Prof. Dr. M. Kateri,
Dr. W. Herff,
L. Kaufmann, M.Sc.

RWTH Aachen, SS 2022
Release date: May 2nd, 2022
Solution: Week starting from May 16th, 2022

Applied Data Analysis

R-Laboratory 5

Implementing Normal Linear Models – Linear Models Beyond Normality Analysis of Variance

Useful packages and functions:

- `model.matrix()`
- `qr()`
- `qr.solve()`
- `chol2inv()`
- `qt()`
- `anova()`
- `rexp()`
- `density()`
- `anova()`
- `car`
- `car::leveneTest()`
- `aov()`
- `TukeyHSD()`
- `par()`

Task 18

(a) Load the `.RData` file of the pre-processed data of *Solar* (Task 10) into the R workspace.

(b) Create four boxplots for the attribute `Pmax`, one for each batch.

(c) Carry out an analysis of variance for the attribute `Pmax` regarding the factor `batch`.

(d) Analyze the model assumptions as in Task 14 (c). Additionally, test on level $\alpha = 0.05$, if there is evidence against the assumption of equal variance in each group of factor level.

Hint: Use a Levene-Test with the R function `leveneTest` from the package `car` to test the equality of the variances.

(e) Test on the significance level $\alpha = 0.05$, if the null hypothesis of equal means of `Pmax` for the four batches is rejected.

Hint: You may call the function `summary` with the fitted model as argument.

(f) Carry out a pairwise comparison of the batches regarding `Pmax` using a Tukey-Test on significance level $\alpha = 0.1$. Create a plot of the computed confidence intervals.

Task 19

(a) Consider the data set `ToothGrowth` from the package `datasets` and transform the attribute `dose` to type `factor`.

(b) Create a boxplot for the value of `len`, separated on all factor combinations of `supp` and `dose`. What is your impression?

(c) Fit a linear model for `len`, where `supp` and `dose` (inclusive interaction) are explanatory variables.

(d) Analyze the model assumptions as in Task 14 (c). Additionally, test on level $\alpha = 0.05$, if there is evidence against the assumption of equal variance in each group of factor combination.

(e) If the model assumptions are sufficiently satisfied, test on level $\alpha = 0.05$, if there is any influence of the explanatory variables on the value of `len`.

(f) If there is any influence of explanatory variables, analyze it concretely: Test on overall $\alpha = 0.05$, if the interaction of `supp` and `dose` and possibly their main effect has an influence on the value of `len`.

Hint: Use a Bonferroni correction to ensure that all tests together satisfy the significance level α. This means that you can compare each p-value with an adapted significance level $\tilde{\alpha} = \frac{\alpha}{k}$, where $k \in \mathbb{N}$ is the number of tests.

(g) If some effects are not significantly different from zero, fit a new model with the removed terms to get the final model for interpretation.

Task 20

Let $\boldsymbol{y} = (y_1, \ldots, y_n)'$ be a realization of the random sample $\boldsymbol{Y} = (Y_1, \ldots, Y_n)'$. For $\boldsymbol{y}$, consider the linear model $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where $\boldsymbol{X}$ is a fixed $(n \times d)$-model (or design) matrix with $d < n$ and rank d, $\boldsymbol{\beta}$ is a $(d \times 1)$-vector of model parameters and $\boldsymbol{\varepsilon} \sim \mathcal{N}_n(\boldsymbol{0}, \sigma^2 I_n)$ with $\sigma^2 > 0$.

Implement your own linear model R-function. This function should use a model formula and a `data.frame` object as input and should deliver the following output: least squares estimates $\hat{\boldsymbol{\beta}}$ of model parameters $\boldsymbol{\beta}$, $\hat{\text{var}}(\hat{\boldsymbol{\beta}})$, the unbiased estimator $\hat{\sigma}^2$ of σ^2, and $R^2 = \frac{SSR}{SST}$.

Remark: Here we write $\|\boldsymbol{y} - y^*\|_2^2 = \|\hat{\boldsymbol{\mu}} - y^*\|_2^2 + \|\boldsymbol{y} - \hat{\boldsymbol{\mu}}\|_2^2$ as SST = SSR + SSE (total sum of squares = sum of squares due to the regression + sum of squared errors), where $y^* = \overline{y}$ for a model with intercept parameter and $y^* = 0$ for models without.

Hints:

(a) Use the function `model.matrix` to create the design matrix from the right hand side of the given formula.

(b) Let $\boldsymbol{X} = \boldsymbol{Q}\boldsymbol{R}$ be a $\boldsymbol{Q}\boldsymbol{R}$-decomposition of the matrix $\boldsymbol{X}$ into a upper triangular $(d \times d)$-matrix $\boldsymbol{R}$ and a matrix $\boldsymbol{Q}$ of the first d columns of an orthogonal $(n \times n)$-matrix. Use the fact that the least squares estimator is given by $\hat{\boldsymbol{\beta}} = \boldsymbol{R}^{-1}\boldsymbol{Q}'\boldsymbol{y}$ to implement a numerically more stable procedure than that using the classical representation $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$.

Test your function on the data set `Solar` and calculate 90%-confidence intervals for the differences $\alpha_{i+1} - \alpha_i$, $i = 1, \ldots, 3$, of the factor levels $1, \ldots, 4$ of `batch` treated as factor (cf. Task 15).

Task 21

Set the seed to 2020 and repeat the following procedure $n = 1000$ times for $N = 10, 20, 50$:

(a) Generate samples $x_{11}, \ldots, x_{1N}$ from a uniform distribution on $(0, 40)$.

(b) Generate samples $x_{21}, \ldots, x_{2N}$ from $\mathcal{N}(15, 10^2)$.

(c) Generate error values $\varepsilon_1, \ldots, \varepsilon_N$ from the distribution of the random variable ε, where

$$\frac{\varepsilon}{5} + 1 \sim \text{Exp}(1)$$

and Exp(1) denotes the exponential distribution with parameter 1.

(d) Generate a sample $y_1, \ldots, y_N$ by setting

$$\mu_i = \beta_1 + \beta_2 x_{1i} + \beta_3 x_{2i}, \qquad i = 1, \ldots, N,$$

where $\beta_1 = 35$, $\beta_2 = 0.5$, $\beta_3 = -0.1$ and $y_i = \mu_i + \varepsilon_i$.

(e) Estimate $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_3)$ using the least squares estimator $\hat{\boldsymbol{\beta}}$.

(f) Compute the norm-difference $\|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\|$ of the true parameter vector and its estimate.

Store the values of $\hat{\beta}_2$ and $\|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\|$ for all n generated data sets and all $N = 10, 20, 50$. Then illustrate some results using the following graphics.

(i) Create boxplots of the $\|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\|$ for the three values of N.

(ii) Create plots of the estimated densities of $\hat{\beta}_2$ for the three values of N and add the curve of the $\mathcal{N}(\hat{\mu}, \hat{\sigma}^2)$ density, where $\hat{\mu}, \hat{\sigma}^2$ are the mean value and the standard deviation of the sample of $\hat{\beta}_2$ values for the current N.

What do you observe?

```
######################
########TASK15########
######################

timeW.fit = lm(timeW ~ climb + distance, data = race_task15)

#b)

timeW.fit2 = lm(timeW ~ distance, data = race_task15)
#
# Analysis of Variance Table
#
# Model 1: timeW ~ distance
# Model 2: timeW ~ climb + distance
# Res.Df   RSS Df Sum of Sq     F    Pr(>F)
# 1     66 30686
# 2     65 12675  1     18011 92.36 4.223e-14 ***
#   ---
#   Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
# reject the null hypothesis that climb is not sigfinicant

distance = race_task15$distance
climb = race_task15$climb
ones_task15 = rep(1, nrow(race_task15))

mens_dat = cbind(ones_task15, climb, distance)

predicted_timeM = mens_dat %*% timeW.fit$coefficients
# which(race_task15["timeW"] == 490.05)
# [1] 41
# > predicted_timeM[41]
# [1] 456.1487
#

# #d)
# > cor(race_task15$timeM, race_task15$timeW)
# [1] 0.9958732

cor(Races$timeW,Races$timeM)
# we have strong positive correlation
# this indicates that in a race where the males need more time, the females will also need
more time

#e)
fit.timeMW = lm(timeM ~ -1 + timeW, data = race_task15)
```

```
######################
########TASK18########
######################
solar_task18 = read.table("R-Lab-Datasets/solar_task10.csv", header = TRUE,
sep = ",")

solar_task18$batch = as.factor(solar_task18$batch)

#b) boxplot for Pmax for each batch
ggplot(solar_task18, aes(x = batch, y =  Pmax, fill=(batch), group = batch)) +
geom_boxplot(outlier.colour="red", outlier.shape=8,outlier.size=4)
#or this
#solar_boxplot = boxplot(solar_task18$Pmax ~ solar_task18$batch)

#anova of the Pmax corresponding for each batch
solar_aov = aov(solar_task18$Pmax ~ solar_task18$batch)

#levene test
levene.test.solar = leveneTest(solar_task18$Pmax ~ solar_task18$batch)

#plot the 4
# fitted vs residuals
# fitted vs standard residuals
# normal QQ
# residual vs leverage
par(mfrow = c(2,2))

plot(solar_aov)

par(mfrow = c(1,1))

#shapiro test
shapiro.test(residuals(solar_aov))

#Tukey test and its plot
plot(TukeyHSD(solar_aov, conf.level=0.9))
```

```
########################
#########TASK19########
########################

#a) load toothgrowth data
tooth.data = ToothGrowth

#to factorize the dose
tooth.data$dose = as.factor(tooth.data$dose)

#b)
par(mfrow = c(1,1))
box_tooth_len_dose = boxplot(tooth.data$len ~ tooth.data$dose *
tooth.data$supp)

#c) fit a model for len based on supp and dose

len.fit = lm(len ~ dose * supp, data = tooth.data)

#d)
par(mfrow = c(2,2))
plot(len.fit)

par(mfrow = c(1,1))
aov.len = aov(len.fit)

#f) pairwise t test and fit another model by
# excluding less effective parameter
new.len.fit = lm(len ~ dose + supp, data = tooth.data)
pairwise.t.test(tooth.data$len, tooth.data$dose, p.adjust.method = "bonferroni")
```

```
########################
#########TASK21########
########################

set.seed(2020)

#a)
unif.sample = runif(50, 0, 40)
#b)
norm.sample = rnorm(50, 15, 10)
#c)
exp.sample = (rexp(50, 1)-1)*5
#d)
beta1 = 35
beta2 = 0.5
beta3 = -0.1

betas = c(beta1, beta2, beta3)

mu = beta1 + unif.sample*beta2 + norm.sample*beta3
Y = mu + exp.sample

#e)
pseudo_fit = lm(Y ~ unif.sample + norm.sample)

beta.hat = pseudo_fit$coefficients

#f)
diff = sum((beta.hat - betas)^2)
```

```
#######################
########TASK20########
#######################

my.lm = function(formula, data){

  Y = data[,as.character(formula[[2]])]#achieve the target column
  X = model.matrix(formula, data = data)#generate the design matrix

  parameter.name = colnames(data)
  model.with.intercep=any(colnames(X)=="(Intercept)")
  X=matrix(X,ncol=ncol(X))
  p.columns=ncol(X)
  n.observations=nrow(X)
  degrees.of.freedom=n.observations-p.columns

  qr.decomp.X=qr(X)
  beta.hat=qr.solve(qr.decomp.X,y)
  beta.hat=matrix(beta.hat)
  mu.hat=X%*%beta.hat
  residuals=y-mu.hat
  SSE=sum((residuals)**2)
  y.star=ifelse(model.with.intercep,mean(y),0)
  TSS=sum((y-y.star)**2)
  sigma.hat.2=SSE/degrees.of.freedom
  R.2=(TSS-SSE)/TSS
  beta.hat.cov=sigma.hat.2 * chol2inv(qr.decomp.X$qr)

  rownames(beta.hat)=param.names
  rownames(beta.hat.cov)=param.names
  colnames(beta.hat.cov)=param.names


erg=list(beta.hat=beta.hat,beta.hat.cov=beta.hat.cov,R.2=R.2,sigma.hat.2=sigm
hat.2)

}
```