Korrelation und Regression

Schmidt, Sophie C.

26 Juni, 2019

knitr::opts\_chunk$set(  
 collapse = TRUE,  
 warning = FALSE,  
 message = FALSE,  
 echo = TRUE,  
 comment = "#>",  
 fig.path = "../figures/",  
 fig.width=6,   
 fig.height=6  
)

Pakete für heute:

library(dplyr)  
library(tidyr)  
  
  
library(archdata)  
data("BACups")

## Korrelation

Um Zusammenhänge zu erfassen, gibt es für jedes Skalenniveau unterschiedliche Methoden:

* für nominale Daten: Cramérs V
* für ordinale Daten: Kendalls Tau und Spearmans Rho
* für metrische Daten: Pearson-Bravais r

### für nominale Daten kurz wiederholt:

Cramérs V kennen wir schon. Nachdem der Chi-Qudrat-Test einen Zusammenhang aufgezeigt hat, können wir Cramérs V nutzen, um die Stärke des Zusammenhangs zu untersuchen. Chi-Quadrat ist in base vorinstalliert, für Cramérs V brauchen wir das Paket “lsr”.

Wir machen hier die gleichen Schritte, wie vor ein paar Wochen, aber mit etwas anderen Daten. Der EndScraper-Datensatz eignet sich. Schauen wir, ob die Art der Rundung (Curvature) der Kratzer etwas mit der Art der Seiten zu tun hat.

Diese Art der Datenumformung braucht man häufig, deswegen machen wir das hier noch einmal.

# 1. Zusammenfassen der Freq (Häufigkeiten) nach Curvature und Sides.  
end\_curv\_sides <- aggregate(EndScrapers$Freq, by = list(Curv = EndScrapers$Curvature, Sides = EndScrapers$Sides), FUN = sum)  
# dran denken, die Freq-Spalte wird jetzt x genannt  
  
# 2. anschauen  
head(end\_curv\_sides)  
#> Curv Sides x  
#> 1 Round Convergent 110  
#> 2 Medium Convergent 107  
#> 3 Shallow Convergent 87  
#> 4 Round Parallel 554  
#> 5 Medium Parallel 1166  
#> 6 Shallow Parallel 976  
  
# 3. die Angaben unter x werden meine Werte in der Tabelle, die Sides-Angaben sollen zu Rownames werden und die Curvature-Angaben werden meine Spaltenköpfe:  
end\_cs <- end\_curv\_sides %>% spread(value = "x", key = "Curv")  
  
# 4. alles klar?  
head(end\_cs)  
#> Sides Round Medium Shallow  
#> 1 Convergent 110 107 87  
#> 2 Parallel 554 1166 976  
  
# 5. a) rownames ist der Befehl: der data frame end\_sr bekommt die rownames, die unter end\_sr$Sides liegen   
rownames(end\_cs) <- end\_cs$Sides  
# anschauen:  
head(end\_cs)  
#> Sides Round Medium Shallow  
#> Convergent Convergent 110 107 87  
#> Parallel Parallel 554 1166 976  
  
# 5. b) die Spalte löschen, die wir nicht mehr brauchen  
end\_cs <- end\_cs[,-1]  
#anschauen  
head(end\_cs)  
#> Round Medium Shallow  
#> Convergent 110 107 87  
#> Parallel 554 1166 976  
  
chisq.test(end\_cs)  
#>   
#> Pearson's Chi-squared test  
#>   
#> data: end\_cs  
#> X-squared = 38.782, df = 2, p-value = 3.789e-09  
  
cramersV(end\_cs)  
#> [1] 0.1136986

Wie kann man das Ergebnis interpretieren?

### Kendalls Tau

Rechnen wir doch einfach das Bsp aus der Präsentation nach: Wir haben einen Datensatz mit 4 Siedlungen, deren Anzahl von Öfen und Größe. Wir brauchen für den Test zwei Vektoren, die in der richtigen Reihenfolge diese beiden Wertereihen darstellen:

ha <- c(2, 1, 4, 3.5)  
nOfen <- c(2, 3, 5, 1)

Jetzt geben wir diese in die Korrelationstestfunktion ein. R wählt automatisch die benötigte Version (a ist standard und b, wenn es Bindungen gibt). Über “alternative” kann man definieren, ob man schon glaubt, dass die erste oder zweite Gruppe größere Ränge einnimmt als die andere. Wir testen “two.sided”, d.h. wir wissen das nicht:

cor.test(ha, nOfen, method = "kendall", alternative = "two.sided")  
#>   
#> Kendall's rank correlation tau  
#>   
#> data: ha and nOfen  
#> T = 3, p-value = 1  
#> alternative hypothesis: true tau is not equal to 0  
#> sample estimates:  
#> tau   
#> 0

Sie mal einer an! Das Ergebnis ist ein anderes, wenn man meinen Rechenfehler nicht macht.

Ein letztes Wort zu Kendalls Tau: Wenn man sehr viele Daten hat (sehr lange Vektoren), die man testen möchte, kann die Berechnung sehr lange dauern (da ja jedes Paar gegeneinander getestet werden muss). Dann informiert euch über Spearmans Rho (<https://www.crashkurs-statistik.de/spearman-korrelation-rangkorrelation/>), der ist eigentlich wie der folgende (Pearson und Bravais r), aber an Rängen wie Kendalls Tau. Er gilt als “weniger genau”, aber für große Datensätze besser geeignet.

Der Code ist simpel:

cor.test(ha, nOfen, method = "spearman", alternative = "two.sided")  
#>   
#> Spearman's rank correlation rho  
#>   
#> data: ha and nOfen  
#> S = 8, p-value = 0.9167  
#> alternative hypothesis: true rho is not equal to 0  
#> sample estimates:  
#> rho   
#> 0.2

Jetzt aber geht es noch um metrische Daten:

# Pearson und Bravais’ R

Für diesen Test können wir wieder den gleichen Befehl nutzen, müssen nur die Methode ändern. Und den Datensatz, denn mit Vektoren der Länge 4 ist der Pearson-und Bravais’ R nicht glücklich.

Nehmen wir doch also einfach Rand- und Nackendurchmesser der bronzezeitlichen Tassen:

cor.test(BACups$RD, BACups$ND, method = "pearson", alternative = "two.sided")  
#>   
#> Pearson's product-moment correlation  
#>   
#> data: BACups$RD and BACups$ND  
#> t = 53.869, df = 58, p-value < 2.2e-16  
#> alternative hypothesis: true correlation is not equal to 0  
#> 95 percent confidence interval:  
#> 0.9835071 0.9941300  
#> sample estimates:  
#> cor   
#> 0.990154

Das Ergebnis sagt uns folgendes:

Der p-Wert ist sehr sehr klein, er wird über eine Testgröße t berechnet, die den Wert 53,87 hat. Ein dafür wichtiger Freiheitsgrad (df) hat den Wert 58.

Das Konfidenzintervall liegt bei einem R-Wert zwischen 0,98 und 0,99.

Der in diesem Test errechnete Korrelationskoeffizient ist 0,99.

Wir haben also einen sehr sicheren und sehr starken Zusammenhang zwischen Rand- und Nackendurchmesser.

Applaus!

Jetzt erstellen wir noch eine Linare Regression daraus:

## Lineare Regression

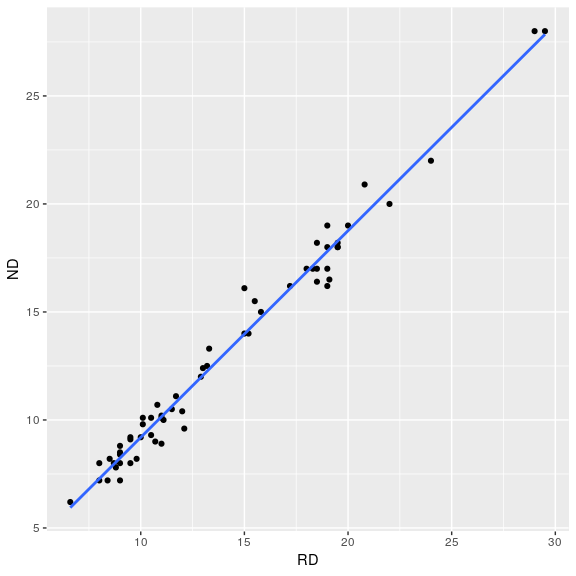
Die lineare Regression legt eine “best-fit”-Linie zwischen die Punkte. Sie soll möglichst gut den Punktverlauf abbilden.

In R ist das ziemlich einfach, in dem ich einem Streudiagramm den geom\_smooth-Befehl mit der Methode “lm” (linear model) hinzufüge. Der Befehl “se = FALSE” sagt aus, dass ich jetzt gern *kein* Konfidenzintervall (standard error) dazu visualisieren möchte.

if (!require("ggpmisc")) install.packages("ggpmisc")  
library(ggpmisc)

library(ggplot2)

ggplot(data = BACups)+  
 geom\_point(aes(x = RD, y = ND))+  
 geom\_smooth( aes(x = RD, y = ND), method = "lm",  
 se = FALSE)



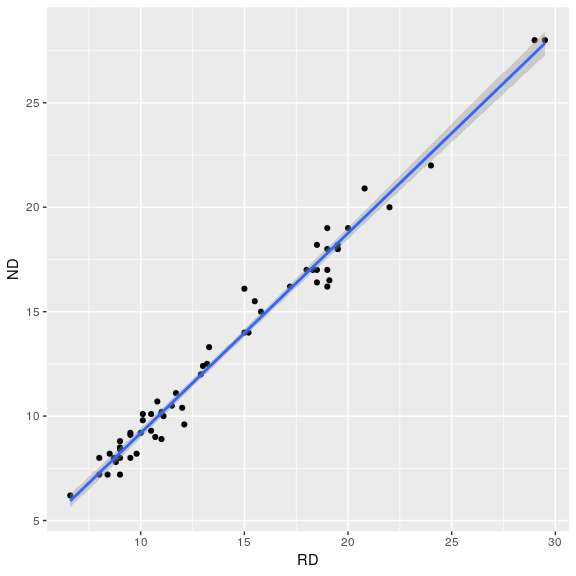
Einfach oder?

Dieses Diagramm können wir jetzt noch ein bisschen verbessern. Wir könnten

1. doch das Konfidenzintervall angeben
2. dazuschreiben, wie diese Linie mathematisch beschrieben werden kann und angeben, wie der R²-Wert der Linie aussieht.

Also, fangen wir an mit 1.:

ggplot(data = BACups)+  
 geom\_point(aes(x = RD, y = ND))+  
 geom\_smooth( aes(x = RD, y = ND), method = "lm",  
 se = TRUE)

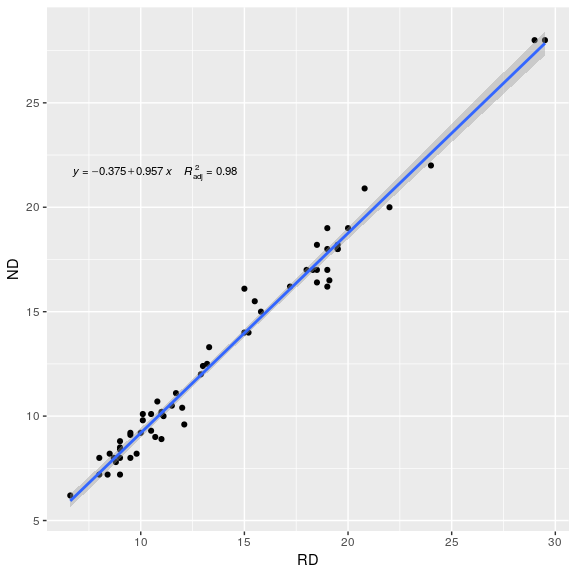
 Denkbar einfach. Das Konfidenzintervall ist sehr schmal, was für eine gute Anpassung der Linie an die Punkte spricht.

Schauen wir uns doch einmal an, wie diese Linie mathematisch beschrieben werden und zusammen mit dem R²-Wert dem plot hinzugefügt werden kann.

Wir brauchen dafür das Paket “ggpmisc”.

Dann fügen wir dem bisherigen Plot (am besten ihr kopiert das bisherige einfach mit strg+c und strg+v) den Befehl “stat\_poly\_eq” hinzu. “Stat\_poly\_eq” kann die Statistik der equation (Formel) einfügen. Dafür braucht es erst einmal noch die Eingabewerte der Regression (x = und y = ), dann die Information, welcher Text als “label” in den Graphen hinzugefügt werden soll: Paste (füge ein) die Formel (..eq.label..,) und den R²-Wert (..adj.rr.label..) und separiere die beiden mit vier Leerzeichen (symbolisiert durch die Tilde). Formula sind eine bestimmte Art von Objekten in R. An dieser Stelle sagt man mit “formula = y~x”, dass y die abhängige Variable sein soll. “parse = TRUE” bedeutet “ja bitte schreib es hin” und “size” gibt die Schriftgröße an. “label.y.npc” platziert die Schrift und zwar auf der y-Achse nach Prozent (also in diesem Fall bei 70% der Y-Achse).

ggplot(data = BACups)+  
 geom\_point(aes(x = RD, y = ND))+  
 geom\_smooth( aes(x = RD, y = ND), method = "lm",  
 se = TRUE)+  
 stat\_poly\_eq(aes(x = RD, y = ND, label = paste(..eq.label..,   
 ..adj.rr.label..,  
 sep = "~~~~")),  
 formula = y~x, # y sei die abhängige Variable  
 parse = TRUE,  
 size = 3,  
 label.y.npc = 0.7)



Cool oder?

Wenn man diese Grafik noch mit Titel und schöner Achsenbeschriftung versieht, hat man echt eine publikationswürdige Grafik.