R-Funktionen

1 R-Funktionen

Diese Seite listet all die R-Funktionen auf die ich schonmal benutzt habe.

Mit ? können wird die Dokumentation von R aufgerufen und weitere zusätzliche Informationen werden angezeigt.

`c, nchar, data, str, dim, names, row.names, head, and tail.`

`mtcars$mpg -- Gibt die Werte der Reihe mpg des Datenframes mtcars aus. (Also $ als Symbol).`

```
mtcars$mpg
```

```
##  [1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3 15.2
## [15] 10.4 10.4 14.7 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.4
## [29] 15.8 19.7 15.0 21.4
```

2 Funktionen die das Zentrum beschreiben

```
# Gibt den Mittelwert der Daten an.
mean(mtcars$mpg)
```

```
## [1] 20.09
```

```
# Gibt den Median der Daten an.
median(mtcars$mpg)
```

```
## [1] 19.2
```

```
# Überblick über alle Daten
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2
##  1st Qu.:12.0   1st Qu.: 26
##  Median :15.0   Median : 36
##  Mean   :15.4   Mean   : 43
##  3rd Qu.:19.0   3rd Qu.: 56
##  Max.   :25.0   Max.   :120
```

2.1 Funktionen um Datenframes zu laden

```r
#Zeigt das Verzeichnis an in welchen wir uns befinden.
getwd()
```

```
## [1] "/Users/user/AllGitHub/ProgrammingKnowledge"
```

```r
#Wechsel des Verzeichnises setwd('~/Downloads') immer in ''.
setwd('~/Downloads')
#cvs-Datei einlesen.
read.csv('reddit.csv')
```

2.1.0.1 Andere Funktionen

```r
subset(stateInfo, pobulation==3100)   # stateInfo ist ein Datenset und population ein
```

```r
# Zeigt einen Überblick an
summary(mtcars)
```

```
##       mpg             cyl             disp             hp       
##  Min.   :10.4   Min.   :4.00   Min.   : 71.1   Min.   : 52.0  
##  1st Qu.:15.4   1st Qu.:4.00   1st Qu.:120.8   1st Qu.: 96.5  
##  Median :19.2   Median :6.00   Median :196.3   Median :123.0  
##  Mean   :20.1   Mean   :6.19   Mean   :230.7   Mean   :146.7  
##  3rd Qu.:22.8   3rd Qu.:8.00   3rd Qu.:326.0   3rd Qu.:180.0  
##  Max.   :33.9   Max.   :8.00   Max.   :472.0   Max.   :335.0  
##       drat             wt             qsec             vs        
##  Min.   :2.76   Min.   :1.51   Min.   :14.5   Min.   :0.000  
##  1st Qu.:3.08   1st Qu.:2.58   1st Qu.:16.9   1st Qu.:0.000  
##  Median :3.69   Median :3.33   Median :17.7   Median :0.000  
##  Mean   :3.60   Mean   :3.22   Mean   :17.8   Mean   :0.438  
##  3rd Qu.:3.92   3rd Qu.:3.61   3rd Qu.:18.9   3rd Qu.:1.000  
##  Max.   :4.93   Max.   :5.42   Max.   :22.9   Max.   :1.000  
##        am              gear            carb      
##  Min.   :0.000   Min.   :3.00   Min.   :1.00  
##  1st Qu.:0.000   1st Qu.:3.00   1st Qu.:2.00  
##  Median :0.000   Median :4.00   Median :2.00  
##  Mean   :0.406   Mean   :3.69   Mean   :2.81  
##  3rd Qu.:1.000   3rd Qu.:4.00   3rd Qu.:4.00  
##  Max.   :1.000   Max.   :5.00   Max.   :8.00
```

```r
# neue Spalte mit year als überschrifft und 1974 für alle
mtcars$year <- 1974
```

```r
# Spalte löschen
mtcars <- subset(mtcars, select = -year)
```

```r
mtcars$wt
```

```
##  [1] 2.620 2.875 2.320 3.215 3.440 3.460 3.570 3.190 3.150 3.440 3.440
## [12] 4.070 3.730 3.780 5.250 5.424 5.345 2.200 1.615 1.835 2.465 3.520
## [23] 3.435 3.840 3.845 1.935 2.140 1.513 3.170 2.770 3.570 2.780
```

```
cond <- mtcars$wt < 3
cond
```

```
##  [1]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [12] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE FALSE
## [23] FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE
```

```
mtcars$weight_class <- ifelse(cond, 'light', 'average')
mtcars$weight_class
```

```
##  [1] "light"   "light"   "light"   "average" "average" "average" "average"
##  [8] "average" "average" "average" "average" "average" "average" "average"
## [15] "average" "average" "average" "light"   "light"   "light"   "light"
## [22] "average" "average" "average" "average" "light"   "light"   "light"
## [29] "average" "light"   "average" "light"
```

```
cond <- mtcars$wt > 3.5
mtcars$weight_class <- ifelse(cond, 'heavy', mtcars$weight_class)
mtcars$weight_class
```

```
##  [1] "light"   "light"   "light"   "average" "average" "average" "heavy"
##  [8] "average" "average" "average" "average" "heavy"   "heavy"   "heavy"
## [15] "heavy"   "heavy"   "heavy"   "light"   "light"   "light"   "light"
## [22] "heavy"   "average" "heavy"   "heavy"   "light"   "light"   "light"
## [29] "average" "light"   "heavy"   "light"
```

```
# entfernt code aus dem arbeitsbereich
rm(cond)
rm(efficient)
```

```
## Warning: Objekt 'efficient' nicht gefunden
```

```
# zeigt die Anzahl der Fahrzeuge mit bestimmten Werten an
table(mtcars$mpg)
```

```
##
## 10.4 13.3 14.3 14.7   15 15.2 15.5 15.8 16.4 17.3 17.8 18.1 18.7 19.2 19.7
##    2    1    1    1    1    2    1    1    1    1    1    1    1    2    1
##   21 21.4 21.5 22.8 24.4   26 27.3 30.4 32.4 33.9
##    2    2    1    2    1    1    1    2    1    1
```

```
# Für Faktrone als Datentypen
levels(reddit$age.range)

# Um eine Plot zu zeichnen
install.packages('ggplot2', dependencies = T)
library(ggplot2)
gplot(data= reddit, x=age.range)
```

2.2 Datentypen

1. Vektoren

Ein Beispiel für Vektoren

```
a <- c(1,2,5.3,6,-2,4) # numeric vector
b <- c("one","two","three") # character vector
c <- c(TRUE,TRUE,TRUE,FALSE,TRUE,FALSE) #logical vector
```

2. Matrizen

Ein Beispiel für Matrizen

```
# generates 5 x 4 numeric matrix
y<-matrix(1:20, nrow=5,ncol=4)

# another example
cells <- c(1,26,24,68)
rnames <- c("R1", "R2")
cnames <- c("C1", "C2")
mymatrix <- matrix(cells, nrow=2, ncol=2, byrow=TRUE,
  dimnames=list(rnames, cnames))
mymatrix[]
```

```
##    C1 C2
## R1  1 26
## R2 24 68
```

3. Arrays

Sind wie Matrizen aufgebaut, nur sind mehrere Dimensionen möglich Arrays are similar to matrices but can have more than two dimensions. See help(array) for details.

4. Data Frames

A data frame is more general than a matrix, in that different columns can have different modes (numeric, character, factor, etc.). This is similar to SAS and SPSS datasets.

```
d <- c(1,2,3,4)
e <- c("red", "white", "red", NA)
f <- c(TRUE,TRUE,TRUE,FALSE)
mydata <- data.frame(d,e,f)
names(mydata) <- c("ID","Color","Passed") # variable names
mydata
```

```
##   ID Color Passed
## 1  1   red   TRUE
## 2  2 white   TRUE
## 3  3   red   TRUE
## 4  4  <NA>  FALSE
```

5. List

An ordered collection of objects (components). A list allows you to gather a variety of (possibly unrelated) objects under one name.

```
# example of a list with 4 components -
# a string, a numeric vector, a matrix, and a scaler
w <- list(name="Fred", mynumbers=a, mymatrix=y, age=5.3)
```

```
# example of a list containing two lists
v <- c(list1,list2)
```

6. Factors

Tell R that a variable is nominal by making it a factor. The factor stores the nominal values as a vector of integers in the range [1... k] (where k is the number of unique values in the nominal variable), and an internal vector of character strings (the original values) mapped to these integers.

- Nominale Variablen

```
# variable gender with 20 "male" entries and
# 30 "female" entries
gender <- c(rep("male",20), rep("female", 30))
gender <- factor(gender)
# stores gender as 20 1s and 30 2s and associates
# 1=female, 2=male internally (alphabetically)
# R now treats gender as a nominal variable
summary(gender)
```

```
## female   male
##     30     20
```

- Ordinale Variablen

An ordered factor is used to represent an ordinal variable.

```
# variable rating coded as "large", "medium", "small'
rating <- c(rep("large"), rep("medium"), rep("small"))
rating <- ordered(rating)

summary(rating)
```

```
##  large medium  small
##      1      1      1
```

```
# recodes rating to 1,2,3 and associates
# 1=large, 2=medium, 3=small internally
# R now treats rating as ordinal
```

R will treat factors as nominal variables and ordered factors as ordinal variables in statistical proceedures and graphical analyses. You can use options in the factor( ) and ordered( ) functions to control the mapping of integers to strings (overiding the alphabetical ordering). You can also use factors to create value labels. For more on factors see the UCLA page.

7. Nützliche Funktionen

```
length(object) # number of elements or components
str(object)    # structure of an object
class(object)  # class or type of an object
names(object)  # names

c(object,object,...)       # combine objects into a vector
cbind(object, object, ...) # combine objects as columns
rbind(object, object, ...) # combine objects as rows

object     # prints the object

ls()       # list current objects
rm(object) # delete an object

newobject <- edit(object) # edit copy and save as newobject
fix(object)               # edit in place
```

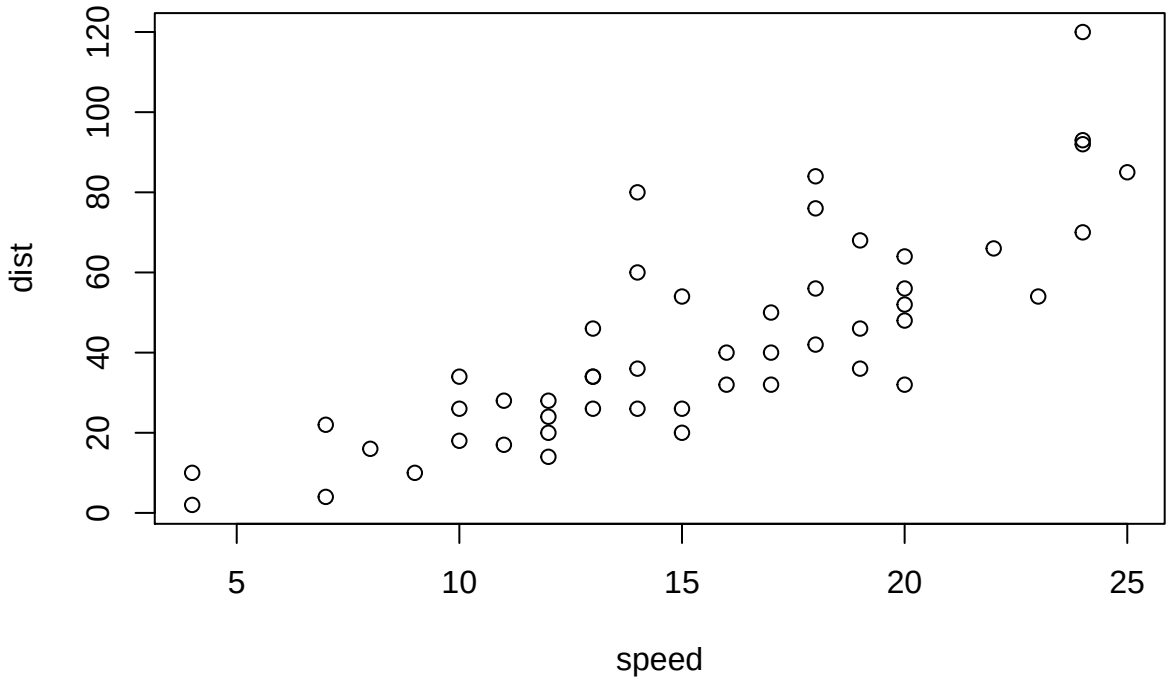
3 Tabellen

Zum Einbinden von Tabellen eignet sich R-Markdown ebenfalls, ess muss nur die Tabelle in der Datei haben ein Beispiel:

```
man kann die Layouts verändern, wenn man dies in die Meta daten der Datei schreibt

rmarkdown::tufte_handout:
  highlight: zenburn

wenn man die Folgenden Daten im Chunkout schreibt wird die Tabele im R ausgeführt.

library(xtable)
options(xtable.comment = FALSE)
options(xtable.booktabs = TRUE)
xtable(head(mtcars[, 1:6]), caption = "First rows of mtcars")
```

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

3.0.0.2 So macht man links

```
[Udacity website](https://www.udacity.com/course/viewer#!/c-ud651/l-729069797/e-804129319/m-811719066)
```

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.