Lucerne University of
Applied Sciences and Arts

HOCHSCHULE
LUZERN

Informatik

MODULENDPRÜFUNG

STATISTICS FOR DATA SCIENCE HS18

DAUER: 90 MINUTEN

Name, Vorname: _____

Stick-Nummer: _____

Unterschrift: .....

Aufgabe	**max. Punktezahl**	**erreichte Punktzahl**
Aufgabe 1	30	
Aufgabe 2	30	
Aufgabe 3	30	
Total	90	

Viel Erfolg!
M. Birbaumer

HINWEISE

Erlaubte Hilfsmittel

- `Python`-Referenzkarte, resp. `R`-Referenzkarte
- eine eigenhändig handgeschriebene Zusammenfassung im Umfang von 12 einseitig beschriebenen A4-Seiten
- Laptop mit `Python`, resp. mit Statistiksoftware `R`
- Taschenrechner
- (leeres) Papier und Schreibzeug

Daten

- Die Daten zur Aufgabe 2 befinden sich im Verzeichnis `Austausch` des USB-Sticks.

Prüfung

- Nennen Sie die Python-Datei `name_vorname.py`, resp. die R-Datei `name_vorname.R` in Ihren Namen um, und speichern Sie diese im Verzeichnis `Austausch` mit den von Ihnen benützten `Python`, resp. `R`-Befehlen.
- Schreiben Sie alle Lösungen zu den Aufgaben vollständig auf Papier.
- Die Prüfung ist in sauberer Schrift und übersichtlicher Darstellung zu schreiben. Die Lösungswege müssen immer klar erkennbar und kontrollierbar sein, beispielsweise unterstützt mit Formeln, Stichworten oder Skizzen.
- Die Bestnote kann bereits ab 80 Punkten erreicht werden.

Aufgabe 1: Erwartungswert und Schätzmethoden ..... 30 Punkte

Betrachten Sie die Zufallsvariable X mit der folgenden Dichtefunktion:

$$f(x;m) = \begin{cases} 0 & \text{falls } x < 0 \\ c \cdot x^m & \text{falls } 0 \leq x \leq 1 \\ 0 & \text{falls } x > 1 \end{cases}$$

wobei m den Verteilungsparameter darstellt und c ein Faktor ist, der von m abhängt.

(a) (5 Punkte) Bestimmen Sie den Faktor c, so dass $f(x;m)$ eine Wahrscheinlichkeitsdichtefunktion ist.

(b) (6 Punkte) Berechnen Sie den Erwartungswert $\mathrm{E}[X]$.

(c) (7 Punkte) Berechnen Sie die Varianz $\mathrm{Var}[X]$.

(d) (5 Punkte) Es wurden folgende Datenpunkte beobachtet: $x_1 = 0.3, x_2 = 0.5$ und $x_3 = 0.7$. Schätzen Sie den Parameter m mit Hilfe der Momentenmethode.

(e) (7 Punkte) Schreiben Sie die Likelihood-Funktion für die Beobachtungen $x_1 = 0.3, x_2 = 0.5$ und $x_3 = 0.7$. Beschreiben Sie in 2-3 Sätzen, wie Sie mit der Maximum-Likelihood-Methode den Parameter m schätzen.

Aufgabe 2: An die wahren Helden der WM18 - Lebensdauer von Schiedsrichtern . 30 Punkte

Als ein Schiedsrichter im Jahre 1990 bei den U.S. Major League Baseball Games zusammenbrach und verstarb, wurde spekuliert, dass der mit diesem Beruf verbundene Stress ein ernsthaftes Gesundheitsrisiko darstellt. Forscher haben darauf historische und damals aktuelle Daten über Schiedsrichter gesammelt, um deren Lebenserwartung zu untersuchen (Cohen et al., *Life expectancy of Major League Baseball Umpires*, The Physician and Sportsmedicine, 28(5) (2000):83-89).

Die in dieser Studie erhobenen Daten beinhalten eine Liste von 227 Schiedsrichtern, die entweder gestorben oder aber pensioniert und immer noch am Leben waren. Sie finden in der Datei **`Lebenserwartung_Schiedsrichter`** auf dem USB-Stick im Verzeichnis *Austausch* die erhobenen Daten: In der ersten Spalte ist die beobachtete Lebenszeit (Alter) in Jahren aufgeführt, in der zweiten Spalte befindet sich die Angabe, ob der betreffende Schiedsrichter tot (= 0) oder noch am Leben (= 1) ist. In der dritten und letzten Spalte ist die aufgrund von einer Lebensversicherung geschätzte erwartete Lebenszeit aufgeführt. Berücksichtigen Sie in Teilaufgaben (a) bis (d) bloss die Schiedsrichter, die bereits gestorben sind. *Hinweis:* Falls Sie die toten Schiedsrichter nicht von den lebendigen unterscheiden können, dann berücksichtigen Sie alle, ob tot oder lebendig (Abzug 3 Punkte).

(a) (10 Punkte) Führen Sie einen t-Test auf dem 1 %-Signifikanzniveau durch, um zu ermitteln, ob Schiedsrichter eine kleinere beobachtete Lebenszeit (Alter) haben als erwartet. Formulieren Sie die Null- und Alternativhypothese, und geben Sie den Verwerfungsbereich an. Wie lautet das Testergebnis?

(b) (5 Punkte) Erklären Sie die Teststatistik im t-Test in 4-5 Sätzen. Verwenden Sie in Ihrer Argumentation die folgenden Begriffe:

i) Teststatistik

ii) Zufallsvariable

iii) standardisierte Zufallsvariable

iv) Abhängigkeit der Teststatistik vom Stichprobenumfang n

v) Zentraler Grenzwertsatz

(c) (4 Punkte) Geben Sie ein 99%-Vertrauensintervall an, und erklären Sie, wie Sie das Testergebnis daraus bestimmen.

(d) (7 Punkte) Führen Sie einen Vorzeichen- oder Wilcoxon-Test durch. Erläutern Sie die Annahmen an die Teststatistik, und formulieren Sie die Null- und Alternativhypothese. Kommen Sie zum selben Testergebnis wie beim t-Test?

(e) (4 Punkte) Geben Sie die mittlere beobachtete Lebenszeit (Alter) der noch lebenden Schiedsrichter mit relativem Fehler an.

Aufgabe 3: Varianzanalyse ..... 30 Punkte

Eine trendige Weinbar führt ein Experiment durch, in welchem die Qualität von 3 Weinen bewertet werden. Dazu werden fünf Weinkenner eingeladen und gebeten, jeden Wein zu probieren und mit einer Bewertung zwischen 0 und 10 zu versehen. Die Reihenfolge der Degustation war randomisiert, und die Testleiter wussten nicht, welchen Wein sie tranken. Die folgende Tabelle zeigt die gesammelten Daten :

	Wein 1	Wein 2	Wein 3
Person 1	1	7	5
Person 2	0	4	0
Person 3	1	6	4
Person 4	1	5	2
Person 5	1	8	10

In **Python** kann die Tabelle folgendermassen eingelesen werden:

```
from pandas import DataFrame
from statsmodels.formula.api import ols
from patsy.contrasts import Sum
from statsmodels.stats.anova import anova_lm
import numpy as np
import warnings
warnings.filterwarnings("ignore")
df = DataFrame({"Person": np.repeat(["P1", "P2", "P3","P4","P5"],3),
"Wein": np.tile(["W1", "W2", "W3"],5),
"Y": np.array([1,7,5,0,4,0,1,6,4,1,5,2,1,8,10])
})
fit = ols("Y ~ C(Person,Sum)+C(Wein,Sum)",data=df).fit()
```

(a) (8 Punkte) Schreiben Sie das der **Python**-Ausgabe zugrundeliegende (allgemeine) Gruppenmit Modell auf. Welche Nebenbedingungen wurden benützt? Um welches Versuchsdesign handelt es sich hier? Welche Rolle haben die einzelnen Faktoren im Modell?

(b) (6 Punkte) Hat die Weinsorte einen Effekt auf die Bewertung? Wie lautet die Nullhypothese in Bezug auf die entsprechenden Parameterwerte? Geben Sie die Teststatistik, deren Bedeutung und Verteilung und den p-Wert der Realisierung der Teststatistik unter der Nullhypothese an.

(c) (4 Punkte) Wie gross ist die geschätzte Differenz der Bewertung zwischen Weinsorte 1 und Weinsorte 2?

(d) (3 Punkte) Muss der Effekt der Testperson im Gruppenmittel-Modell berücksichtigt werden? Begründen Sie Ihre Antwort.

(e) (6 Punkte) Ihre Kollegin möchte folgenden **Python**-Code anwenden: Die **Python**-Ausgabe sieht folgendermassen aus:

```
from pandas import DataFrame
from statsmodels.formula.api import ols
from patsy.contrasts import Sum
from statsmodels.stats.anova import anova_lm
import numpy as np
import warnings
warnings.filterwarnings("ignore")
df = DataFrame({"Person": np.repeat(["P1", "P2", "P3","P4","P5"],3),
"Wein": np.tile(["W1", "W2", "W3"],5),
"Y": np.array([1,7,5,0,4,0,1,6,4,1,5,2,1,8,10])
})
fit2 = ols("Y ~ C(Person,Sum) * C(Wein,Sum)", data=df).fit()
```

Welches Modell passt sie an die Daten an? Kann sie statistische Tests durchführen? Begründen Sie Ihre Antwort.

(f) (3 Punkte) Beurteilen Sie graphisch, ob es einen Interaktionseffekt gibt.

MEP Statistics for Data Science HS18

Musterlösungen

Lösung 1: ..... 30 Punkte

(a)

$$\begin{aligned}\int_0^1 f(x;m)\mathrm{d}x &= \int_0^1 c \cdot x^m \mathrm{d}x \\ &= \left[c \cdot \frac{x^{m+1}}{m+1}\right]_0^1 \\ &= c \cdot \frac{1-0}{m+1} \\ &\stackrel{!}{=} 1 \quad \textbf{3P}\end{aligned}$$

Somit folgt

$$c = m+1 \quad \textbf{2P}$$

(b) Für den Erwartungswert ergibt sich

$$\begin{aligned}\mathrm{E}[X] &= \int_0^1 x f(x;m)\mathrm{d}x \quad \textbf{2P} \\ &= \int_0^1 (m+1) \cdot x^{m+1}\mathrm{d}x \\ &= \left[(m+1) \cdot \frac{x^{m+2}}{m+2}\right]_0^1 \quad \textbf{2P} \\ &= \frac{m+1}{m+2} \quad \textbf{2P}\end{aligned}$$

(c) Für die Varianz berechnen wir als erstes

$$\begin{aligned}\mathrm{E}[X^2] &= \int_0^1 x^2 f(x;m)\mathrm{d}x \quad \textbf{2P} \\ &= \int_0^1 (m+1) \cdot x^{m+2}\mathrm{d}x \\ &= \left[(m+1) \cdot \frac{x^{m+3}}{m+3}\right]_0^1 \\ &= \frac{m+1}{m+3} \quad \textbf{2P}\end{aligned}$$

Somit ergibt sich für die Varianz

$$\begin{aligned}\text{Var}[X] &= \text{E}[X^2] - \text{E}[X]^2 \quad \textbf{2P} \\ &= \frac{m+1}{m+3} - \left(\frac{m+1}{m+2}\right)^2 \quad \textbf{1P}\end{aligned}$$

(d) Mit der Momentenmethode folgt

$$\text{E}[X] = \frac{m+1}{m+2} \stackrel{!}{=} \overline{x} = 0.5 \quad \textbf{3P}$$

Somit ist $\widehat{m} = 0$ **2P**

(e) Die Likelihood-Funktion lautet

$$l(m) = \Big(m+1\Big)^3 (0.3)^m \cdot (0.5)^m \cdot (0.5)^m \quad \textbf{3P}$$

Als nächstes wird man den Logarithmus dieser Funktion zur Bestimmung der Log-likelihood-Funktion schreiben. Der Logarithmus eines Produkts ergibt eine Summe von Ausdrücken, die Funktionen von m sind. Die log-Likelihood-Funktion leitet man nach m ab und setzt sie gleich null. Durch Auflösen dieser Gleichung erhält man den Log-Likelihood-Schätzer. **4P**

Lösung 2: ..... 30 Punkte

(a) Wir bilden die Differenz D_i, indem wir von der tatsächlichen Lebenszeit des $i-$ten Schiedsrichters die erwartetet Lebenszeit subtrahieren. Die Nullhypothese lautet, dass der Beruf des Schiedsrichters zu keiner Veränderung der Lebenszeit führt, also dass der Erwartungswert der Differenz zwischen tatsächlicher Lebenszeit und erwarteter Lebenszeit null beträgt :

$$H_0 : \mu_{D,0} = 0 \quad \textbf{1P}$$

Die Alternativhypothese entspricht unserer Vermutung, dass der Beruf des Schiedsrichters zu einer Verkürzung der Lebenszeit führt:

$$H_A : \mu_D < \mu_{D,0} \quad \textbf{1P}$$

Der Mittelwert der Differenz $\overline{D}_n$ (n ist in dieser Aufgabe 195) ist eine Zufallsvariable und folgt der Verteilung $\mathcal{N}(\mu_{\overline{D}_n} = \mu_D, \sigma^2_{\overline{D}_n} = \frac{\sigma^2_D}{n})$. Da die Standardabweichung σ_D nicht bekannt ist, und wir diese schätzen müssen, betrachten wir die folgende Teststatistik

$$T = \frac{\overline{D}_n - \mu_D}{\widehat{\sigma}_D / \sqrt{n}} \quad \textbf{2P}$$

Unter der Nullhypothese folgt die Teststatistik T einer t-Verteilung mit $n-1$ Freiheitsgraden. Der Verwerfungsbereich auf dem Signifikanzniveau $\alpha = 0.01$ ist gegeben durch

$$\{-\infty, t_{n-1;\alpha}\} = \{-\infty, t_{195-1;0.01}\} \quad \textbf{3P}$$

wobei die 1%-Quantile der t-Verteilung gegeben ist durch

```
from scipy.stats import norm, t
print(t.ppf(q=0.01, df=194))

## -2.34572280577
```

Der **R**-Code lautet:

```
qt(0.01, df = 194)

## [1] -2.345723
```

Also ist der Verwerfungsbereich

$$\{-\infty, -2.345723\} \quad \textbf{1P}$$

Die Realisierung der Testatistik T ergibt $t = -0.9870368$. **1P**

```
import pandas as pd
import numpy as np
schiedsrichter = pd.read_csv(r"./DatenFS18/Schiedsrichter_Lebenserwartung.txt",sep=" ", header=None)
schiedsrichter.columns=['Alter', 'lebendig', 'Lebenserwartung']
schiedsrichter_tot = schiedsrichter[schiedsrichter['lebendig']==0]
differenz_lebenszeit = schiedsrichter_tot['Alter'] - schiedsrichter_tot['Lebenserwartung']
t_value = differenz_lebenszeit.mean()/(differenz_lebenszeit.std()/np.sqrt(differenz_lebenszeit.size))
print(t_value)

## -0.987036775437
```

Der **R**-Code lautet:

```
schiedsrichter <- read.table(file = "./DatenFS18/Schiedsrichter_Lebenserwartung.txt")
schiedsrichter.tot <- schiedsrichter[schiedsrichter[,
    2] == 0, ]
differenz.lebenszeit <- schiedsrichter.tot[, 1] - schiedsrichter.tot[,
    3]
mean(differenz.lebenszeit)/(sd(differenz.lebenszeit)/sqrt(length(differenz.lebenszeit)))

## [1] -0.9870368
```

Da die Realisierung der Teststatistik nicht im Verwerfungsbereich liegt, belassen wir die Nullhypothese. Es ist also statistisch auf dem Signifikanzniveau nicht nachgewiesen, dass Schiedsrichter kürzer als ihre erwartete Lebenszeit leben (**1P**).

Der Testentscheid liesse sich auch mit Hilfe des p-Wertes fällen:

```
# alternativ mit:
# t_value = st.ttest_1samp(a=differenz_lebenszeit, popmean=0).statistic
# print(t.cdf(t_value, df=differenz_lebenszeit.size-1))
```

```
## 0.16242701837
```

Da in Python der zweiseitige P-Wert berechnet wird, muss die Ausgabe des zweiseitigen P-Wertes noch durch zwei dividiert werden.

Der dazugehörige **R**-Code lautet:

```
schiedsrichter <- read.table(file = "./DatenFS18/Schiedsrichter_Lebenserwartung.txt")
schiedsrichter.tot <- schiedsrichter[schiedsrichter[,
    2] == 0, ]
differenz.lebenszeit <- schiedsrichter.tot[, 1] - schiedsrichter.tot[,
    3]
t.test(differenz.lebenszeit, mu = 0, alternative = "less")

##
##  One Sample t-test
##
## data:  differenz.lebenszeit
## t = -0.98704, df = 194, p-value = 0.1624
## alternative hypothesis: true mean is less than 0
## 95 percent confidence interval:
```

```
##       -Inf 0.3631666
## sample estimates:
##  mean of x
## -0.5384615
```

Da der p-Wert 0.1624 beträgt, lässt sich die Nullhypothese auf dem Signifikanzniveau von 1% nicht verwerfen.

(b) Der Mittelwert der Differenz $\overline{D}_n = \overline{D}_{195}$ ist eine Zufallsvariable und folgt der Verteilung $\mathcal{N}(\mu_{\overline{D}_n} = \mu_D, \sigma^2_{\overline{D}_n} = \frac{\sigma^2_D}{n})$(**2P**). Dass die Summe, respektive der Mittelwert, gemäss einer Normalverteilung verteilt ist, folgt aus dem Zentralen Grenzwertsatz (**1P**). Da die Standardabweichung σ_D nicht bekannt ist, und wir diese schätzen müssen, betrachten wir die folgende Teststatistik

$$T = \frac{\overline{D}_n - \mu_D}{\widehat{\sigma}_D / \sqrt{n}} \quad \textbf{1P}$$

Je grösser der Stichprobenumfang n ist, desto kleiner sind die Werte von T, d.h. desto kleiner ist die Streuung der Teststatistik T. Unter der Nullhypothese folgt die Teststatistik T einer t-Verteilung mit $n-1$ Freiheitsgraden (**1P**). Wäre die Standardabweichung bekannt und nicht geschätzt (was in der Realität nicht der Fall ist), so wäre die Teststatistik standardnormalverteilt.

(c) Wir entnehmen der **Python**-Ausgabe

```
## (-1.8181317009452997, 0.74120862402222298)
```

dass das (einseitige) 99%-Vertrauensintervall gegeben ist durch

$$\{-\infty, 0.7412086\}$$

Der dazugehörige **R**-Code lautet:

```
schiedsrichter <- read.table(file = "./DatenFS18/Schiedsrichter_Lebenserwartung.txt")
schiedsrichter.tot <- schiedsrichter[schiedsrichter[,
    2] == 0, ]
differenz.lebenszeit <- schiedsrichter.tot[, 1] - schiedsrichter.tot[,
    3]
t.test(differenz.lebenszeit, mu = 0, alternative = "less",
    conf.level = 0.99)

##
##  One Sample t-test
##
## data:  differenz.lebenszeit
## t = -0.98704, df = 194, p-value = 0.1624
## alternative hypothesis: true mean is less than 0
## 99 percent confidence interval:
##       -Inf 0.7412086
## sample estimates:
##  mean of x
## -0.5384615
```

Da das 99%-Vertrauensintervall die Null enthält, kann nicht ausgeschlossen werden, dass die wahre Differenz zwischen tatsächlicher Lebenlänge und erwarteter Lebenslänge null ist.

(d) Die Teststatistik V sei die Anzahl Schiedsrichter, die vor erwarteter Lebenszeit gestorben sind.

```
import pandas as pd
import numpy as np
import scipy.stats as st
from scipy.stats import t
schiedsrichter = pd.read_csv("./DatenFS18/Schiedsrichter_Lebenserwartung.txt",sep=" ", header=None)
```

```
schiedsrichter.columns=['Alter', 'lebendig', 'Lebenserwartung']
schiedsrichter_tot = schiedsrichter[schiedsrichter['lebendig']==0]
differenz_lebenszeit = schiedsrichter_tot['Alter']-schiedsrichter_tot['Lebenserwartung']
V = np.sum(differenz_lebenszeit < 0)
print(V)

## 82
```

Der R-Code lautet:

```
schiedsrichter <- read.table(file = "./DatenFS18/Schiedsrichter_Lebenserwartung.txt")
schiedsrichter.tot <- schiedsrichter[schiedsrichter[,
    2] == 0, ]
differenz.lebenszeit <- schiedsrichter.tot[, 1] - schiedsrichter.tot[,
    3]
sum(differenz.lebenszeit < 0)

## [1] 82
```

Diese Teststatistik folgt einer Binomialverteilung mit Parameterwerten $n = 195$ und Wahrscheinlichkeit für frühzeitigen Tod π (**2P**). Nun stellt sich die Frage, wie wahrscheinlich das Ereignis ist, dass 82 oder mehr Schiedsrichter von den 195 betrachteten Schiedsrichtern vor ihrer erwarteten Lebenszeit sterben, wenn die Nullhypothese lautet: Ein Schiedsrichter stirbt mit gleicher Wahrscheinlichkeit vor der erwarteten Lebenszeit wie nach der erwarteten Lebenszeit (**1P**). Somit ist die Wahrscheinlichkeit π unter der Nullhypothese gegeben durch 0.5 (**1P**). Die Alternativhypothese lautet: Die Wahrscheinlichkeit, dass ein Schiedsrichter vor der erwarteten Lebenszeit stirbt, ist grösser als 0.5 (**1P**). Die Wahrscheinlichkeit, dass unter der Nullhypothese 82 oder mehr Schiedsrichter sterben, ermitteln wir im Rahmen eines Binomialtests, und zwar mit Hilfe des p-Wertes

```
import scipy.stats as st
p_wert = 1 - st.binom.cdf(k=82, n=195, p=0.5)
print(p_wert)

## 0.984289042755
```

Der R-Code lautet:

```
binom.test(x = 82, n = 195, p = 0.5, alternative = "greater")

##
##  Exact binomial test
##
## data:  82 and 195
## number of successes = 82, number of trials =
## 195, p-value = 0.9892
## alternative hypothesis: true probability of success is greater than 0.5
## 95 percent confidence interval:
##  0.3610158 1.0000000
## sample estimates:
## probability of success
##              0.4205128
```

Der p-Wert ergibt 0.9892 **1P**. Somit behalten wir die Nullhypothese bei (**1P**). Es gibt also auch aufgrund des Vorzeichentests keine statistische Evidenz, dass Schiedsrichter signifikant früher sterben.

Wir gelangen zum selben Testergebnis mit dem Wilcoxon-Test:

```
import scipy.stats as st
from scipy.stats import norm, t, binom
import numpy as np
from math import sqrt
import pandas as pd
schiedsrichter = pd.read_csv("./DatenFS18/Schiedsrichter_Lebenserwartung.txt",sep=" ", header=None)
schiedsrichter.columns=['Alter', 'lebendig', 'Lebenserwartung']
schiedsrichter_tot = schiedsrichter[schiedsrichter['lebendig']==0]
differenz_lebenszeit = schiedsrichter_tot['Alter']-schiedsrichter_tot['Lebenserwartung']
p_wert = st.wilcoxon(x=np.array(differenz_lebenszeit), zero_method='wilcox', correction=True).pvalue
```

```
print(p_wert/2)

## 0.329816071419
```

Der entsprechende R-Code lautet:

```
schiedsrichter <- read.table(file = "./DatenFS18/Schiedsrichter_Lebenserwartung.txt")
schiedsrichter.tot <- schiedsrichter[schiedsrichter[,
    2] == 0, ]
differenz.lebenszeit <- schiedsrichter.tot[, 1] - schiedsrichter.tot[,
    3]
wilcox.test(differenz.lebenszeit, alternative = "less")

##
##  Wilcoxon signed rank test with continuity
##  correction
##
## data:  differenz.lebenszeit
## V = 8640, p-value = 0.6707
## alternative hypothesis: true location is less than 0
```

(e) Der relative Fehler ist gegeben durch

```
import pandas as pd
import numpy as np
import scipy.stats as st
from scipy.stats import t
schiedsrichter = pd.read_csv("./DatenFS18/Schiedsrichter_Lebenserwartung.txt",sep=" ", header=None)
schiedsrichter.columns=['Alter', 'lebendig', 'Lebenserwartung']
schiedsrichter_lebend = schiedsrichter[schiedsrichter['lebendig']==1]
alter_schiedsrichter_lebend = schiedsrichter_lebend['Alter']
schiedsrichter_lebend_mittel = alter_schiedsrichter_lebend.mean()
abs_fehler =  alter_schiedsrichter_lebend.std()/np.sqrt(alter_schiedsrichter_lebend.size)
rel_fehler = abs_fehler/schiedsrichter_lebend_mittel
print(rel_fehler)

## 0.0168578165902
```

Dazu noch den R-Code:

```
schiedsrichter <- read.table(file = "./DatenFS18/Schiedsrichter_Lebenserwartung.txt")
alter.schiedsrichter.lebend <- schiedsrichter[schiedsrichter[,
    2] == 1, 1]
mittel.schiedsrichter.lebend <- mean(alter.schiedsrichter.lebend)
mittel.schiedsrichter.lebend

## [1] 59.125

abs.fehler <- sd(alter.schiedsrichter.lebend)/sqrt(length(alter.schiedsrichter.lebend))
abs.fehler

## [1] 0.9967184

rel.fehler <- abs.fehler/mittel.schiedsrichter.lebend
rel.fehler

## [1] 0.01685782
```

Die mittlere Lebenszeit der noch lebenden Schiedsrichter beträgt

$$59\text{Jahre} \pm 1.7\% \quad \textbf{4P}$$

Lösung 3: ..... 30 Punkte

(a) Das Gruppenmittelmodell lautet:

$$Y_{ijk} = \alpha_i + \beta_j + \varepsilon_{ijk} \quad \textbf{2P}$$

wobei α_i die Behandlungseffekte der i-ten Weinsorte und β_j die Effekte der j-ten Person bezeichnen. β_j hat hier die Rolle einer Blockvariable: die Personen interessieren eigentlich nicht, deren Effekt soll aber berücksichtigt werden (**2P**). Die Nebenbedingungen sind gegeben durch

$$\sum_i \alpha_i = \sum_j \beta_j = 0 \quad \textbf{2P}$$

Der Versuchsplan ist ein vollständig randomisiertes Block-Design (RCBD) (**2P**) .

(b)
```
from pandas import DataFrame
from statsmodels.formula.api import ols
from patsy.contrasts import Sum
from statsmodels.stats.anova import anova_lm
import numpy as np
import warnings
warnings.filterwarnings("ignore")
df = DataFrame({"Person": np.repeat(["P1", "P2", "P3","P4","P5"],3),
"Wein": np.tile(["W1", "W2", "W3"],5),
"Y": np.array([1,7,5,0,4,0,1,6,4,1,5,2,1,8,10])
})
fit = ols("Y ~ C(Person,Sum)+C(Wein,Sum)",data=df).fit()
anova_lm(fit)
print(anova_lm(fit))

##                   df     sum_sq    mean_sq          F    PR(>F)
## C(Person, Sum)   4.0  42.000000  10.500000   3.281250  0.071681
## C(Wein, Sum)     2.0  69.733333  34.866667  10.895833  0.005200
## Residual         8.0  25.600000   3.200000        NaN       NaN
```

Der entsprechende **R**-Code lautet:

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## wine         2   69.7    34.9   10.90 0.0052 **
## person       4   42.0    10.5    3.28 0.0717 .
## Residuals    8   25.6     3.2
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Ja, die Weinsorte hat einen Effekt auf die Bewertung (**1P**). Die Nullhypothese lautet : $\alpha_1 = \alpha_2 = \alpha_3 = 0$ (**1P**). Die Teststatistik F ist gegeben durch

$$F = \frac{\text{MS}_G}{\text{MS}_E} \quad \textbf{1P}$$

wobei sich MS_G auf die Streuung der Gruppenmittelwerte bezieht und MS_E auf die Streuung innerhalb der Gruppen (**1P**). Unter der Nullhypothese hätte die Teststatistik F einen Wert nahe 1. In diesem Experiment hat die Teststatistik den Wert 10.9 und somit einen p-Wert von 0.0052. Die Nullyhpothese wird auf dem 5%-Niveau verworfen (**2P**).

(c) Der **Python**-Code lautet:

```
from pandas import DataFrame
from statsmodels.formula.api import ols
from patsy.contrasts import Sum
from statsmodels.stats.anova import anova_lm
import numpy as np
import warnings
warnings.filterwarnings("ignore")
df = DataFrame({"Person": np.repeat(["P1", "P2", "P3","P4","P5"],3),
"Wein": np.tile(["W1", "W2", "W3"],5),
"Y": np.array([1,7,5,0,4,0,1,6,4,1,5,2,1,8,10])
})
fit = ols("Y ~ C(Person,Sum)+C(Wein,Sum)",data=df).fit()
print(fit.summary())

##                             OLS Regression Results
## ==============================================================================
```

```
## Dep. Variable:                      Y   R-squared:                       0.814
## Model:                            OLS   Adj. R-squared:                  0.674
## Method:                 Least Squares   F-statistic:                     5.819
## Date:                Fri, 25 Jan 2019   Prob (F-statistic):             0.0131
## Time:                        11:15:19   Log-Likelihood:                -25.293
## No. Observations:                  15   AIC:                             64.59
## Df Residuals:                       8   BIC:                             69.54
## Df Model:                           6
## Covariance Type:            nonrobust
## ===============================================================================================
##                           coef    std err          t      P>|t|      [95.0% Conf. Int.]
## -----------------------------------------------------------------------------------------------
## Intercept               3.6667      0.462      7.939      0.000         2.602     4.732
## C(Person, Sum)[S.P1]    0.6667      0.924      0.722      0.491        -1.464     2.797
## C(Person, Sum)[S.P2]   -2.3333      0.924     -2.526      0.035        -4.464    -0.203
## C(Person, Sum)[S.P3] 2.026e-15      0.924   2.19e-15      1.000        -2.130     2.130
## C(Person, Sum)[S.P4]   -1.0000      0.924     -1.083      0.311        -3.130     1.130
## C(Wein, Sum)[S.W1]     -2.8667      0.653     -4.389      0.002        -4.373    -1.360
## C(Wein, Sum)[S.W2]      2.3333      0.653      3.572      0.007         0.827     3.840
## ==============================================================================
## Omnibus:                        2.108   Durbin-Watson:                   1.461
## Prob(Omnibus):                  0.349   Jarque-Bera (JB):                0.546
## Skew:                           0.375   Prob(JB):                        0.761
## Kurtosis:                       3.559   Cond. No.                         2.24
## ==============================================================================
##
## Warnings:
## [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## wine         2   69.7    34.9   10.90 0.0052 **
## person       4   42.0    10.5    3.28 0.0717 .
## Residuals    8   25.6     3.2
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Full coefficients are
##
## (Intercept):         3.67
## wine:                   1         2         3
##                    -2.867     2.333     0.533
## person:                 1         2         3
##                  6.67e-01 -2.33e+00  9.99e-16
##
## (Intercept):
## wine:
##
## person:                 4         5
##                 -1.00e+00  2.67e+00
```

Der Unterschied zwischen Weinsorte 1 und Weinsorte 2 beträgt 5.2

(d) Die Blockvariable Person β_j könnte weggelassen werden, da der p-Wert 0.0717 beträgt und somit auf dem 5% Signifikanzniveau nicht signifikant ist (**3P**).

(e) Sie berücksichtigt in ihrem Modell Interaktion (**2P**). Um in einem Modell mit Wechselwirkungseffekt die Standardabweichung der Residuen zu schätzen, bräuchte sie mindestens 2 Replikate pro Run. Dies ist hier nicht der Fall. Also kann sie in diesem Fall keine statistischen Tests durchführen (**4P**).

```
from pandas import DataFrame
from statsmodels.formula.api import ols
from patsy.contrasts import Sum
from statsmodels.stats.anova import anova_lm
import numpy as np
import warnings
warnings.filterwarnings("ignore")
df = DataFrame({"Person": np.repeat(["P1", "P2", "P3","P4","P5"],3),
"Wein": np.tile(["W1", "W2", "W3"],5),
"Y": np.array([1,7,5,0,4,0,1,6,4,1,5,2,1,8,10])
})
fit2 = ols("Y ~ C(Person,Sum)*C(Wein,Sum)",data=df).fit()
```

```
print(fit2.summary())
```

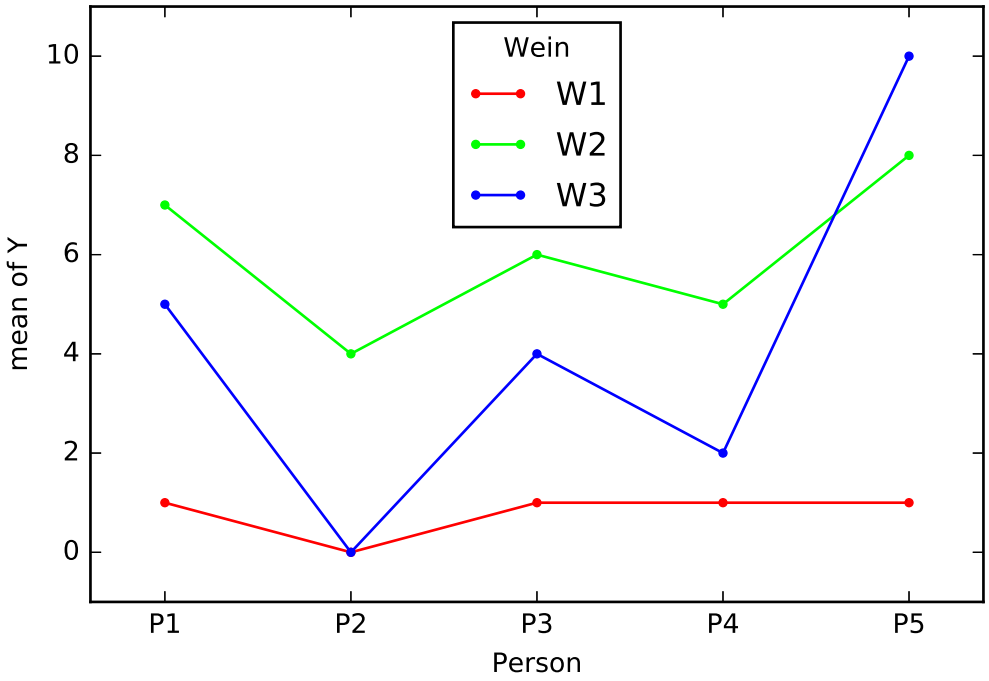
```
##                             OLS Regression Results
## ==============================================================================
## Dep. Variable:                      Y   R-squared:                       1.000
## Model:                            OLS   Adj. R-squared:                    nan
## Method:                 Least Squares   F-statistic:                     0.000
## Date:                Fri, 25 Jan 2019   Prob (F-statistic):               nan
## Time:                        11:15:19   Log-Likelihood:                484.16
## No. Observations:                  15   AIC:                            -938.3
## Df Residuals:                       0   BIC:                            -927.7
## Df Model:                          14
## Covariance Type:            nonrobust
## ===================================================================================================================
##                                                   coef    std err          t      P>|t|      [95.0% Conf. Int.]
## -------------------------------------------------------------------------------------------------------------------
## Intercept                                      3.6667        inf          0        nan           nan       nan
## C(Person, Sum)[S.P1]                           0.6667        inf          0        nan           nan       nan
## C(Person, Sum)[S.P2]                          -2.3333        inf         -0        nan           nan       nan
## C(Person, Sum)[S.P3]                       -5.274e-16        inf         -0        nan           nan       nan
## C(Person, Sum)[S.P4]                          -1.0000        inf         -0        nan           nan       nan
## C(Wein, Sum)[S.W1]                            -2.8667        inf         -0        nan           nan       nan
## C(Wein, Sum)[S.W2]                             2.3333        inf          0        nan           nan       nan
## C(Person, Sum)[S.P1]:C(Wein, Sum)[S.W1]       -0.4667        inf         -0        nan           nan       nan
## C(Person, Sum)[S.P2]:C(Wein, Sum)[S.W1]        1.5333        inf          0        nan           nan       nan
## C(Person, Sum)[S.P3]:C(Wein, Sum)[S.W1]        0.2000        inf          0        nan           nan       nan
## C(Person, Sum)[S.P4]:C(Wein, Sum)[S.W1]        1.2000        inf          0        nan           nan       nan
## C(Person, Sum)[S.P1]:C(Wein, Sum)[S.W2]        0.3333        inf          0        nan           nan       nan
## C(Person, Sum)[S.P2]:C(Wein, Sum)[S.W2]        0.3333        inf          0        nan           nan       nan
## C(Person, Sum)[S.P3]:C(Wein, Sum)[S.W2]     2.914e-15        inf          0        nan           nan       nan
## C(Person, Sum)[S.P4]:C(Wein, Sum)[S.W2]    -4.441e-16        inf         -0        nan           nan       nan
## ==============================================================================
## Omnibus:                        1.253   Durbin-Watson:                   2.468
## Prob(Omnibus):                  0.534   Jarque-Bera (JB):                0.902
## Skew:                           0.305   Prob(JB):                        0.637
## Kurtosis:                       1.965   Cond. No.                         3.87
## ==============================================================================
##
## Warnings:
## [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Der entsprechende **R**-Code lautet:

```
wine_tasting <- data.frame(y = c(1, 0, 1, 1, 1, 7,
    4, 6, 5, 8, 5, 0, 4, 2, 10), wine = factor(rep(c(1,
    2, 3), each = 5)), person = factor(rep(c(1, 2,
    3, 4, 5), 3)))
options(contrasts = c("contr.sum", "contr.sum"))
fit2 <- aov(y ~ wine * person, data = wine_tasting)
summary(fit2)
```

(f) Wir erstellen in **Python** wie folgt einen Interaktionsplot:

```
df = DataFrame({"Person": np.repeat(["P1", "P2", "P3","P4","P5"],3),
"Wein": np.tile(["W1", "W2", "W3"],5),
"Y": np.array([1,7,5,0,4,0,1,6,4,1,5,2,1,8,10])
})
interaction_plot(x=df["Person"], trace=df["Wein"],
response=df["Y"])
plt.show()
```



Aufgrund des Interaktionsplot würde man auf eine Interaktion zwischen Wein und Testperson schliessen. Ausschlaggebend ist Weinsorte 3 (**3 Punkte**).