Züchtungslehre - Übung 3

Peter von Rohr

October 14, 2015

Aufgabe 1 (8)

Im Gegensatz zu Aufgabe 2 aus Übung 2 betrachten wir in dieser Aufgabe eine multiple Regression, in welcher die enthaltenen Variablen alle einen Beitrag leisten zur Erklärung der abhängigen $y-$ Variablen.

Mit dieser Aufgabe soll demonstriert werden, dass multiple Regression nicht durch mehrere Regressionen einzelner Variablen ersetzt werden darf. Das mag logisch erscheinen, aber frühe Paper auf dem Gebiet der genomischen Selektion, so zum Beispiel Meuwissen et al. [1], wollten das Problem der vielen Regressionsvariablen durch eine Kombination von mehreren Regressionen mit wenigen Variablen lösen. Simulationsstudien unter anderem in Meuwissen et al. [1] zeigten aber klar, dass diese Methode nicht funktioniert.

In einem einfach Beispiel mit simulierten Daten wollen wir den gleichen Effekt auch demonstrieren. Die Datei mit den simulierten Daten sind auf der Vorlesungswebseite unter
`http://charlotte-ngs.github.io/LivestockBreedingAndGenomics/w5/simgenphen.csv`
verfügbar. Der Datensatz kann mit

```
> dfSimGenPhenDat <- read.csv2(file =
+   "http://charlotte-ngs.github.io/LivestockBreedingAndGenomics/w5/simgenphen.csv",
+                              as.is = TRUE)
```

in R eingelesen werden. Der Befehl

```
> names(dfSimGenPhenDat)
```

```
[1] "id"     "locus1" "locus2" "locus3" "y"
```

zeigt die Kolonnenüberschriften im Datensatz. Die Kolonne "id" enthält die Nummern der Tiere im Datensatz und werden in der Auswertung nicht gebraucht. Die Kolonnen "locus1" bis "locus3" enthalten Genotypen der Tiere an

drei verschiedenen Orten im Genom. Die Genotypen sind wie folgt kodiert.

Genotyp	Wert
A_2A_2	-1
A_1A_2	0
A_1A_1	1

Dabei nehmen wir an, dass an allen drei Loci A_1 das favorisierte Allel ist. Die Kolonne "y" enthält die Merkmalswerte.

Vergleich multiple und einfache Regression

Ihre Aufgabe ist es jetzt die multiple Regression von Merkmal "y" auf die drei x-Variablen "locus1" bis "locus3" zu vergleichen mit den drei einfach Regressionen von "y" auf die drei x-Variablen "locus1" bis "locus3". Den Vergleich machen Sie am besten anhand des Output der Funktion `summary()` für die gefitteten Regressionsmodelle.

Vergleich der Resultate zu den Simulationsparameter

Folgende Parameter wurden zur Simulation der Daten verwendet. Diese sollten mit den Resultaten der multiplen Regression recht gut übereinstimmen. Allfällige tiefe Signifikanzwerte können durch die geringe Anzahl an Tieren im Datensatz begründet werden.

Locus	additiver genotypischer Wert (a)
1	10
2	5
3	2

Als Intercept wurde ein Wert von -10.76 angenommen. Die Standardabweichung der Fehler betrug 2.5.

Aufgabe 2 (5)

Funktionen sind die wahrscheinlich wichtigsten Werkzeuge in R. Wir haben schon ein paar interne Funktionen, wie `lm()`, `plot()`, `sqrt()`, usw. kennen gelernt. Eine der grossen Stärken von R liegt darin, dass wir als Benutzer selber Funktionen erzeugen können und diese dann auch einsetzen können. Unsere eigenen Funktionen können wir auch mit anderen Benutzern austauschen. Also, falls wir ein Problem lösen möchten, dann lohnt es sich immer zuerst ein Suche nach schon bekannten Lösungen zu starten.

Das Ziel dieser Aufgabe soll es sein selber eine Funktion in R zu schreiben. Da wir das in der Vorlesung noch nicht behandelt haben, hier zuerst ein kleines Beispiel einer Funktion. Dieses soll zeigen, wie wir als Benutzer selber Funktionen erschaffen können. Angenommen wir möchten eine Funktion, welche uns

für eine beliebe Zahl n das Quadrat $m = n^2$ der Zahl n berechnet. Als erstes brauchen wir einen Namen für unsere neue Funktion. Wir nennen unsere Funktion einfach "quadratzahl". Als Input für unsere Funktion dient unsere Zahl n, welche wir quadrieren wollen. Jetzt können wir unsere Funktion wie folgt definierten

```
> quadratzahl <- function(n){
+   resultat <- n * n
+   return(resultat)
+ }
```

Unsere Funktion ist erst richtig nützlich, wenn wir sie auch gebrauchen können. Dies passiert mit einem Funktionsaufruf, der wie folgt aussieht

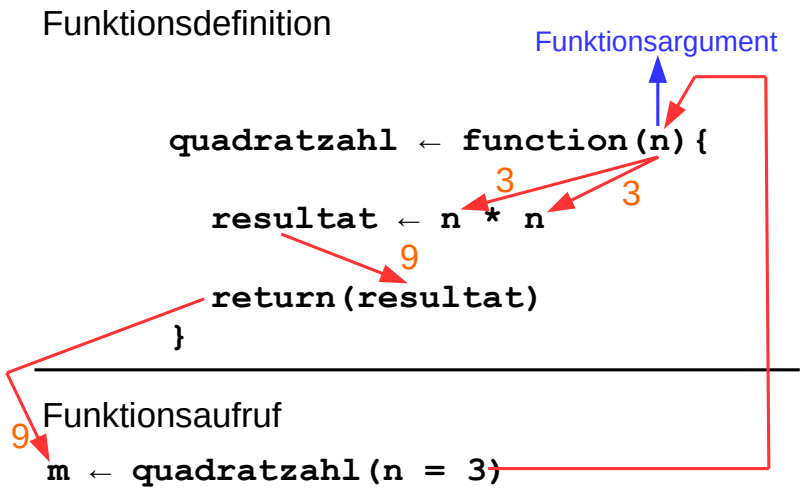
```
> quadratzahl(n = 3)

[1] 9

> quadratzahl(n = 14)

[1] 196
```

Was beim Aufruf genau passiert, kann aufgrund des folgenden Diagrams gezeigt werden.



Ihre eigene Funktion

Ihre Aufgabe wird es nun sein eine eigene Funktion zu schreiben. Diese soll als Input (d.h. als Funktionsargument) einen Temperaturwert in Grad Celsius übernehmen und als Ausgabe oder als Resultat soll die Temperatur in Grad Fahrenheit zurueckgeben werden.

Hinweis

Die Formel zur Verwandlung von Grad Celsius (c) in Fahrenheit (f) lautet

$$f = 32 + (9/5) * c$$

References

[1] T. H. E. Meuwissen, B. J. Hayes and M. E. Goddard 2001. Prediction of Total Genetic Value Using Genome-Wide Dense Marker Maps. Genetics 157: 1819âĂŞ1829.