Hausarbeit

Organisatorisches

- Diese Hausarbeit besteht aus 4 Aufgaben.
- Durch die Abgabe dieser Hausarbeit können maximal 85 % der erreichbaren Gesamtpunktzahl für den Kurs „Programmieren mit statistischer Software“ im Sommersemester 2016 erreicht werden.
- Die erreichte Gesamtpunkzahl für den Kurs „Programmieren mit statistischer Software“ setzt sich aus der erreichten Punktzahl durch abgegebene Übungsaufgaben (maximal 15 %) und der erreichten Punktzahl durch die Abgabe dieser Hausarbeit (maximal 85 %) zusammen.
- Eine verbindliche Anmeldung zur Veranstaltung (d.h. Scheinerwerb) erfolgt mit Abgabe dieser Hausarbeit.
- **Letztmöglicher Abgabetermin: Freitag, 16. September 2016 (23:55 Uhr).**

Hinweise zur Bearbeitung

- Bitte geben Sie die Hausarbeit über Moodle ab. Eine Abgabe ist nur möglich, falls Sie sich für den Kurs eingeschrieben haben. Ihre erreichte Note können Sie nach der Korrektur ebenfalls über Moodle abrufen.
- Soweit nicht anders vorgegeben, verwenden Sie für Ihre Lösung ausschließlich die Vorlagendatei „ProgStat16_Nachname_Vorname.R“. Benennen Sie die Datei entsprechend mit Ihrem Nach- und Vornamen.
- Tragen Sie in die R-Syntax-Datei „ProgStat16_Nachname_Vorname.R“ an ausgewiesener Stelle alle prüfungsrelevanten Informationen (vollständiger Name, Matrikelnummer, Studiengang, Prüfungsordnung) ein.
- Bitte geben Sie nur Ihre R-Syntax-Dateien ab.
- Machen Sie jeweils deutlich, welcher Programm-Code zu welcher (Teil-) Aufgabe gehört.
- Achten Sie darauf, dass Ihr Programm-Code grundsätzlich nachvollziehbar ist.
- Alle Teilschritte, die aufgrund eines fehlerhaften Programm-Codes eine nicht beabsichtigte Fehlermeldung erzeugen, werden mit 0 Punkten bewertet.
- **Geben Sie eine sinnvoll und ordentlich kommentierte Syntax ab.**

Aufgabe 1 (Datenanalyse und Grafiken)

Laden Sie sich von der GESIS-Homepage[1] den ALLBUS-Datensatz ZA5240_v2-1-0.sav und die zugehörige Variablenliste herunter. Dazu müssen Sie sich *vorher* bei GESIS registrieren. Bitte geben Sie als Verwendungszweck für den Download der Daten „Ich nutze die Daten im Rahmen des Studiums“ an. Die für die nachfolgenden Aufgaben relevanten Variablen sind in folgender Tabelle kurz beschrieben.

Variable	Beschreibung
V14	FREIZEIT: DAS INTERNET NUTZEN
V17	FREIZEIT: EINFACH NICHTS TUN, FAULENZEN
V19	FREIZEIT: YOGA, MEDITATION, AUTOG. TRAINING
V27	FREIZEIT: AKTIVE SPORTLICHE BETAETIGUNG
V225	GESUNDHEITSZUSTAND BEFR.
V275	KOERPERGROESSE IN CM, BEFRAGTE<R>
V277	GEWICHT IN KG, BEFRAGTE<R>

a) Lesen Sie den Datensatz mit folgendem Code ein. (Hinweis: Ignorieren Sie die dabei entstehenden Warnmeldungen.)

```
library(foreign)
allbus <- read.spss("ZA5240_v2-1-0.sav", to.data.frame=TRUE, use.
    value.labels=TRUE)
```

b) Erstellen Sie daraus einen neuen Teildatensatz `teil.allbus`, der nur die Variablen aus obiger Tabelle enthält. Weisen Sie den Variablen des Datensatzes `teil.allbus` aussagekräftige Namen zu. Kodieren Sie die Variablen, falls nötig, in geeigneter Weise um.

Verwenden Sie für alle folgenden Teilaufgaben den Datensatz `teil.allbus`.

c) Verschaffen Sie sich, unter Verwendung geeigneter Funktionen, einen Überblick über den Datensatz.

d) Wie viele Ausprägungen hat die Variable `V19`? Welche Ausprägungen hat die Variable `V225`?

e) Geben Sie die Anzahl der fehlenden Werte pro Variable in Form einer Tabelle aus.

f) Erstellen Sie eine neue Variable `BMI`, die den BMI der Befragten beinhaltet.
Hinweis: $BMI = \frac{Gewicht\ in\ kg}{(Körpergröße\ in\ m)^2}$

g) Entfernen Sie aus dem Datensatz alle Beobachtungen mit einem fehlenden Wert in der Variable `BMI`.

h) Erstellen Sie mithilfe des Pakets `ggplot2` eine Grafik (inkl. aussagekräftiger Beschriftung/Legende), die den `BMI` in Form von Boxplots getrennt nach der sportlichen Aktivität (`V27`) darstellt. Die Ausprägungen der Variable `V27` sollen dabei zusätzlich in verschiedenen Farben hervorgehoben werden. Die Breite der Box soll jeweils proportional zur (Wurzel der) Anzahl der Beobachtungen in den Gruppen der Variable `V27` sein.

i) Berechnen Sie das mittlere Gewicht der Befragten, die einen sehr guten Gesundheitszustand haben und sich täglich aktiv sportlich betätigen.

j) Erzeugen Sie eine Kreuztabelle mit der Verteilung (absolute Häufigkeiten) der Variablen `V14`, `V17`, `V19` und `V27` (in den Spalten). Verwenden Sie dazu eine geeignete Funktion der `apply`-Familie.

k) Erstellen Sie eine neue R-Datei „uebersicht_nachname_vorname.R“ und speichern Sie diese im Ordner, in dem Ihre Lösungsdatei „ProgStat16_Nachname_Vorname.R“ liegt.

[1] `https://dbk.gesis.org/DBKSearch/SDESC2.asp?no=5240&tab=3&db=D&dab=0`

l) Schreiben Sie in der Datei „uebersicht_nachname_vorname.R“ eine Funktion `uebersicht`, die als Input einen Datensatz verlangt und folgenden Grundstruktur aufweist:

```
uebersicht <- function(daten){
...
...
...
}
```

Die Funktion soll folgende Informationen ausgeben:

- Die Objektklasse des Datensatzes
- Die Anzahl der Beobachtungen des Datensatzes
- Die Anzahl der Variablen des Datensatzes
- Den Anteil der fehlenden Werte des Datensatzes.

Der Output der Funktion in der Konsole soll (in etwa) wie folgt aussehen:

```
Das Objekt gehört der Klasse data.frame an und beinhaltet 312 Beobachtungen
    von 4 Variablen. 10.25 Prozent der Einträge sind fehlende Werte.
```

Des Weiteren soll die Funktion für jede Variable des Datensatzes eine Grafik auf dem Bildschirm ausgeben. Für jede numerische Variable soll der Output ein Boxplot und für jede Faktor-Variable ein Balkendiagramm sein. Als Überschrift der Grafiken soll der jeweilige Variablenname verwendet werden.

Definieren Sie dafür eine geeignete plot-Methode für die Objektklasse `numeric`. Machen Sie sich mit dem Grafikparameter `ask` vertraut und verwenden Sie ihn an geeigneter Stelle in Ihrer Funktion.

m) Rufen Sie, mithilfe geeigneter Syntax, die von Ihnen in „uebersicht_nachname_vorname.R“ definierte Funktion aus l) auf und wenden Sie die Funktion auf den Datensatz `teil.allbus` an.
Hinweis: Denken Sie daran, auch die R-Syntax-Datei „uebersicht_nachname_vorname.R“ über Moodle abzugeben.

Aufgabe 2 (Text und S3)

Gegeben ist ein beliebiger Text, wie zum Beispiel:

```
‘‘Ich wusste wohl, mein Brutus, dass, als ich das, was die geistreichsten und
gelehrtesten Philosophen in griechischer Sprache behandelt hatten, in lateinischer
wiedergab, meine Arbeit mancherlei Tadel finden würde. Denn manchen und nicht gerade
ungelehrten Männern gefällt das Philosophiren überhaupt nicht; andere wollen eine
mässige Tätigkeit hier wohl gestatten, aber meinen, dass man nicht so grossen Fleiss
und so viele Mühe darauf verwenden dürfe. Auch giebt es Männer, die, mit den Schriften
der Griechen vertraut, die lateinischen verachten und sagen, dass sie ihre Mühe
lieber auf jene verwenden mögen. Endlich werden auch Einige mich vermuthlich an andere
Wissenschaften verweisen, weil diese Art von Schriftstellerei, trotz des Scharfsinns,
doch nach ihrer Meinung meiner Person und Würde nicht gezieme.’’
```

oder

```
‘‘D1353 M1TT31LUNG Z31GT D1R, ZU W3LCH3N GRO554RT1G3N L315TUNG3N UN53R G3H1RN F43H1G
15T! 4M 4NF4NG W4R 35 51CH3R NOCH 5CHW3R, D45 ZU L353N, 483R M1TTL3W31L3 K4NN5T DU
D45 W4HR5CH31NL1ICH 5CHON G4NZ GUT L353N, OHN3 D455 35 D1CH W1RKL1CH 4N5TR3NGT. D45
L315T3T D31N G3H1RN M1T 531N3R 3NORM3N L3RNF43HIGKEIT. 8331NDRUCK3ND, OD3R?’’
```

a) Schreiben Sie eine Funktion `analyze_text`, die eine deskriptive Beschreibung der Buchstaben, Ziffern und Satzzeichen eines Textes zurückgibt.

   Die Grundstruktur der Funktion soll folgendermaßen aussehen:

   ```
   analyze_text <- function(text){
   ...
   ...
   ...
   }
   ```

   Rückgabe der Funktion ist eine Liste mit folgenden Elementen:

   - Die Gesamtzahl an Zeichen des Textes (ohne Leerzeichen und Zeilenumbrüche)
   - Die Anzahl an Buchstaben des Textes
   - Die Anzahl an Ziffern des Textes
   - Die Anzahl an Satzzeichen des Textes
   - Tabelle der im Text vorhandenen Buchstaben, nach absteigender Häufigkeit geordnet
   - Tabelle der im Text vorhandenen Ziffern, nach absteigender Häufigkeit geordnet
   - Tabelle der im Text vorhandenen Satzzeichen, nach absteigender Häufigkeit geordnet

   Die Funktion soll außerdem folgende Spezifikationen erfüllen:

   - Die Elemente des Outputs sollen sinnvoll bezeichnet werden.
   - Die Funktion soll auch einen sinnvollen Output erzeugen, falls eine der Komponenten (Buchstaben, Ziffern, Satzzeichen) gar nicht im Text vorkommt.
   - Der Output der Funktion soll von der Objektklasse `textAnalyze` sein.

b) Definieren Sie eine print-Methode für die Objektklasse `textAnalyze`, die die Ergebnisse der Funktion `analyze_text` übersichtlich in der Konsole ausgibt.

c) Führen Sie für die beiden obigen Beispieltexte eine Textanalyse mit der von Ihnen definierten Funktion in a) aus und geben Sie die Ergebnisse in der Konsole mithilfe von `print` aus.

Aufgabe 3 (Simulationen)

Sie möchten in einer Simulation untersuchen, wie sich der Fehler 1. Art beim Testen mehrerer Hypothesen verhält. Gehen Sie dabei wie folgt vor.

a) Schreiben Sie eine Funktion `simulation.data`, die einen Datensatz mit n Beobachtungen aus einer multivariaten Normalverteilung (Dimension r) mit Mittelwertsvektor $\boldsymbol{\mu}$ und Kovarianz-Matrix $\boldsymbol{\Sigma}$ generiert.

```
simulation.data <- function(n, mu, Sigma){
...
...
...
}
```

Output der Funktion ist eine Matrix der Dimension $n \times r$.

b) Schreiben Sie eine Funktion `simulation.test`, die für jede Spalte der Daten aus a), mithilfe eines zweiseitigen t-Tests (Signifikanzniveau α), den empirischen Mittelwert mit dem wahren Mittelwert vergleicht und die zugehörigen p-Werte ausgibt.

```
simulation.test <- function(data, mu, alpha){
...
...
...
}
```

Output der Funktion ist ein Vektor der Länge r.

c) Schreiben Sie eine Funktion `simulation.repeat`, die die Funktionen `simulation.data` und `simulation.test` wiederholt (n_{rep} mal) aufruft und die entsprechenden Ergebnisse ausgibt.

```
simulation.repeat <- function(n.rep, n, mu, Sigma, alpha){
...
...
...
}
```

Output der Funktion ist eine Matrix der Dimension $n_{\text{rep}} \times r$.

d) Schreiben Sie eine Funktion `simulation.result`, die für den Output der Funktion `simulation.repeat` für jeden Testdurchlauf (zeilenweise) überprüft, ob *mindestens ein* signifikantes Ergebnis vorliegt.

```
simulation.result <- function(p.values){
...
...
...
}
```

Output der Funktion ist der Anteil der Testdurchläufe mit *mindestens einem* signifikaten Ergebnis.

e) Schreiben Sie eine Funktion `simulation`, die die Funktionen `simulation.repeat` und `simulation.result` n_{sim} mal aufruft und jeweils den Anteil an signifikanten Testentscheidungen ausgibt.

```
simulation <- function(n.sim, n.rep, n, mu, Sigma, alpha){
...
...
...
}
```

Output der Funktion ist ein Vektor der Länge n_{sim}.

f) Rufen Sie die Funktion `simulation` mit folgenden Parametern auf:

- $\boldsymbol{\Sigma} = 1$, $\boldsymbol{\mu} = 0$, $n = 100$, $n_{\text{rep}} = 200$, $n_{\text{sim}} = 100$, $\alpha = 0.05$

Achten Sie darauf, dass Ihre Ergebnisse reproduzierbar sind.
Stellen Sie das Simulationsergebnis in geeigneter Weise grafisch dar und interpretieren Sie es.

g) Rufen Sie die Funktion `simulation` mit folgenden Parametern auf:

- $\boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$, $\boldsymbol{\mu} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$, $n = 100$, $n_{\text{rep}} = 200$, $n_{\text{sim}} = 100$, $\alpha = 0.05$

Achten Sie darauf, dass Ihre Ergebnisse reproduzierbar sind.
Stellen Sie das Simulationsergebnis in geeigneter Weise grafisch dar und vergleichen Sie es mit dem theoretischen Wert 0.226. Wie lässt sich dieses Ergebnis erklären?

Aufgabe 4 (Schiffe versenken)

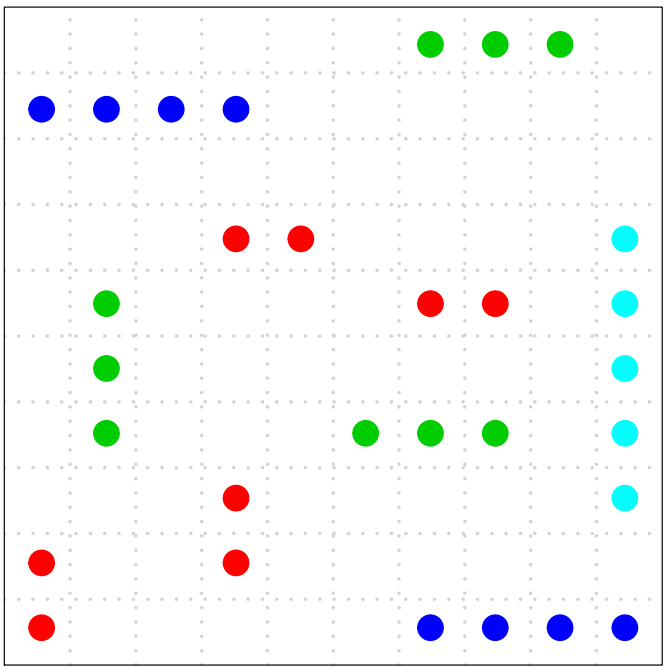
Vor Beginn des Spiels **Schiffe versenken** fertigt sich jeder Spieler einen 10×10 Kästchen großen Plan an. Dieser stellt das eigene Kampfgebiet dar. In das eigene Kampfgebiet trägt man, ohne dass der Mitspieler dies sieht, seine Flotte ein. Dies geschieht, indem man Gebilde von unterschiedlicher Kästchenlänge einzeichnet. Folgende Spielregeln müssen dabei eingehalten werden:

- Die Schiffe dürfen nicht aneinander stoßen.
- Die Schiffe dürfen nicht über Eck gebaut sein oder Ausbuchtungen besitzen.
- Die Schiffe dürfen auch am Rand liegen.
- Die Schiffe dürfen nicht diagonal aufgestellt werden.
- Die Flotte besteht aus insgesamt 10 Schiffen:
  - Ein Schlachtschiff (5 Kästchen)
  - Zwei Kreuzer (je 4 Kästchen)
  - Drei Zerstörer (je 3 Kästchen)
  - Vier U-Boote (je 2 Kästchen)

a) Schreiben Sie eine Funktion `combat_zone`, die ein Kampfgebiet gemäß der obigen Spielregeln zufällig generiert und als Grafik auf dem Bildschirm ausgibt. Argumente der Funktion sind die Anzahl an Schlachtschiffen (n5), Kreuzern (n4), Zerstörern (n3) und U-Booten (n2). Die Grundstruktur der Funktion soll folgendermaßen aussehen:

```
combat_zone <- function(n5,n4,n3,n2){
...
...
...
}
```

Der grafische Output der Funktion könnte beispielsweise folgendermaßen aussehen:



Die Funktion soll außerdem folgende Spezifikationen erfüllen:

- Ausgangspunkt ist ein leeres Kampfgebiet.
- Alle Schiffe werden nacheinander der Größe nach platziert (beginnend mit dem Größten).
- In jedem Schritt wird die Position und die Richtung des neu zu platzierenden Schiffes zufällig bestimmt.
- Bedingt durch die Platzierung der bisherigen Schiffe kann es vorkommen, dass kein möglicher Platz mehr für das nächste Schiff in der gewählten Richtung exisitiert. In einem solchen Fall, soll die Platzierung der Schiffe neu gestartet werden, beginnend mit einem leeren Kampfgebiet. Desweitern soll beispielsweise (in etwa) folgende Meldung in der Konsole ausgegeben werden:

  ```
  Error: Das Boot der Länge 2 hat in vertikaler Richtung keinen Platz mehr!
  ```

- Wurde die Platzierung der Schiffe 10 mal ohne Erfolg gestartet, soll die Funktion abbrechen und (in etwa) folgendes in der Konsole ausgeben:

  ```
  Error: Platzierung der Schiffe nicht möglich!
  ```

- Wurden alle Schiffe platziert, wird das Kampfgebiet als Grafik auf dem Bildschirm ausgegeben und zusätzlich die Anzahl an benötigten Versuche in der Konsole ausgegeben, zum Beispiel:

  ```
  Benötigte Versuche: 3
  ```

- Schiffe derselben Länge werden in der Grafik jeweils in derselben Farbe dargestellt.

b) Generieren Sie, mithilfe der von Ihnen definierten Funktion in a), drei Kampfgebiete für die oben angegebene Flotte und speichern Sie diese als pdf-Datei in Ihrem Arbeitsverzeichnis ab.