Modul Statistische Aspekte der Analyse molekularbiologischer und genetischer Daten

R-Blatt 3: Regressionsmodelle in R

Janne Pott

WS 2021/22

In dieser Übung wird R genutzt, um verschiedene Regressionsmodelle zu testen.

Lineare Regression

Beispiele

In dem ersten Teil der Übung beschäftigen wir uns mit einfacher linearer Regression. Dazu wird der Datensatz *SNP.RData* verwendet.

```
loaded1<-load("data/SNP.RData")
loaded1
```

```
## [1] "myDat"
```

```
var<-colnames(myDat)
dummy1<-c("Durchlaufende ID-Nummer","Merkmal/Phänotyp"  ,"Geschlecht","Genotyp")
dummy2<-c(NA,"kontinuierlich","1 = Mann; 2 = Frau","0 = AA; 1 = AB; 2 = BB")
dumTab<-data.frame(var,dummy1,dummy2)
knitr::kable(dumTab, position = "!b",
             caption = "Parameterbeschreibung zu Aufgabe 1 (Blatt 3)",
             col.names = c("Variable","Beschreibung","Codierung / Einheit"))
```

Table 1: Parameterbeschreibung zu Aufgabe 1 (Blatt 3)

Variable	Beschreibung	Codierung / Einheit
id	Durchlaufende ID-Nummer	NA
trait	Merkmal/Phänotyp	kontinuierlich
sex	Geschlecht	1 = Mann; 2 = Frau
SNP	Genotyp	0 = AA; 1 = AB; 2 = BB

Eine einfache lineare Regression $y_i = \beta_0 + \beta_1 * x_i + \epsilon_i$ für $i = 1, ..., n$ kann mit dem Befehl *lm()* durchgeführt werden. Die abhängige Variable ist in diesem Beispiel der *trait*, die unabhängigen Variablen sind *sex* und *SNP*. Wenn man *summary()* auf ein *lm-Objekt* anwendet, werden die Schätzer der Regression ausgegeben.

Die Güte des Models kann am adjustierten r^2 (wie viel Varianze der abhängigen Variablen wird erklärt? Je größer, desto besser), Log-Likelihood (je größer desto besser) oder Akaikes "An Information Criterion" (AIC, je kleiner desto besser) abgelesen werden.

```
mod1<-lm(trait~sex,data=myDat)
summary(mod1)
```

```
##
## Call:
## lm(formula = trait ~ sex, data = myDat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -20.6254  -3.3577  -0.0289   3.3917  20.9575
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.50256    0.11181   380.1   <2e-16 ***
## sex         12.00060    0.07101   169.0   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.021 on 19998 degrees of freedom
## Multiple R-squared:  0.5881, Adjusted R-squared:  0.5881
## F-statistic: 2.856e+04 on 1 and 19998 DF,  p-value: < 2.2e-16
```

```
summary(mod1)$adj.r.squared
```

```
## [1] 0.5881264
```

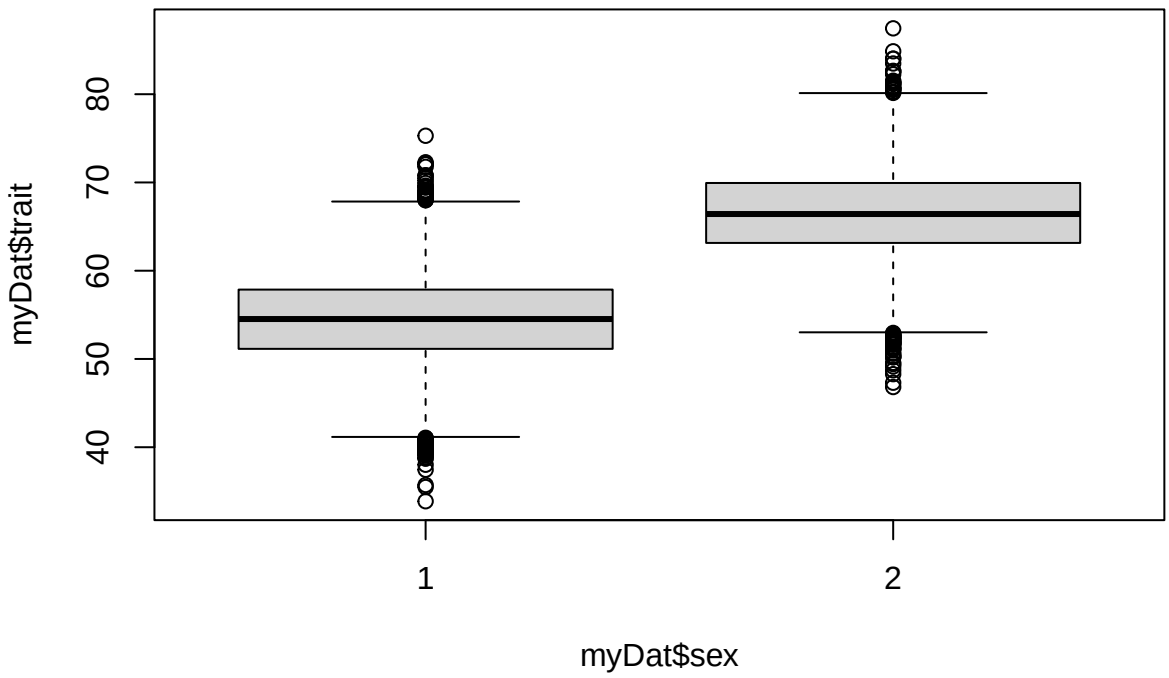
```
logLik(mod1)
```

```
## 'log Lik.' -60649.9 (df=3)
```

```
AIC(mod1)
```

```
## [1] 121305.8
```

```
boxplot(myDat$trait ~ myDat$sex)
```



Diese Regression zeigt uns, dass das Geschlecht einen starken Einfluss auf den *trait* hat: Frauen haben deutlich höhere Werte als Männer.

Um eine multiple lineare Regression zu bestimmen, werden die zusätzlichen unabhängigen Variablen einfach mit "+" ergänzt. Um eine Interaktion dieser zu bestimmen, nutzt man "*".

Um genetische Modelle zu testen, kann man den SNP in Wahrscheinlichkeitsvektoren aufspalten. Anschließend kann man diese verwenden, um zum Beispiel das rezessive Modell (bzgl. Allel B) zu testen.

```
wskBB<-c()
wskBB[myDat$SNP==2]<-1
wskBB[myDat$SNP!=2]<-0
mod2<-lm(myDat$trait ~ wskBB)
summary(mod2)
```

```
##
## Call:
## lm(formula = myDat$trait ~ wskBB)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.175  -4.827  -0.277   4.790  24.466
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 58.13748    0.05958  975.85   <2e-16 ***
## wskBB        7.48313    0.10788   69.36   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.024 on 19998 degrees of freedom
## Multiple R-squared:  0.1939, Adjusted R-squared:  0.1939
## F-statistic:  4811 on 1 and 19998 DF,  p-value: < 2.2e-16
```

Aufgaben

a) Untersuchen Sie die Effekte von *sex* und *SNP* auf *trait* mittels linearer Regression!

b) Untersuchen Sie die multiplen Effekte und die Interaktion der Einflussvariablen *sex* und *SNP*!

c) Welche von den vier Modellen ist besser geeignet, um *trait* zu beschreiben? Begründen Sie Ihre Entscheidung!

d) Welches genetische Modell wird hier verwendet?

e) Erstellen Sie je einen Wahrscheinlichkeits-Vektor pro Genotyp **AA**, **AB** und **BB** (1 = dieser Genotyp trifft zu, 0 = trifft nicht zu).

f) Erstellen Sie nun je einen Vektor pro genetischen Modell (additiv, dominant, rezessiv). (Hinweis: Überlegen Sie zuerst, welchen der Vektoren Sie bereits haben (Teilaufgabe d))

g) Untersuchen Sie die Effekte der verschiedenen Modelle auf *trait* mittels lineare Regression (univariat)! Welches Modell ist am besten geeignet? Wie könnten Sie Ihr Ergebnis testen?

h) Können wir annehmen, dass der SNP auf den Autosomen liegt, oder könnte es sich auch um einen X-chromosomalen SNP handeln? Begründen Sie Ihre Antwort.

Logistische / Proportional Odds Regression

Beispiele

In dieser Aufgabe soll der kontinuierlicher Phänotyp aus der vorherigen Aufgabe in einen binären bzw. kategorialen zerlegt werden. Ein praktisches Beispiel, wo so etwas genutzt wird, ist der BMI. Er wird als kontinuierliches Merkmal bestimmt, aber je nach Analyse will man nur wissen, ob es Unterschiede zwischen Gruppen (Untergewicht (<20), Normalgewicht (20-25), Übergewicht (25-30) oder Adipositas (>30)) gibt, oder auch nur zwischen Übergewicht ja/nein. Die Ergebnisse ähneln sich meistens bezüglich der Signifikanz, allerdings sagen die Schätzer etwas anderes aus.

In diesem Beispiel wird aus *trait* zwei neue Variablen definiert: eine binäre Variabel (>mean), und eine Variable mit drei Kategorien (Terzile).

```
summary(myDat$trait)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   33.88   54.33   60.33   60.42   66.44   87.46
```

```
setDT(myDat)

myDat[trait<mean(trait),trait_bin:=0]
myDat[trait>mean(trait),trait_bin:=1]

quantile(myDat$trait,probs = c(1/3,2/3))
```

```
## 33.33333% 66.66667%
##  56.29975  64.53686
```

```
dum<-quantile(myDat$trait,probs = c(1/3,2/3))
myDat[,trait_cat:=0]
myDat[trait>dum[1],trait_cat:=trait_cat+1]
myDat[trait>dum[2],trait_cat:=trait_cat+1]

table(myDat$trait_bin,myDat$trait_cat)
```

```
##
##        0    1    2
##   0 6667 3397    0
##   1    0 3269 6667
```

Mit diesen beiden Variablen kann eine logistische mit *glm()* bzw. proportional odds Regression mit *polr()* aus dem Paket **MASS** durchgeführt werden. Wichtig: für die prop. odds Regression muss die abhängige Variable als Faktor übergeben werden!

```
modB1<-glm(trait_bin ~ SNP, data=myDat, family = "binomial")
modD1<-polr(as.factor(trait_cat) ~ SNP, data=myDat, Hess = T)
summary(modB1)
```

```
##
## Call:
```

```
## glm(formula = trait_bin ~ SNP, family = "binomial", data = myDat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8307  -0.6461  -0.6461   0.6438   1.8272
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.46063    0.02791  -52.34   <2e-16 ***
## SNP          1.46451    0.02295   63.81   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 27725  on 19999  degrees of freedom
## Residual deviance: 22493  on 19998  degrees of freedom
## AIC: 22497
##
## Number of Fisher Scoring iterations: 4
```

```
summary(modD1)
```

```
## Call:
## polr(formula = as.factor(trait_cat) ~ SNP, data = myDat, Hess = T)
##
## Coefficients:
##     Value Std. Error t value
## SNP 1.264     0.0188   67.22
##
## Intercepts:
##     Value   Std. Error t value
## 0|1  0.3692  0.0221    16.7257
## 1|2  2.0868  0.0272    76.7666
##
## Residual Deviance: 38827.64
## AIC: 38833.64
```

Hier gibt es keine erklärte Varianz mehr, aber AIC und Log-Likelihood kann ebenfalls bestimmt werden.

In der Summary zum polr gibt es keinen direkten p-Wert für die unabhängigen Variablen. Doch diesen kann man einfach aus den t values mittels *pnorm()* bestimmen. Neben der Normalverteilung sind weiter Verteilungen bereits in R definiert, z.B. *pchisq*, *pbinom* und *ppois*, zusammen mit den zugehörigen Dichtefunktionen (*dnorm*), Quantilfunktionen (*qnorm*) und eine Zufallsgenerator (*rnorm*).

```
ctable1 <- coef(summary(modD1))
p <- pnorm(abs(ctable1[, "t value"]), lower.tail = FALSE) * 2
ctable1 <- cbind(ctable1, "p value" = p)
ctable1
```

```
##        Value Std. Error  t value      p value
## SNP 1.263953 0.01880320 67.22012 0.000000e+00
## 0|1 0.369199 0.02207375 16.72570 8.516961e-63
## 1|2 2.086827 0.02718405 76.76659 0.000000e+00
```

Aufgaben

a) Berechnen Sie den **Median** von *trait* und nutzen Sie diesen als Cut-off, um *trait* in einen binären Phänotyp *trait2* zu zerlegen.

b) Untersuchen Sie die univariaten und multivariaten Effekte von *sex* und *SNP* auf *trait2* mittels logistischer Regression! (Hinweis: Funktion *glm()* mit *family="binomial"*)

c) Bestimmen Sie die **Quartile** von *trait*! Zerlegen Sie *trait* nun in einen 4-stufigen Phänotypen *trait3*, in dem Sie die Quartile als Kategorien nutzen.

d) Untersuchen Sie die univariaten und multivariaten Effekte von *sex* und *SNP* auf *trait3* mittels proportional odds regression! (Hinweis: Funktion *polr()* aus dem Paket **MASS** mit *Hess=T*)

e) Vergleichen Sie Ihre Ergebnisse von b) und d) mit den Ergebnissen von Aufgabe 1

Nichtlineare Regression

Dieses Kapitel ist nur ein Zusatz, da es in dieser Veranstaltung nicht direkt benötigt wird.

Beispiele

Ein bekanntes Beispiel für ein nichtlineares Regressionsmodell ist die Michaelis-Menten-Gleichung aus der Enzymkinetik: $v = \frac{V_{max} * c(S)}{c(S)+K_m}$. Sie beschreibt eine Sättigungsfunktion: Umsatzgeschwindigkeit v einer enzymatischen Reaktion in Abhängigkeit der Substratkonzentration *c(S)*. Dabei ist V_{max} die maximale Geschwindigkeit und K_m die Michaelis-Menten-Konstante. Die Geschwindigkeit ist zusätzlich vom Milieu abhängig (z.B. Zelltyp, ph-Wert, o.ä.).

Table 2: Parameterbeschreibung zu Aufgabe 3 (Blatt 3)

Variable	Beschreibung	Codierung / Einheit
cS	Substratkonzentration	in 10^-5 mol
vA	Umsatzgeschwindigkeit, gemessen in Erwachsenen	in mikromol/(mg Enzym)*min
vE	Umsatzgeschwindigkeit, gemessen in Embryonen	in mikromol/(mg Enzym)*min

In der nichtlineare Regression möchte man V_{max} und K_m gleichzeitig schätzen. Dies kann man in R mittels der Funktion *nls()* tun. Als Startwerte kann man z.B. die maximale Geschwindigkeit im Datensatz und die Hälfte davon nehmen.

```
vmaxA<-max(myDat$vA)
kmA<-vmaxA/2
modA<-nls(vA ~ vmax*cS/(cS+km), data=myDat,start = list(km=kmA,vmax=vmaxA))
summary(modA)
```

```
##
## Formula: vA ~ vmax * cS/(cS + km)
##
## Parameters:
##      Estimate Std. Error t value Pr(>|t|)
## km   30.09194    0.10508   286.4   <2e-16 ***
## vmax 20.03563    0.04408   454.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.006654 on 8 degrees of freedom
##
## Number of iterations to convergence: 6
## Achieved convergence tolerance: 4.241e-07
```

Aufgaben

a) Bestimmen Sie V_{max} und K_m für Erwachsene und Embryonen getrennt, indem Sie die Funktion *nls()* und folgende Startwerte nutzen: $V_{max} = max(v)$ und $K_m = \frac{1}{2}max(v)$

b) Was passiert wenn man die Startwerte weglässt?

c) Fassen Sie die Ergebnisse in einem Plot zusammen und interpretieren Sie diesen!