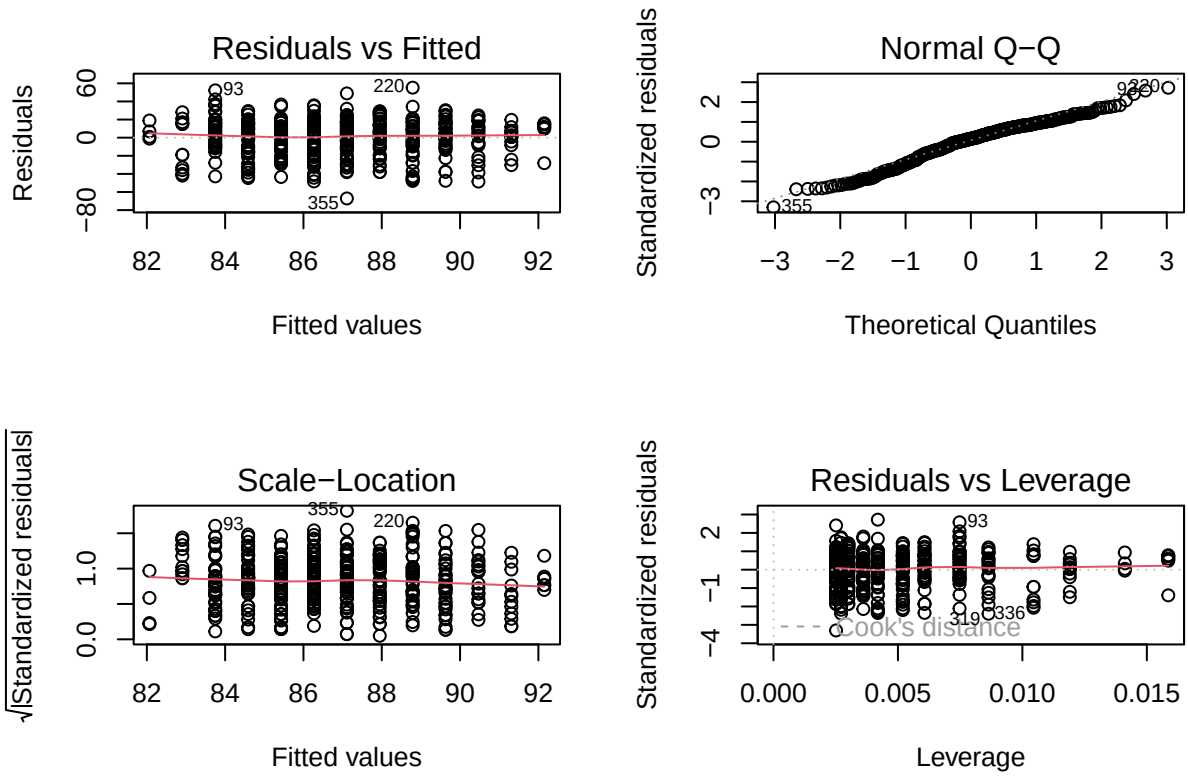
```
plot(child_iq$momage, child_iq$ppvt,
     main = "Relation between momage at birth and child's IQ score",
     ylab = "Momage",
     xlab ="Child's IQ score")
abline(model, col = 5)
```



Interpretation des Steigungskoeffizientes (slope coefficient): Wenn Mutter bei Geburt ein Jahr älter ist, erhöht dies erwarteten IQ des Kindes um 0.8403.

Wenn wir uns ansehen wollen, ob Modellannahmen verletzt wurden, können wir einfach Regressionsmodell ploten und uns QQ -Plot und residuals vs Fitted Plot näher anschauen.

```
par(mfrow = c(2, 2))
plot(model)
```



QQ-Plot sagt uns, dass die Verteilung etwas linksschief ist, was die Normalitätsannahme verletzt. Residuen sind gegen 0 zentriert und deuten auf kein Muster hin. Homoskedastizität ist erfüllt.

Wie alt soll die Mutter bei Geburt sein, damit ihr drei-jähriges Kind IQ über 90 hat?

```
best_age_is_after <- (90 - coefficients(model)[1]) / coefficients(model)[2]
best_age_is_after
```

```
## (Intercept)
##     26.4406
```

```
#max(child_iq$momage)
```

Mutter sollte älter als 26.4406 sein, damit ihr Kind IQ über 90 hat. In unserem Dataset ist aber die älteste Mutter 29 und daher können wir keine Annahmen darüber treffen, wie Verhältnis vom IQ des Kindes und Mutteralter ist, wenn Mutter bei Geburt über 30 ist. Anhand unseres Datensets würde ich vorschlagen, dass die Mutter bei Geburt zwischen 27 und 29 sein soll.

b.

```
model2 <- lm(ppvt ~ I(momage) + as.numeric(educ_cat),
             data = child_iq)
summary(model2)
```

```
##
```

```
## Call:
## lm(formula = ppvt ~ I(momage) + as.numeric(educ_cat), data = child_iq)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -61.763 -13.130   2.495  14.620  55.610
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)           69.1554     8.5706   8.069 8.51e-15 ***
## I(momage)              0.3433     0.3981   0.862 0.389003
## as.numeric(educ_cat)   4.7114     1.3165   3.579 0.000388 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.05 on 397 degrees of freedom
## Multiple R-squared:  0.04309,    Adjusted R-squared:  0.03827
## F-statistic: 8.939 on 2 and 397 DF,  p-value: 0.0001594
```

Wenn muter bei Geburt ein Jahr älter ist, wird IQ des Kindes um 0.3433 höher sein. Wenn die Bildung der Mutter um ein Niveau höher ist, dann wird IQ des Kindes um 4.7114 höher sein. Wir wissen widerum nicht, wie sich Beziehung zwischen dem IQ des Kindes und *momage + educ_cat* verhalten wird, wenn Mutter bei Geburt älter als 29 ist.

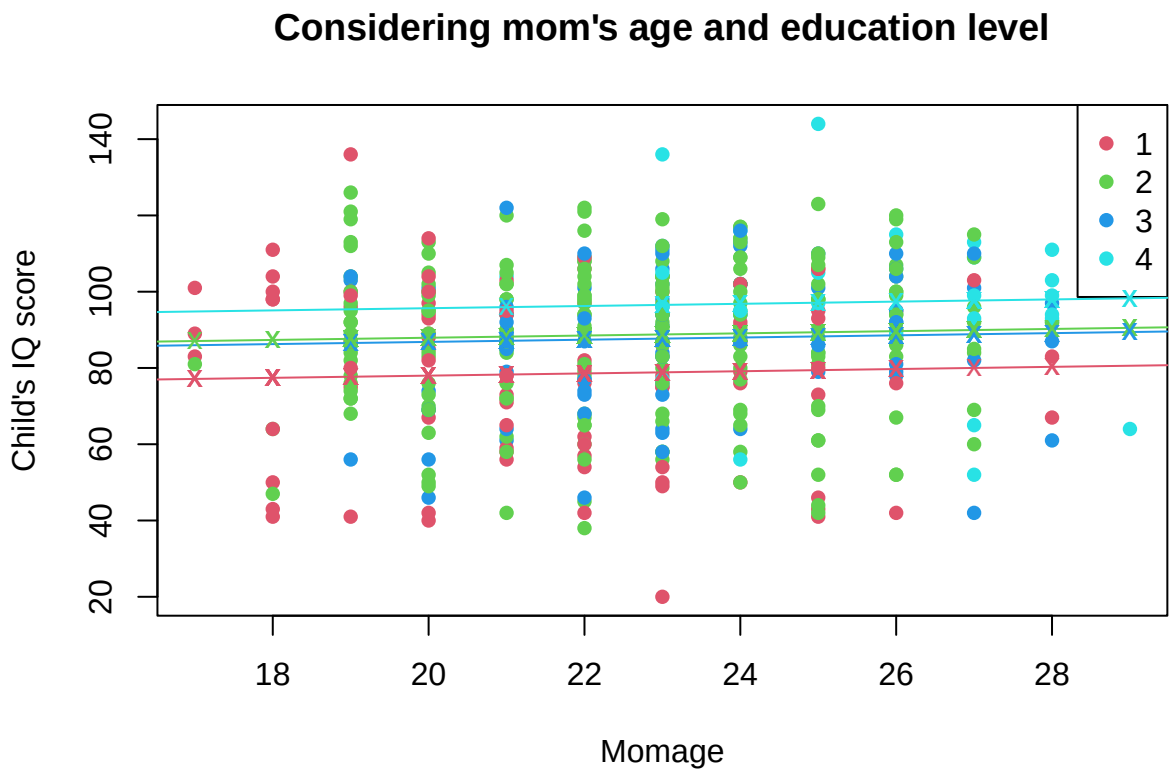
Effekt von Alter ist nicht signifikant, daher macht es keinen Sinn, in diese Richtung Empfehlungen zu machen.

c.

```
model3 <- lm(ppvt ~ momage + as.factor(educ_cat),
             data = child_iq)
summary(model3)
```

```
##
## Call:
## lm(formula = ppvt ~ momage + as.factor(educ_cat), data = child_iq)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -58.853 -12.112   1.779  14.687  58.298
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)           72.2360     8.8700   8.144 5.07e-15 ***
## momage                 0.2877     0.3985   0.722 0.470736
## as.factor(educ_cat)2   9.9365     2.5953   3.829 0.000150 ***
## as.factor(educ_cat)3   8.8416     3.2203   2.746 0.006316 **
## as.factor(educ_cat)4  17.6809     4.7065   3.757 0.000198 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.92 on 395 degrees of freedom
## Multiple R-squared:  0.0598, Adjusted R-squared:  0.05028
## F-statistic:  6.28 on 4 and 395 DF,  p-value: 6.59e-05
```

```r
cols <- c(2:6)

plot(ppvt ~ momage, data = child_iq,
          pch = 16,
          col=cols[as.numeric(educ_cat)],
          xlab ="Momage",
          ylab = "Child's IQ score",
          main = "Considering mom's age and education level")


legend("topright",bg = "white",
       col=cols, pch = 16,
       levels(as.factor(child_iq$educ_cat)))

abline(coef(model3)[1],coef(model3)[2],col=cols[1])
abline(coef(model3)[1]+coef(model3)[3],coef(model3)[2],col=cols[2])
abline(coef(model3)[1]+coef(model3)[4],coef(model3)[2],col=cols[3])
abline(coef(model3)[1]+coef(model3)[5],coef(model3)[2],col=cols[4])

with(child_iq, points(momage,fitted(model3),
                      col=cols[as.numeric(educ_cat)],
                      pch = "x"))
```



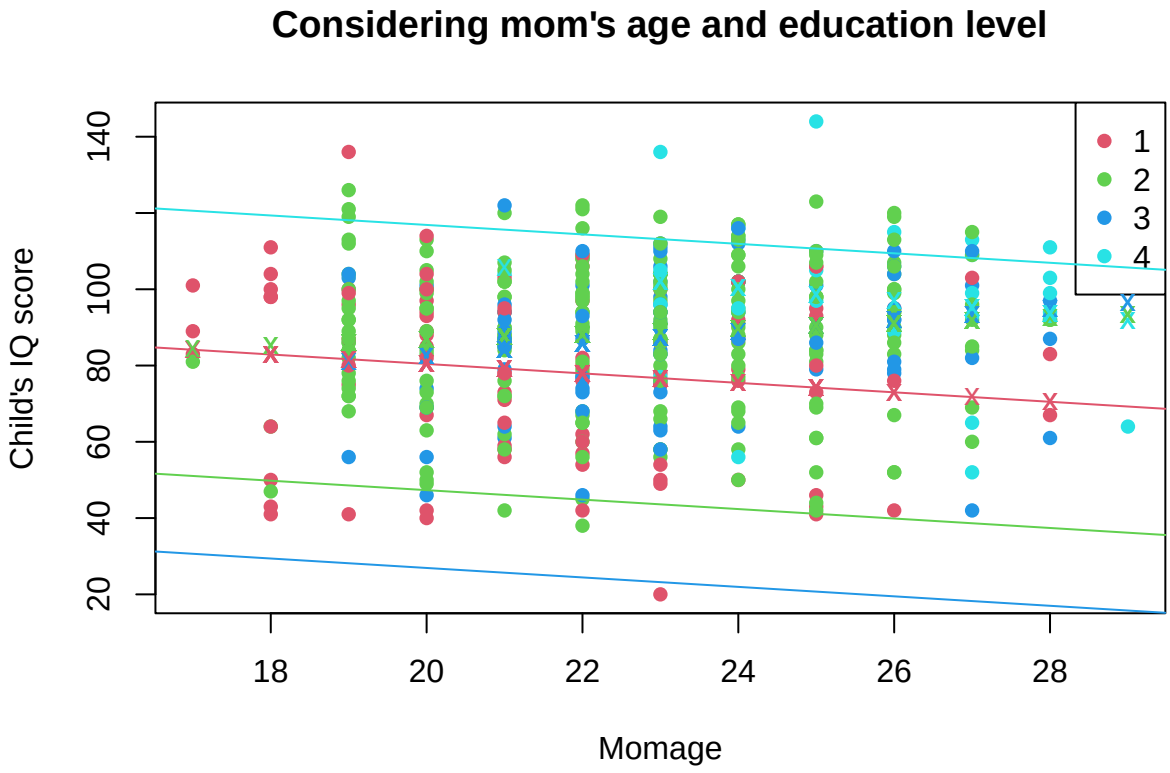

```r
#the following will fit regression with different slopes and
#different intercepts for the different education groups.
```

```r
#I wasn't sure if i shold have done this or just previous model, so here it comes
model4 <- lm(ppvt ~  momage*as.factor(educ_cat),
             data = child_iq)
summary(model4)
```

```
##
## Call:
## lm(formula = ppvt ~ momage * as.factor(educ_cat), data = child_iq)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -56.70 -11.80   2.07  14.58  54.34
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   105.2202    17.6127   5.974  5.2e-09 ***
## momage                         -1.2402     0.8097  -1.532   0.1264
## as.factor(educ_cat)2          -33.0929    21.5732  -1.534   0.1258
## as.factor(educ_cat)3          -53.4970    27.9460  -1.914   0.0563 .
## as.factor(educ_cat)4           36.4537    49.5065   0.736   0.4620
## momage:as.factor(educ_cat)2    1.9704     0.9764   2.018   0.0443 *
## momage:as.factor(educ_cat)3    2.7862     1.2293   2.266   0.0240 *
## momage:as.factor(educ_cat)4   -0.4799     1.9635  -0.244   0.8070
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.81 on 392 degrees of freedom
## Multiple R-squared:  0.07705,    Adjusted R-squared:  0.06057
## F-statistic: 4.675 on 7 and 392 DF,  p-value: 4.756e-05
```

```r
cols <- c(2:6)

plot(ppvt ~ momage, data = child_iq,
          pch = 16,
          col=cols[as.numeric(educ_cat)],
          xlab ="Momage",
          ylab = "Child's IQ score",
          main = "Considering mom's age and education level")


legend("topright",bg = "white",
       col=cols, pch = 16,
       levels(as.factor(child_iq$educ_cat)))

abline(coef(model4)[1], coef(model4)[2], col = cols[1])
abline(coef(model4)[1] + coef(model4)[3], coef(model4)[2],
       col = cols[2])
abline(coef(model4)[1] + coef(model4)[4],
       coef(model4)[2], col = cols[3])
abline(coef(model4)[1] + coef(model4)[5],
       coef(model4)[2], col = cols[4])

with(child_iq, points(momage, fitted(model4),
```

```
                col = cols[as.numeric(educ_cat)],
                pch = "x"))
```



Aufagbe 24:

```
data("RailTrail", package ="mosaicData")
```

a.

```
#remove duplicated observations if any
RailTrail <- RailTrail[!duplicated(RailTrail), ]

#remove observations with missing values if any
RailTrail <- RailTrail[complete.cases(RailTrail), ]

#remove columns which have same names
RailTrail <- RailTrail[, !duplicated(colnames(RailTrail))]

#columns weekday and daytype basicaly hold the same information,
#so we are going to remove column weekday
RailTrail <- subset(RailTrail, select = -c(weekday))

#column avgtemp is calculated as the average of hightemp and lowtemp,
```

```
#so this column is redudant, as it can easyly be computed.
#We are going to delete this column from dataset
RailTrail <- subset(RailTrail, select = -c(avgtemp))

#factor season
RailTrail$season <- factor(RailTrail$spring + 2*RailTrail$summer
                           + 3*RailTrail$fall)
levels(RailTrail$season) <-  c("Spring", "Summer", "Fall")

RailTrail <- subset(RailTrail, select = -c(spring, summer, fall))
```

b.

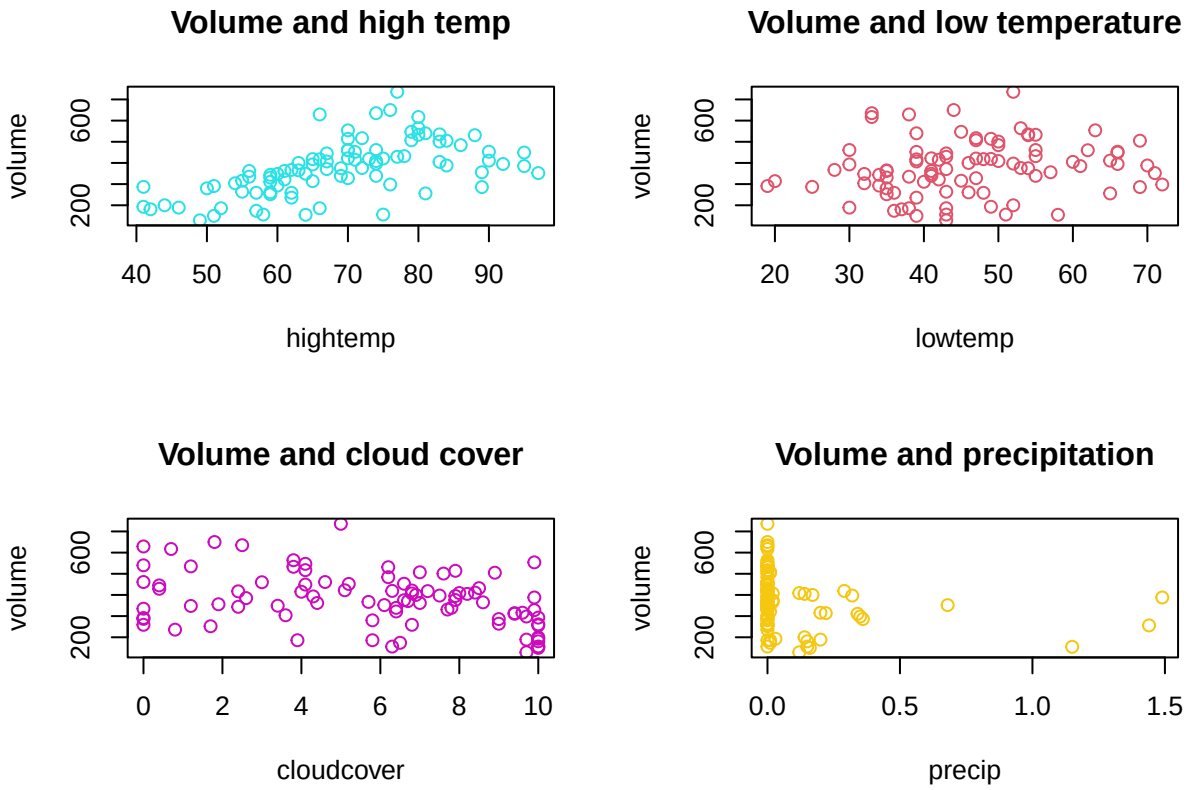
Summary:

```
summary(RailTrail)
```

```
##     hightemp        lowtemp        cloudcover         precip
##  Min.   :41.00   Min.   :19.00   Min.   : 0.000   Min.   :0.00000
##  1st Qu.:59.25   1st Qu.:38.00   1st Qu.: 3.650   1st Qu.:0.00000
##  Median :69.50   Median :44.50   Median : 6.400   Median :0.00000
##  Mean   :68.83   Mean   :46.03   Mean   : 5.807   Mean   :0.09256
##  3rd Qu.:77.75   3rd Qu.:53.75   3rd Qu.: 8.475   3rd Qu.:0.02000
##  Max.   :97.00   Max.   :72.00   Max.   :10.000   Max.   :1.49000
##      volume        dayType             season
##  Min.   :129.0   Length:90          Spring:53
##  1st Qu.:291.5   Class :character   Summer:25
##  Median :373.0   Mode  :character   Fall  :12
##  Mean   :375.4
##  3rd Qu.:451.2
##  Max.   :736.0
```

Scatterplots:

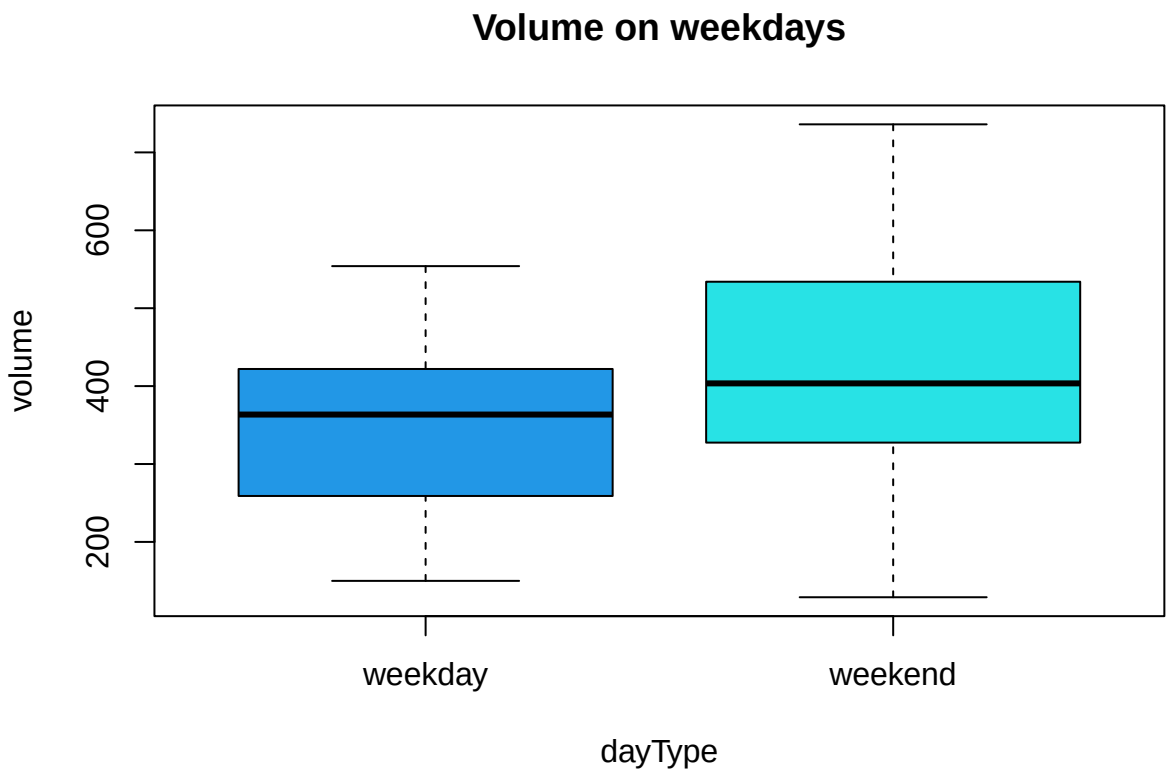
```
par(mfrow = c(2, 2))
plot(volume ~ hightemp, data = RailTrail,
     main = "Volume and high temp", col = 5)
plot(volume ~ lowtemp, data =  RailTrail,
     main = "Volume and low temperature", col = 2)
plot(volume ~ cloudcover, data = RailTrail,
     main ="Volume and cloud cover", col = 6)
plot(volume ~ precip, data = RailTrail,
     main = "Volume and precipitation", col = 7)
```



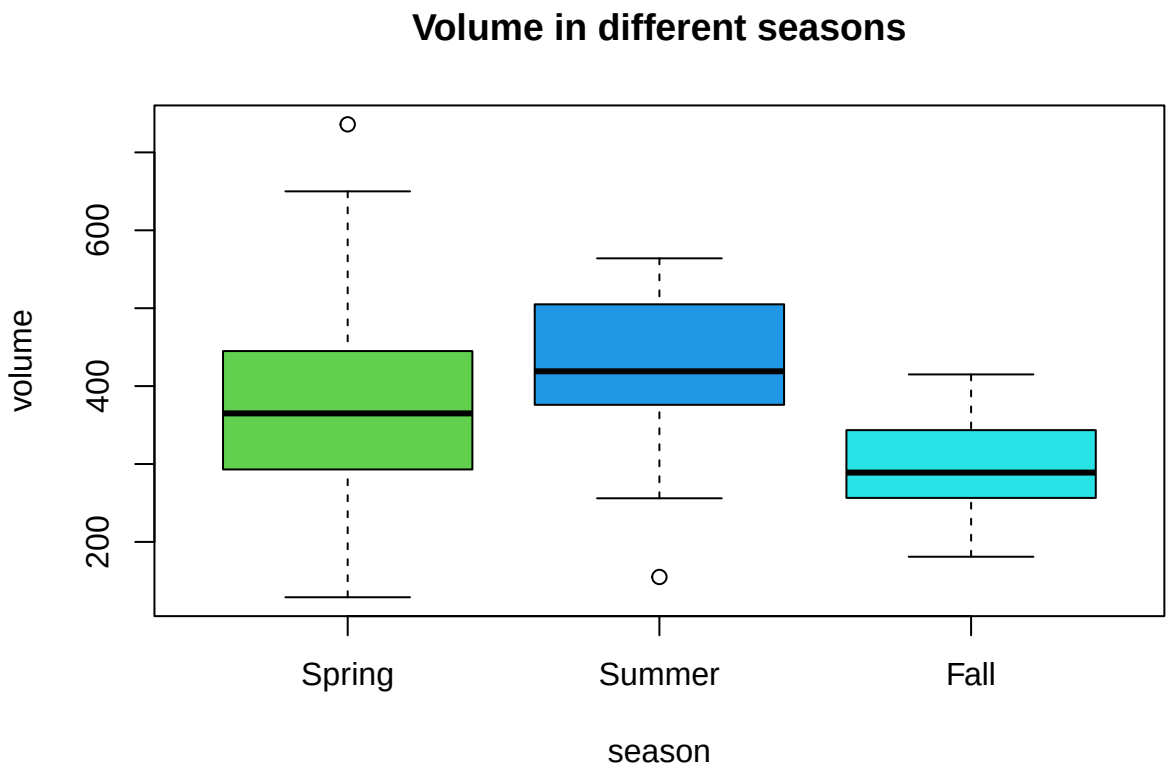
Korrelation zwischen *hightemp, lowtemp* and *volume* is positiv, da je größer *high-* bzw. *lowtemp* desto größer *volume.* Umgekehrt ist mit *cloudcover* and *percip.* Je größer diese zwei Variablen Verden, desto kleiner ist *volume.* das weist auf eine ***negative Korrelation*** **hin.**

Boxplots:

```
boxplot(volume ~ dayType, data = RailTrail,
        main ="Volume on weekdays", col = c(4:5))
```

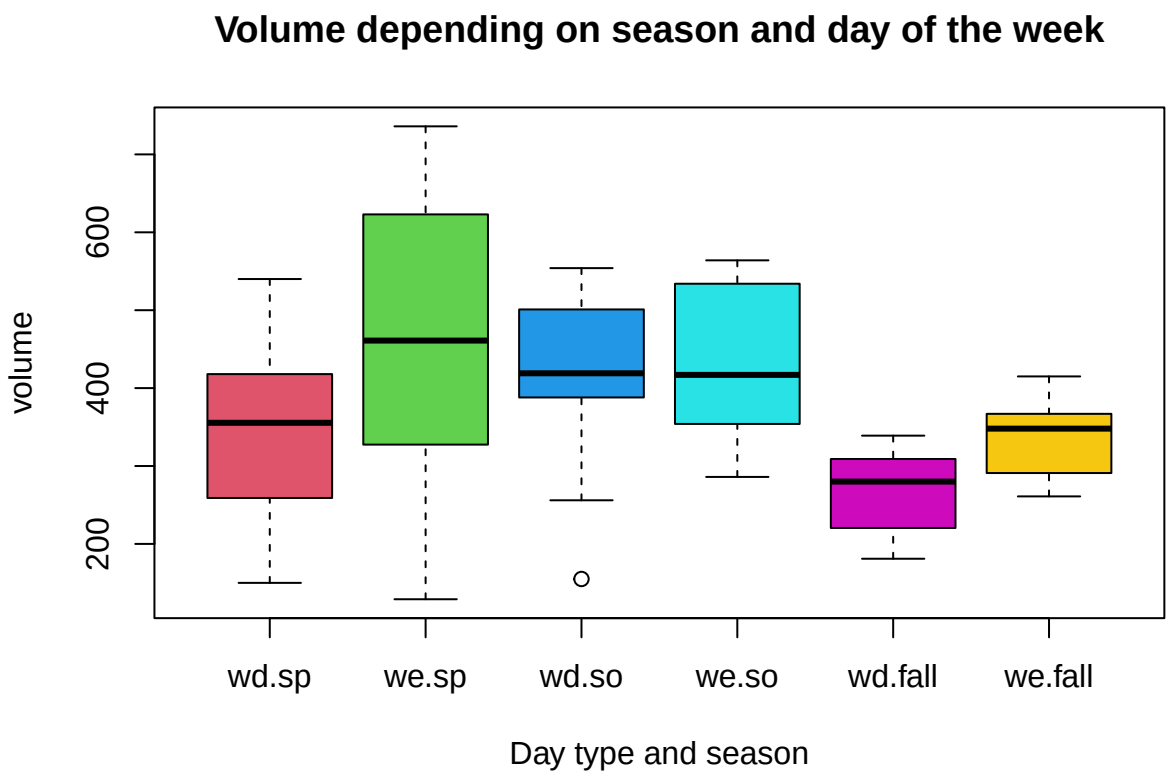



```
boxplot(volume ~ season, data = RailTrail,
        main = "Volume in different seasons",
        col = c(3:6))
```



Wir können hier auch Variablen *dayType* und *season* zusammenfassen. So bekommen wir eine Boxplot-Darstellung, wo wir gleich alle mögliche Paare von *dayType* und *Season* haben.

```
boxplot(volume ~ dayType*season, data = RailTrail,
        main ="Volume depending on season and day of the week",
        col = c(2:7),
        names =c("wd.sp", "we.sp", "wd.so", "we.so", "wd.fall", "we.fall"),
        xlab = "Day type and season")
```



Wenn man sich Boxplot-Darstellung der Beziehung von *dayType* und *volume* anschaut, fällt einem leicht auf, dass *volume* am Wochenende im Durchschnitt höher ist als an einem Arbeitstag. So erreicht *volume* im Sommer seinen höchsten Wert. Wenn man sich Beziehung von *dayType*season* und *volume* ansehen will, stellt sich heraus, dass im Schnitt *volume* am größten an Wochenenden in Frühling ist.

c.

```
model_rt <- lm(formula = volume ~ hightemp + lowtemp + cloudcover + precip +
    dayType + season,
    data = RailTrail)
summary(model_rt)
```

```
##
## Call:
## lm(formula = volume ~ hightemp + lowtemp + cloudcover + precip +
##     dayType + season, data = RailTrail)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -261.976  -39.110    8.216   45.242  269.753
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      53.537     85.580   0.626   0.5333
## hightemp          6.052      1.290   4.690 1.08e-05 ***
## lowtemp          -1.018      1.571  -0.648   0.5187
```

```
## cloudcover       -7.252      3.843  -1.887   0.0627 .
## precip          -95.697     42.573  -2.248   0.0273 *
## dayTypeweekend   35.904     22.429   1.601   0.1133
## seasonSummer    -11.761     35.774  -0.329   0.7432
## seasonFall      -35.915     32.993  -1.089   0.2795
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 92.84 on 82 degrees of freedom
## Multiple R-squared:  0.5112, Adjusted R-squared:  0.4695
## F-statistic: 12.25 on 7 and 82 DF,  p-value: 1.279e-10
```

```
anova(model_rt)
```

```
## Analysis of Variance Table
##
## Response: volume
##            Df Sum Sq Mean Sq F value    Pr(>F)
## hightemp    1 490744  490744 56.9330 5.516e-11 ***
## lowtemp     1 111176  111176 12.8980 0.0005593 ***
## cloudcover  1  56061   56061  6.5038 0.0126245 *
## precip      1  49695   49695  5.7653 0.0186085 *
## dayType     1  18822   18822  2.1836 0.1433192
## season      2  12647    6324  0.7336 0.4832955
## Residuals  82 706813    8620
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

d.

Hypothese H_0: Lineares Modell, welches *cloudcover* einbezieht ist **nicht besser** als eines, des *cloudcover* nicht einbezieht

Alternative H_1: Lineares Modell, welches *cloudcover* einbezieht ist **besser** als eines, des *cloudcover* nicht einbezieht

```
rail_model <- lm(volume ~ hightemp+lowtemp+cloudcover+precip,
            data = RailTrail)
rail_model_1 <- lm(volume ~ hightemp + lowtemp + precip,
                   data = RailTrail)

anova(rail_model, rail_model_1)
```

```
## Analysis of Variance Table
##
## Model 1: volume ~ hightemp + lowtemp + cloudcover + precip
## Model 2: volume ~ hightemp + lowtemp + precip
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     85 738282
## 2     86 771236 -1    -32954 3.7941 0.05473 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Da, p-Wert 0.05473 ist, kann die Hypothese H_0 auf Signifikanzniveau $\alpha = 0.1$ nicht verworfen werden. Der Unterschied zwischen zwei Modellen ist nicht signifikant und man muss *cloudcover* ***ins Modell nicht einbeziehen.***

e.

$H_0 : \beta_{hightemp} - \beta_{lowtemp} = 0$

$H_1 : \beta_{hightemp} - \beta_{lowtemp} \neq 0$

```
model_fitted <- lm (volume ~ I(hightemp + lowtemp) + hightemp + precip,
                    data = RailTrail)
summary(model_fitted)
```

```
##
## Call:
## lm(formula = volume ~ I(hightemp + lowtemp) + hightemp + precip,
##     data = RailTrail)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -285.31  -48.56   -4.01   46.04  303.22
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -15.812     54.594  -0.290  0.77280
## I(hightemp + lowtemp)   -2.639      1.221  -2.161  0.03345 *
## hightemp                10.247      2.063   4.968 3.41e-06 ***
## precip                -118.759     41.681  -2.849  0.00548 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 94.7 on 86 degrees of freedom
## Multiple R-squared:  0.4666, Adjusted R-squared:  0.448
## F-statistic: 25.08 on 3 and 86 DF,  p-value: 9.448e-12
```