1 Generalisierte lineare Modelle (V)

Aufgabe 1

```
# Daten einlesen
foodstamp <- read.table("foodstamp.dat", header=F)

# Variablennamen definieren (Reihenfolge stimmt)
names(foodstamp) <- c("y","TEN","SUP","INC")

# Überblick
str(foodstamp)

## 'data.frame': 150 obs. of  4 variables:
##  $ y  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ TEN: num  1 1 1 1 0 1 1 1 0 1 ...
##  $ SUP: num  0 0 1 0 0 0 0 0 0 0 ...
##  $ INC: num  271 287 714 521 0 ...

# Auffällig: fünfte Beobachtung hat INC = 0
# foodstamp£INC

# Logarithmieren der Variable INC (+1, da der Wert 0 vorkommt)
foodstamp$LMI <- log(foodstamp$INC+1)

# Anzahl der Beobachtungen
n <- length(foodstamp$y)
```

a) Logit Modell

```
foodstampLogit <- glm(y ~ TEN + SUP + LMI, data=foodstamp, family=binomial)
summary(foodstampLogit)

##
## Call:
## glm(formula = y ~ TEN + SUP + LMI, family = binomial, data = foodstamp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5874  -0.5248  -0.3197  -0.2620   2.5769
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.9264     1.6229   0.571  0.56813
## TEN          -1.8502     0.5347  -3.460  0.00054 ***
## SUP           0.8961     0.5009   1.789  0.07365 .
## LMI          -0.3328     0.2729  -1.219  0.22280
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 131.9  on 149  degrees of freedom
## Residual deviance: 106.4  on 146  degrees of freedom
## AIC: 114.4
##
## Number of Fisher Scoring iterations: 5
```

Die Einflussgröße `TEN` ist signifikant von 0 verschieden. Die Chance für die Teilnahme am Essensmarkenprogramm ist für Personen mit Mietverhältnis geringer als für Personen ohne. `SUP` ist nur "schwach" signifikant (Chancen bei Bezug von Unterstützungseinkommen höher). Der Parameter für `LMI` hat zwar das erwartete Vorzeichen, ist aber nicht signifikant.

Zur genaueren Quantifizierung der Effekte:

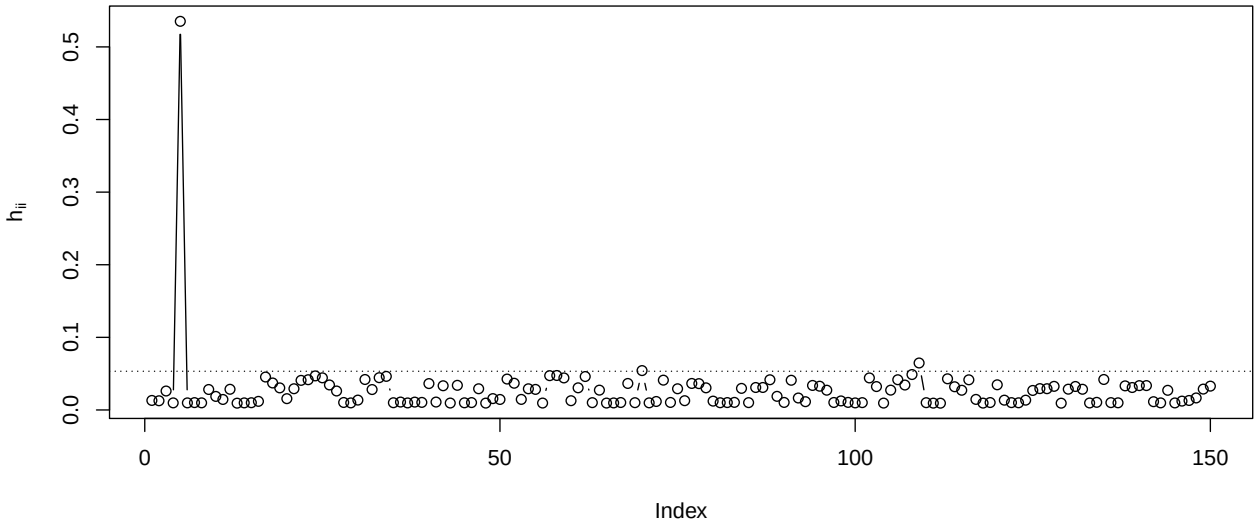
```
exp(foodstampLogit$coef)
```

```
## (Intercept)         TEN         SUP         LMI
##   2.5253512   0.1572037   2.4499390   0.7169480
```

z.B.: Chance (Wkt. durch 1- Wkt.) für Bezug von Essensmarken 2.45 mal größer, falls Ergänzungseinkommen bezogen wird.

b) Übungsmitschrift und ...
Berechnung der hatvalues mit implementierter Funktion (in R wird für GLMs eine Case-Deletion-Diagnostics Approximationen aus Williams, 1987, verwendet):

```
help(hatvalues)
hii <- hatvalues(foodstampLogit)

plot(1:n, hii, type="b", xlab="Index", ylab=expression(h[ii]))
p <- length(foodstampLogit$coef)
abline(h = 2*p/n, lty=3)
```



Man erkennt einen extrem hohen Wert für h_{ii} für Beobachtung 5 (mit `INC=0`), es handelt sich um einen High-Leverage-Punkt (alle Wert mit $h_{ii} > 2p/n$).

Indices der Punkte mit $h_{ii} > 2p/n$ (Kandidaten für High-Leverage-Punkte)

```
candHL <- which(hii > 2*p/n)
candHL
```

```
##   5  70 109
##   5  70 109
```

Auch Beobachtungen 70 und 109 haben eine vergleichsweise große Hebelwirkung.
Man beachte allerdings, dass der Einfluss von Punkten mit extremer Lage im Design-Raum auf die Parameterschätzung im Logit-Modell auch noch vom Residuum abhängt; schließlich hängt die Hat-Matrix auch vom linearen Prädiktor η ab (über $W(\eta)$).

Alternativ: Berechnung der hatvalues mit selbst geschriebener Funktion

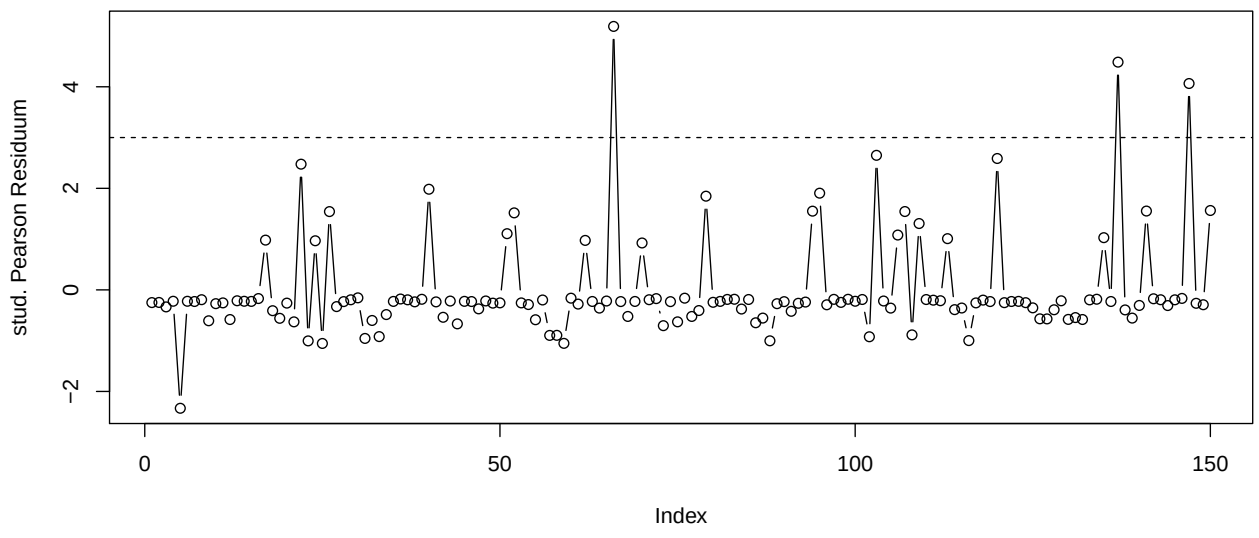
```
hatglm <- function(glmobject){

  # Gewichte aus dem terminalen IWLS-Schritt
  w <- glmobject$weights
  # Extrahieren der Design-Matrix X
  X <- model.matrix(glmobject)
  # Berechnen von W^(t/2)*X
  wsqrtX <- sqrt(w)*X
  # Inverse Fisher-Matrix: hier gleich der skalierten Kovarianzmatrix von \beta
  Finv <- summary(glmobject)$cov.scaled
  hii <- diag(wsqrtX%*%Finv%*%t(wsqrtX))
  return(hii)
}
hiiR <- hatglm(foodstampLogit)
```

Der Vergleich zeigt: nur numerischeÜnterschiede zu `hatvalues()`:

```
sum((hii-hiiR)^2)

## [1] 5.035163e-31

all.equal(hii,hiiR)

## [1] TRUE
```

c) Studentisierten Pearson-Residuen: Berechunung + Plot
Motivation: die asymptotische Varianz der Pearson-Residuen ist gegeben durch $1 - h_{ii}$, deswegen werden die Pearson Residuen damit "studentisiert"

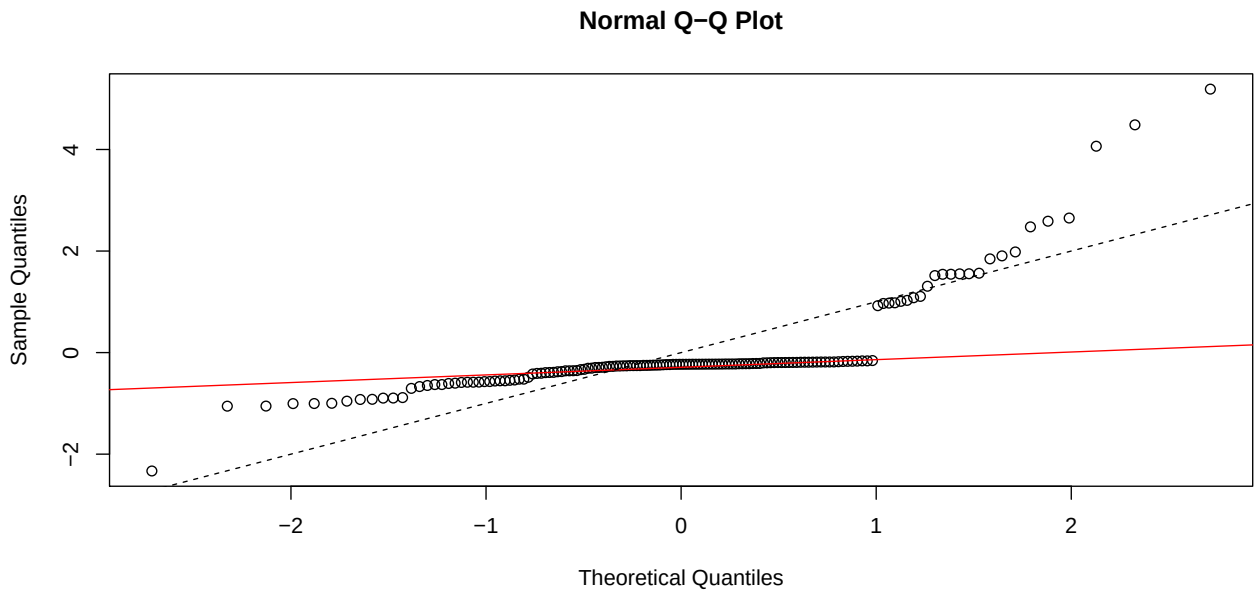
```
pearsonSt <- resid(foodstampLogit,type="pearson")/sqrt(1-hii)
plot(1:n,pearsonSt,type="b",xlab="Index",ylab="stud. Pearson Residuum")
abline(h=3,lty=2)
```



```r
#Große positive Residuen (> 3) für r_s,i^P?
which(pearsonSt > 3)

##  66 137 147
##  66 137 147

#Großes negative Residuen (< -2) ?
which(pearsonSt< -2)

## 5
## 5
```

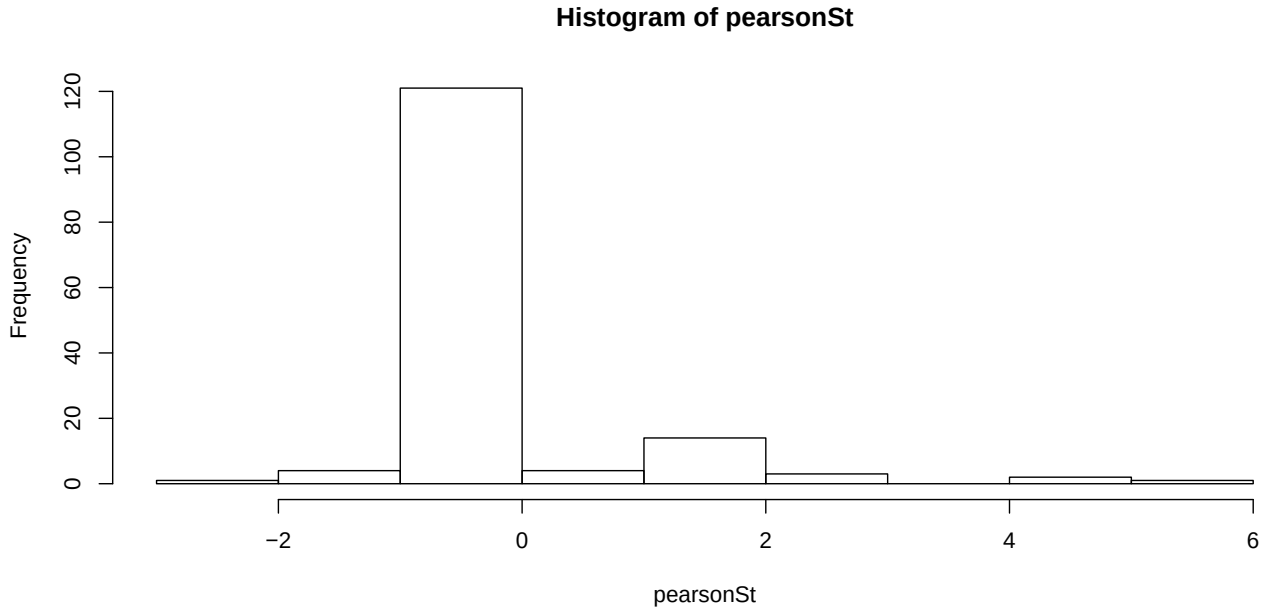
$r_{s,i}^P$ sollten bei gruppierten Daten (für großes n_i) normalverteilt sein. Überprüfung durch NQ-Plot (Plot der emp. Quantile der $r_{s,i}^P$ gegen theoretische Quantile der $N(0,1)$-Verteilung). Im Idealfall sollten alle Punkte auf der Winkelhalbierenden liegen.

```r
qqnorm(pearsonSt)
qqline(pearsonSt,col=2)
abline(c(0,1),lty=2)
```



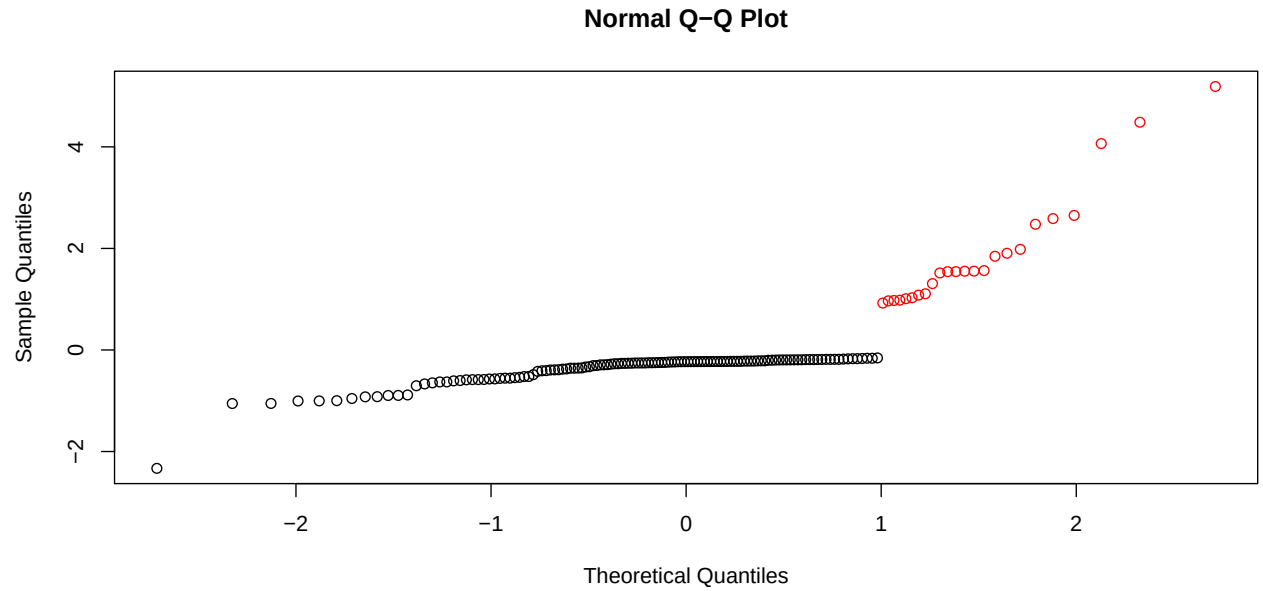
Hier liegen keine gruppierten Daten vor, die $r_{s,i}^P$ sind deutlich entfernt von einer Normalverteilung; die Verteilung ist offensichtlich linkssteil:

```r
hist(pearsonSt)
```



Wenn man den QQ-Plot dennoch zur Diagnostik verwendet: Man identifiziert die zuvor angesprochenen Punkte als Ausreisser. Auffällig ist weiterhin, dass es eine große Gruppe mit betragsmäßig kleineren, negativen Werten für $r^P_{s,i}$ gibt, während eine weitere, kleinere Gruppe positive, betragsmäßig größere Werte von $r^P_{s,i}$ aufweist, die deutlich von den anderen entfernt liegen. Hier handelt es sich um die 24 Punkte mit (y=1), die hier so auffällig sind, da sie nur einen relativ kleinen Anteil an der Gesamtstichprobe von n=150 haben:

```
color <- rep(1,n)
color[foodstamp$y==1] <- 2
qqnorm(pearsonSt,col=color)
```



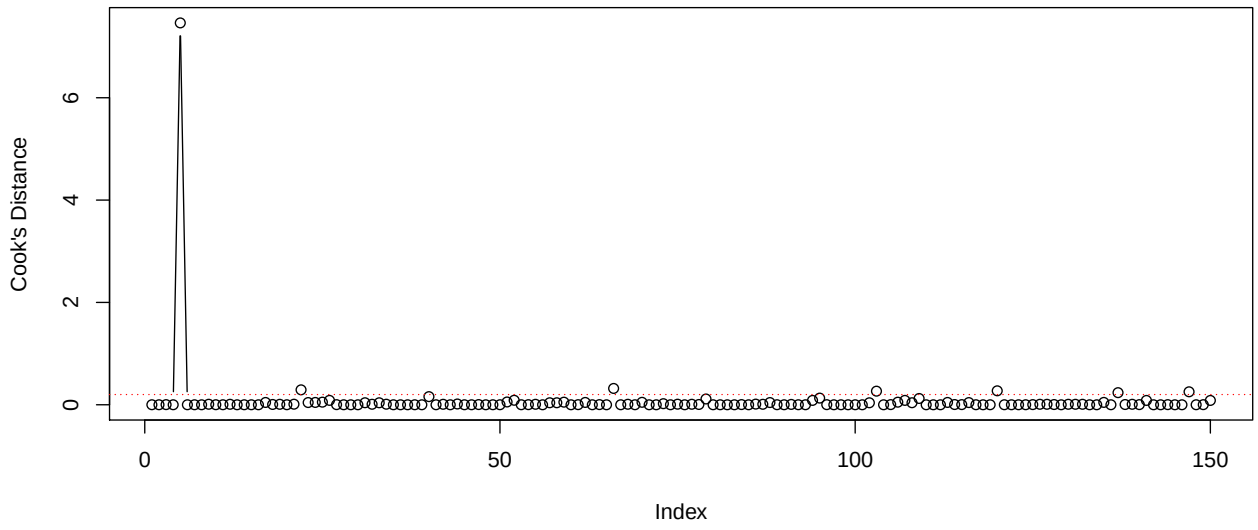
d) Cook's distance: Maß, wie sich der Parameterschätzer $\hat{\beta}$ verändert, wenn man die i-te Beobachtung weglässt (hoher Einfluss ⇒ starke Veränderung), vgl. Formel, diese erhält man aus Taylor-Approximation der Log-Likelihood-Distanz.
Funktion zur Berechnung von Cook's Distance für GLMs:

```
CookGLM <- function(glmobject){

  # Response-Werte
  y <- as.vector(model.frame(glmobject)[,1])

  # Stichprobenumfang
```

```
  n <- length(y)

  # Definition des Ergebnisvektors
  cookres <- numeric(n)

  # mit allen Daten geschätzte Koeffizienten
  beta0 <- glmobject$coef

  # Kovarianzmatrix von beta0
  cov0 <- summary(glmobject)$cov.scaled

  # Inversion: Fisher-Matrix des vollen Modells
  covchol <- chol(cov0)
  Fish0 <- chol2inv(covchol)

  # Berechnung aller n Cook's Distances
  for(i in 1:n)
    {

      # Schätzung bei Weglassen der i-ten Beobachtung
      logittemp <- update(glmobject, data=model.frame(glmobject)[-i,])

      #zugehöriger Parametervektor
      betatemp <- logittemp$coef

      # Cook's Distance
      cookres[i] <- t(betatemp-beta0) %*% Fish0 %*% (betatemp-beta0)
    }
   return(cookres)
}
```

Berechnung und Plot von Cook's Distances

```
cookdistance <- CookGLM(foodstampLogit)
plot(1:n,cookdistance,type="b",xlab="Index",ylab="Cook's Distance")
abline(h=0.2,lty=3,col=2)
```




```
which(cookdistance>0.2)

## [1]   5  22  66 103 120 137 147

1 - pchisq(cookdistance[5],4)

## [1] 0.1133646
```

Die Werte von Cook's Distance sind theoretisch χ^2_{df}-Quadratverteilt mit $df = length(\beta)$, hier also 4 DFs (3 Einfl.gr. + Intercept). Aber Achtung, was hier die Nullhypothese ist ist nicht ganz klar. Testaussagen sind hier nicht so sinnvoll.
Vor allem das Entfernen von Beobachtung 5 verändert die Schätzung stark (wurde auch in (b) und (c) als Ausreisser identifiziert). Augenscheinlich etwas höhere c_i's (aber nur ¿0.2) für Beobachtunen 22, 66, 103, 120, 137, 147 (davon sind 66, 137 und 147 zuvor schon aufgefallen).

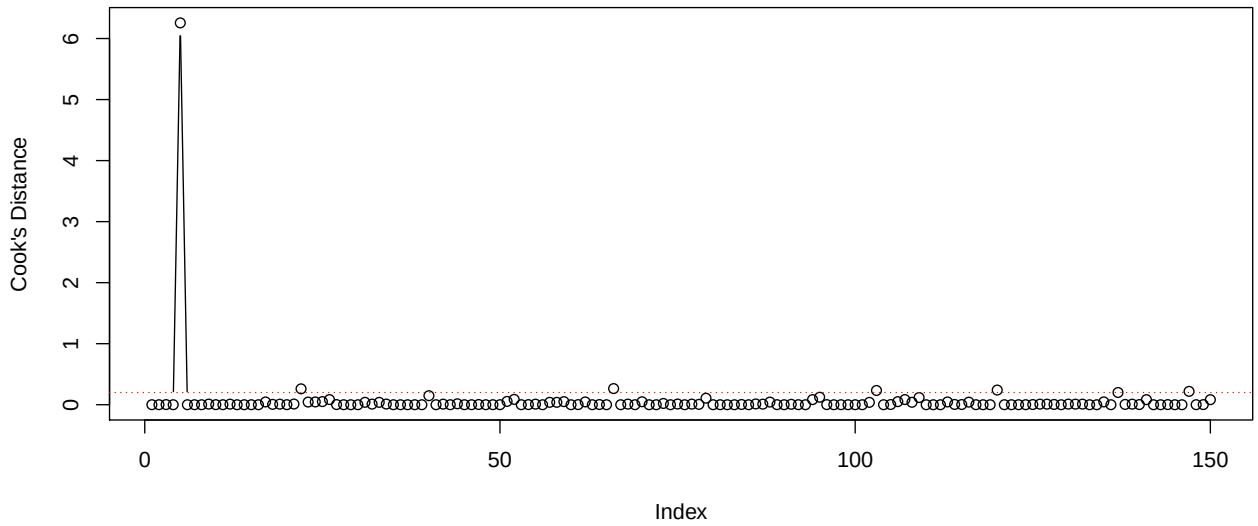
e) siehe Übungsmitschrift

f) Funktion zur Berechnung der Approximation

```
CookGLMapprox <- function(glmobject){

  # Berechnung der h_ii
  hii <- hatglm(glmobject)

  # Berechnen der studentisierten Pearson-Residuen
  pearsonSt <- resid(glmobject,type="pearson")/sqrt(1-hii)

  # approx. Cook's Distances
  c1 <- pearsonSt^2*hii/(1-hii)
  c1
}
```

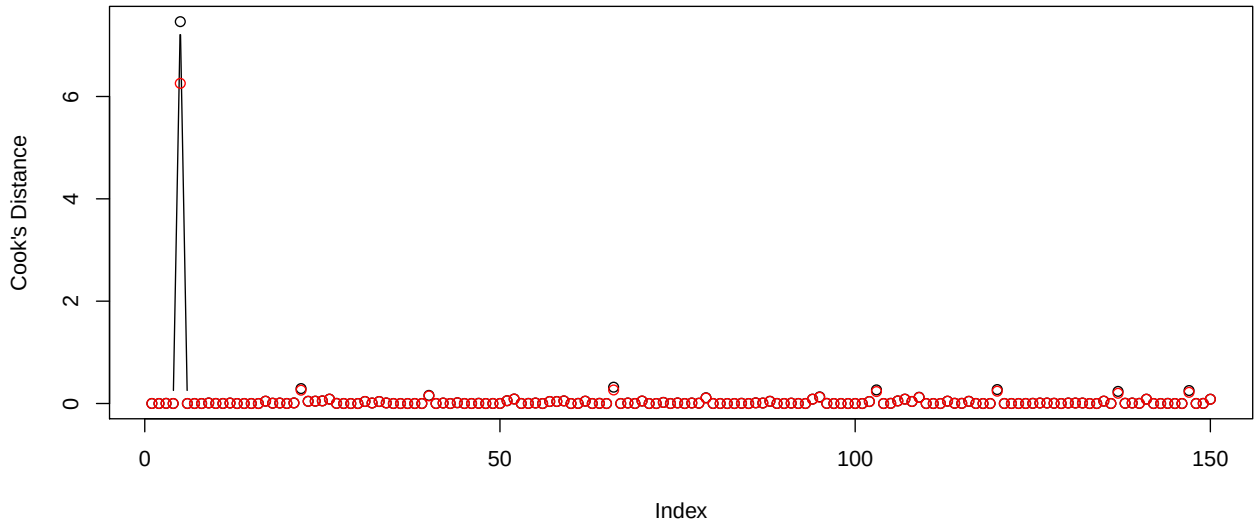
Berechnung und Plot

```
cookdistapprox <- CookGLMapprox(foodstampLogit)
plot(1:n,cookdistapprox,type="b",xlab="Index",ylab="Cook's Distance")
abline(h=0.2,lty=3,col=2)
```



Vergleich mit (d)

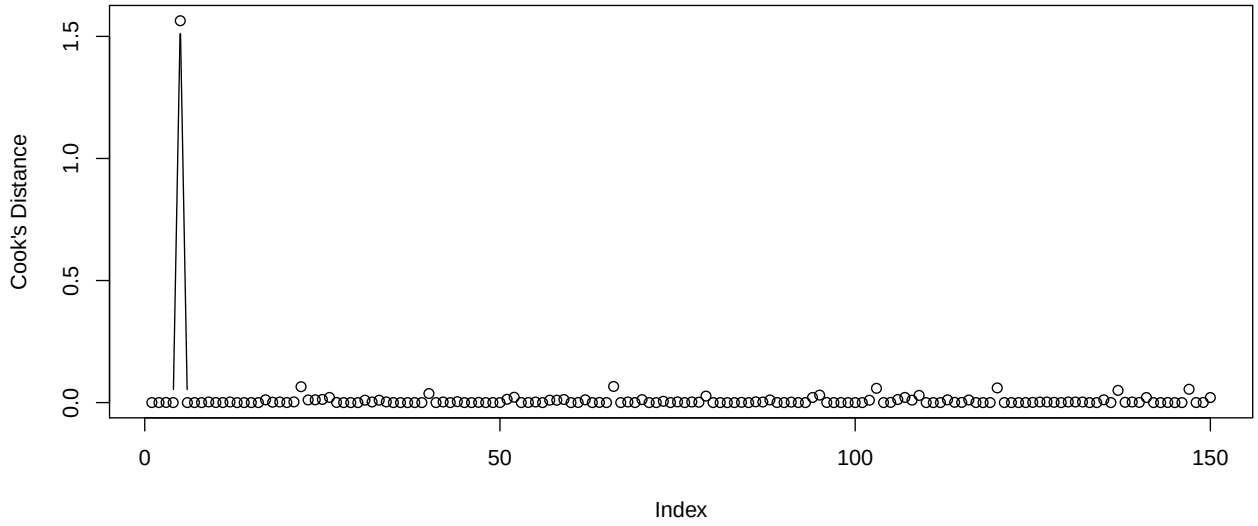
```
plot(1:n,cookdistance,type="b",xlab="Index",ylab="Cook's Distance")
points(1:n,cookdistapprox, col=2)
```




```
help(cooks.distance)
```

Die Funktion `cooks.distance()` aus R verwendet obige Approximation (Williams, 1987; siehe oben), aber eine andere Skalierung:

```
cookdistanceR <- cooks.distance(foodstampLogit)
plot(1:n,cookdistanceR,type="b",xlab="Index",ylab="Cook's Distance")
```




```
#cookdistapprox/cookdistanceR
summary(cookdistapprox/cookdistanceR)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       4       4       4       4       4
```