Hausübung 3

Organisatorisches zur Abgabe

- Bitte geben Sie die Hausübung über Moodle ab. Bitte geben Sie nur ihre R-Syntax-Datei ab.
- Bennen Sie die R-Syntax-Datei bitte 'ProgStat15_H3_Nachname_Vorname.R', wobei Sie Nachname und Vorname durch Ihren eigenen Namen ersetzen.
- Gruppenarbeit von bis zu drei Personen ist möglich. Pro Gruppe bitte nur einmal abgeben! (Jede Hausübung kann prinzipiell in einer neuen Gruppe abgegeben werden.)
- Um den Schein zu erwerben, muss jede Hausübung einzeln positiv bestanden werden.
- **Letztmöglicher Abgabetermin: Mittwoch, den 16. September 2015 (23:55 Uhr)**

Vorbereitung

- Legen Sie folgende Ordnerstruktur an: Hauptordner 'H3' mit Unterordnern 'Daten', 'Programme', 'Ergebnisse'. Speichern Sie in 'Daten' Ihre Daten, in 'Programme' Ihr Programm und in 'Ergebnisse' Ihre Ergebnisse.
- Öffnen Sie eine neue Programmdatei und speichern Sie diese im Programme-Ordner unter folgendem Dateinamen: 'ProgStat15_H3_Nachname_Vorname.R', wobei Sie Nachname und Vorname durch Ihren eigenen Namen ersetzen. (Sie können auch die Vorlage auf der Homepage verwenden.)
- Geben Sie bitte in den ersten Zeilen der R-Datei die Namen aller Gruppenmitglieder an. (Ordnen Sie diese bitte möglichst alphabetisch nach dem Nachnamen. Verwenden Sie den ersten Namen als Dateinamen, falls Sie als Gruppe abgeben.)
- Verwenden Sie bitte den H3-Ordner als Working Directory.
- Schreiben Sie bitte zu jeder 'Lösung' die Aufgabennummer.
- 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

(50 Punkte)

Sie möchten den Zusammenhang zwischen einer skalaren Zielgröße y und einer kategorialen Einflussgröße mit K Kategorien $x \in \{1, \ldots, K\}$ mittels eines linearen Regressionsmodells beschreiben. Dabei gilt:

$$y_i = \beta_0 + \sum_{k=1}^{K-1} \beta_k Z_{ik} + \epsilon_i, \quad i = 1, \ldots, n, \quad \epsilon_i \sim N(0, \sigma^2),$$

wobei $Z_{ik} = \begin{cases} 1, & \text{falls } Z_i = k \\ 0, & \text{falls } Z_i \neq k \end{cases}$ Dummy-Variablen darstellen. Kategorie K dient als Referenzkategorie.

a) Schreiben Sie eine Funktion 'sim_data', die Daten gemäß des obigen Modells für beliebige Werte von n, σ^2, den wahren Regressionskoeffizienten β_0 und $\boldsymbol{\beta}$ und $\boldsymbol{\pi}$ erzeugt. Der Vektor $\boldsymbol{\pi}$ enthält dabei die Wahrscheinlichkeiten für die Kategorien $k = 1. \ldots, K$, die Länge von $\boldsymbol{\beta}$ entspricht $K - 1$.

```
sim_data <- function(n, sigma2, beta0, beta, pi){
...
...
...
}
```

Die Funktion gibt den data.frame ($\mathbf{y}$,$\mathbf{Z}$) zurück, der in der ersten Spalte den Vektor der Zielgröße und in den folgenden Spalten die Matrix der Dummy-Variablen der kategorialen Einflussgröße mit Dimension n x $(K-1)$ beinhaltet. Die Spalten des Datensatzes werden mit $y, Z\_1, Z\_2, \ldots, Z\_K-1$ benannt.

b) Schreiben Sie eine Funktion 'est_model', die ein lineares Modell der Daten aus a) berechnet und die Koeffizienten der Schätzung zurückgibt.

```
est_model <- function(data){
...
...
...
}
```

Output der Funktion ist ein Vektor der geschätzten Regressionskoeffizienten der Länge K.

c) Schreiben Sie eine Funktion 'est_MSE', die den MSE der Schätzung des Modells aus b) basierend auf den Daten aus a) gemäß eines iterativen Prozesses zurückgibt. Machen Sie explizit Gebrauch von Ihrer Funktion aus Teilaufgabe b).

```
est_MSE <- function(data, m){
...
...
...
}
```

Gehen Sie dabei wie folgt vor:

1. Teilen Sie den vollständigen Datensatz zufällig in m gleichgroße Teile. Dabei muss sichergestellt sein, dass in jedem Teildatensatz Beobachtungen aus allen Kategorien $k = 1, \ldots, K$ vorhanden sind.

2. Lassen Sie 1/m der Daten weg und schätzen Sie das Modell auf Basis der restlichen Daten.
3. Berechnen Sie den MSE unter Verwendung der vorher geschätzten Regressionskoeffizienten auf den in diesem Schritt nicht verwendeten Daten.
4. Wiederholen Sie Schritt 2 und 3 für alle m Teile.
5. Berechnen Sie den geschätzten MSE der Daten als den Durchschnitt der MSEs der m Schätzungen.

Die Funktion gibt den geschätzten MSE zurück.

d) Schreiben Sie eine Funktion 'est_KI', die die geschätzen Regressionskoeffizienten $\hat{\beta}_0$ und $\hat{\boldsymbol{\beta}}$ des Modells aus b) basierend auf den Daten aus a) sowie ein geschätztes 95%-Konfidenzintervall für jeden Regressionskoeffizienten zurückgibt. Machen Sie explizit Gebrauch von Ihrer Funktion aus Teilaufgabe b).

```
est_KI <- function(data, s){
...
...
...
}
```

Gehen Sie dabei wie folgt vor:

1. Schätzen Sie das Modell auf den vollständigen Daten.
2. Ziehen Sie aus den Daten eine Stichprobe der Größe n mit Zurücklegen. Dabei muss sichergestellt sein, dass in der Stichprobe Beobachtungen aus allen Kategorien $k = 1, \ldots, K$ vorhanden sind.
3. Schätzen Sie das Modell erneut auf der Stichprobe und speichern Sie die resultierenden Regressionskoeffizienten.
4. Wiederholen Sie Schritt 2 und 3 für eine bestimmte Anzahl an Wiederholungen s.
5. Bestimmen Sie das 2.5%-Quantil und das 97.5%-Quantil der s Wiederholungen für jeden Regressionskoeffizienten.
6. Die resultierenden Intervalle aus Schritt 5 entsprechen approximativen 95%-Konfidenzintervallen für die Regressionskoeffizienten aus Schritt 1.

Die Funktion gibt eine Matrix mit drei Spalten zurück. Die erste Spalte beinhaltet die geschätzten Regressionskoeffizienten basierend auf den vollständigen Daten. Die zweite Spalte beinhaltet die geschätzten unteren Schranken des Konfidenzintervalls und die dritte Spalte die geschätzten oberen Schranken des Konfidenzintervalls.

e) Schreiben Sie eine Funktion 'repetitions', die alle bisher implementierten Berechnungen für eine feste Kombination der Parameter n, σ^2, β_0, $\boldsymbol{\beta}$, $\boldsymbol{\pi}$, m, s und eine beliebige Anzahl an Wiederholungen w ausführt. Machen Sie dazu explizit Gebrauch von Ihren Funktionen aus den vorherigen Teilaufgaben.

```
repetitions <- function(n, sigma2, beta0, beta, pi, m, s, w){
...
...
...
}
```

Die Funktion gibt eine Liste mit drei Elementen zurück. Das erste Element ist ein dreidimensionales array, das die Datenmatrix aller Wiederholungen beinhaltet. Das zweite Element ist ein Vektor, der die MSEs aller Wiederholungen beinhaltet. Das dritte Element ist ein dreidimensionales array, das den Output der Funktion 'est_KI' für jede Wiederholung beinhaltet.

f) Rufen Sie die Funktion aus e) mit folgenden Parameterkombinationen auf:

- $n = 100$, $\sigma^2 = 1$, $\beta_0 = 2$, $\boldsymbol{\beta} = (-1, 1, 2)$, $\boldsymbol{\pi} = (0.25, 0.25, 0.25, 0.25)$, $m = 10$, $s = 250$, $w = 100$
- $n = 500$, $\sigma^2 = 2$, $\beta_0 = 1$, $\boldsymbol{\beta} = (-2, -1, 1, 2)$, $\boldsymbol{\pi} = (0.2, 0.15, 0.3, 0.1, 0.25)$, $m = 5$, $s = 150$, $w = 100$

g) Werten Sie (Teile der) Ergebnisse aus f) auf sinnvolle Weise graphisch aus.

Hinweis: Es empfiehlt sich, jede Funktion der Teilaufgaben a) bis d) auch einzeln mit sinnvollen Eingaben zu testen.

Aufgabe 2

(33 Punkte)

Die folgenden Teilaufgaben beziehen sich auf einen Datensatz zur Funktionseinschränkung von Patienten mit Multipler Sklerose (siehe `https://de.wikipedia.org/wiki/Multiple_Sklerose`). Ein Ausschnitt des Datensatzes kann von der Homepage heruntergeladen werden.

Erklärung **der für die Aufgabe relevanten** Variablen:

- id_study_centre: Die ID des Studienzentrums, als dreistellige Zahl
- id_patient: Die ID des Patienten innerhalb des Studienzentrums, als dreistellige Zahl
- age: Alter des Patienten (in Jahren)
- sex: Geschlecht des Patienten (1=weiblich, 2=männlich)
- MS_Form: Art der MS
  - 1 = RRMS (schubförmig remittierende MS)
  - 2 = PPMS (primär progrediente MS)
  - 3 = SPMS (sekundär progrediente MS)
- ICF-Kategorien der Komponente Aktivitäten und Partizipation (alle Variablen, deren Name mit d beginnt, gefolgt von 3 Zahlen; aber nicht die Variablen, die auf _C enden), z.B. d110, d115, d140, usw.
  - 0 = keine Beeinträchtigung
  - 1 = leichte Beeinträchtigung
  - 2 = mittlere Beeinträchtigung
  - 3 = große Beeinträchtigung
  - 4 = vollständige Beeinträchtigung
  - 8 = nicht spezifiziert
  - 9 = nicht anwendbar

  Jede dieser ICF-Kategorien beschreibt das Ausmaß der Beeinträchtigung in einem bestimmten Lebensbereich.
- EDSS: Expanded Disability Status Scale - Leistungsskala, die Auskunft über den Schweregrad der Behinderung bei Multiple Sklerose-Patienten gibt. Die Skala beginnt bei 0.0 (= normal) und endet bei 10.0 (Tod infolge MS).
  Siehe dazu auch: `https://de.wikipedia.org/wiki/Expanded_Disability_Status_Scale`

- BDI: Beck Depression Inventory - Score zur Messung des Schweregrads einer Depression. Der BDI-II besteht aus 21 Fragen, von denen jede auf einer Skala von 0 bis 3 beantwortet werden kann. Aus diesen wird der Score berechnet, der folgendermaßen interpretiert werden kann. Umso höher der Scorewert, desto schwerer ist die Depression.
  Siehe dazu auch: `https://en.wikipedia.org/wiki/Beck_Depression_Inventory`

a) Speichern Sie den Datensatz im Unterordner für die Daten. Lesen Sie die Daten von dort ein. (Verwenden Sie dazu ihr Working Directory!) Nennen Sie den Datensatz 'data'.

b) Im Datensatz ist die ID des Studienzentrums und die ID des Patienten angegeben. Bilden Sie eine neue Variable 'id_observation', die eine eindeutige ID für jeden Patienten enthält. Diese soll die ID des Studienzentrums, dann einen Unterstrich, und dann die ID des Patienten enthalten. (Es macht nichts, wenn die führenden Nullen in der ID des Patienten dabei verloren gehen.) Diese neue Variable soll Teil des gesamten Datensatzes sein.

c) Erstellen Sie einen Vektor namens 'ICF_d', der alle Variablennamen enthält, die mit d beginnen und auf welches dann genau 3 Zahlen folgen, also d110, d115, d140, usw.. (Erstellen Sie diesen Vektor automatisiert, indem Sie die Struktur der Variablennamen verwenden. Kopieren Sie nicht die Variablennamen von Hand hintereinander!)

d) Kodieren Sie auf effiziente Art alle Ausprägungen 8 und 9 in den Variablen 'ICF_d' in fehlende Werte um.

e) Kodieren Sie alle relevanten kategorialen Variablen (sex, MS_Form, ICF-Kategorien der Komponente Aktivitäten und Partizipation) explizit als solche und weisen Sie deren Wertelabel zu. (Setzen Sie dies für die ICF-Kategorien effizient um.)

f) Schreiben Sie eine Funktion namens 'bivariate_regression'. Diese soll für eine metrische Zielgröße und mehrere Einflussgrößen, die sowohl metrisch als auch kategorial sein können, jeweils ein lineares Regressionsmodell zur Beschreibung des bivariaten Zusammenhangs zwischen der Zielgröße und jeweils einer Einflussgröße berechnen.

   Die Funktion hat folgende Parameter:

   - data: der Name des Datensatzes
   - y_var: der Name der Zielgröße
   - x_vars: Vektor der Namen der Einflussgrößen

```
bivariate_regression <- function(y_var, x_vars, data){
...
...
...
}
```

   Die Funktion gibt eine Tabelle zurück, die folgendermaßen aufgebaut ist.

   - In den Spalten stehen in dieser Reihenfolge:
     - der Variablenname der unabhängigen Variablen,
     - die jeweilige Ausprägung der Variable bei kategorialen Variablen (leer bei metrischen Variablen),
     - der geschätzte Regressionskoeffizient (gerundet auf 4 Nachkommastellen) für diese Variable bzw. Ausprägung der Variable,

  - das 95%-Konfidenzintervall für den Regressionskoeffizienten in folgender Darstellung: [untere Grenze (gerundet auf 4 Nachkommastellen); obere Grenze (gerundet auf 4 Nachkommastellen)], z.B. [0.2203; 1.4423]
  - der p-Wert (gerundet auf 5 Nachkommastellen).
- In den Zeilen stehen die einzelnen Einflussgrößen in der Reihenfolge, in der sie in x_vars enthalten sind. Der Regressionskoeffizient für den Intercept wird nicht berücksichtigt. Für jede metrische Einflussgröße gibt es eine Zeile. Für jede kategoriale Einflussgröße mit K Ausprägungen gibt es $K - 1$ Zeilen (für alle Werte ausgenommen der Referenzkategorie).
- Überlegen Sie sich, ob Sie die Funktionalität der Funktion sinnvoll in weitere Funktionen aufsplitten können. Implementieren Sie dies gegebenenfalls geeignet. (Möglichkeit, aber kein Muss.)

g) Rufen Sie die Funktion mit den aufbereiteten MS-Daten auf. Verwenden Sie EDSS als Zielgröße und sex, age, MS_Form und d460, d475 und d850 in dieser Reihenfolge als Einflussgrößen.

h) Schreiben Sie eine weitere Funktion, die die Funktion 'bivariate_regression' für mehrere Zielgrößen aufruft und die Ergebnisse geeignet kombiniert. (Benennen Sie die Spalten geeignet.)

i) Rufen Sie die neue Funktion aus Teilaufgabe h) mit den beiden Zielgrößen EDSS und BDI auf und denselben Einflussgrößen wie in Teilaufgabe g).

Aufgabe 3

(40 Punkte)

Vier Gewinnt ist ein Zweipersonen-Strategiespiel.
Das Spiel wird auf einem senkrecht stehenden hohlen Spielbrett gespielt, in das die Spieler abwechselnd ihre Spielsteine fallen lassen. Jeder Spieler hat Spielsteine in genau einer Farbe (z.B. schwarz und rot). Wenn ein Spieler einen Spielstein in eine Spalte fallen lässt, besetzt dieser den untersten freien Platz der Spalte. Gewinner ist der Spieler, der es als erster schafft, vier oder mehr seiner Spielsteine waagerecht, senkrecht oder diagonal in eine Linie zu bringen. Das Spiel endet unentschieden, wenn das Spielbrett komplett gefüllt ist, ohne dass ein Spieler eine Viererlinie gebildet hat.

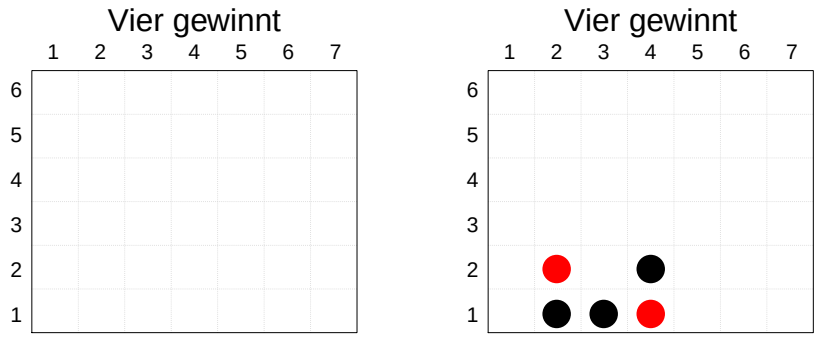
Schreiben Sie eine Funktion 'fourwin', mit welcher zwei (reale) Spieler, auf einem Spielfeld beliebiger Größe mit `n_row` Zeilen und `n_col` Spalten, gegeneinander Vier Gewinnt spielen können.

```
fourwin <- function(n_row, n_col){
...
...
...
}
```

Zu Beginn des Spiels wird zufällig entschieden, ob Spieler 1 oder Spieler 2 beginnt. Ein Spielzug erfolgt durch Eingabe der gewählten Spalte, in die der Spieler seinen Stein setzen möchte, in die Konsole. Der Output der Konsole während des Spielverlaufs, könnte (ausschnittsweise) folgendermaßen aussehen:

```
> Spieler 1:
> 3
> Spieler 2:
> 4
> Spieler 1:
> 2
```

Das Spielfeld und der Spielverlauf (das Setzen von Steinen) wird als sich-veränderliche Graphik auf dem Bildschirm ausgegeben. Der Output zu Spielbeginn und nach fünf gespielten Zügen könnte beispielsweise folgendermaßen aussehen:



Das Spiel ist beendet, sobald ein Spieler gewonnen hat oder das Spielfeld komplett befüllt ist. Wenn dies der Fall ist, so wird in der Konsole in Form von Text eine Nachricht ausgeben. Der Output bei Ende des Spiels könnte (ausschnittsweise) folgendermaßen aussehen:

```
> Spieler 2:
> 5
[1] "Spieler 2 hat gewonnen!"
```

oder

```
> Spieler 1:
> 3
[1] "Unentschieden!"
```

Die Funktion muss folgende Spezifikationen des Spiels erfüllen:

- Die Maximalzahl an Steinen, die gesetzt werden können, ist die Anzahl an Spielfeldern.
- Die Maximalzahl an Steinen, die pro Spalte gesetzt werden können, ist die Anzahl an Zeilen.
- In jedem Spielzug muss sichergestellt werden, dass die Eingabe des Spielers valide ist.
- Nach jedem Spielzug muss überprüft werden, ob einer der Spieler gewonnen hat. Eine Linie, die zum Sieg führt, kann nur vom aktuell gesetzten Stein ausgehen.

Folgende Hinweise können zur Implementierung des Spiels hilfreich sein:

- In jedem Spielzug muss nur die gewünschte Spalte angegeben werden, die jeweilige Zeile ist automatisch durch die vorherigen Spielzüge festgelegt.
- Nutzen Sie die in `R` existierende Funktion `scan()`.
- Lagern Sie (eigenständige) Teile der Funktion 'fourwin' in Hilfsfunktionen aus.
- Testen Sie Ihre Funktion durch mehrmaliges Spielen mit unterschiedlichen Spielzügen (nur zur Selbstkontrolle).