Aufgabenblatt
Abgabe: bis Montag, 25.01.2009; 10:00 Uhr

Diese Zettel sollen Sie mit dem Statistikprogramm R vertraut machen. Zunächst werden die grundlegende Bedienung und wichtige Befehle erklärt. Im Anschluss daran finden sich Aufgaben.
Durch das regelmäßige Bearbeiten dieser Aufgaben kann ein Teilnahmeschein erworben werden.
Alle Aufgaben sind Programmieraufgaben. Bitte kopieren Sie die eingegebenen Befehle und die Ausgaben von R in eine Textdatei (Kommentare können hinter der Raute `#` eingegeben werden), und senden Sie diese zur Abgabe per e-Mail an
`mentemeier@uni-muenster.de`
Eine schriftliche Abgabe ist nicht erforderlich.

Das Statistikprogramm R ist frei verfügbar, und kann von `http://www.r-project.org/` heruntergeladen werden. Zum Starten in der Uni siehe gesonderte Anleitung.

Theorie

Generalisierte Lineare Modelle

In GLM können auch nicht-stetigverteilte Zufallsgrößen behandelt werden, z.B. binomialverteilte Zufallsgrößen. Da ein lineares Modell für 0-1-Variablen wenig Sinn ergibt, wird hier vielmehr angenommen, dass die Erfolgswahrscheinlichkeit *in gewisser Weise* einem linearen Modell genügt. Für eine Zielgröße X und Einflussgrößen $k_1, \dots k_n$ nehmen wir also an, dass

$$\pi_i = P(X_i = 1) = F(\theta_0 + \theta_1 k_{i1} + \dots \theta_n k_{in})$$

mit zu schätzenden Parametern θ_0 bis θ_n gilt. Beachte, dass keine Fehler ϵ_i auftauchen, der Zufall kommt bereits dadurch ins Spiel, dass X_i ja nur $B(1, \pi_i)$-verteilt ist. Beachte weiterhin, dass wir π_i als Funktion F des linearen Prädiktors $\theta_0 + \theta_1 k_{i1} + \dots \theta_n k_{in}$ annehmen. Nun, wir wollen Wahrscheinlichkeiten schätzen, und diese müssen zwischen 0 und 1 liegen. Darum wird (mit einiger Plausibilität) F als (streng monotone) Verteilungsfunktion gewählt. Ist F die logistische Verteilungsfunktion, so spricht man vom *Logit*-Modell, und vom *Probit*-Modell, wenn F die Verteilungsfunktion der Standardnormalverteilung ist.

GLM sind in der Funktion `glm()` implementiert, die vor allem der Parameter `family` von `lm()` unterscheidet. In diesem Parameter wird angegeben, welcher Verteilungsfamilie die Zielgröße angehört, im obigen Fall müssten wir `familiy=binomial` setzen. Die Funktion F wird durch ihre Inverse F^{-1}, die sogenannte *Linkfunktion* eingebaut, und zwar als Parameter der Verteilungsfamilie. Die Beziehung wird in der für lineare Modelle bekannten Syntax eingegeben, mit obigen Bezeichnungen würde der Aufruf

```
glm(X ~ k1 + k2 + ... + kn, family = binomial(link="logit"))
```

lauten.

In der folgenden Aufgabe wird außerdem der Parameter `subset` benötigt, dort können wir einen Boolschen Vektor angeben, der eine Auswahl der Daten vornimmt, bspw. `subset=(jahr==1985 & patus==0)`.

Aufgaben

Aufgabe 1. (6 Punkte)
Lesen Sie den auf der Homepage verlinkten Datensatz `Patentdaten` ein. Untersuchen Sie die Wirkung der verschiedenen erfassten Einflussgrößen auf die Wahrscheinlichkeit der Zielgröße *Einspruch gegen das Patent ja / nein* in einem generalisierten linearen Modell, indem Sie wie folgt vorgehen:

(a) Untersuchen Sie zunächst den Datensatz auf untypische Daten, indem Sie sich die `summary` anzeigen lassen, und identifizieren Sie (bspw. mittels Boxplots) zwei Einflussgrößen mit deutlichen Ausreißern.

(b) Schätzen Sie mit `glm()` den Einfluss der Größen *jahr, azit, ansp, uszw, patus, patdsg, aland* auf die Wahrscheinlichkeit für einen Einspruch bei einem Computertechnologie-Patent (`biopharm==0`) in einem Logit-Modell.

(c) Führen Sie die gleiche Schätzung mit zensierten Daten durch: Bestimmen Sie für die in (a) ausgewählten Einflussgrößen 99%-Quantile, und nehmen Sie nur die Datensätze in die Untersuchung mit auf, bei denen diese Merkmale kleiner gleich diesen Quantile sind.

(d) Untersuchen Sie die zensierten Daten nun auch in einem Probit-Modell. Sie werden feststellen, dass die geschätzten Koeffizienten deutlich abweichen. Berechnen Sie in beiden Modellen (Logit und Probit) die Quotienten aus den geschätzten Koeffizienten und dem `Intercept`. Was fällt Ihnen auf?

Bemerkung: Die von R bei Aufruf von `summary(glm(...))` angegebenen Signifikanzniveaus beziehen sich auf die Hypothese, dass der jeweilige Koeffizient 0 sei.

Aufgabe 2 (Konsistenz von Schätzern)**.** (4 Punkte)
Lesen Sie (diesmal mit dem Befehl `source`) die in `Testdaten1.txt` hinterlegte Variable ein.

(a) Die Variable `x` enthält 200 Realisierungen einer Laplace-Verteilung mit Erwartungswert 0. Vergleichen Sie die Konvergenz der beiden in diesem Fall erwartungstreuen Schätzer *Stichprobenmittel* und *Stichprobenmedian* (`mean` und `median`) gegen den tatsächlichen Parameterwert 0, indem Sie sukzessive Stichprobenmittel bzw. -median der ersten i Werte aus `x` berechnen, $1 \leq i \leq 200$, und je einen Vektor mit deren quadratischen Abweichungen vom wahren Parameterwert (0) füllen. Stellen Sie den Verlauf dieser Abweichungen grafisch dar.

(b) Erstellen Sie zwei Vektoren mit je 200 Realisierungen einer N(0,1) bzw. N(0,5)-verteilten Zufallsgröße. Berechnen Sie wie in (a) sukzessive die jeweiligen quadratischen Abweichungen des Stichprobenmittels vom tatsächlichen Erwartungswert, und stellen Sie diese grafisch dar.

(c) Wiederholen Sie die Befehlsfolge aus (b) mehrmals, bis Sie einen aussagekräftigen Plot erhalten.

Aufgabe 3. (4 Punkte)

(a) Schreiben Sie drei Funktionen `fac.for`, `fac.while` und `fac.rek`, die der Fakultätsfunktion

$$f(n) = n \cdot (n-1) \cdot (n-2) \cdots 2 \cdot 1, \quad n \in \mathbb{N}$$

entsprechen. Dabei soll `fac.for` eine `for`-Schleife benutzen, `fac.while` eine `while`-Schleife und `fac.rek` eine rekursive Funktion sein.

(b) Schreiben Sie drei Funktionen `ZM.2fors`, `ZM.1for` und `ZM`, die jeweils die drei Parameter `n, m` und `dist` erhalten und eine `n` × `m` Matrix zurückgeben, deren Einträge gemäß der Funktion `dist` gesampelt wurden.

Der Parameter `dist` ist demnach eine Funktion der Form

`dist(size)  return(Vektor der Länge size)`,

etwa `dist = rnorm` für standardnormalverteilte Matrix-Einträge.

Die drei Funktionen erzeugen jeweils eine Zufallsmatrix. `ZM.2fors` soll dabei zwei `for`-Schleifen benutzen und dabei jeden Matrizeneintrag einzeln sampeln, `ZM.1for` soll nur eine `for`-Schleife nutzen (etwa indem Sie mit `dist(m)` einen Zufallsvektor der Länge `m` erzeugen und diesen in die Matrix einfüllen) und die Funktion `ZM` soll komplett auf Schleifen verzichten (etwa indem Sie mit `dist(n*m)` einen Zufallsvektor erzeugen und diesen in die Matrix einfüllen).

Vergleichen Sie anschließend die Laufzeiten der Funktionen mit Hilfe von `system.time`. Wählen Sie dabei `n=m=1000` und `dist` als die Gleichverteilung auf $\{-1, 1\}$. Was fällt Ihnen auf?

(c) Schreiben Sie zwei Funktionen `det.sapply` und `det.for`, die jeweils zwei Parameter `n` und `anz` erhalten. Beide Funktionen sollen mit Hilfe von `ZM(n,n,dist=rnorm)` `anz`-viele Zufallsmatrizen sampeln und das Mittel der Determinanten (vgl. `help(det)`) bestimmen. `det.for` soll dabei mit einer `for`-Schleife arbeiten, `det.sapply` dagegen soll den Befehl `sapply` benutzen.

Vergleichen Sie anschließend wieder die Laufzeiten beider Funktionen. Wählen Sie dabei `n=10` und `anz=1000000`.

Aufgabe 4. (6 Punkte)
Marc, Tom und Annette spielen ein Würfelspiel. Jeder Spieler besitzt einen Spielzettel, auf dem die Zahlen $1, 2, \ldots, 10$ notiert sind. Jeder Spieler würfelt 10 Runden mit jeweils 3 Würfeln. In jeder Runde muss der Spieler die gewürfelte Augenzahl unter einer der Zahlen auf dem Spielzettel notieren. Multipliziert man diese Zahl mit der gewürfelten Augenzahl, so erhält man die Punkte, die der Spieler für diese Runde erhält. Unter jeder Zahl des Spielzettels darf nur eine gewürfelte Augenzahl stehen, so dass nach 10 Runden der Spielzettel des Spielers vollständig ausgefüllt ist. Am Ende werden die Punkte addiert und der Spieler mit den meisten Punkten gewinnt das Spiel.

Beispiel: Marc beginnt das Spiel und würfelt eine 15. Diese Zahl schreibt er auf seinem Spielzettel unter die 9. Er erhält für diesen Wurf also $9 \cdot 15$ Punkte. Mit dem nächsten Wurf erzielt er eine 5 und schreibt diese unter die 3 des Spielzettels. Dafür erhält er dann $5 \cdot 3$ Punkte, u. s. w. Nach 10 Würfen sieht sein Spielzettel etwa so aus:

1	2	3	4	5	6	7	8	9	10
4	7	5	12	10	11	15	11	15	18

Damit erzielt Marc $1 \cdot 4 + 2 \cdot 7 + 3 \cdot 5 + 4 \cdot 12 + 5 \cdot 10 + 6 \cdot 11 + 7 \cdot 15 + 8 \cdot 11 + 9 \cdot 15 + 10 \cdot 18 = 705$ Punkte.

Jeder der drei Spieler hat seine eigene Spielstrategie:

Marc nervt das Spiel. Er wählt in jedem Wurf zufällig (gleichverteilt) einen leeren Platz auf dem Spielzettel und trägt dort seinen Wurf ein.

Tom spielt wie folgt:

- Bei einer $3, 4, 5$ oder 6 schreibt er diese Augenzahl stets unter die kleinste freie Stelle auf dem Spielplan.
- Bei einer $7, 8, 9$ oder 10 schreibt er diese Augenzahl stets unter die kleinste freie Stelle auf dem Spielplan, die ≥ 4 ist. Erfüllt dies keine Stelle, so wählt er die größte freie Stelle.
- Bei einer $11, 12, 13$ oder 14 schreibt er diese Augenzahl stets unter die kleinste freie Stelle auf dem Spielplan, die ≥ 6 ist. Erfüllt dies keine Stelle, so wählt er die größte freie Stelle.
- Bei einer $15, 16, 17$ oder 18 schreibt er diese Augenzahl stets unter die kleinste freie Stelle auf dem Spielplan, die ≥ 8 ist. Erfüllt dies keine Stelle, so wählt er die größte freie Stelle.

Annette spielt wie folgt:

- Bei einer $3, 4$ oder 5 schreibt sie diese Augenzahl stets unter die kleinste freie Stelle auf dem Spielplan.
- Bei einer 6 oder 7 entscheidet sie sich spontan (d. h. wählt gleichverteilt) für eine freie Stelle zwischen 4 und 7. Ist dort nichts mehr frei, so wählt sie die größte freie Stelle ≤ 3. Ist auch dort schon alles belegt, so wählt sie die kleinste freie Stelle (die dann mindestens die Zahl 8 ist).

- Bei einer 8 oder 9 entscheidet sie sich ebenfalls spontan (d. h. wählt gleichverteilt) für eine freie Stelle zwischen 4 und 7. Ist dort nichts mehr frei, so wählt sie die kleinste freie Stelle ≥ 8. Ist auch dort schon alles belegt, so wählt sie die größte freie Stelle (die dann aber höchstens die Zahl 3 ist).

- Bei einer $10, 11$ oder 12 entscheidet sie sich für die kleinste freie Stelle auf dem Spielplan, die ≥ 5 ist. Erfüllt dies keine Stelle, so wählt sie die größte freie Stelle.

- Bei einer 13 oder 14 entscheidet sie sich spontan (d. h. wählt gleichverteilt) für eine freie Stelle zwischen 4 und 7. Ist dort nichts mehr frei, so wählt sie die größte freie Stelle ≤ 3. Ist auch dort schon alles belegt, so wählt sie die kleinste freie Stelle.

- Bei einer 15 oder 16 entscheidet sie sich ebenfalls spontan (d. h. wählt gleichverteilt) für eine freie Stelle zwischen 4 und 7. Ist dort nichts mehr frei, so wählt sie die kleinste freie Stelle ≥ 8. Ist auch dort schon alles belegt, so wählt sie die größte freie Stelle.

- Eine 17 oder 18 schreibt sie stets auf die größte freie Stelle.

Programmieren Sie nun drei Funktionen `Sim.Tom`, `Sim.Marc` und `Sim.Annette`, die jeweils einen Spieldurchlauf für die zugehörigere Spielstrategie simuliert.

Schreiben Sie anschließend eine Funktion `Sim.Game(n)`, die `n` Spiele simuliert. Diese Funktion gibt einen Vektor zurück, der die Anzahl der gewonnenen Spiele für jeden Spieler enthält.

Schreiben Sie weitere drei Funktionen `Sim.Game.TomVsMarc(n)`, `Sim.Game.TomVsAnnette(n)` und `Sim.Game.MarcVsAnnette(n)`, die die jeweiligen zwei Spieler gegeneinerander antreten lassen.

Rufen Sie die vier Simulationsfunktionen für `n=1000` auf und stellen Sie die Ergebnisse mit Hilfe von Tortendiagrammen graphisch dar.